Huawei Cloud Astro Zero

Best Practices

Issue	01
Date	2025-09-04

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2025. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Cloud Computing Technologies Co., Ltd.

Trademarks and Permissions


and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Cloud Computing Technologies Co., Ltd.

Address: Huawei Cloud Data Center Jiaoxinggong Road
Qianzhong Avenue
Gui'an New District
Gui Zhou 550029
People's Republic of China

Website: https://www.huaweicloud.com/intl/en-us/

Contents

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1 Huawei Cloud Astro Zero Best Practices

This section outlines the best practices for common scenarios. Each one includes a clear description and easy-to-follow steps.

Table 1-1 Huawei Cloud Astro Zero best practices

Best Practice	Description
Associating Customer Information with Order Data and Synchronizing the Modification	In certain order systems, customer information and order data need to be associated to process orders and update inventory. This practice explains how to associate these two objects to keep their data in sync.
Adding, Deleting, and Modifying Object Data in Frontend Tables	This practice describes how to add, delete, and modify object data on a frontend page by adding a toolbar.
Limiting Form Submissions with Scripts	When creating frontend pages, you can use scripts to limit form submissions. This improves user experience and data security. Set a delay time in the script. If a form is submitted before the delay, it shows "Submission failed: Not PortalUser!" If submitted after the delay, it shows "Submission failed: Submitted too late."
Adding and Deleting Table Data with Scripts	This practice describes how to use scripts to add and delete table data.
Using the File Template Function to Generate Contracts	Huawei Cloud Astro Zero enables quick creation of **Word**, **Excel**, **Email**, and **SMS** templates, enhancing document efficiency and standardization. This practice uses a **Word** template as examples. You can create print templates as required.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Best Practice	Description
Using Email Templates to Send Email Notifications	Huawei Cloud Astro Zero enables quick creation of **Word**, **Excel**, **Email**, and **SMS** templates, enhancing document efficiency and standardization. This practice uses an **Email** template as an example. Developers can set the email format and framework in advance to avoid repetitive work.
Creating a Business Trip Approval Workflow Using Templates	Huawei Cloud Astro Zero has developed its own Business Process Management (BPM), following the BPMN 2.0 industry standards. This practice walks you through creating a business trip approval workflow to familiarize you with Huawei Cloud Astro Zero's BPMs.
Implementing the Priority Approval Function with User Tasks in BPMs	This practice explains how to configure approval and termination actions for a user task in a BPM to enable priority approval during countersigning.
Adding Links to Data in Standard Page Tables	On standard pages, adding hyperlinks to data in tables enhances user experience and interaction. This practice explains how to move the cursor to the "WEB A" entry in the "webName" column of the table. The corresponding link will appear in the lower left corner of the page, and clicking it will take you to the target page.
Adding Computing Capabilities, Such as Summation, to Standard Page Tables	On standard pages, tables can use computing functions such as sum and product to boost efficiency. This practice explains how to set the value in the "Cost" column to "Number of Products × Price + Other Costs".
Displaying Images in Standard Page Tables	Displaying images in a table makes information more intuitive, vivid, and easier to understand. This practice explains how to show images in a table by customizing column display types on a standard page.
Reusing a Submitted Form Page on an Approval Page	You can create pages for submitting and approving data in a table for employee trips, leave requests, or goods movement. You can also approve expense requests. If the pages have similar content, you can use the same functions on one page. This saves time and makes things easier.
Customizing Standard Page Widget Packages	You can customize standard page widgets according to this practice. This lets you add features to the platform and build applications faster and simpler. This practice shows you how to customize using the imgButton widget as an example.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Best Practice	Description
Customizing Widget Properties on Pages	This practice explains how to customize text, checkbox, and select properties for the **widget_demo_property** widget.
Configuring Chinese and English Language Properties for Widgets	You can configure the multi-language property for a widget to ensure it displays correctly in different languages. To do this, you need to modify the internationalization resource file (i18n). This practice explains how to set up Chinese and English for the **widget_demo_i18n**.
Creating Multi-Device Compatible Advanced Pages	Huawei Cloud Astro Zero provides two terminal views (computer and mobile), responsive layouts, and stretch functions to help advanced pages adapt to different device sizes. This practice shows how to develop an item list widget that meets these requirements.
Referencing Third-Party Libraries to Develop Advanced Pages	Libraries are third-party dependencies that enable advanced page widgets to run. Without them, these widgets cannot function properly. This practice explains how to integrate the third-party library MintUI to simplify widget development and enhance their capabilities.
Implementing Image Display and URL Redirection with the Banner Widget on Advanced Pages	The banner widget on advanced pages automatically switches between multiple images and supports adding hyperlinks to individual images. This practice explains how to add a hyperlink to an image in the banner widget, allowing users to click the image and navigate to the help document of Huawei Cloud Astro Zero.
Using Petal Charts to Display Order Data on Advanced Pages	Advanced pages support dynamic data, which is generated in real-time through API calls from scripts, flows, and objects. This practice explains how to display order data in a rose pie chart on an advanced page.
Uploading and Recognizing ID Card Images with a Connector	Huawei Cloud Astro Zero encapsulates various connectors to integrate with external services, enabling their use within applications. This practice explains how to use the OCR connector to recognize and store ID card information.
Uploading Files With Connectors	Huawei Cloud Astro Zero encapsulates various connectors to integrate with external services, enabling their use within applications. This practice shows how to use the OBS connector to save files from a frontend page to an OBS bucket.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Best Practice	Description
Connecting to Third-Party Databases with a Connector	This practice describes how to use a custom connector to integrate third-party databases into Huawei Cloud Astro Zero.
Developing a Login Page for Portal Users	Huawei Cloud Astro Zero provides a default login page for each application. This practice explains how to customize a login page.
Using the AI Assistant Widget to Build a Knowledge Base	If you have a large number of documents that are hard to use and query efficiency is low, you can use the AI assistant widget to call the foundation model API to get answers. Additionally, RAG technology works with the knowledge base to provide a more intelligent and natural dialog experience, improving the accuracy and relevance of information retrieval.
Calling the Foundation Model Connector to Talk with the AI Assistant	When you build a standard page, you can connect to an external foundation model API via the AI assistant widget. First, set up the foundation model connection details in the connector, then choose that connector in the AI assistant widget. This practice shows you how to configure the connector and use it in the widget for smart chat functionality.
Setting Option Associations	In applications for data entry, queries, and process approvals, associated filters and option limits are often used to maintain data consistency and reduce user errors. This practice explains how to use associate options.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2 Objects

2.1 Associating Customer Information with Order Data and Synchronizing the Modification

Expected Results

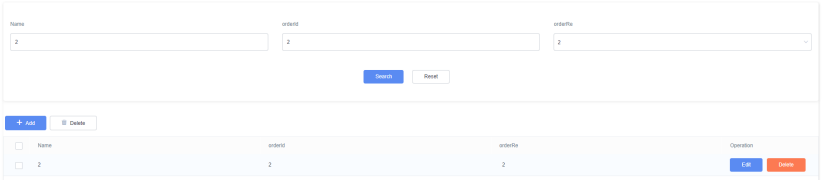
In some order systems, customer information and order data need to be associated for order processing and inventory deduction. For example, order application **A** has two objects: **customerList** and **orderList**. You can establish an association between these two objects to implement the following functions:

- Add and display customer object data.

  Figure 2-1 Adding customer object data

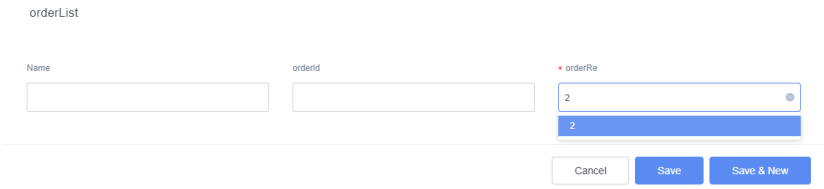
  Figure 2-2 Displaying customer object data

- Add order data and associate it with the existing customer information.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 2-3 Adding order data and associating it with the existing customer information



- After the customer data is deleted, the associated order data is also deleted.

Figure 2-4 Deleting customer data

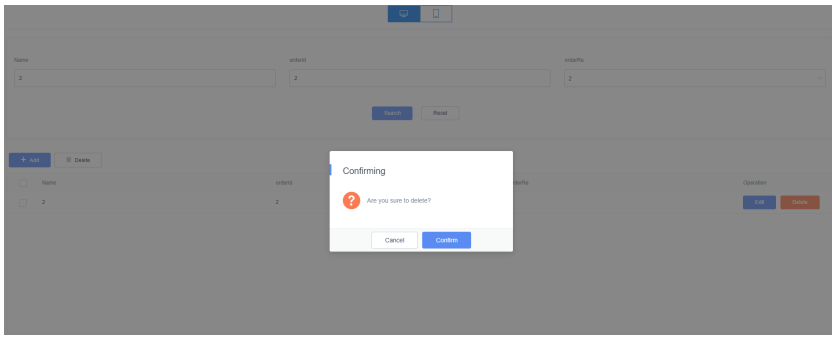
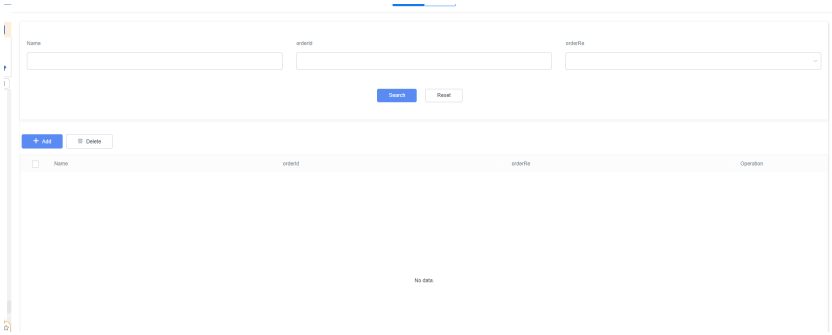


Figure 2-5 Associated order data deleted



Procedure

Step 1 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

Figure 2-6 Creating a blank application

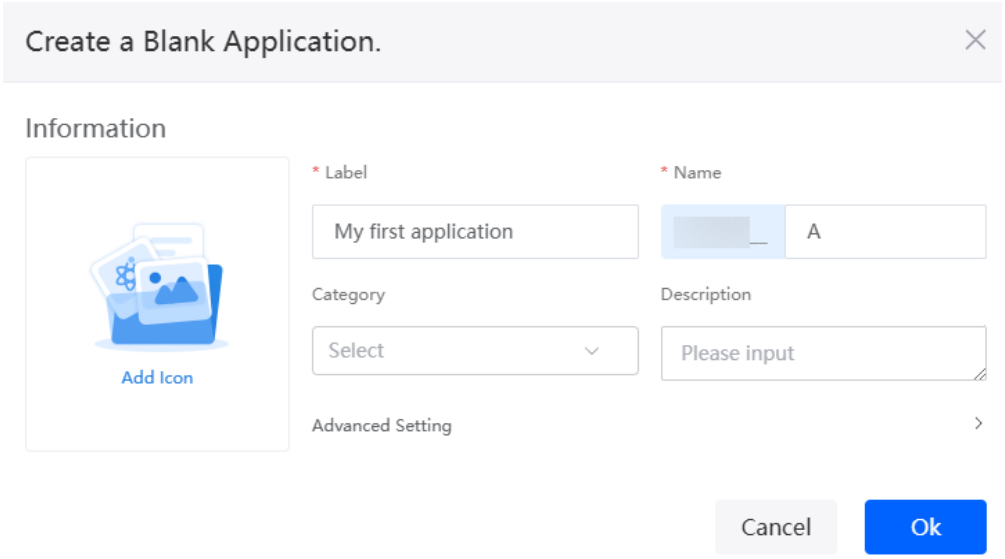


Table 2-1 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 2 Create the customer object **customerList** and the order object **orderList**, and add fields to the objects.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1. In the navigation pane, choose **Data**, and click **+** next to **Object**.
2. Complete the configuration and click the confirm button.

Figure 2-7 Creating the object customerList

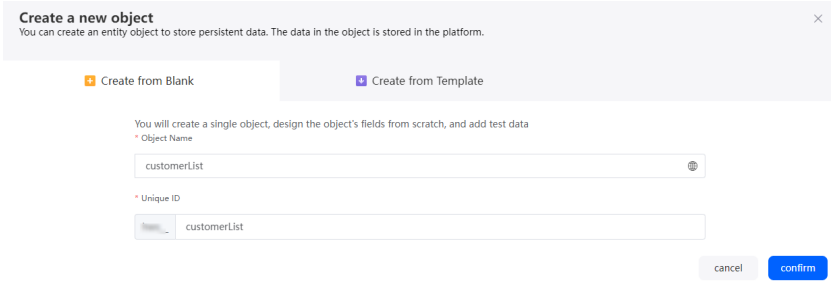


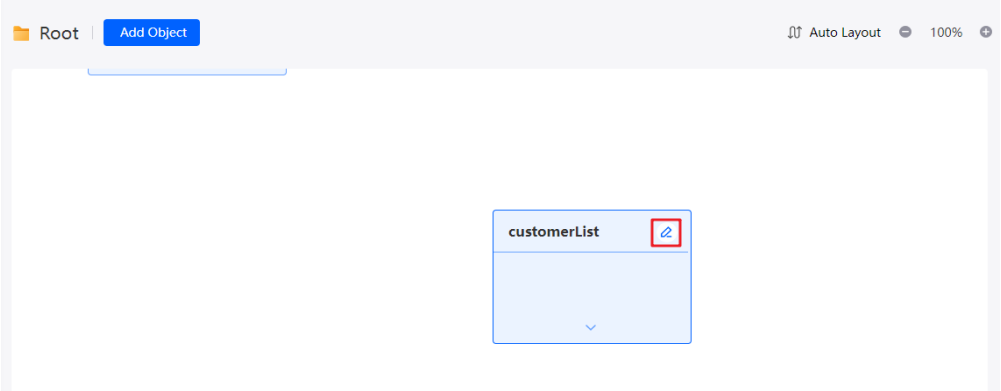
Table 2-2 Parameters for creating the customerList object

Parameter	Description	Example
Object Name	Name of the new object, which can be changed after the object is created. Value: 1–80 characters.	customerList
Unique ID	ID of a new object in the system, which cannot be modified after being created. Naming rules: – Max. 63 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	customerList

3. Click  to go to the object details page.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 2-8 Clicking the edit button



4. On the **Fields** tab page, click **Add** and add the **customerId** field for the object.

Figure 2-9 Adding the customerId field

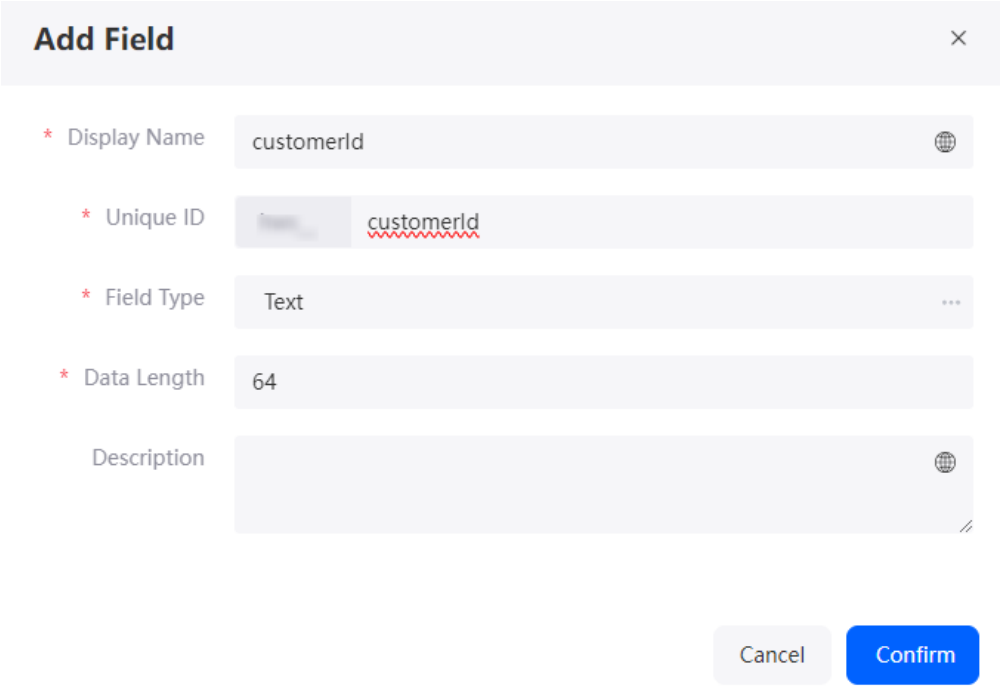


Table 2-3 Parameters for adding the customerId field

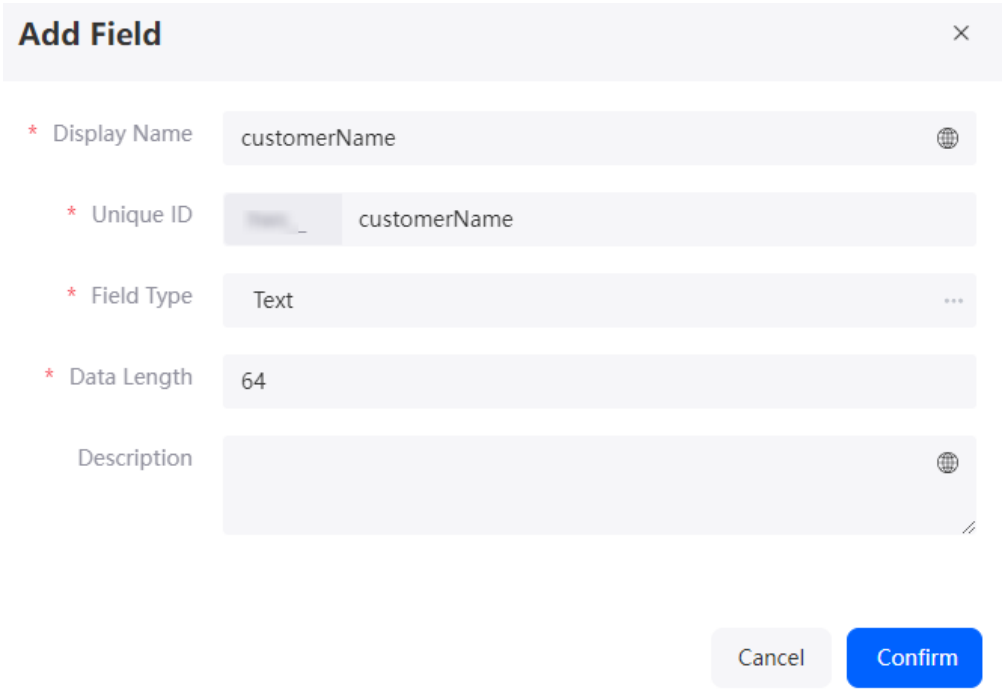
Parameter	Description	Example
Display Name	Name of the new field, which can be changed after the field is created. Value: 1–63 characters.	customerId

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Unique ID	ID of a new field in the system. The value cannot be changed after the field is created. Naming rules: – Max. 63 characters, including the prefix namespace. – Start with a letter and use only letters, digits, and an underscore (_). Do not end with an underscore (_).	customerId
Field Type	Click ***. On the page that is displayed, select the type of the new field based on the parameter description.	Text
Data Length	Length of a field that can be entered.	64

5. On the **Fields** tab page, click **Add** again to add the **customerName** field.

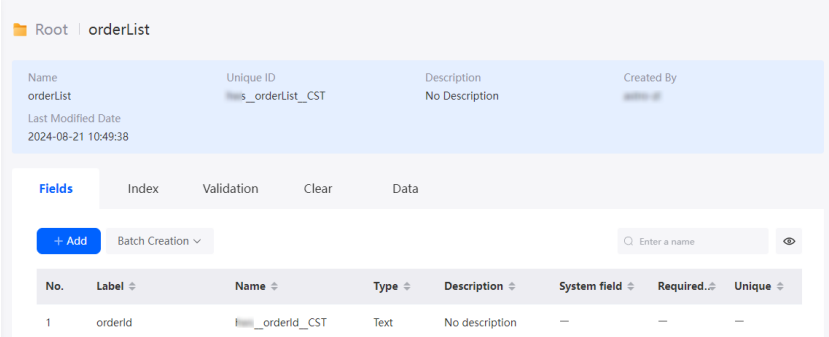
Figure 2-10 Adding the customerName field



6. Repeat the preceding operations to create the order object **orderList** and add the **orderId** field to it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

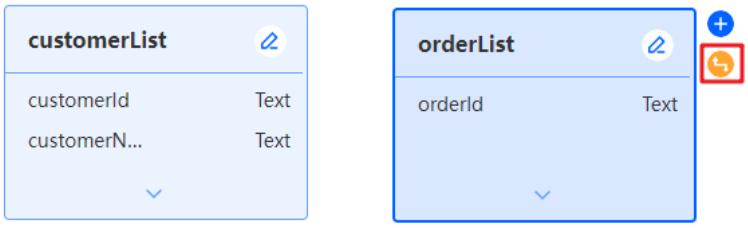
Figure 2-11 Creating the orderList object and adding a field



Step 3 Select the order object and add an association.

1. Click  next to the object. The page for adding an association is displayed.

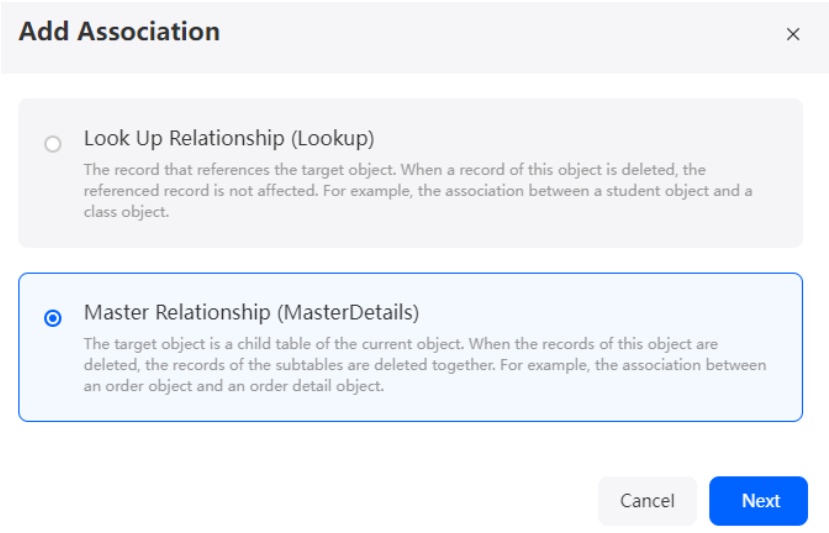
Figure 2-12 Clicking the association button



2. Select **Master-Detail Relationship** and click the next button.

   Master-Detail Relationship: You can establish a master/slave relationship between objects by linking the current field to another object's ID. After a master relationship is defined, the value of the current field can be obtained only from the associated master object. When the records of the object are deleted, the records of the subtables are also deleted.

Figure 2-13 Selecting the master relationship



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3. Add the association and click **Confirm**.

Figure 2-14 Adding an association

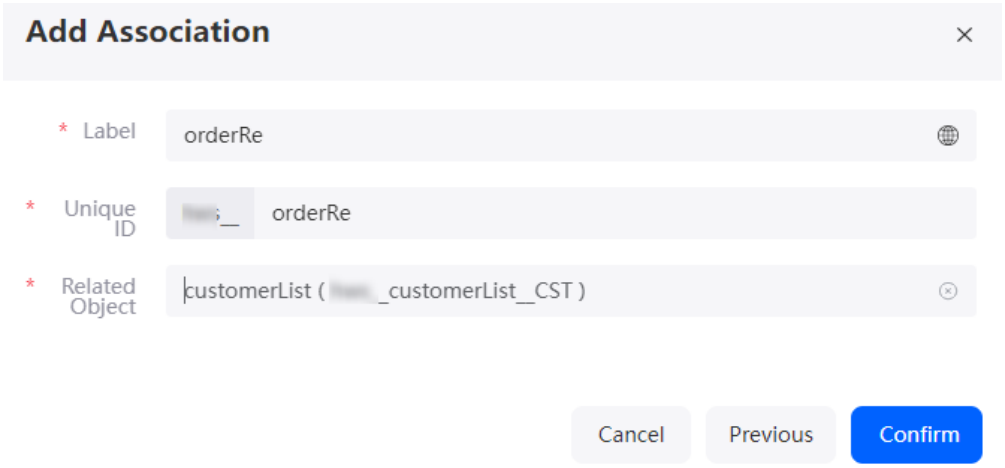


Table 2-4 Parameters for adding the orderRe association

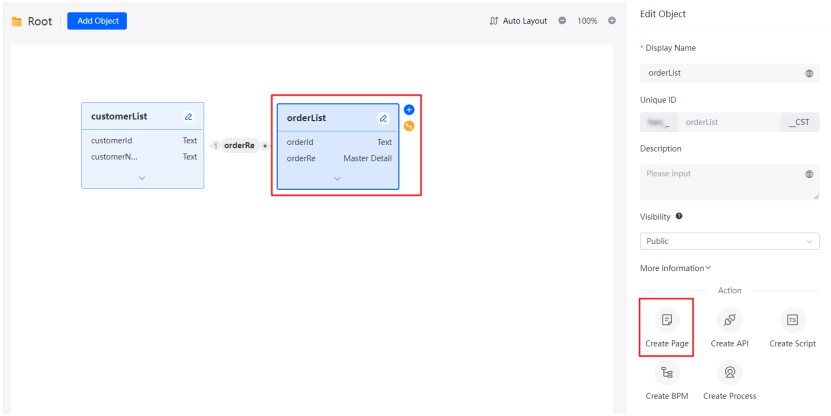
Parameter	Description	Example
Label	Name of the association displayed on the page. The name can be changed after the association is created. Value: 1–80 characters.	orderRe
Unique ID	Unique ID of the association in the system, which cannot be modified after being created. Naming rules: – Max. 63 characters, including the prefix namespace. – Start with a letter and use only letters, digits, and an underscore (_). Do not end with an underscore (_).	orderRe
Related Object	Select an object from the drop-down list.	Select the customer object **customerList** created in **Step 2**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 4 Create pages for the customer and order object

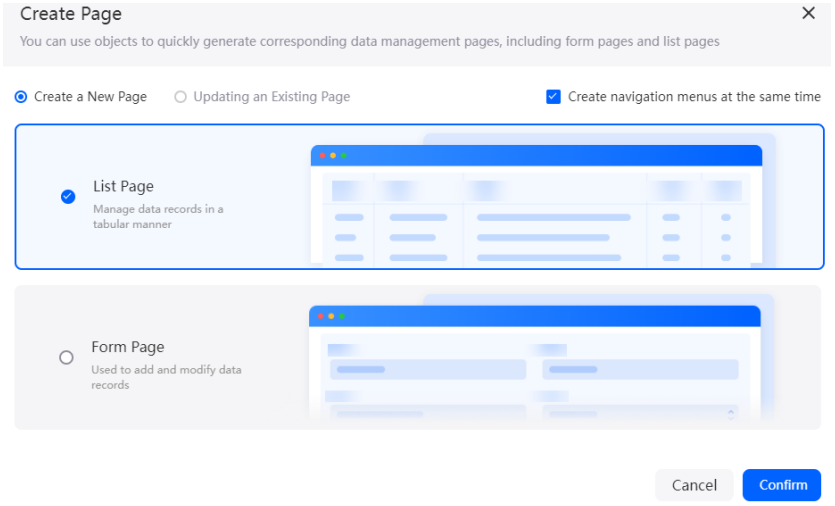
1. Select the order object **orderList** and choose **Action** > **Create Page** in the object editing area on the right.

   Figure 2-15 Clicking Create Page



2. Deselect **Form Page** and retain only **List Page**. Click **Confirm**.

   Figure 2-16 Creating a list page for the orderList object



   After the page is created, a message shows that the page is ready. Choose **Page** > **Standard Page** to view the created list page.

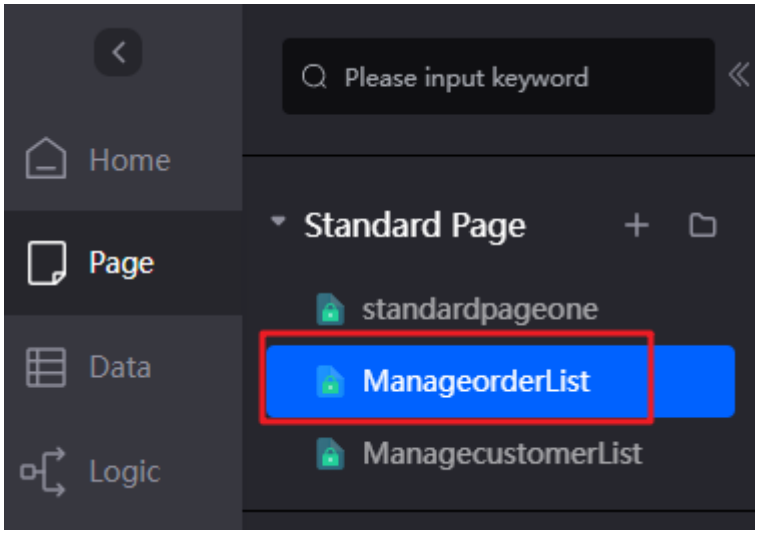
3. Use the same method to create a list page for the customer object **customerList**.

Step 5 On the order object page, set the fields to be displayed for the parent object.

1. In the navigation pane, choose **Page**.
2. Under **Standard Page**, choose **ManageorderList**.

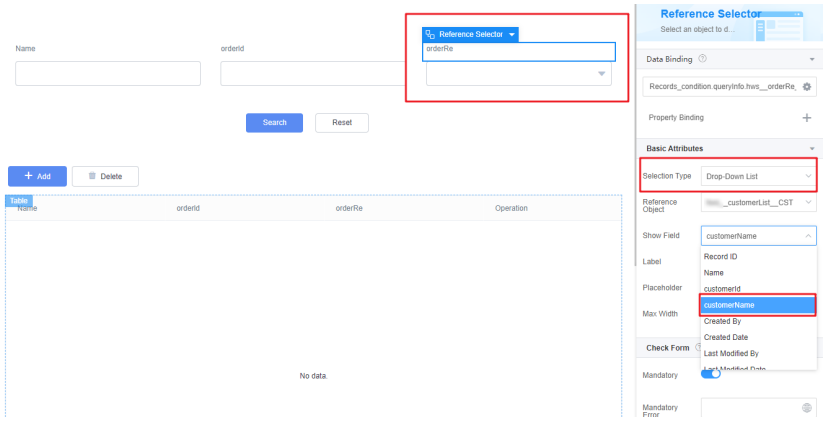
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 2-17 Clicking the order object page orderList



3. Select the **orderRe** field, set **Selection Type** to **Drop-Down List** and **Show Field** to **customerName** in the **Properties** > **Basic Attributes** area.

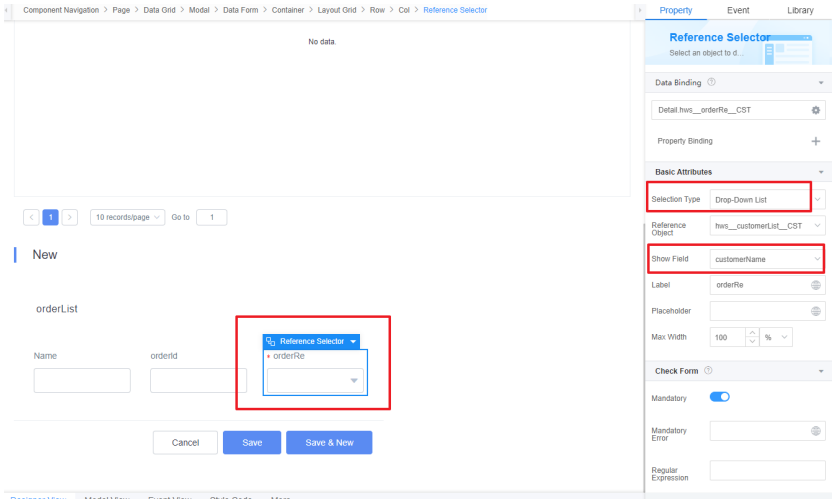
Figure 2-18 Setting the fields to be displayed for the parent object



4. On the **New** area, select the **orderRe** field, set **Selection Type** to **Drop-Down List** and **Show Field** to **customerName** in the **Properties** > **Basic Attributes** area.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 2-19 Setting the fields to be displayed for the parent object on the New area



Step 6 Click  in the upper part of the page to save the page settings.

Step 7 After the configuration is saved, click  in the upper part of the page to view the configuration effect.

----End

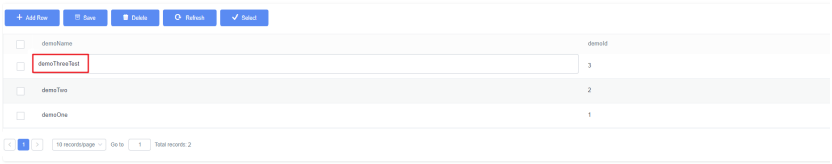
2.2 Adding, Deleting, and Modifying Object Data in Frontend Tables

Expected Results

Add a toolbar to add, delete, and modify object data on the frontend page. The background object data associated with the page will change accordingly.

Implementation effect: Double-click the data bar in **demoName** column to edit the data. After editing, click the save button. The data in the table is updated and the background object is also updated. Select data and click the delete button. After the data is deleted, the data in the background object is also deleted.

Figure 2-20 Updating data on the page



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 2-21 Object data being updated accordingly

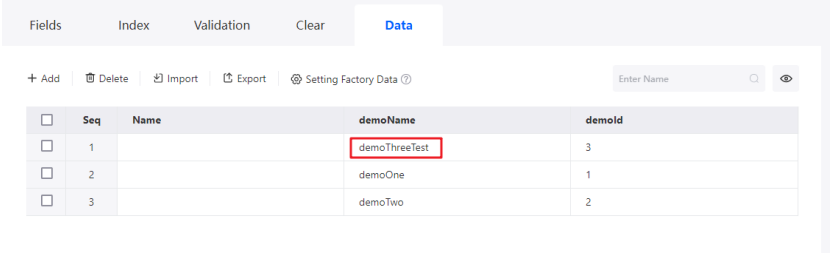


Figure 2-22 Deleting data

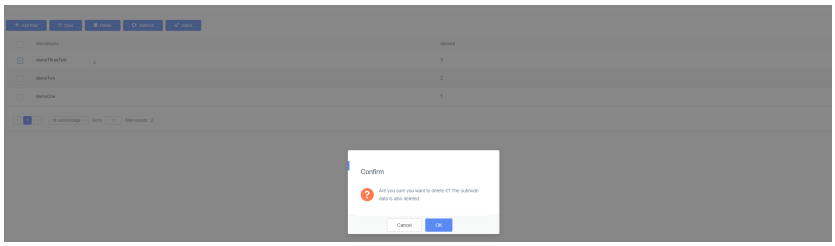


Figure 2-23 Page data being deleted

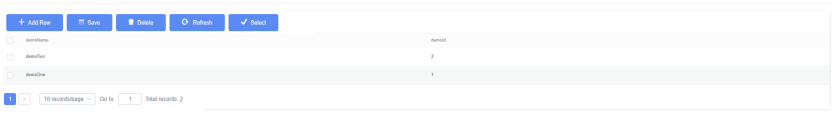
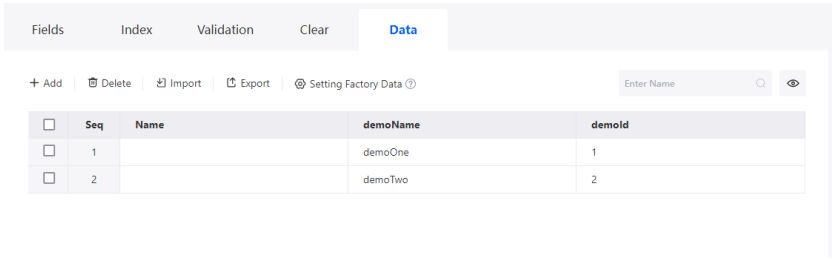


Figure 2-24 Object data being deleted



Implementation Methods

Step 1 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

Figure 2-25 Creating a blank application

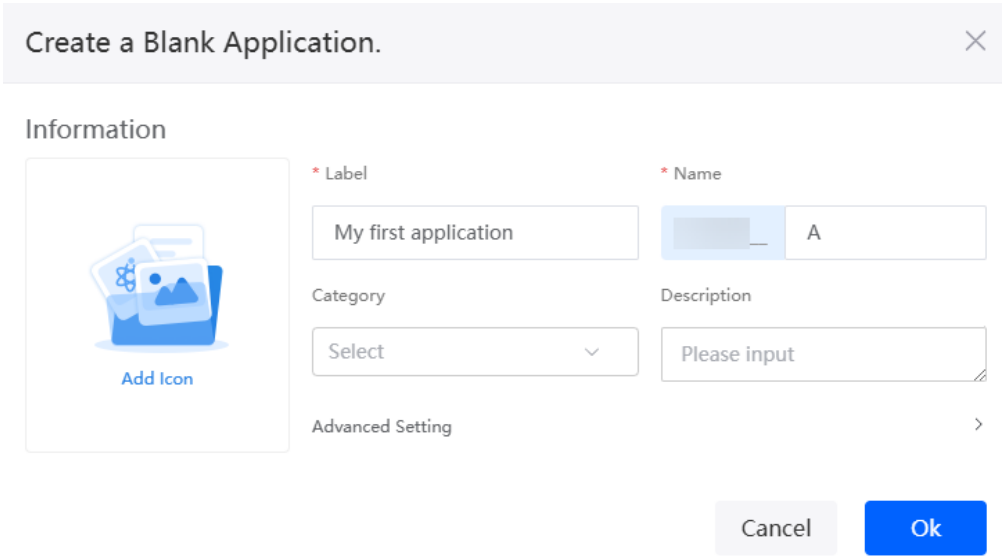


Table 2-5 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 2 Create an object named **demoData** and add fields and data to the object.

1. In the navigation pane, choose **Data**, and click **+** next to **Object**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2. Set **Object Name** and **Unique ID** to **demoData** and click the confirm button.

Figure 2-26 Creating object demoData

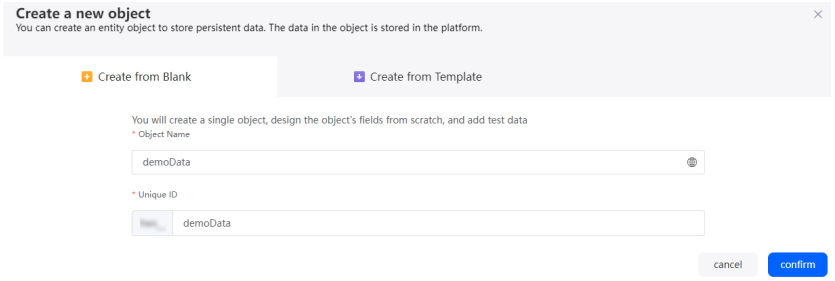


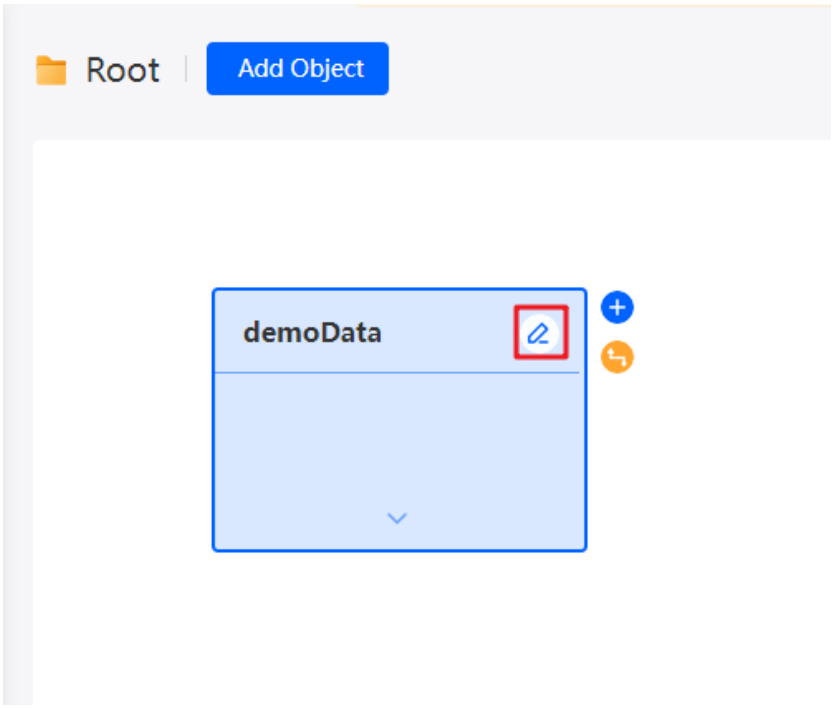
Table 2-6 Parameters for the created object demoData

Parameter	Description	Example
Object Name	Name of the new object, which can be changed after the object is created. Value: 1–80 characters.	demoData
Unique ID	ID of a new object in the system, which cannot be modified after being created. Naming rules: • Max. 63 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. • Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	demoData

3. Click  to go to the object details page.

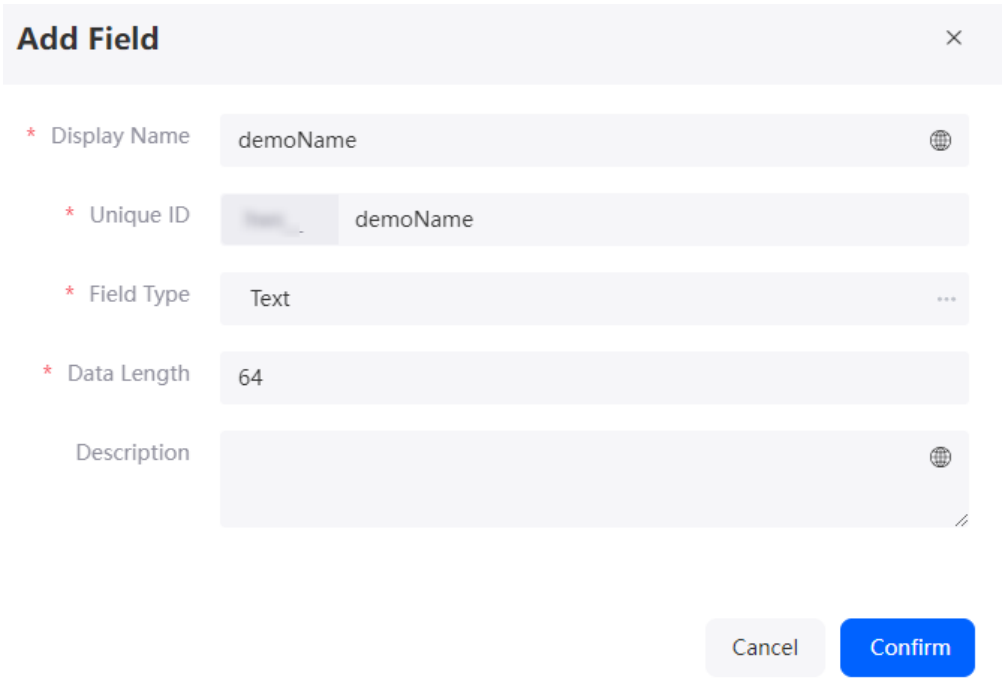
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 2-27 Clicking the edit button



4. On the **Fields** tab page, click **Add** and add the **demoName** field for the object.

Figure 2-28 Adding the demoName field



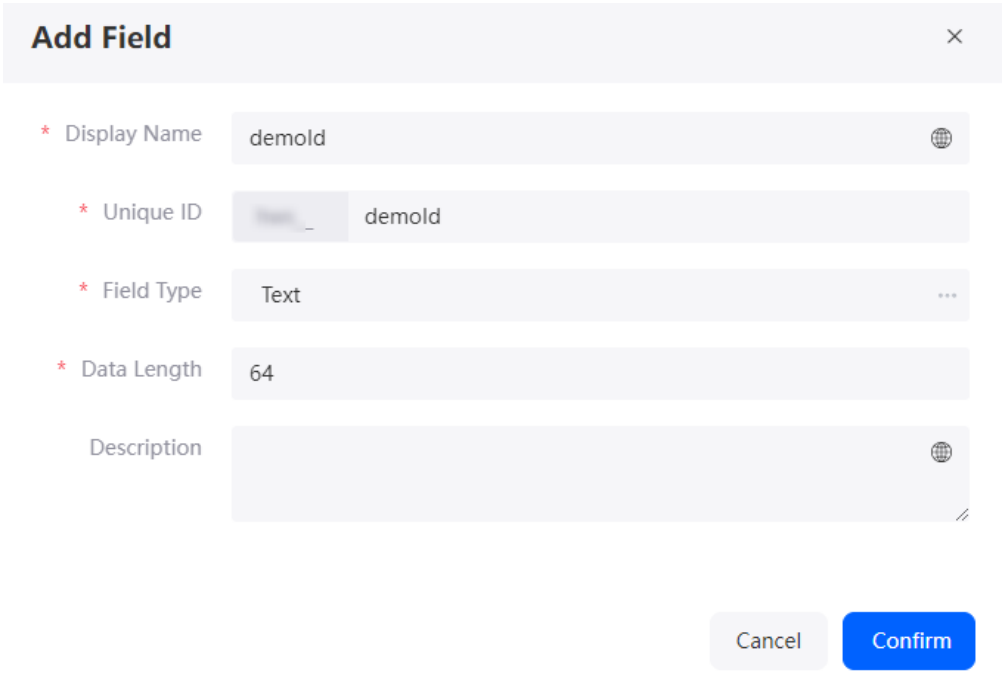
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-7 Parameters for adding the demoName field

Parameter	Description	Example
Display Name	Name of the new field, which can be changed after the field is created. Value: 1–63 characters.	demoName
Unique ID	ID of a new field in the system. The value cannot be changed after the field is created. Naming rules: – Max. 63 characters, including the prefix namespace. – Start with a letter and use only letters, digits, and an underscore (_). Do not end with an underscore (_).	demoName
Field Type	Click ***. On the page that is displayed, select the type of the new field based on the parameter description.	Text

5. On the **Fields** tab, click **Add** again to add the **demoId** field.

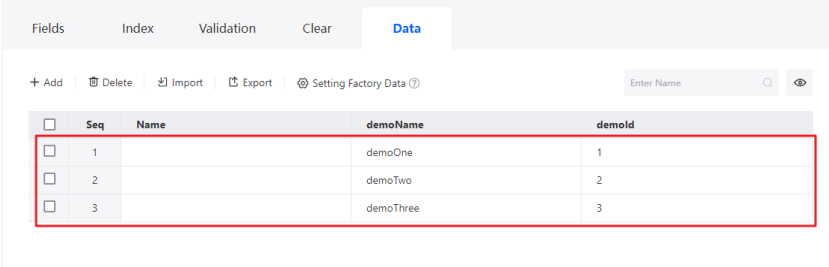
Figure 2-29 Adding the demoId field



6. On the **Data** tab, click **Add** to add data to the object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 2-30 Adding data to an object



Step 3 Create an object model.

1. In the navigation pane, choose **Page**, and click **+** next to **Standard Page**.
2. Enter a label and name and click **Add** to create a standard page.

Figure 2-31 Creating a standard page

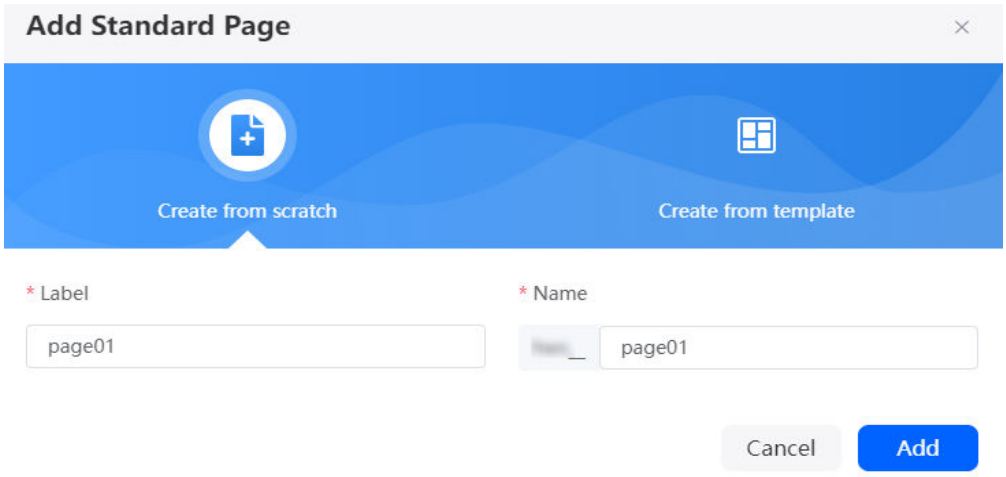


Table 2-8 Parameters for creating a standard page

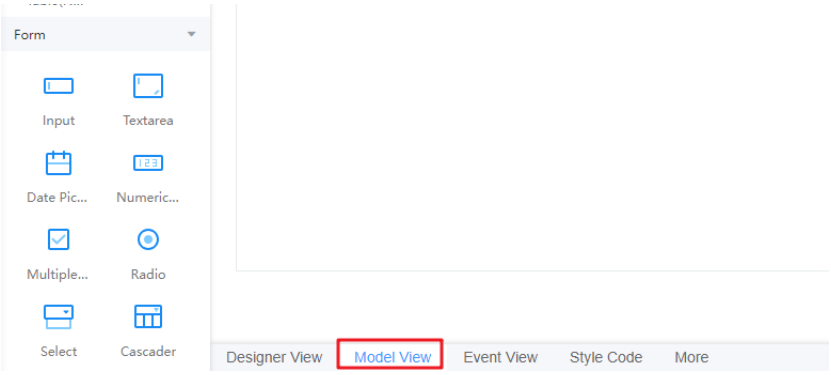
Parameter	Description	Example
Label	Label name of a standard page, which can be changed after being created. Value: 1–64 characters.	page01

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Name	Name of the standard page. The name is the unique identifier of the standard page in the system and cannot be changed after being created. Naming rules: – Max. 64 characters, including the prefix namespace. – Start with a letter and use only letters, digits, and an underscore (_). Do not end with an underscore (_).	page01

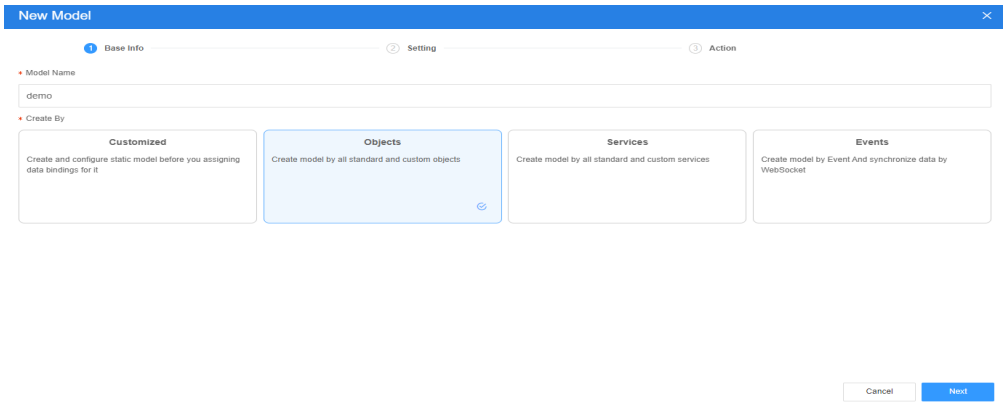
3. At the bottom of the standard page, click **Model View**.

Figure 2-32 Clicking Model View



4. On the displayed page, click **New**. On the displayed page, specify **Model Name** (for example, **demo**), set **Source** to **Objects**, and click **Next**.

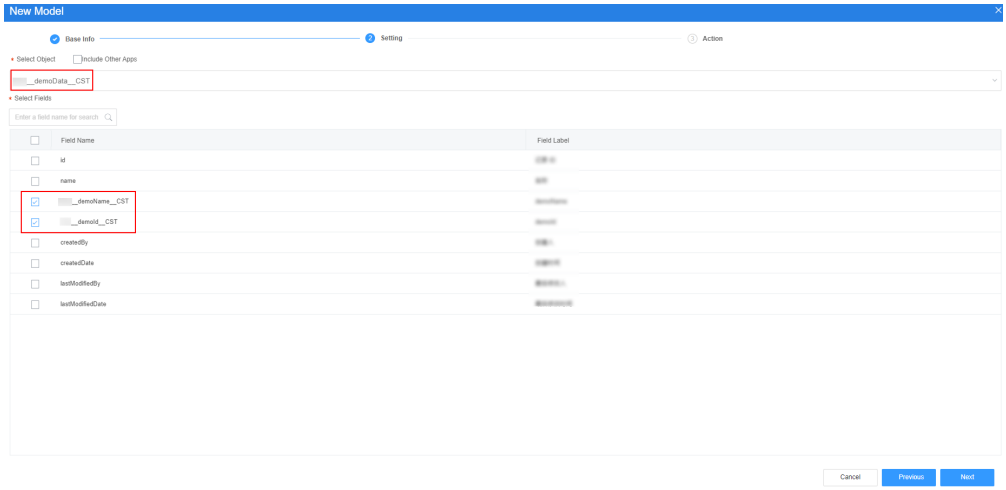
Figure 2-33 Creating a model



5. Select the object created and the fields added in **Step 3**, and click **Next**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 2-34 Selecting objects and fields

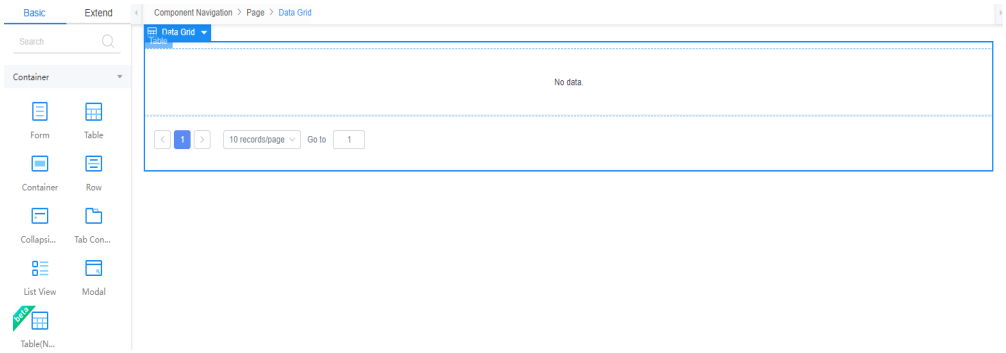


6. Click **OK**.

Step 4 Return to the **Designer View** page and create a table to bind the model.

1. At the bottom of the standard page, click **Designer View**.
2. Drag a table widget to the standard page.

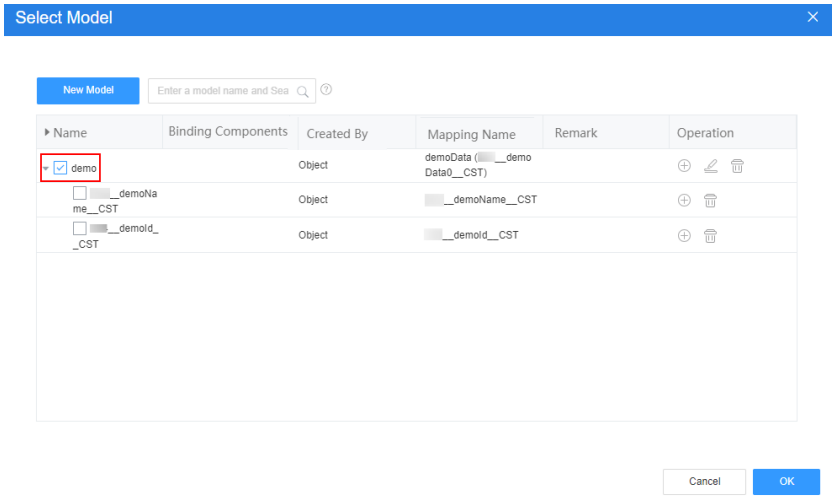
Figure 2-35 Dragging a table widget



3. Select the table widget, choose **Properties** > **Data Binding** and click next to **Value Binding**.
4. Select the model created in **Step 3** and click the confirm button.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

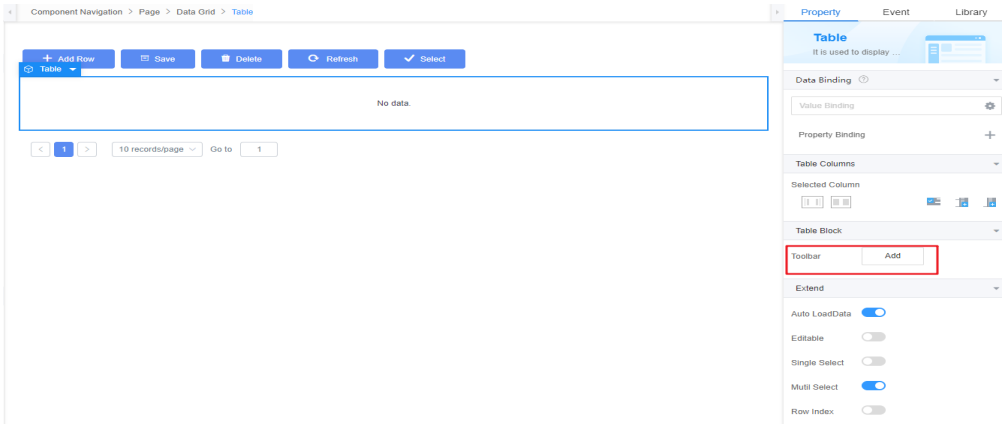
Figure 2-36 Selecting the model



Step 5 Add a toolbar.

1. Select the table widget, choose **Properties** > **Table Block**, and click **Add** next to **Toolbar** to add a toolbar.

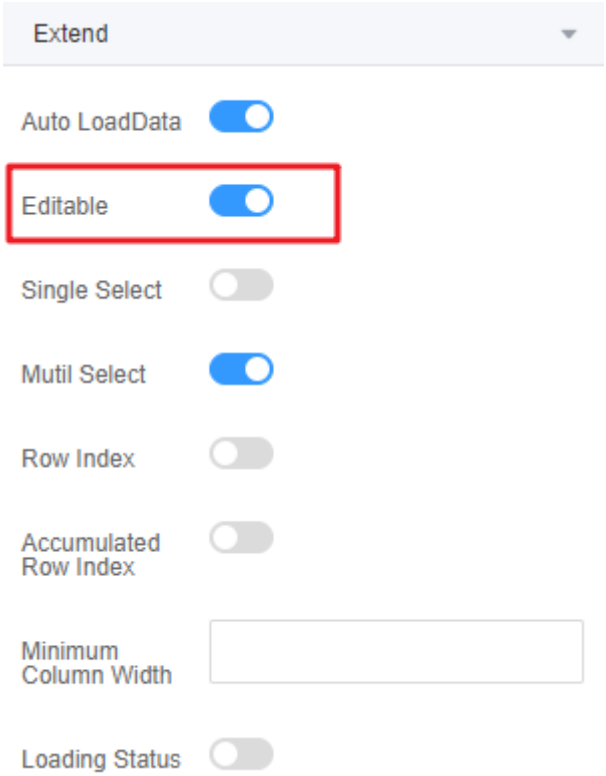
Figure 2-37 Adding a toolbar



2. Expand **Extended Property** to enable **Editable**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

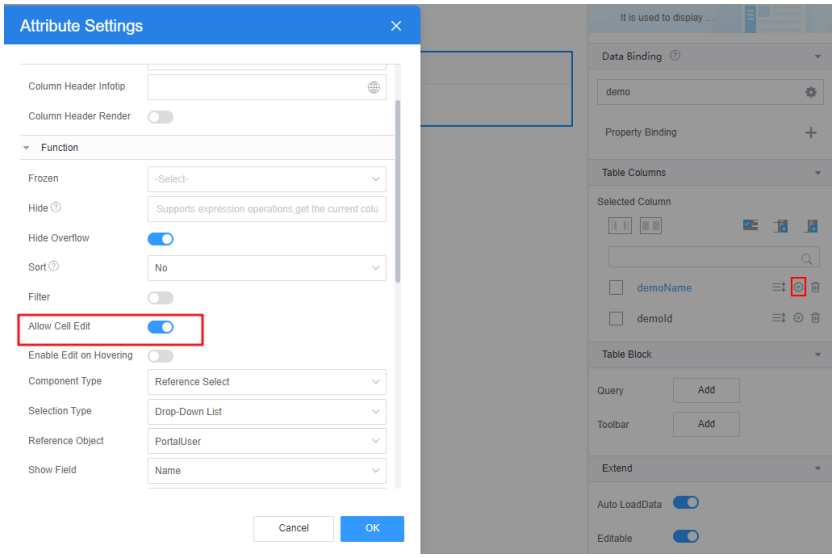
Figure 2-38 Enabling Editable



3. Set the **demoName** and **demoId** columns to be editable.

   Click ⚙ next to the **demoName** column to make it editable. Repeat the preceding operations to set the **demoId** column to be editable.

Figure 2-39 Setting the demoName column to be editable



Step 6 Click 💾 in the upper part of the page to save the page settings.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 7 After the configuration is saved, click  in the upper part of the page to view the configuration effect.

----End

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3 Scripts

3.1 Limiting Form Submissions with Scripts

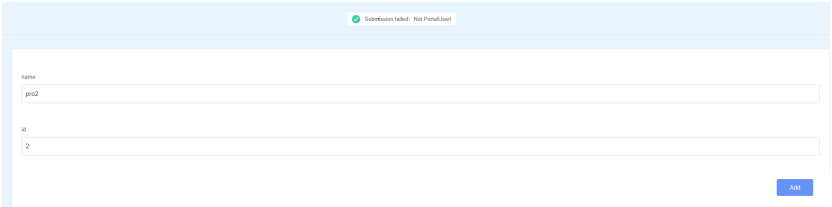
Expected Results

When developing a frontend page, you can add some submission restrictions to the form in the script to improve user experience and data security. For example, you can define a delay time in the script as shown in **Figure 3-1**. When a form is submitted within the specified time, the **Submission failed: Not PortalUser!** is displayed as shown in **Figure 3-2**. When the specified time is exceeded, the **Submission failed: Submitted too late** is displayed as shown in **Figure 3-3**.

Figure 3-1 Script restriction

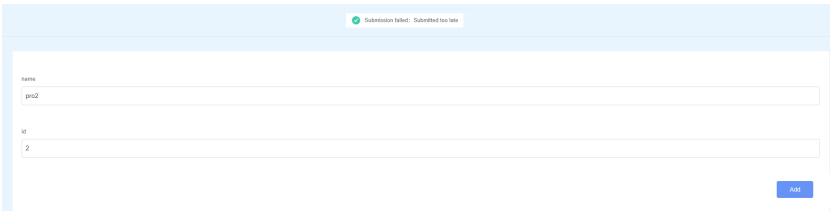
```
try {
    let currentTime = now();
    let date = toDate('2024-04-08 20:08:08', 'yyyy-MM-dd HH:mm:ss');
    if (date.getTime() < currentTime.getTime()) { //
        error.name = "WOERROR";
        error.message = "Submitted too late";
        throw error;
    }
```

Figure 3-2 Submission failed: Not PortalUser!



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 3-3 Submission failed: Submitted too late



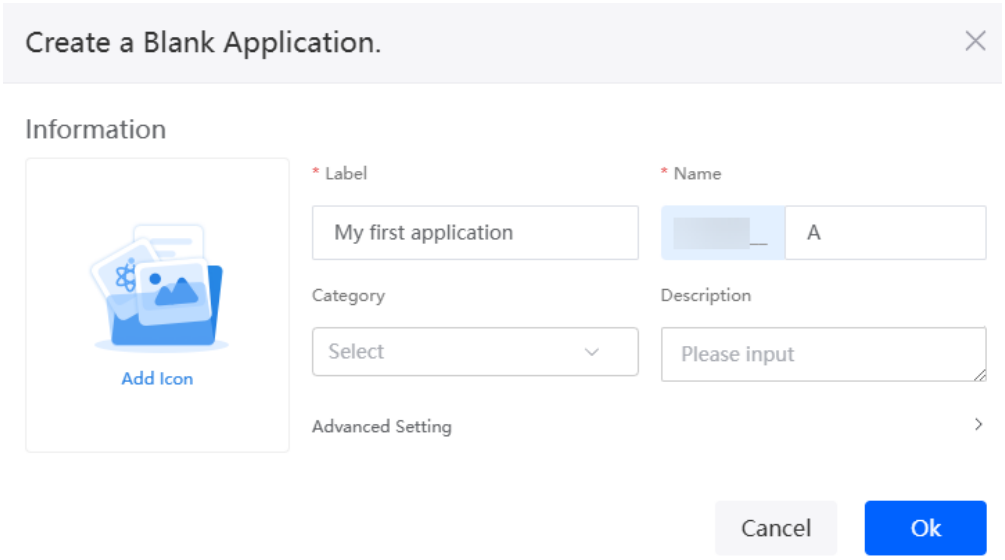
Implementation Methods

Step 1 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

Figure 3-4 Creating a blank application



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

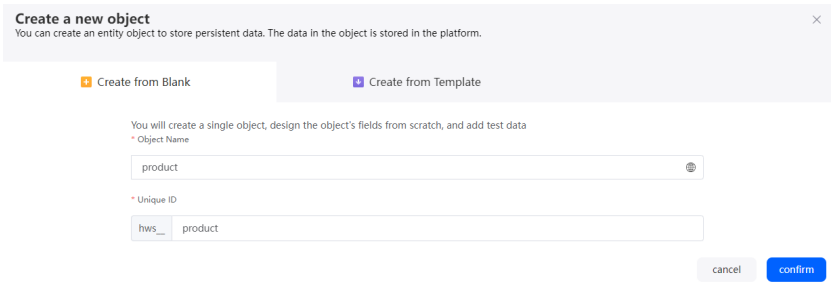
Table 3-1 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 2 Create an object named **product** and add fields to the object.

1. In the navigation pane, choose **Data**, and click **+** next to **Object**.
2. Set **Object Name** and **Unique ID** of the object to **product** and click the confirm button.

Figure 3-5 Creating object product



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-2 Parameters for the created object product

Parameter	Description	Example
Object Name	Name of the new object, which can be changed after the object is created. Value: 1–80 characters.	product
Unique ID	ID of a new object in the system, which cannot be modified after being created. Naming rules: • Max. 63 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. • Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	product

3. Click  to go to the object details page.
4. On the **Fields** tab page, click **Add** and add the **ProName** field for the object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 3-6 Adding the ProName field

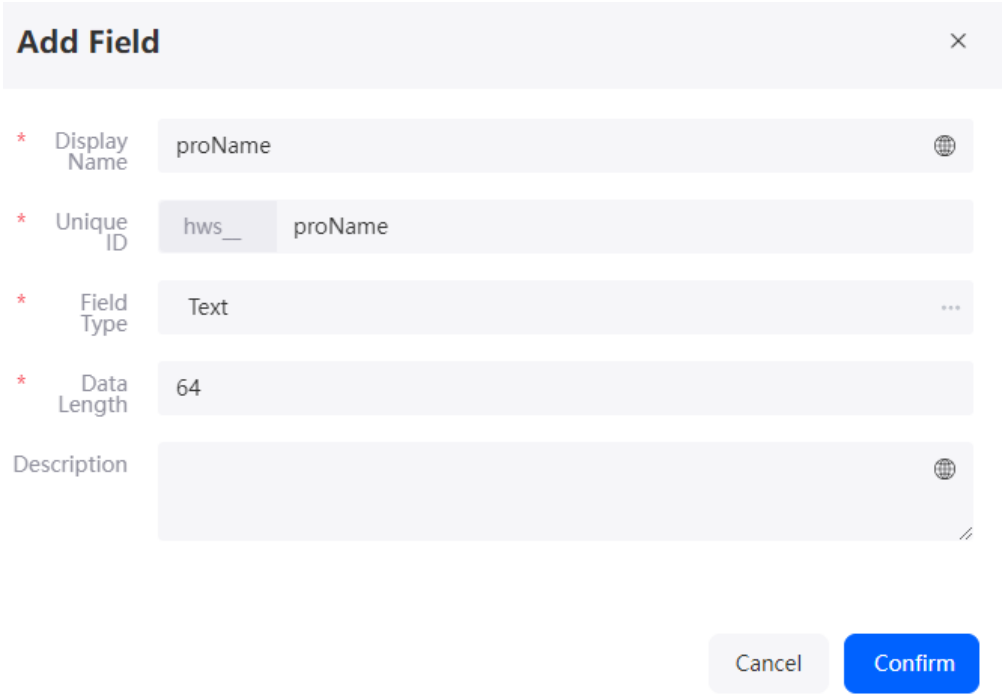


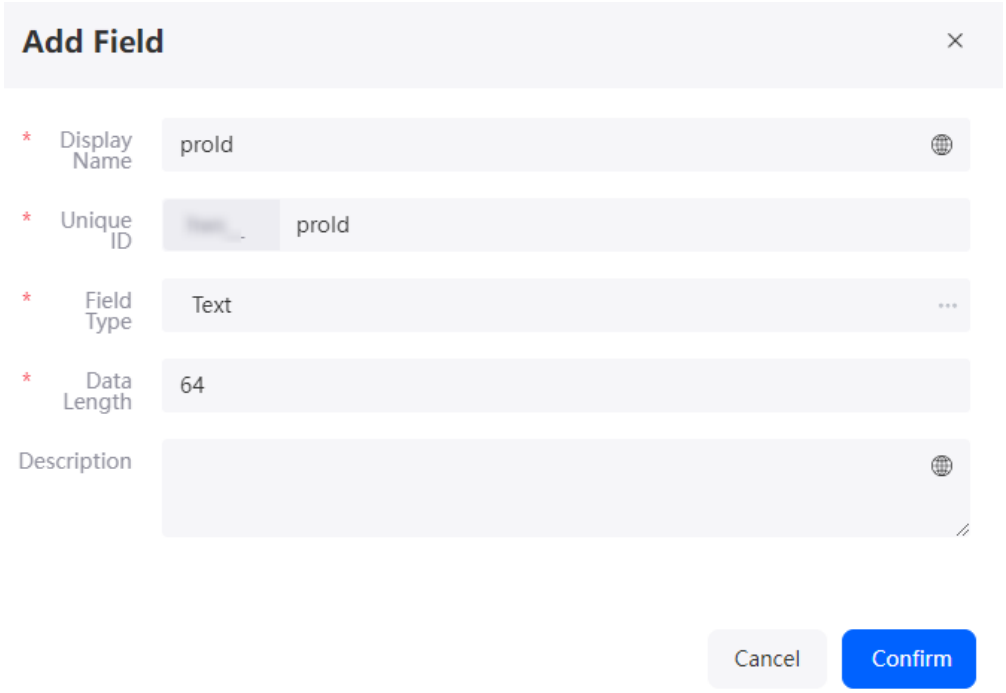
Table 3-3 Parameters for adding the proName field

Parameter	Description	Example
Display Name	Name of the new field, which can be changed after the field is created. Value: 1–63 characters.	proName
Unique ID	ID of a new field in the system. The value cannot be changed after the field is created. Naming rules: – Max. 63 characters, including the prefix namespace. – Start with a letter and use only letters, digits, and an underscore (_). Do not end with an underscore (_).	proName
Field Type	Click ***. On the page that is displayed, select the type of the new field based on the parameter description.	Text

5. On the **Fields** tab, click **Add** again to add the **proId** field.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

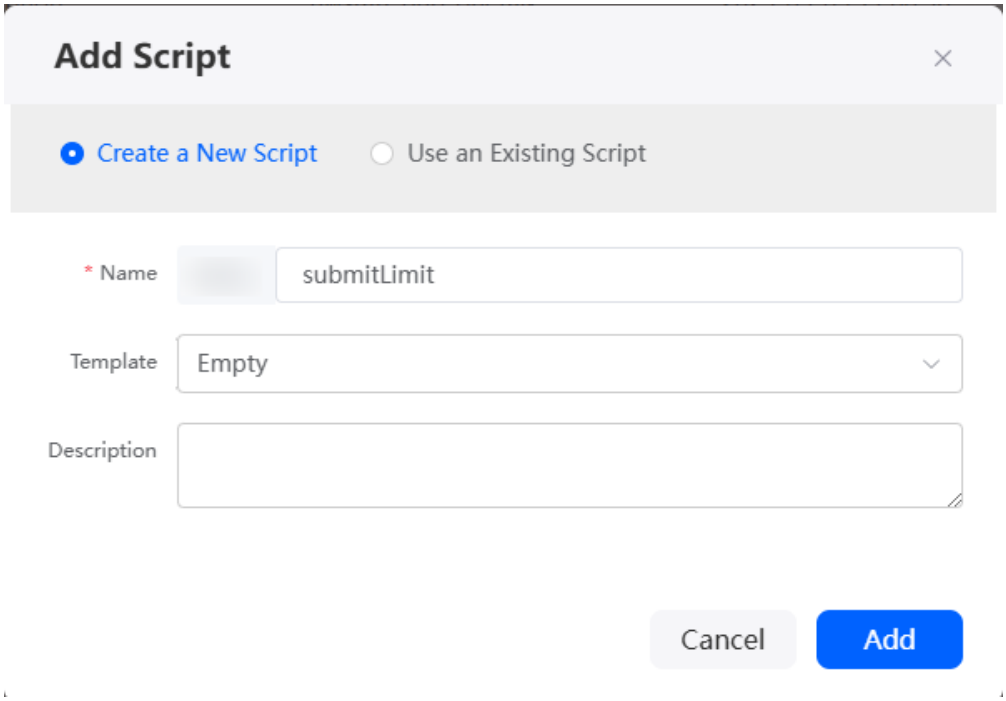
Figure 3-7 Adding the proId field



Step 3 Create a script.

1. In the navigation pane, choose **Logic** and click **+** next to **Script**.
2. Create a script, set the name to **submitLimit**, and click **Add**.

Figure 3-8 Creating the submitLimit script



3. In the script editor, enter the sample code.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The sample code is used to obtain the input of the frontend page, verify the input parameters, and call the object API to add an object instance. In the example, **Namespace __product__CST** is the object created in **Step 2**.

```
//This script is used to submit forms and restrict the submission time and user type.
import * as db from 'db';//Import the standard library related to the object.
import * as context from 'context';//Import the standard library related to the context.
import { now } from 'date';
import { toDate } from 'date';

//Define the input parameter structure.
@action.object({ type: "param" })
export class ActionInput {
    @action.param({ type: 'String', required: true, label: 'String' })
    proId: string;
    @action.param({ type: 'String', required: true, label: 'String' })
    proName: string;
}
//Define the output parameter structure. The output parameter contains one parameter, that is, the record ID of workOrder.
@action.object({ type: "param" })
export class ActionOutput {
    @action.param({ type: 'String' })
    id: string;
}
//Use the data object namespace __product__CST.
@useObject(['Namespace__product__CST'])
@action.object({ type: "method" })
export class CreateWorkOrder {    //Define the API class. The input parameter of the API is ActionInput, and the output parameter is ActionOutput.
    @action.method({ input: 'ActionInput', output: 'ActionOutput' })
    public createWorkOrder(input: ActionInput): ActionOutput {
        let out = new ActionOutput();    //Create an instance of the ActionOutput type as the return value.
        let error = new Error();    //Create an instance of the error type to save the error information when an error occurs.
        try {
            let currentTime = now();
            let date = toDate('2024-04-08 20:08:08', 'yyyy-MM-dd HH:mm:ss');
            if (date.getTime() < currentTime.getTime()) { // Restrict the submission time
                error.name = "WOERROR";
                error.message = "Submitted too late";
                throw error;
            }
            let user = context.getUserType();
            if (user != "PortalUser") { // Restrict the submission user type. Only the PortalUser can submit data.
                error.name = "WOERROR";
                error.message = "Not PortalUser!";
                throw error;
            }

            let productData = new Object();
            productData['Namespace__proName__CST'] = input.proName;   //Assign value to input parameters.
            productData['Namespace__proId__CST'] = input.proId;
            let s = db.object('Namespace__product__CST');    //Obtain the operation instance of the Namespace__product__CST object.
            let id = s.insert(productData);
            if (id) {
                out.id = id;
            } else {
                error.name = "WOERROR";
                error.message = "Unable to create product!";
                throw error;
            }
        } catch (error) {
            console.error(error.name, error.message);
            context.setError(error.name, error.message);
        }
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

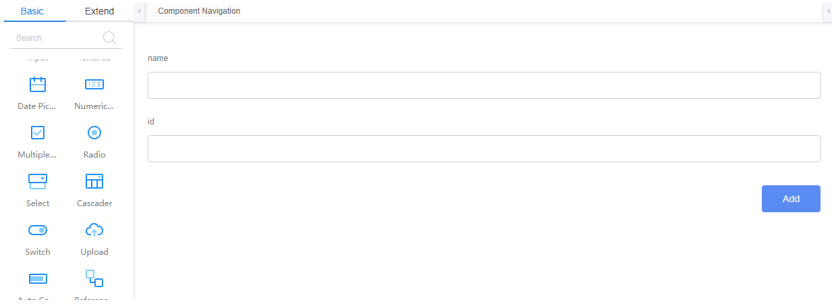
```
        return out;
    }
}
```

4. Click  to save the script. After the script is saved, click  to activate the script.

Step 4 Create a form page to submit form data.

1. In the navigation pane, choose **Page**, and click **+** next to **Standard Page**.
2. Enter the label and name of the page and click **Add** to create a standard page.
3. Drag two **Input** widgets and one **Button** widget to page on the right.
4. Change the **Label** to **name** and **id** for these two **Input** widgets, and set the **Button** widget's **Display Name** to **Add**.

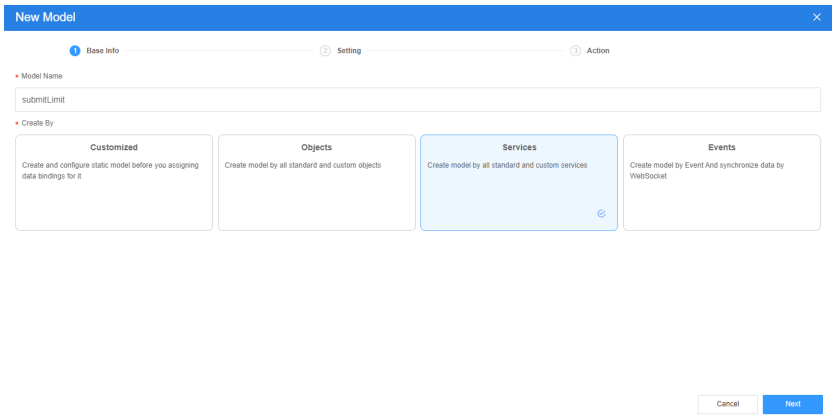
Figure 3-9 Designing a form page



Step 5 Create an object model.

1. At the bottom of the standard page, click **Model View**.
2. Click **New**, specify the **Model Name** (for example, **submitLimit**), select **Services** for **Source**, and click **Next**.

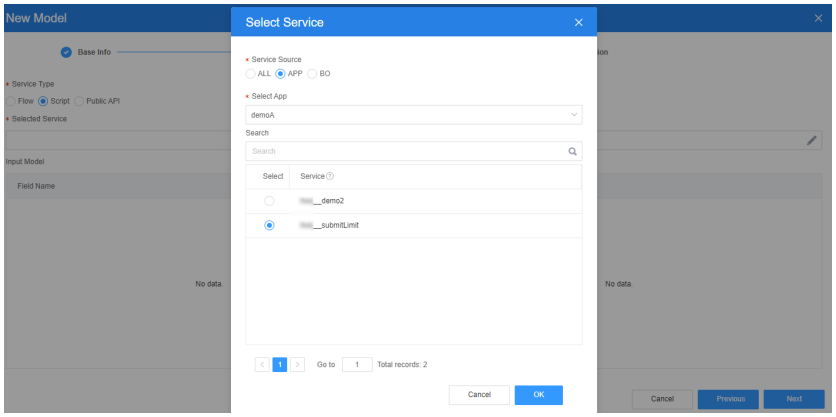
Figure 3-10 Creating a model



3. Set **Service Type** to **Script**. On the displayed **Select Service** page, select the script created in **Step 3** and click **OK**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 3-11 Selecting the script



4. Click **Next** and then click **OK**.

Step 6 Return to **Designer View** and bind the model with the widgets.

1. At the bottom of the standard page, click **Designer View**.
2. Select the **name** input widget, choose **Properties** > **Data Binding** and click  next to **Value Binding**.
3. Select the model (**proName**) created in **Step 5** and click **OK**.

Figure 3-12 Selecting the model

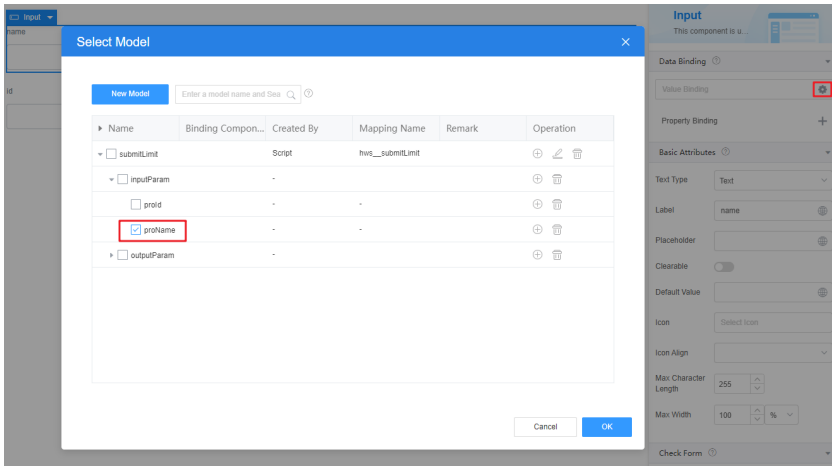
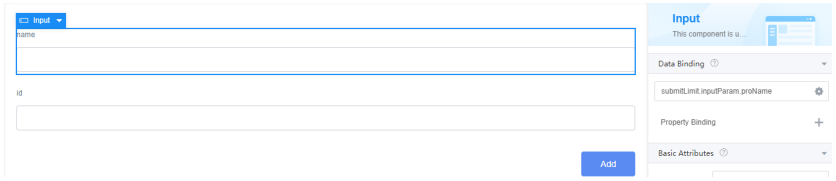


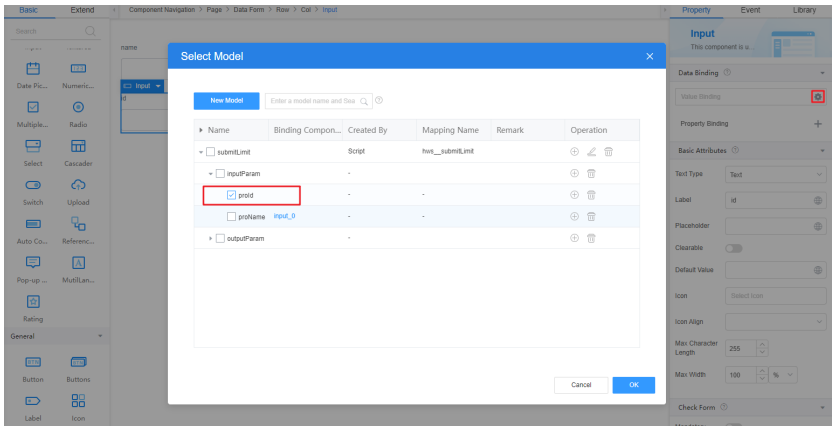
Figure 3-13 Binding effect



4. Repeat the preceding operations to bind the model (**proId**) created in **Step 5** to the **id** input widget.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

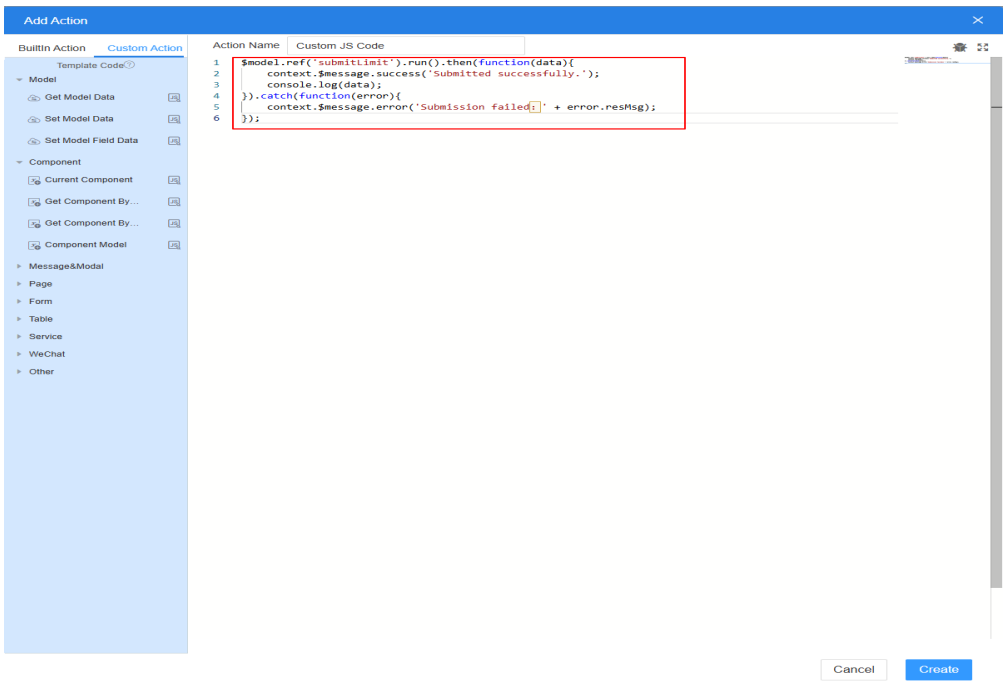
Figure 3-14 Binding a model to the id input widget



Step 7 Add an event for the **Add** button.

1. Select the **Add** button widget and click the **Events** tab on the right of the page.
2. Click **+** next to **on-click**. The page for adding an action is displayed.
3. Under **Custom Action**, enter the custom code and click **Create**.

Figure 3-15 Customizing an action



In this example, the custom JavaScript code is used to obtain the exception information thrown by the script and display the exception information on the page after the submit button is clicked.

```
$model.ref('submitLimit').run().then(function(data){
    context.$message.success('Submitted successfully.');
    console.log(data);
}).catch(function(error){
    context.$message.error('Submission failed: ' + error.resMsg);
});
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 8 Click  in the upper part of the page to save the page settings.

Step 9 After the settings are saved, click  in the upper part of the page to preview the effect.

----End

3.2 Adding and Deleting Table Data with Scripts

Expected Results

Use scripts to add and delete object data on the frontend page. For example, when a piece of data is added to or deleted from a standard page, the data is also added to or deleted from the object associated with the standard page.

Figure 3-16 Adding data on the page

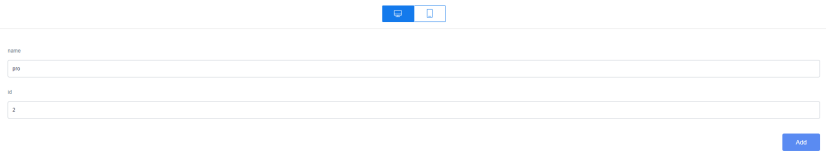


Figure 3-17 Synchronizing new data in an object

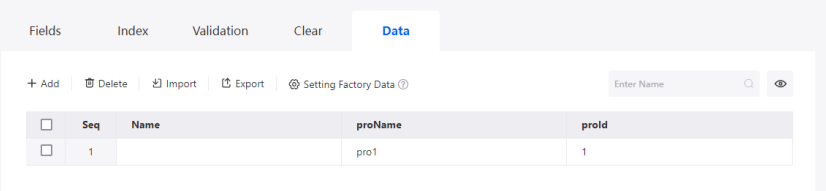


Figure 3-18 Object with two data records

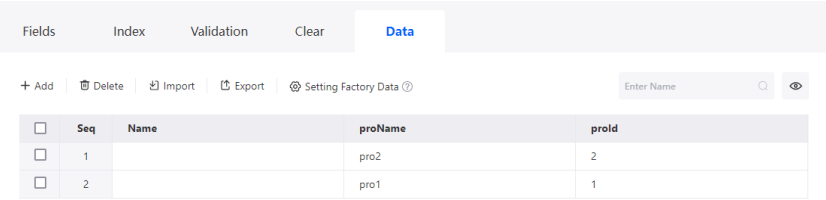


Figure 3-19 Deleting the data record whose proId is 1 on the frontend page

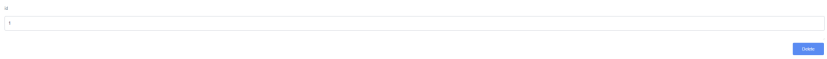
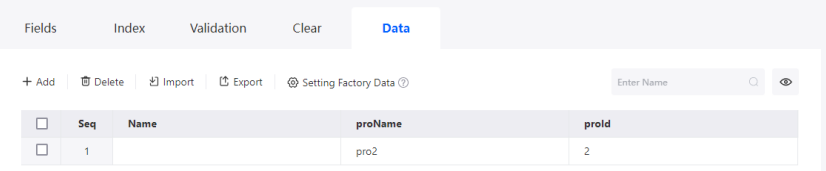


Figure 3-20 Synchronizing deletion of data whose proId is 1 in the object



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Implementation Method (Add)

Step 1 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

Figure 3-21 Creating a blank application

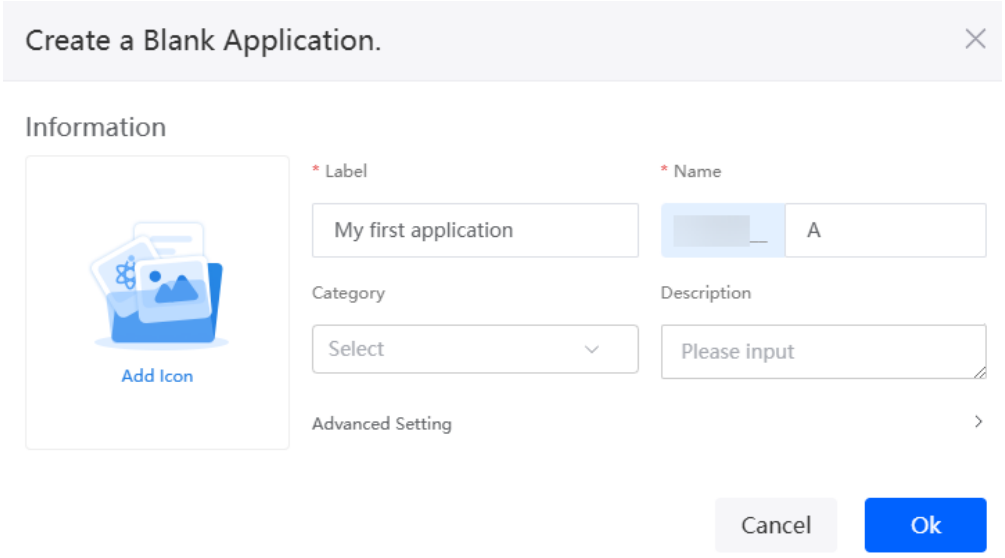


Table 3-4 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 2 Create an object named **product** and add fields to the object.

1. In the navigation pane, choose **Data**, and click **+** next to **Object**.
2. Set **Object Name** and **Unique ID** of the object to **product** and click the confirm button.

Figure 3-22 Creating object product

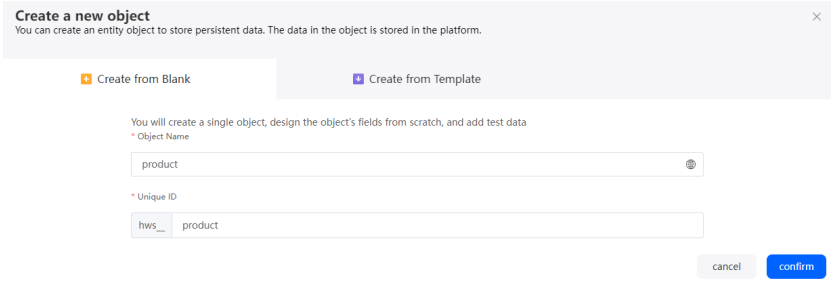


Table 3-5 Parameters for the created object product

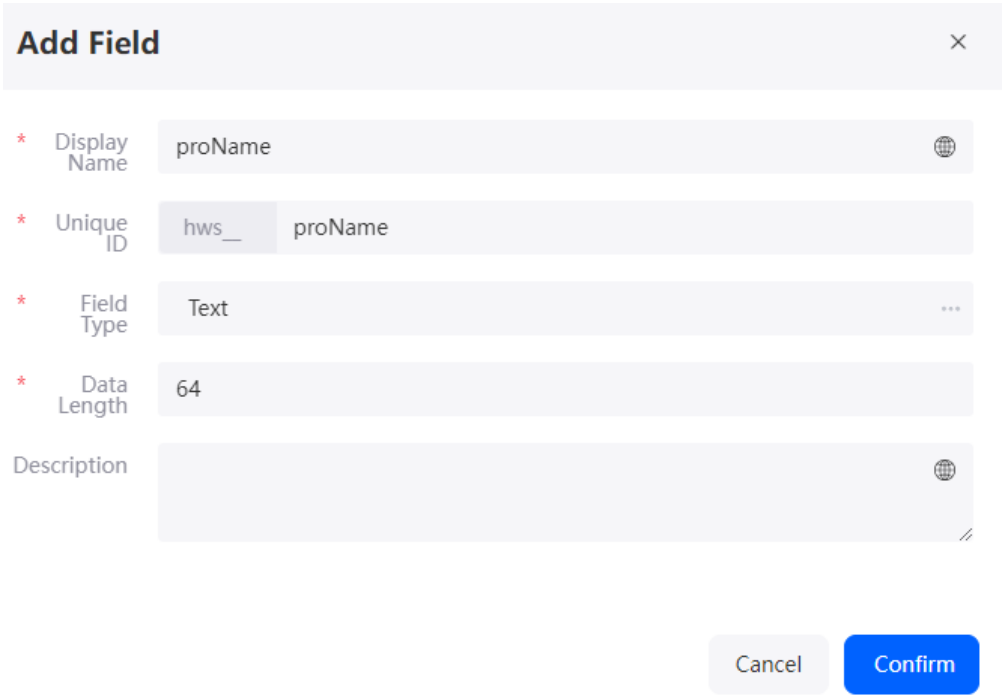
Parameter	Description	Example
Object Name	Name of the new object, which can be changed after the object is created. Value: 1–80 characters.	product

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Unique ID	ID of a new object in the system, which cannot be modified after being created. Naming rules: • Max. 63 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. • Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	product

3. Click  to go to the object details page.
4. On the **Fields** tab page, click **Add** and add the **ProName** field for the object.

Figure 3-23 Adding the ProName field



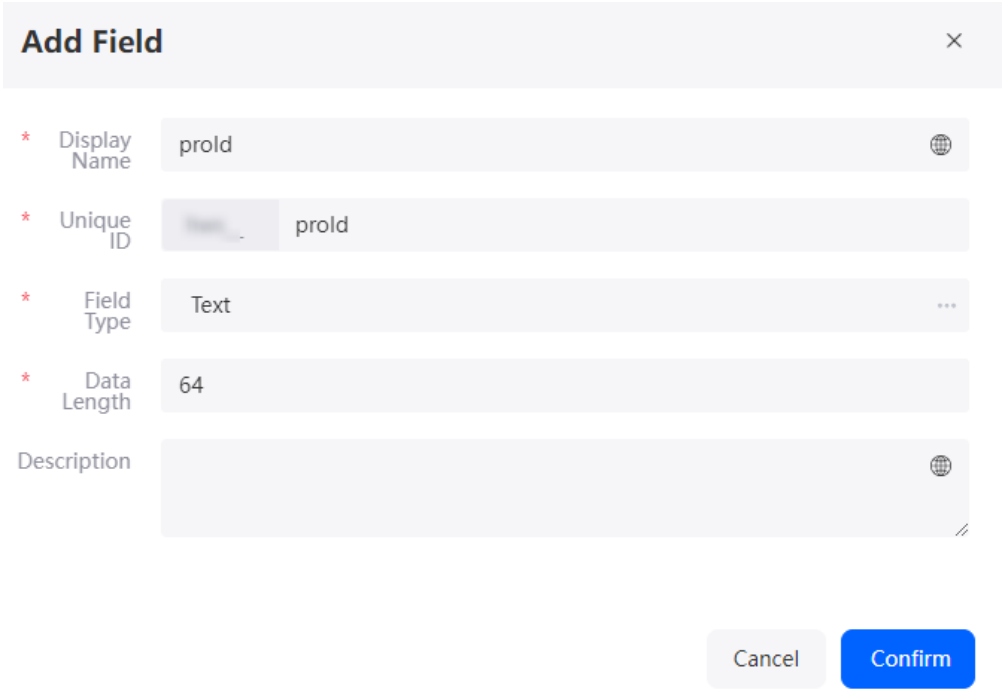
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-6 Parameters for adding the proName field

Parameter	Description	Example
Display Name	Name of the new field, which can be changed after the field is created. Value: 1–63 characters.	proName
Unique ID	ID of a new field in the system. The value cannot be changed after the field is created. Naming rules: – Max. 63 characters, including the prefix namespace. – Start with a letter and use only letters, digits, and an underscore (_). Do not end with an underscore (_).	proName
Field Type	Click ***. On the page that is displayed, select the type of the new field based on the parameter description.	Text

5. On the **Fields** tab, click **Add** again to add the **proId** field.

Figure 3-24 Adding the proId field

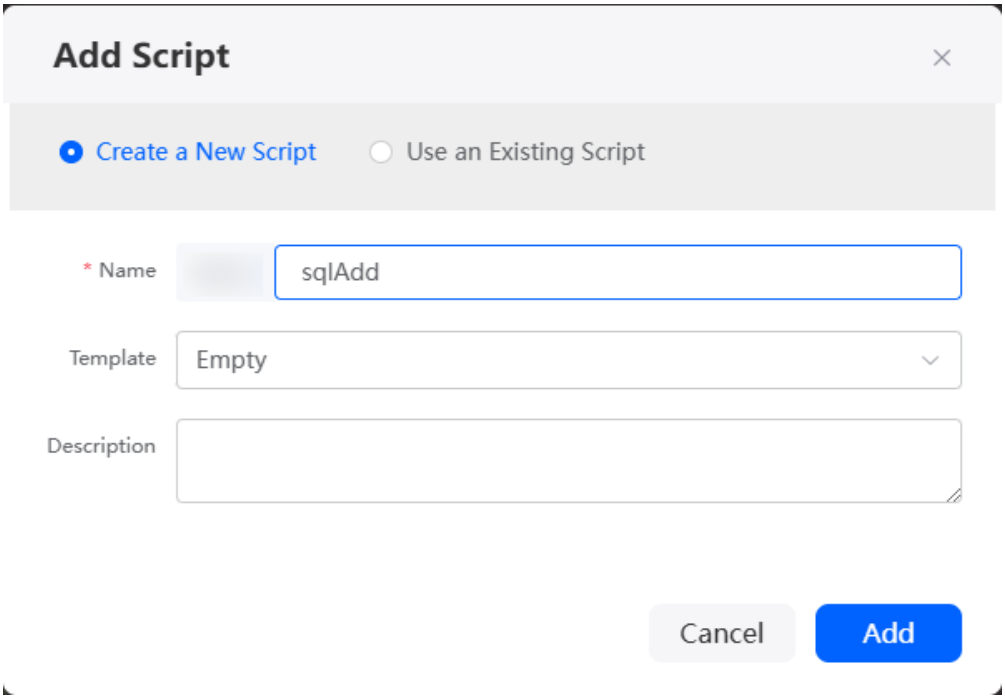


 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 3 Create a script.

1. In the navigation pane, choose **Logic** and click **+** next to **Script**.
2. Create a script, set the name to **sqlAdd**, and click **Add**.

Figure 3-25 Creating the sqlAdd script



3. In the script editor, enter the sample code.

   The sample code implements the following function: When the **Add** button is clicked, data such as the name and ID entered at the frontend is obtained and inserted into the object. If the insertion fails, the failure information is recorded. In the example, **Namespace __product__CST** is the object created in **Step 2**.

```
//This script is used to create a work order.
import * as db from 'db';//Import the standard library related to the object.
import * as context from 'context';//Import the standard library related to the context.

//Define the input parameter structure.
@action.object({ type: "param" })
export class ActionInput {
    @action.param({ type: 'String', required: true, label: 'String' })
    proId: string;
    @action.param({ type: 'String', required: true, label: 'String' })
    proName: string;
}
//Define the output parameter structure. The output parameter contains one parameter, that is, the record ID of workOrder.
@action.object({ type: "param" })
export class ActionOutput {
    @action.param({ type: 'String' })
    id: string;
}
//Use the data object namespace __product__CST.
@useObject(['Namespace__product__CST'])
@action.object({ type: "method" })
export class CreateWorkOrder {    //Define the API class. The input parameter of the API is ActionInput, and the output parameter is ActionOutput.
    @action.method({ input: 'ActionInput', output: 'ActionOutput' })
    public createWorkOrder(input: ActionInput): ActionOutput {
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

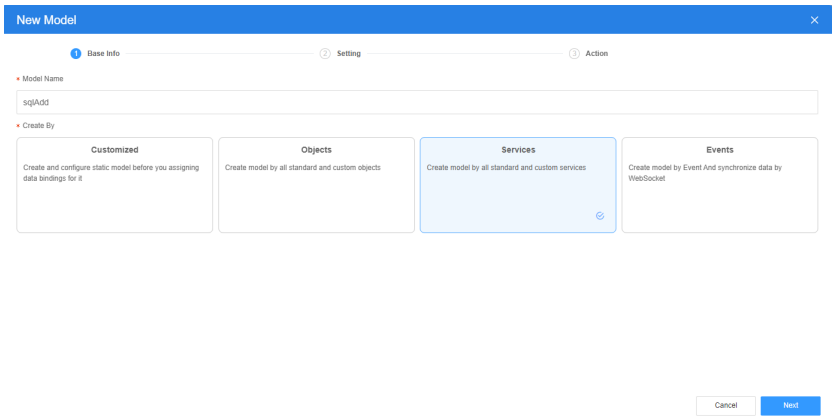
```
        let out = new ActionOutput();    //Create an instance of the ActionOutput type as the return
value.
        let error = new Error();    //Create an instance of the error type to save the error information
when an error occurs.
        try {
            let productData = new Object();
            productData['Namespace__proName__CST'] = input.proName;   //Assign the input parameter
to the productData variable for future use.
            productData['Namespace__proId__CST'] = input.proId;
            let s = db.object('Namespace__product__CST');    //Obtain the operation instance of the
Namespace__product__CST object.
            let id = s.insert(productData);
            if (id) {
                out.id = id;
            } else {
                error.name = "WOERROR";
                error.message = "Unable to create product!";
                throw error;
            }
        } catch (error) {
            console.error(error.name, error.message);
            context.setError(error.name, error.message);
        }
        return out;
    }
}
```

4. Click  to save the script. After the script is saved, click  to activate the script.

Step 4 Create an object model.

1. In the navigation pane, choose **Page**, and click **+** next to **Standard Page**.
2. At the bottom of the standard page, click **Model View**.
3. Click **New**, specify **Model Name** (for example, **sqlAdd**), select **Services** for **Source**, and click **Next**.

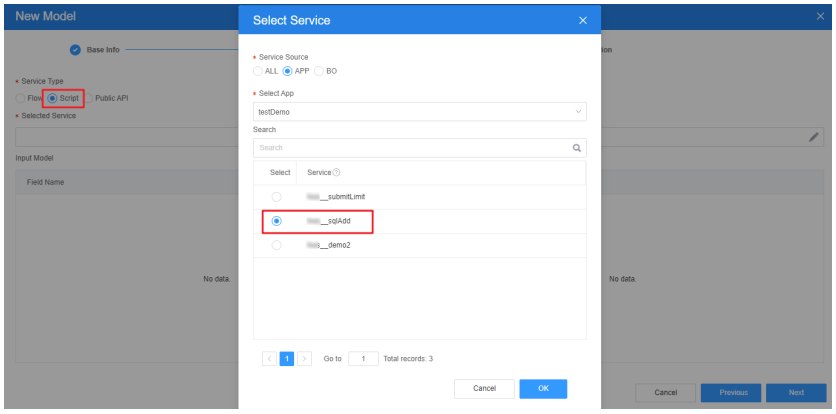
Figure 3-26 Creating a model



4. Select the script created in **Step 3** and click the confirm button.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 3-27 Selecting the script

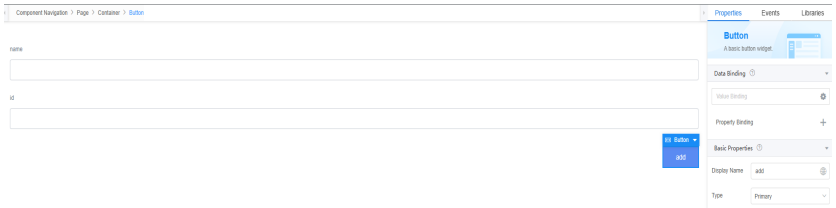


5. Click **Next** and then click **OK**.

Step 5 Return to **Designer View** and bind the model with the widgets.

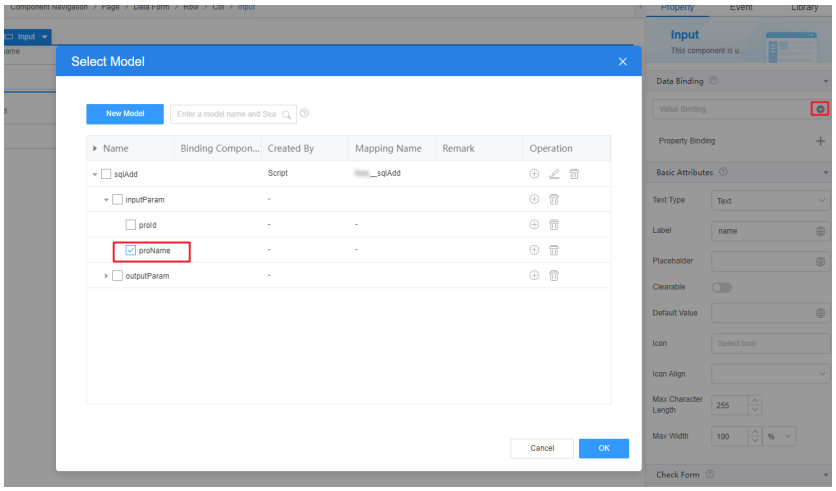
1. On the standard page, drag two **Input** widgets and one **Button** widget, change the **Label** of two input widgets to **name** and **id**, and set the button **Display Name** to **Add**.

Figure 3-28 Final page setting effect



2. Select the **name** input widget, choose **Properties** > **Data Binding** and click next to **Value Binding**.
3. Select the model (**proName**) created in **Step 4** and bind the input widget with the script data.

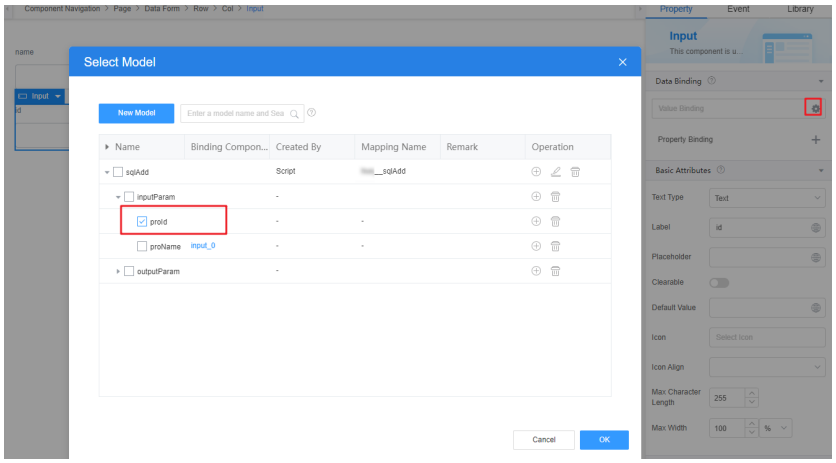
Figure 3-29 Selecting the model



4. Repeat the preceding operations to bind the model (**proId**) created in **Step 4** to the **id** input widget.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

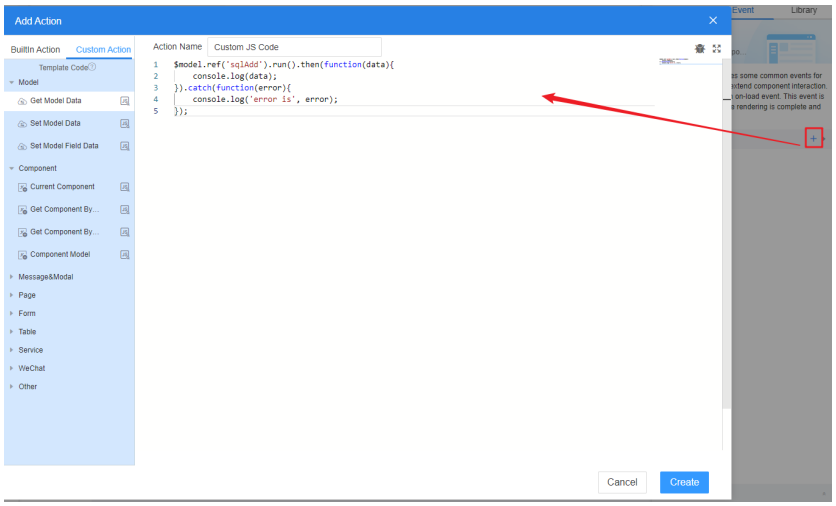
Figure 3-30 Binding a model to the id input widget



Step 6 Add an event for the **Add** button.

1. Select the **Add** button widget and click the **Events** tab.
2. Click **+** next to **on-click**. The page for adding an action is displayed.
3. Under **Custom Action**, enter the custom code and click **Create**.

Figure 3-31 Customizing an action



In this example, the customized JavaScript code is used to call the service after the submit button is clicked. The service calls the script to add records.

```
$model.ref('sqlAdd').run().then(function(data){
    console.log(data);
}).catch(function(error){
    console.log('error is', error);
});
```

Step 7 Return to the standard page and click  to save the page settings. After the page is saved, click  to preview the effect.

----End

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Implementation Method (Delete)

Step 1 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

Figure 3-32 Creating a blank application

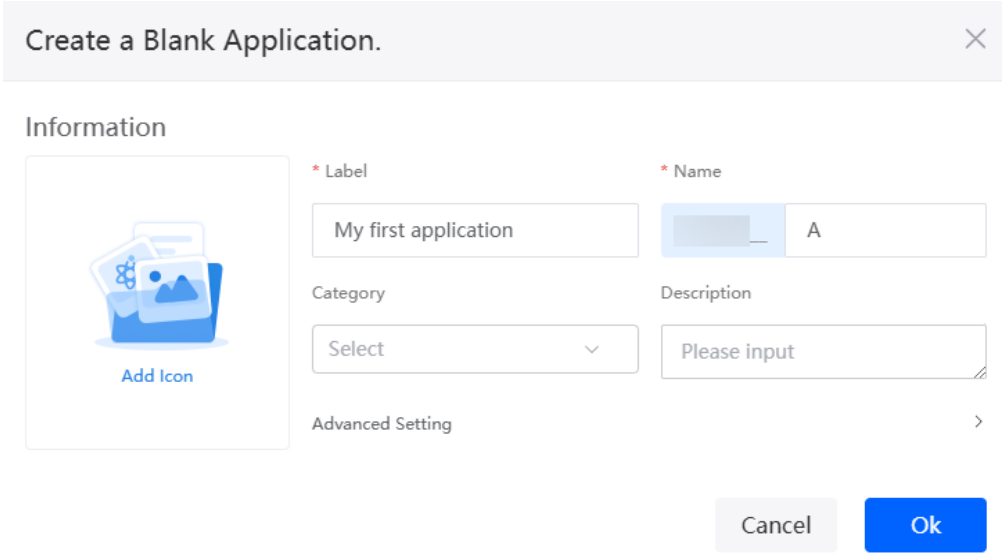


Table 3-7 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application

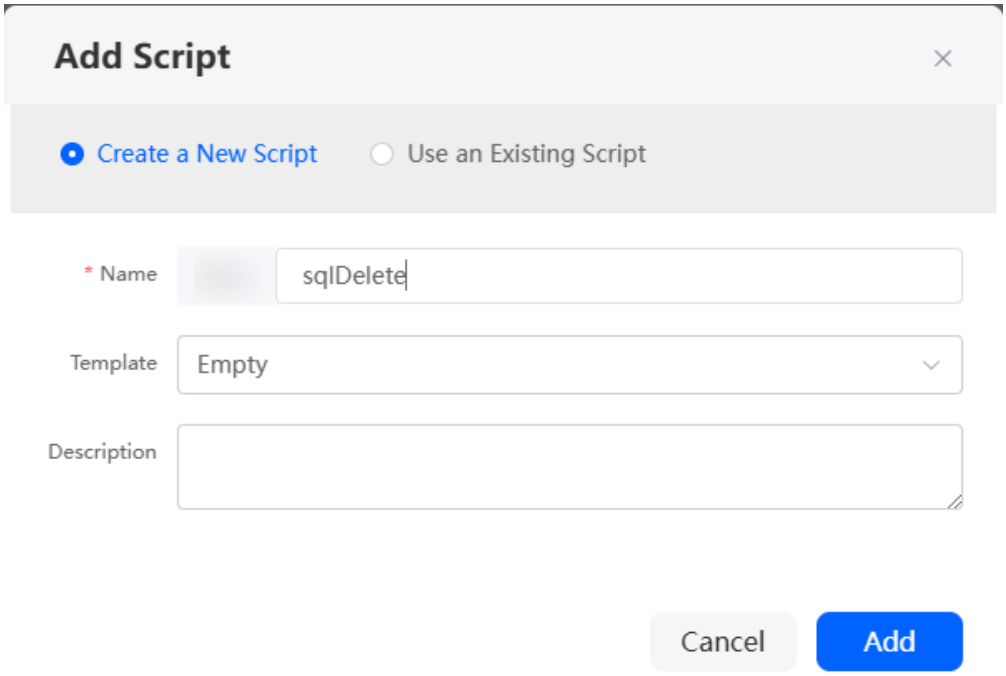
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 2 Create a deletion script.

1. In the navigation pane, choose **Logic** and click **+** next to **Script**.
2. Create a script, set the name to **sqlDelete**, and click **Add**.

Figure 3-33 Creating the sqlDelete script



3. In the script editor, enter the sample code.

   The script in this example is used to delete a data record based on the ID entered on the page through API. If an error is reported, the error information

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

is recorded. In the example, **Namespace __product__CST** is the object created in **Step 2**.

```
//This script is used to delete work orders.
import * as db from 'db';//Import the standard library related to the object.
import * as context from 'context';//Import the standard library related to the context.
//Define the input parameter structure.
@action.object({ type: "param" })
export class ActionInput {
    @action.param({ type: 'String', required: true, label: 'String' })
    id: string;
}
//Define the output parameter structure. The output parameter contains one parameter, that is, the record ID of workOrder.
@action.object({ type: "param" })
export class ActionOutput {
    @action.param({ type: 'String' })
    id: string;
}
//Use the data object namespace __product__CST.
@useObject(['Namespace__product__CST'])
@action.object({ type: "method" })
export class DeleteWorkOrder {    //Define the API class. The input parameter of the API is ActionInput, and the output parameter is ActionOutput.
    @action.method({ input: 'ActionInput', output: 'ActionOutput' })
    public deleteWorkOrder(input: ActionInput): ActionOutput {
        let out = new ActionOutput();    //Create an instance of the ActionOutput type as the return value.
        let error = new Error();    //Create an instance of the error type to save the error information when an error occurs.
        try {
            let id = input.id;
            let s = db.object('Namespace__product__CST');    //Obtain the operation instance of the Namespace__product__CST object.
            //Query condition
            let condition = {
                "conjunction": "AND",
                "conditions": [{
                    "field": "Namespace__proId__CST",
                    "operator": "eq",
                    "value": id
                }]
            };
            let isDeleted = s.deleteByCondition(condition);
            if (isDeleted) {
                out.id = id;
            } else {
                error.name = "WOERROR";
                error.message = "Failed to delete the work order!";
                throw error;
            }
        } catch (error) {
            console.error(error.name, error.message);
            context.setError(error.name, error.message);
        }
        return out;
    }
}
```

4. Click  to save the script. After the script is saved, click  to activate the script.

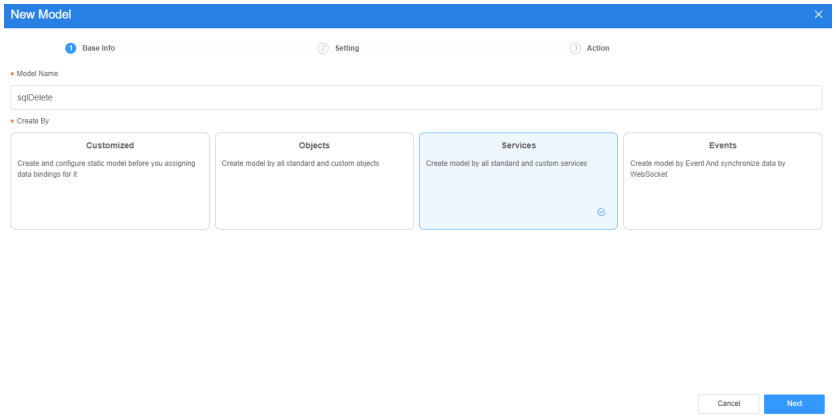
Step 3 Create an object model.

1. In the navigation pane, choose **Page**, and click **+** next to **Standard Page**.
2. Enter the label and name of the page and click **Add** to create a standard page.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

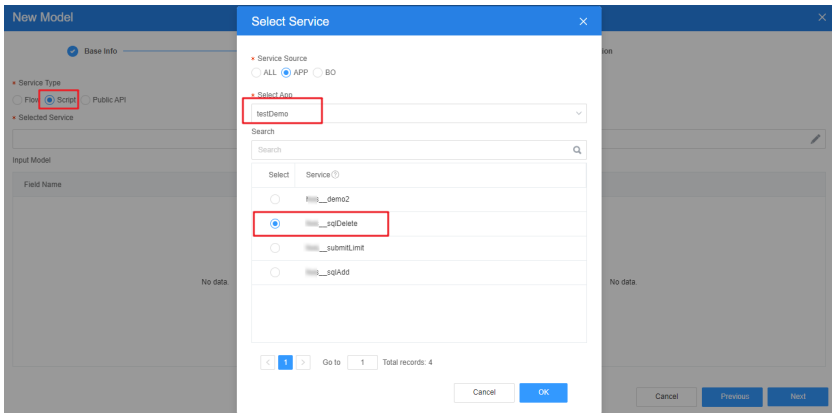
3. At the bottom of the standard page, click **Model View**.
4. Click **New**, specify **Model Name** (for example, **sqlDelete**), select **Services** for **Source**, and click **Next**.

Figure 3-34 Creating a model



5. Set **Service Type** to **Script**. On the displayed **Select Service** page, select the script created in **Step 2** and click **OK**.

Figure 3-35 Selecting the script



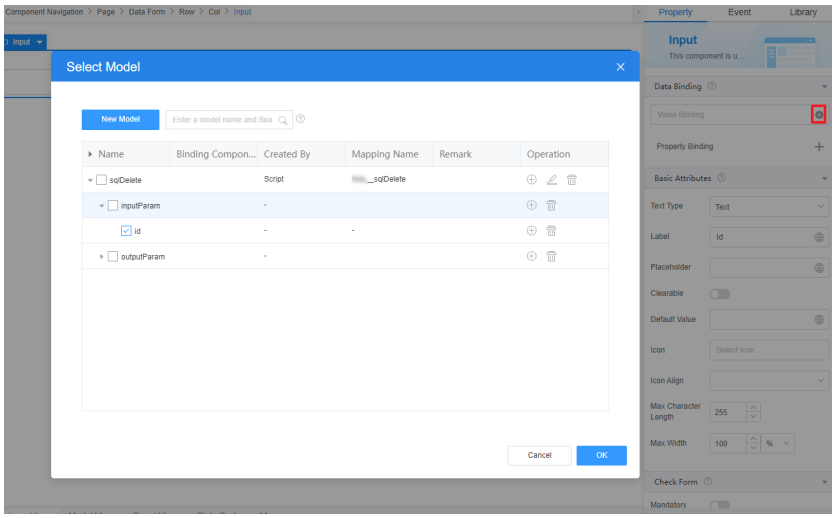
6. Click **Next** and then click **OK**.

Step 4 Return to **Designer View** and bind the model with the widgets.

1. Drag an **Input** widget and a **Button** widget to the standard page, set the **Label** of the input widget to **id**, and set the **Display Name** of the button widget to **Delete**.
2. Select the input widget, choose **Properties** > **Data Binding** and click next to **Value Binding**.
3. Bind the input widget with the input parameter **id** of the **sqlDelete** model created in **Step 3**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

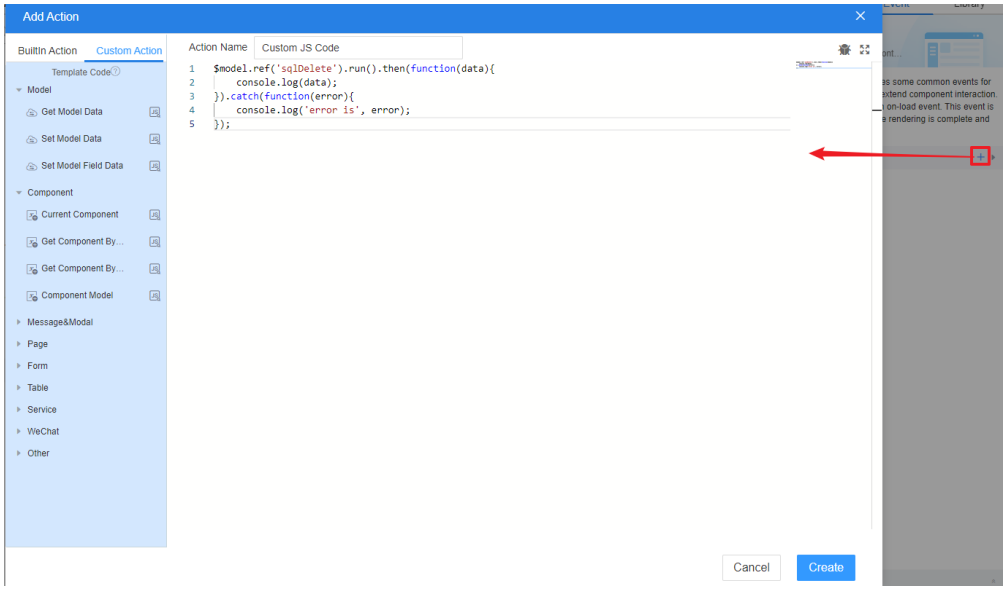
Figure 3-36 Selecting the model



Step 5 Add an event for the **Delete** button.

1. Select the **Delete** button widget and click the **Events** tab.
2. Click **+** next to **on-click**. The page for adding an action is displayed.
3. Under **Custom Action**, enter the custom code and click **Create**.

Figure 3-37 Customizing an action



The custom JavaScript code in the example is used to call the backend service, and the backend service calls the script to delete data.

```
$model.ref('sqlDelete').run().then(function(data){
    console.log(data);
}).catch(function(error){
    console.log('error is', error);
});
```

Step 6 Click in the upper part of the page to save the page settings.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 7 After the settings are saved, click  in the upper part of the page to preview the effect.

----End

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

4 Templates

4.1 Using the File Template Function to Generate Contracts

Application Scenarios

Huawei Cloud Astro Zero enables quick creation of **Word**, **Excel**, **Email**, and **SMS** templates, enhancing document efficiency and standardization.

This practice uses a **Word** template as examples. Developers can create print templates based on customer needs. For example, users can generate a product order list based on information such as products and prices from the product order management system by using this function. After processing an order, the template function can sync this order information to a contract template, creating a contract document for offline signing. Similarly, in finance, templates can be used to generate customized invoices and receipts. If a formal invitation is needed, the same template feature can create and print professional business letters.

Advantages

In a flow, you can call document template nodes and configure their input and output parameters to generate specific documents. This flow can be packaged as an open API for third-party use, or directly called on standard pages for portal users to download.

Constraints

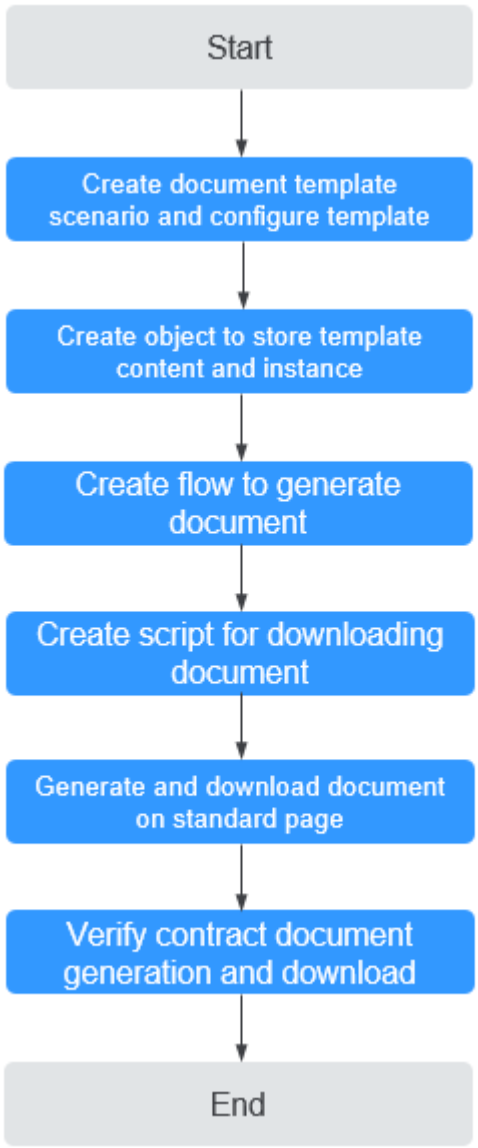
- After uploading a Word document (containing text type variables defined by ${*Parameter*}), you can only preview the document template or generate a specific document, but modifying it directly on the GUI is not allowed. To make changes, modify the template locally on your PC and re-upload it.
- The generated DOCX template supports only text variable replacement. It does not support dynamic replacement of QR codes or images.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Procedure

Figure 4-1 shows the process of generating a contract using a file template.

Figure 4-1 Generating a contract using a file template



Step 1: Create a Document Template Scenario and Configure the Template

Create a template scenario and add a contract template to the scenario. In the template scenario configuration, add the parameters to be replaced in the document template. A template scenario is a collection of service scenarios. Multiple templates can be created in a template scenario. The templates share the data structure.

Step 1 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

Figure 4-2 Creating a blank application

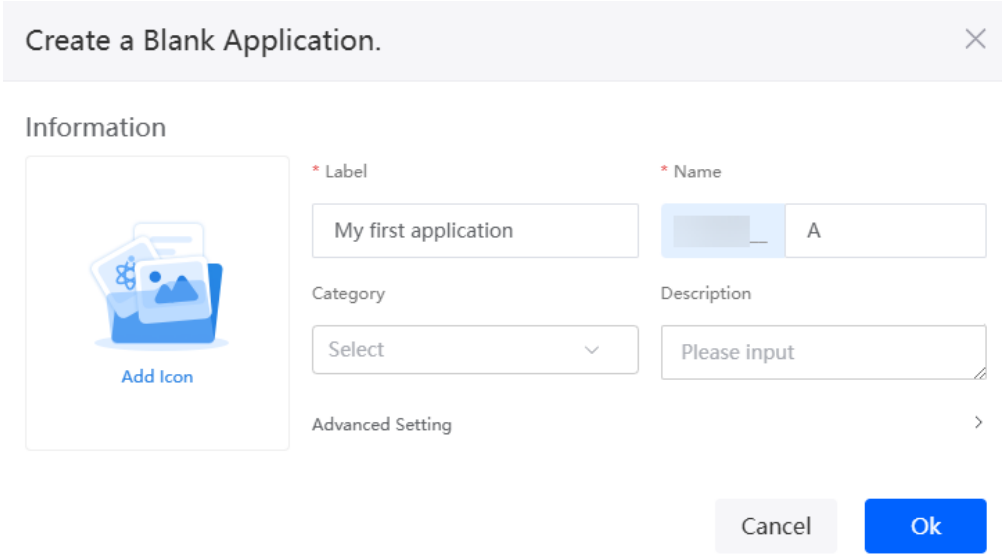


Table 4-1 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

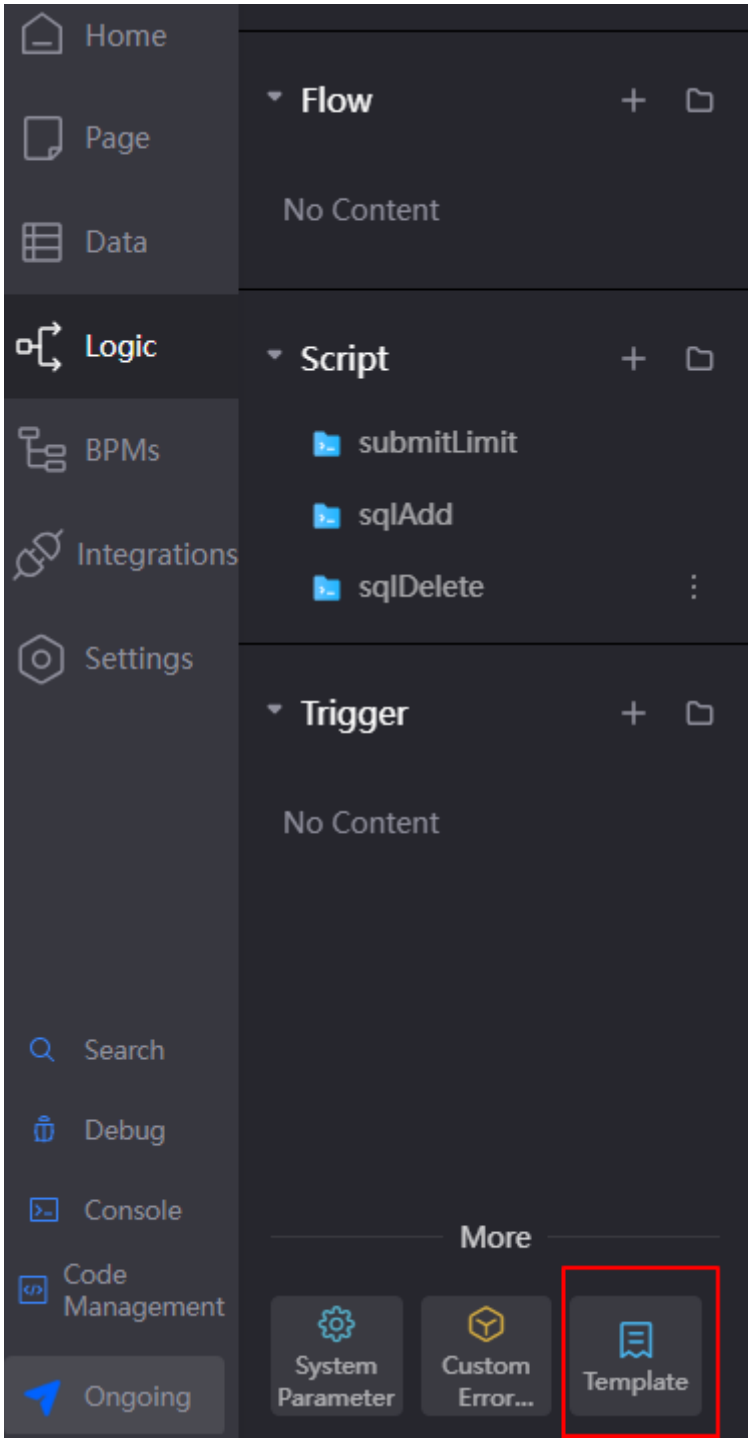
Parameter	Description	Example
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 2 Create a file template scenario.

1. In the navigation pane of the application designer, choose **Logic** > **More** > **Templates**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-3 Clicking Template



2. Click **Add Template Scenario**, set the label and name of the template scenario, and click **Confirm**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-4 Creating a template scenario

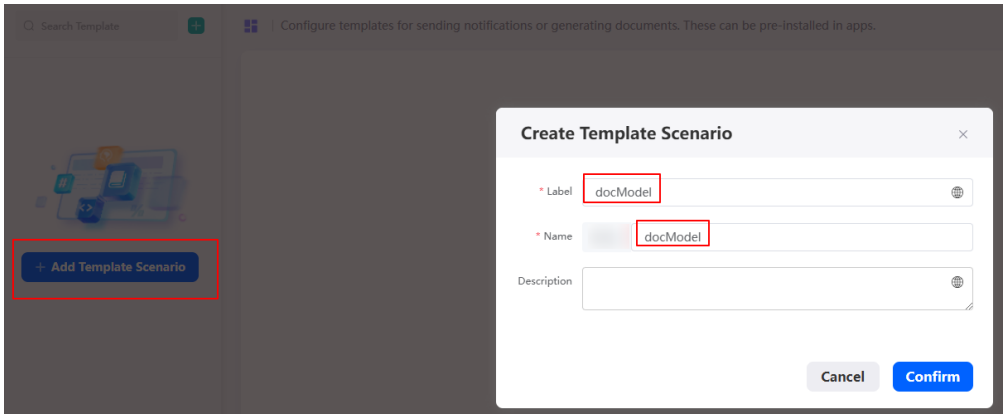


Table 4-2 Parameters for creating a template scenario

Parameter	Description	Example
Label	Name of the new template scenario, which can be changed after the template is created. Value: 1–80 characters.	Document template
Name	ID of a new template scenario in the system. The ID cannot be modified after the template is created. Naming rules: – Max. 64 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	docModel

Step 3 On the **Configurations** tab page, set the storage location of the template file and click **Save**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-5 Configuring file storage

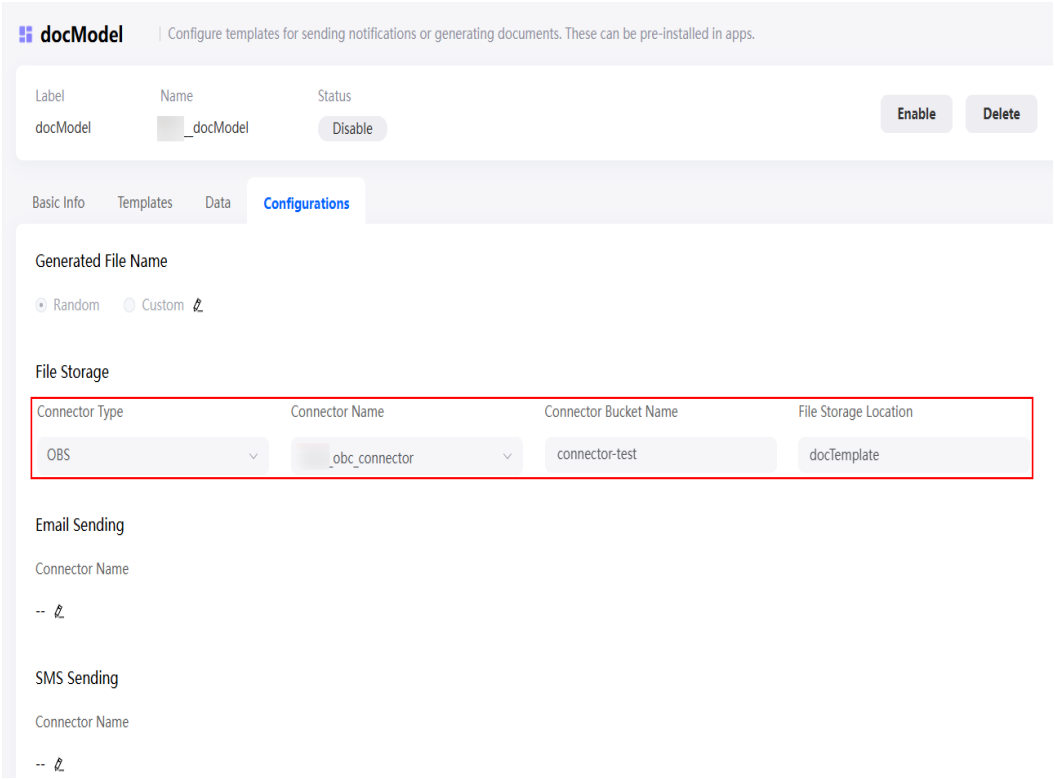


Table 4-3 File storage parameters

Parameter	Description	Example
Connector Type	Connector type of the file storage. Currently, only **OBS** and **MINIO** are supported. In Huawei Cloud Astro Zero, you can create OBS and MinIO connectors to store data in OBS and MinIO, for details, see **Interconnecting with OBS Instances** and **Connecting MinIO for Object and Asset Storage**.	OBS
Connector Name	Name of the OBS or MinIO connector created in . You can choose **Integrations** > **Connector** > **Connector Instance** > **Storage** > **OBS**/**MINIO** to view the name.	*Namespace*__OBS_Connector
Connector Bucket Name	OBS or MinIO bucket name configured during connector creation. .	template-obs
File Storage Location	You can specify the path for storing files in the OBS or MinIO bucket.	docTemplate

Step 4 On the **Templates** tab page, click **Add** to create a contract template.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-6 Creating a contract template

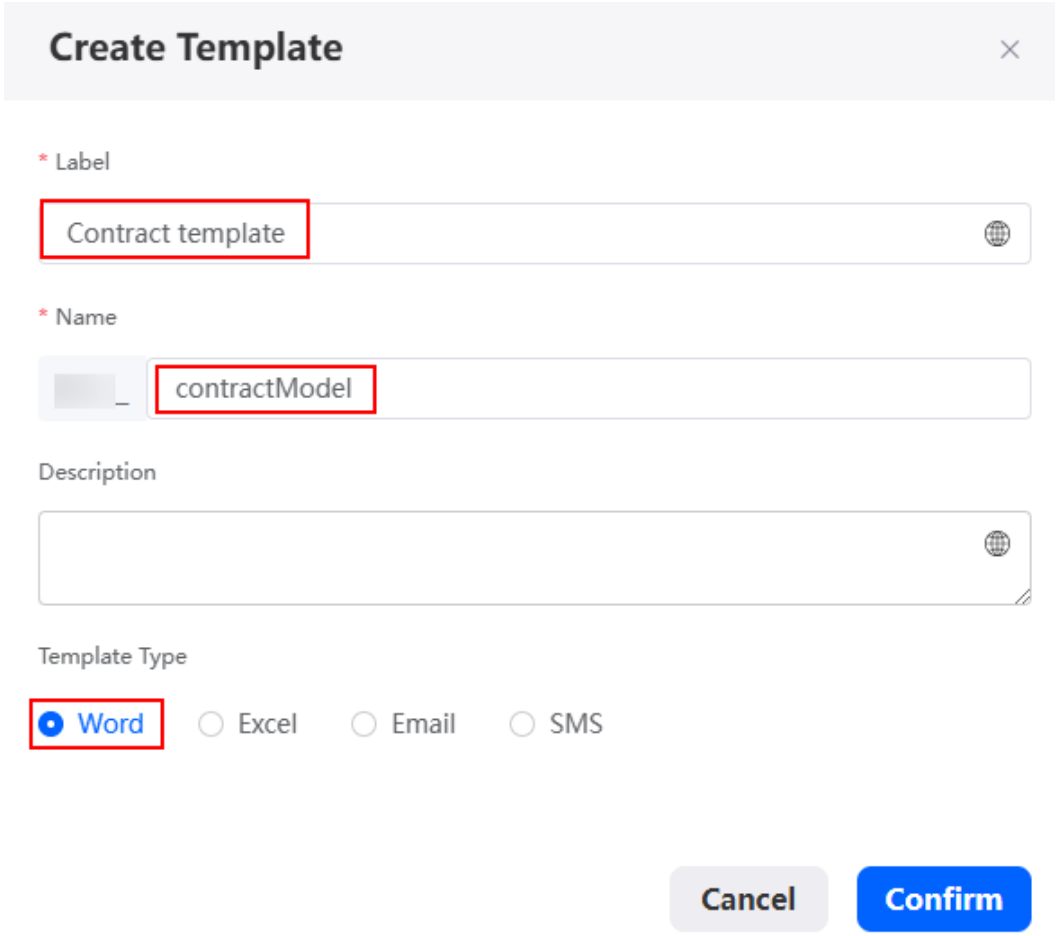


Table 4-4 Parameters for creating a template

Parameter	Description	Example
Label	Name of the new template, which can be changed after the template is created. Value: 1–80 characters.	Contract template

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Name	ID of a new template in the system. The ID cannot be modified after the template is created. Naming rules: • Max. 63 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. • Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	contractModel
Template Type	Type of the document template to be created. • **Word**: You can upload a Word document template. The recommended size of the document is less than 10 MB. • **Excel**: You can upload an Excel document template. The uploaded Excel document cannot contain more than 10 sheets. Each sheet cannot contain more than 200 columns, 1,000 rows, or 500 characters in each cell. • **Email**: You can compile email templates on the template editing page. • **SMS**: You can compile SMS templates on the template editing page.	Word

Step 5 On the **Data** tab page, add parameters in **Table 4-5** for the contract template.

The template parameters added here correspond to the content to be replaced in the document template.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-7 Adding a company name parameter to a contract template

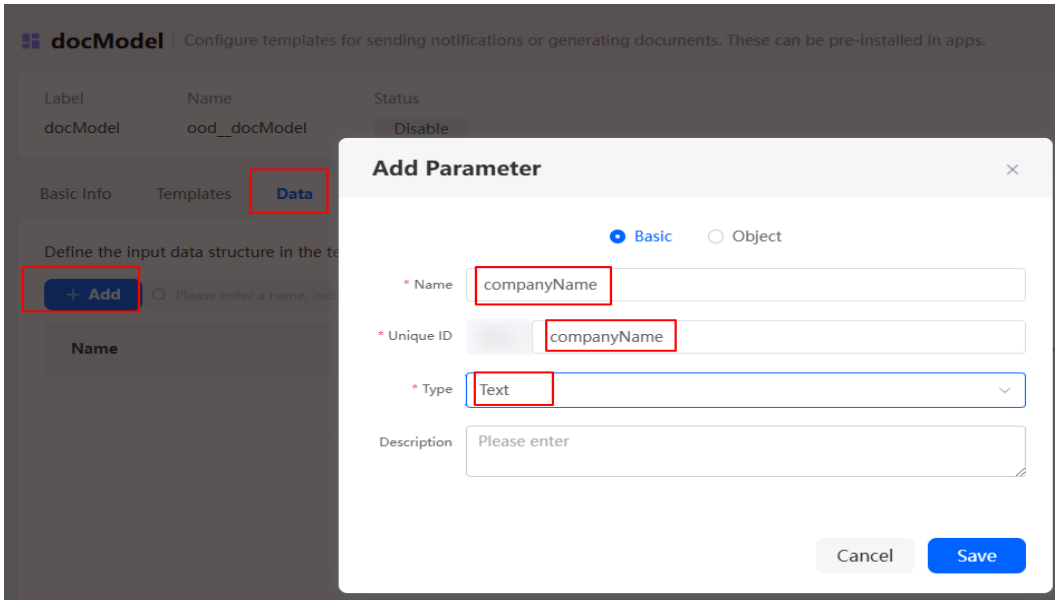


Table 4-5 Parameters to be added

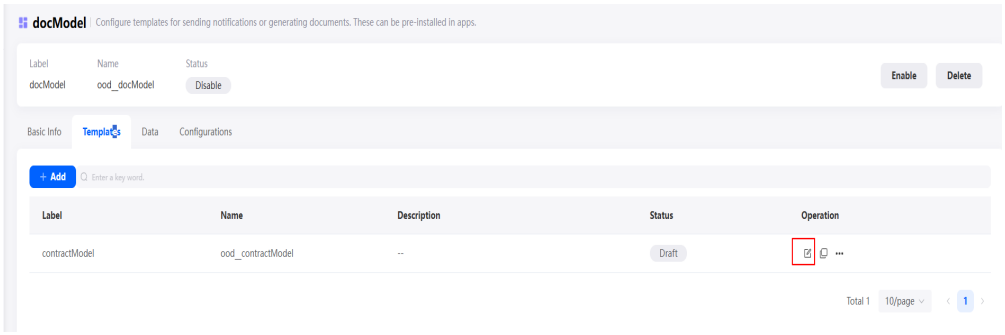
Name	Unique ID	Type
Company name	companyName	Text
Contract amount	amount	Number
Number of orders	orderNum	Number
Contract signer	person	Text
Contract date	date	Date
Contract name	contractName	Text
Company name of party B	otherCompanyName	Text

Step 6 Upload the document template.

1. On the **Templates** tab page, click next to the template created in **Step 4**. The contract template page is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-8 The template editing page



2. Click the upload button, select the Word file to be uploaded, and view the effect after the file is uploaded.

   After the upload is successful, you can view the uploaded Word document in the OBS bucket configured in **Step 3**, as shown in **Figure 4-11**.

Figure 4-9 Uploading a Word document

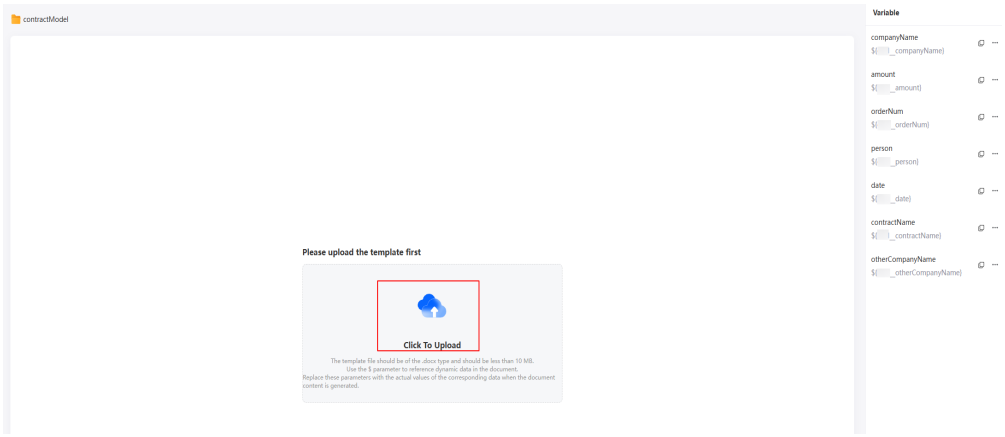
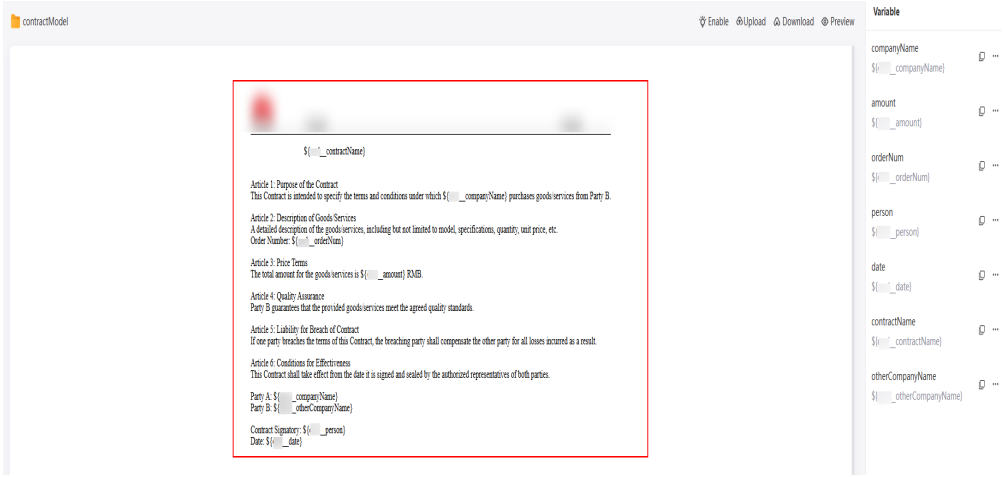
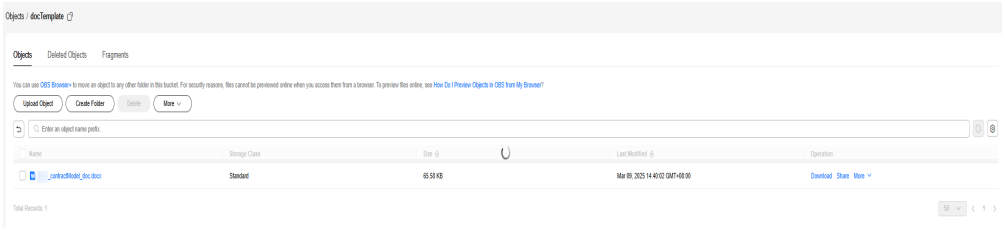


Figure 4-10 Viewing the effect after the document is uploaded



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-11 Checking whether the document is uploaded to the OBS bucket



In this practice, the content of the Word document to be uploaded is as follows. Upload the document based on service requirements.

```
                                ${Namespace__contractName}

Article 1 Purpose of the Contract
This contract outlines the specific terms and conditions under which ${Namespace__companyName} will purchase goods/services from Party B.

Article 2 Product/Service Description
Detailed description of the goods/services, including but not limited to the model, specifications, quantity, and unit price.
The number of orders: ${Namespace__orderNum}

Article 3 Price Clause
The total amount of the goods/services is ${Namespace__amount}.

Article 4 Quality Assurance
Party B warrants that the goods/services provided comply with the agreed quality standards.

Article 5 Liability for Breach of Contract
If one party violates the contract clauses, the breaching party shall compensate the other party for all losses incurred therefrom.

Article 6 Conditions for Entry into Force
This contract shall take effect upon signature and seal by the authorized representatives of both parties.

Party A: ${Namespace__companyName}
Party B: ${Namespace__otherCompanyName}

Contract signed by: ${Namespace__person}
Date: ${Date__date}
```

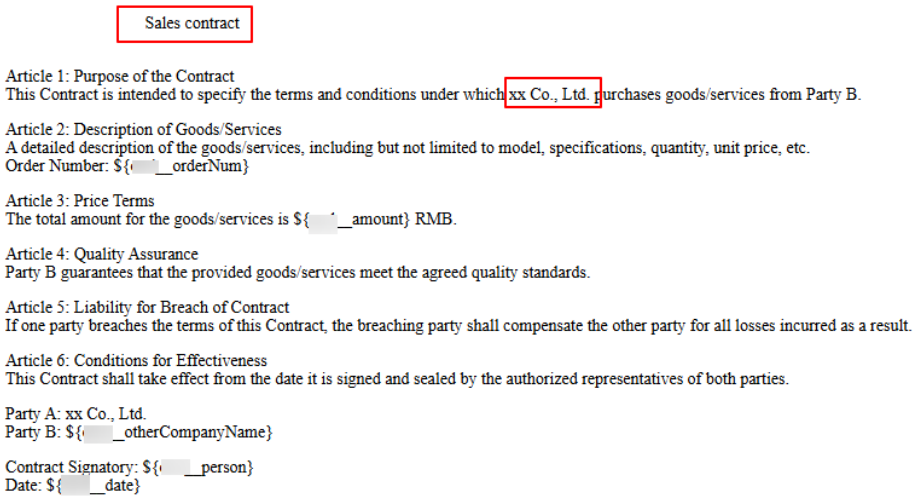
3. Click the preview button in the upper right corner of the page and set parameters to preview the effect.

   For example, enter the following template parameters in the input parameter text box and check whether the corresponding parameters in the contract are replaced.

```
{
   "Namespace__contractName": "Contract",
   "Namespace__companyName": "xx Co., Ltd."
}
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-12 After the parameters in the contract are replaced


Sales contract

Article 1: Purpose of the Contract
This Contract is intended to specify the terms and conditions under which xx Co., Ltd. purchases goods/services from Party B.

Article 2: Description of Goods/Services
A detailed description of the goods/services, including but not limited to model, specifications, quantity, unit price, etc.
Order Number: ${ __orderNum}

Article 3: Price Terms
The total amount for the goods/services is ${ __amount} RMB.

Article 4: Quality Assurance
Party B guarantees that the provided goods/services meet the agreed quality standards.

Article 5: Liability for Breach of Contract
If one party breaches the terms of this Contract, the breaching party shall compensate the other party for all losses incurred as a result.

Article 6: Conditions for Effectiveness
This Contract shall take effect from the date it is signed and sealed by the authorized representatives of both parties.

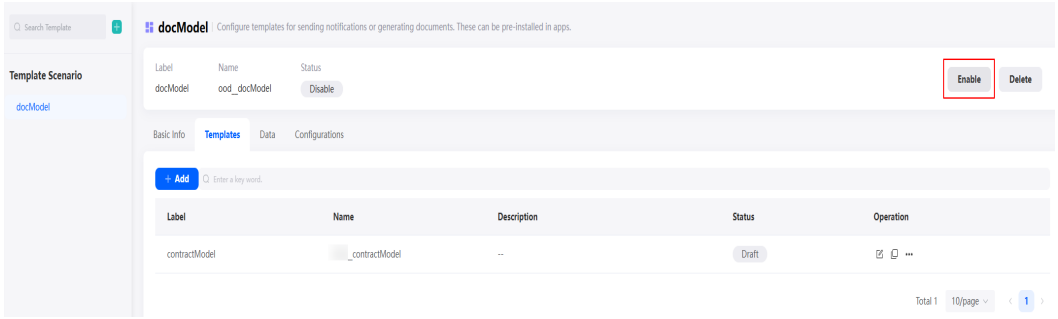
Party A: xx Co., Ltd.
Party B: ${ _otherCompanyName}

Contract Signatory: ${ __person}
Date: ${ __date}

4. If the preview meets your expectation, click **Enable** in the upper right corner of the page to enable the document template.

Step 7 Return to the document template scenario and click the enable button to enable the template scenario.

Figure 4-13 Enabling a template scenario



----End

Step 2: Create an Object to Store Template Content and the Template Instance

Create an object with fields to store both the document template content and the template instance generated later.

Step 1 In the navigation pane, choose **Data**, and click **+** next to **Object**.

Step 2 Complete the configuration and click the confirm button.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-14 Creating docObject

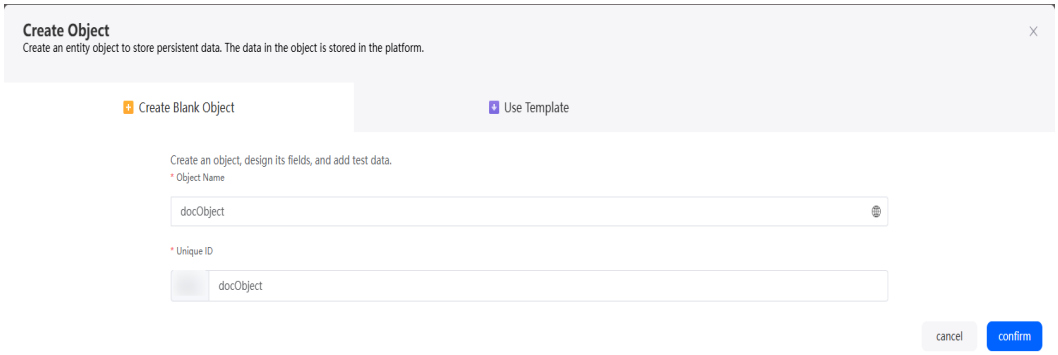


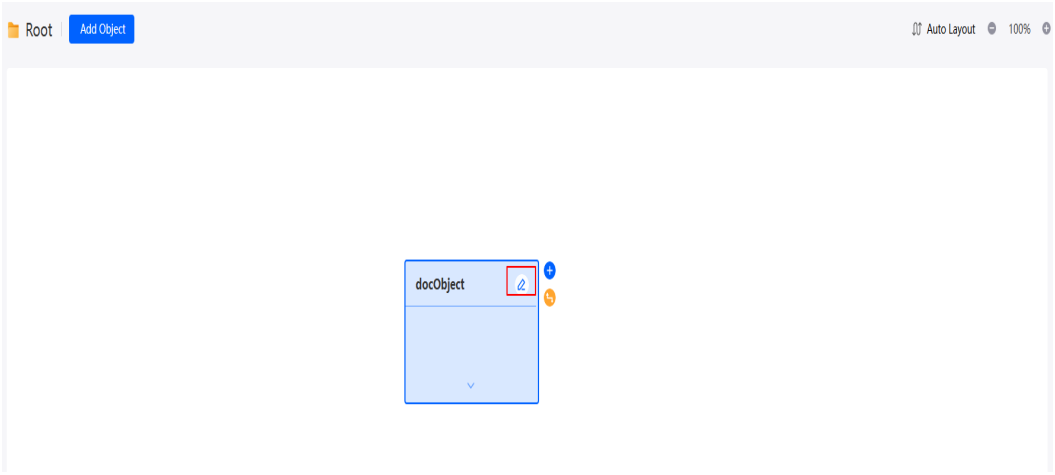
Table 4-6 Parameters for creating docObject

Parameter	Description	Example
Object Name	Name of the new object, which can be changed after the object is created. Value: 1–80 characters.	Document template object
Unique ID	ID of a new object in the system, which cannot be modified after being created. Naming rules: • Max. 63 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. • Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	docObject

Step 3 Click to go to the object details page.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-15 Clicking the edit button



Step 4 On the **Fields** tab page, click **Add** and add the **companyName** field for the object.

Figure 4-16 Adding the companyName field

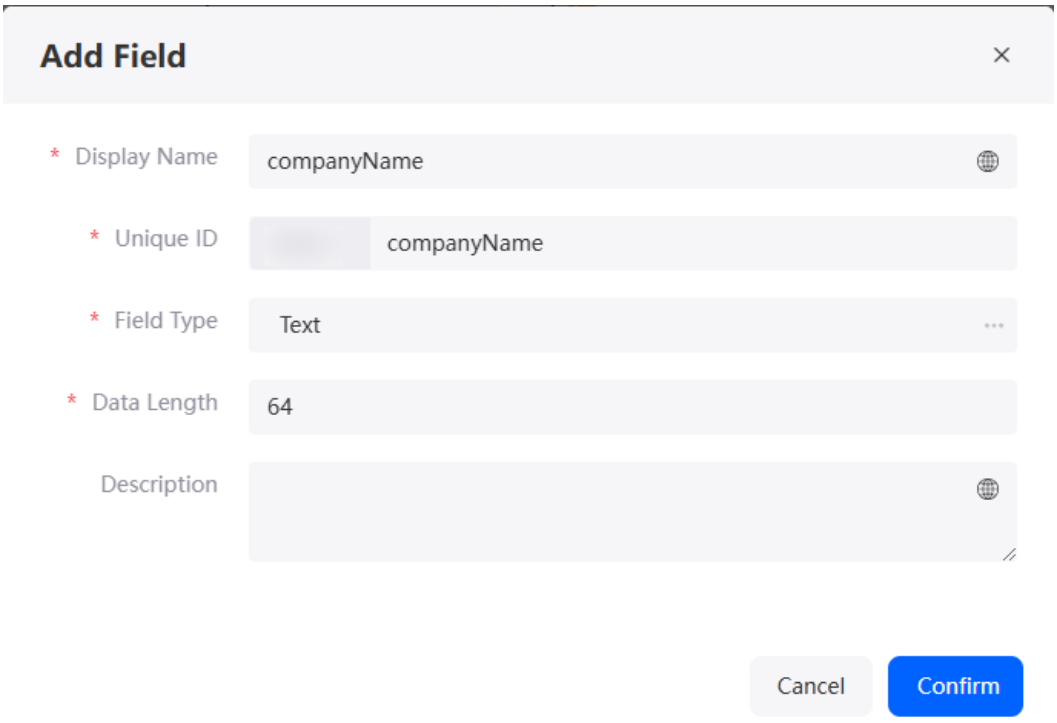


Table 4-7 Parameters for adding the companyName field

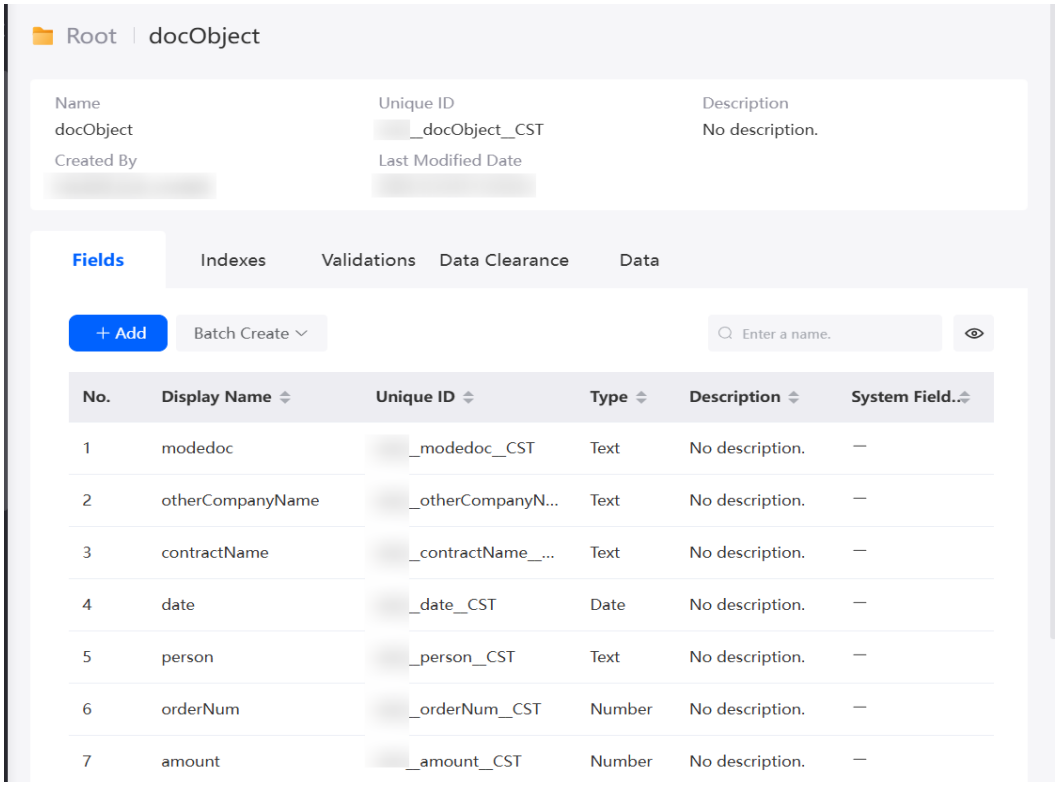
Parameter	Description	Example
Display Name	Name of the new field, which can be changed after the field is created. Value: 1–63 characters.	Company name

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Unique ID	ID of a new field in the system. The value cannot be changed after the field is created. Naming rules: • Max. 63 characters, including the prefix namespace. • Start with a letter and use only letters, digits, and an underscore (_). Do not end with an underscore (_).	companyName
Field Type	Click ***. On the page that is displayed, select the type of the new field based on the parameter description.	Text
Data Length	Length of a field that can be entered.	64

Step 5 Repeat the preceding operations to add the fields in **Table 4-8** to the object.

Figure 4-17 Viewing fields in an object



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-8 Fields to be added to the object

Name	Unique ID	Type
Company name (added)	companyName	Text
Contract amount	amount	Number
Number of orders	orderNum	Number
Contract signer	person	Text
Contract date	date	Date
Contract name	contractName	Text
Company name of party B	otherCompanyName	Text
Contract template instance	modedoc	Text (The data length is set to 255.)

----End

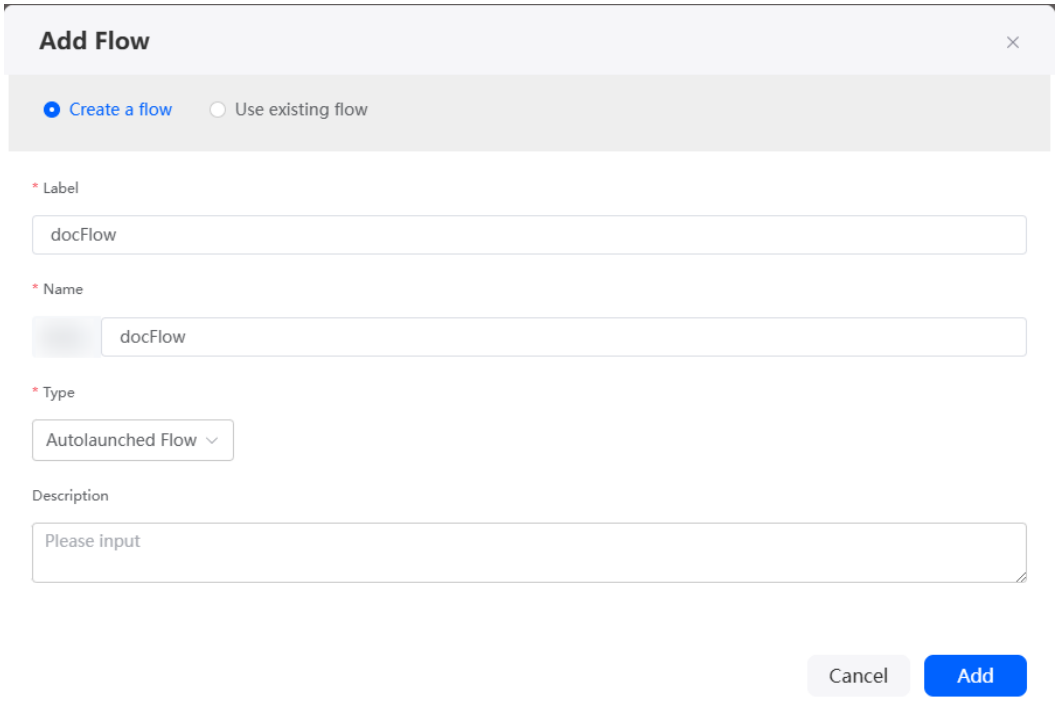
Step 3: Create a Flow to Generate a Document

Create a flow and add the **Invoke Template** and **Record Create** diagram elements to create a document based on the parameters in the contract.

Step 1 In the navigation pane, choose **Logic**, and click **+** next to **Flow**.

Step 2 Specify the label and name and click **Add**.

Figure 4-18 Creating a flow



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-9 Parameters for adding a flow

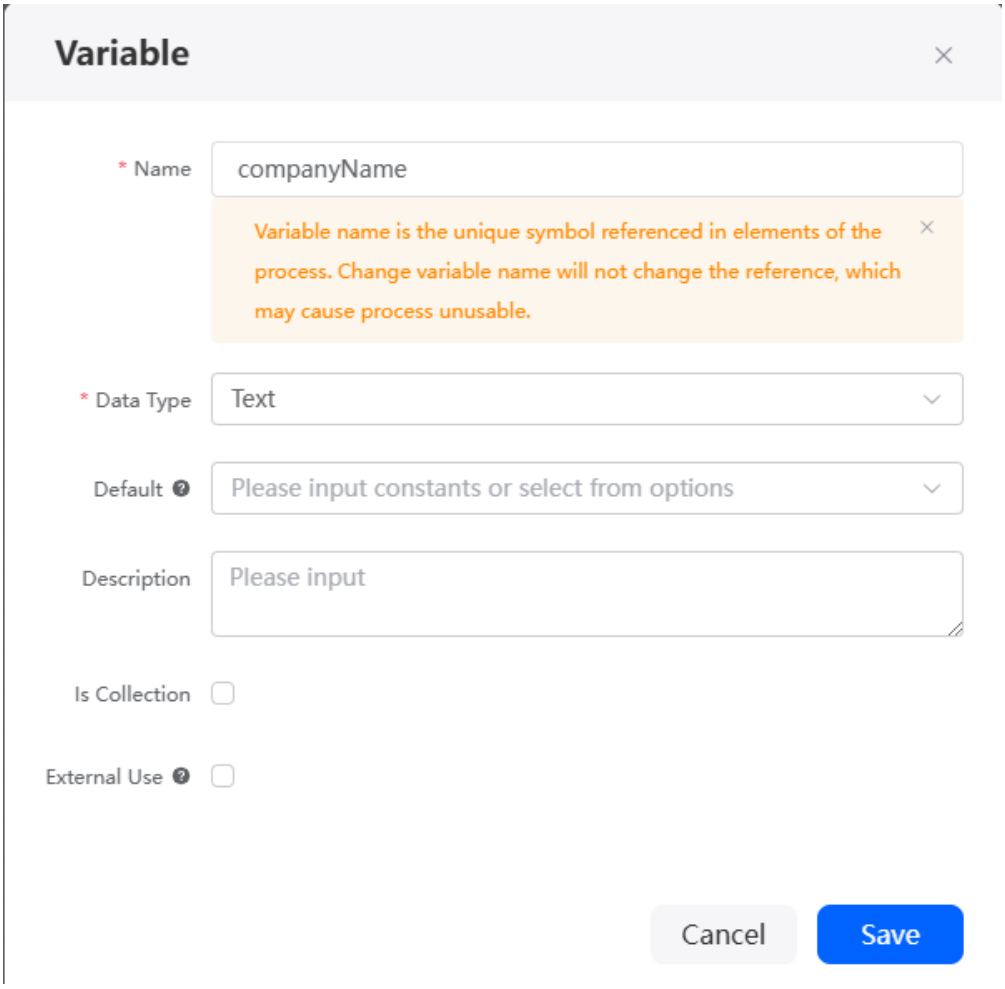
Parameter	Description	Example
Label	Flow label, which is displayed on the page and can be modified after being created. Value: 1–64 characters.	Document instance based on the contract template
Name	Unique ID of a flow in the system, which cannot be modified after being created. Naming rules: • Max. 64 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. • Start with a letter and can contain only letters, digits, and underscores (_). It cannot end with an underscore (_).	docFlow

Step 3 Create a global context variable.

1. On the flow design page, select the start node and click  .
2. Click **Context** and click  next **Variable** to create the variable **variable0**.
3. Click  next to **variable0** and select **Set**.
4. Set **Name** to **companyName** and click **Save**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-19 Creating the companyName variable



5. Repeat the preceding operations to create variables in **Table 4-10**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-20 Viewing created variables

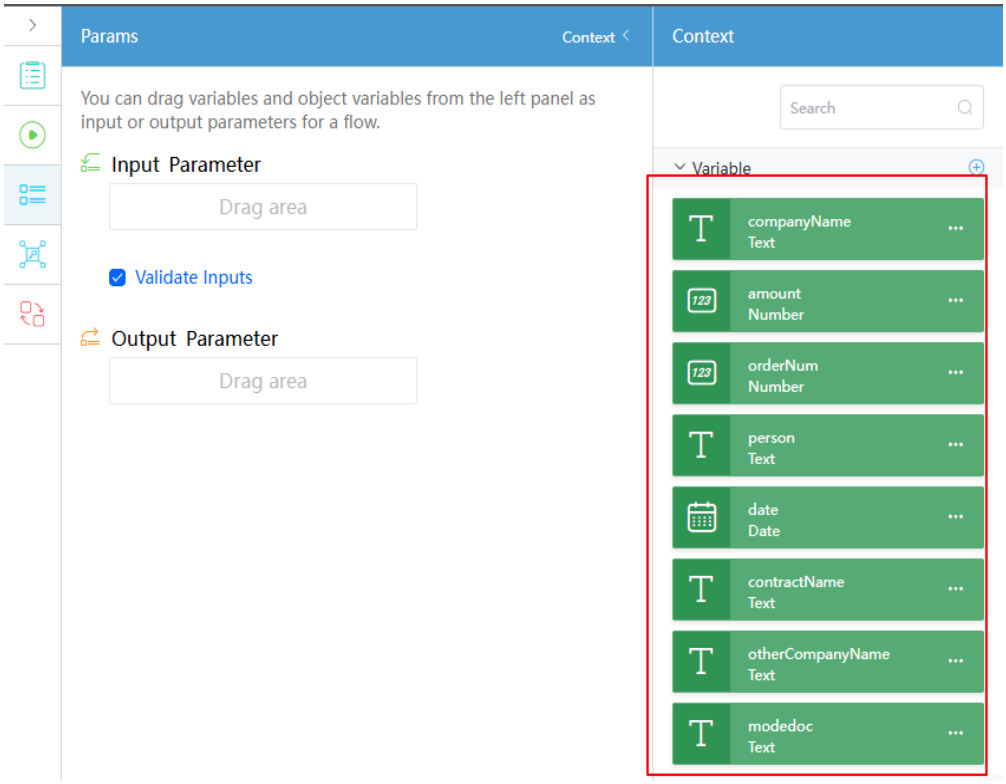


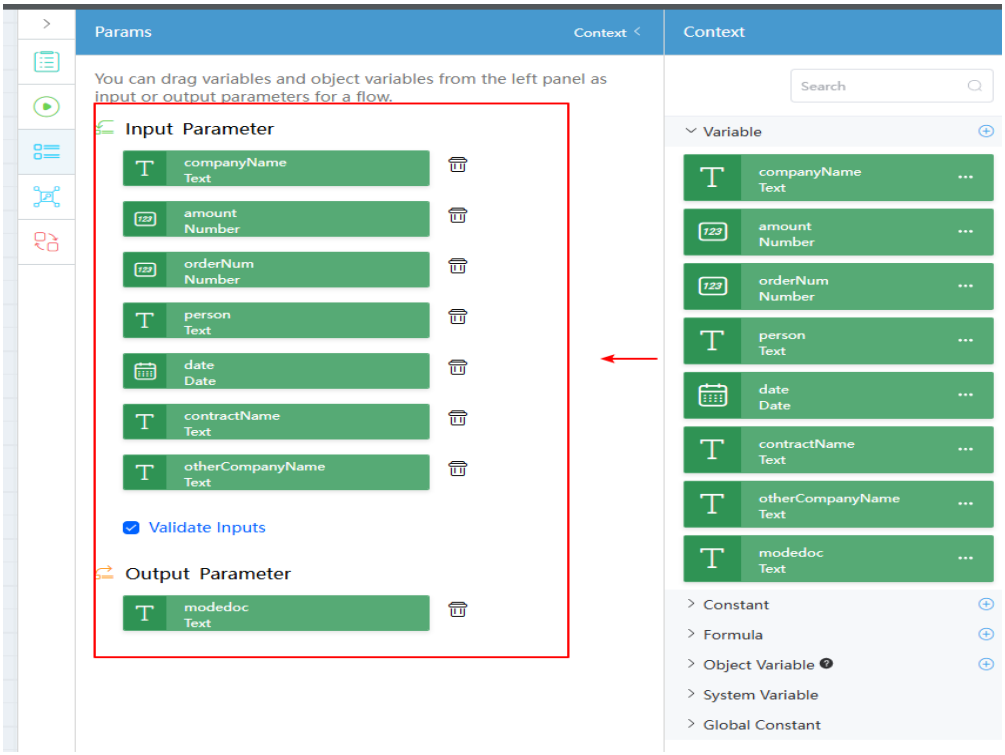
Table 4-10 Variables to be created

Name	Data Type
companyName (created)	Text
amount	Number
orderNum	Number
person	Text
date	Date
contractName	Text
otherCompanyName	Text
modedoc	Text

6. Select the start node and set the input and output parameters of the node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

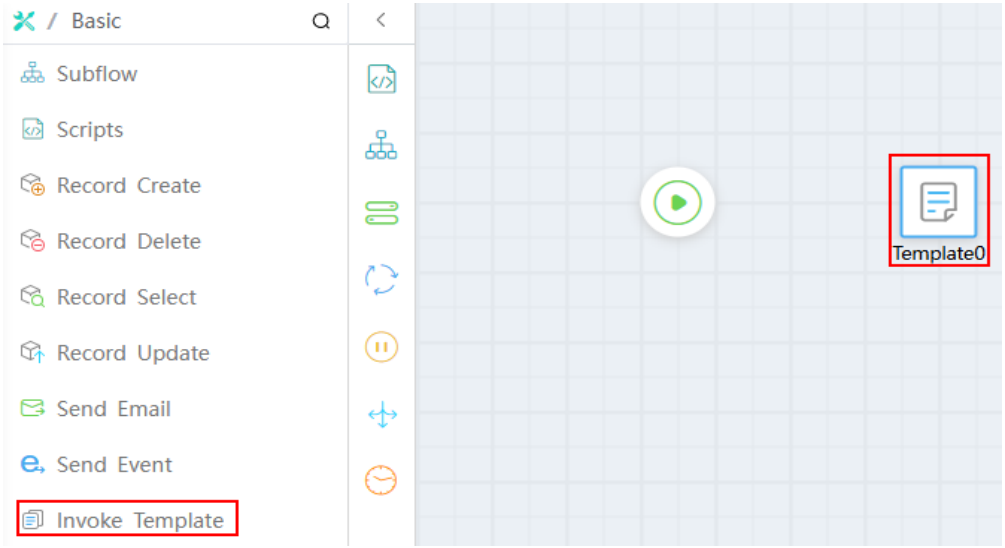
Figure 4-21 Setting input and output parameters



Step 4 Add the **Invoke Template** node.

1. Under **Basic**, drag the **Invoke Template** diagram element to the end of the start diagram element.

Figure 4-22 Dragging the Invoke Template diagram element to the canvas



2. Select the **Invoke Template** diagram element, click , and set the diagram element.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-23 Setting the Invoke Template diagram element

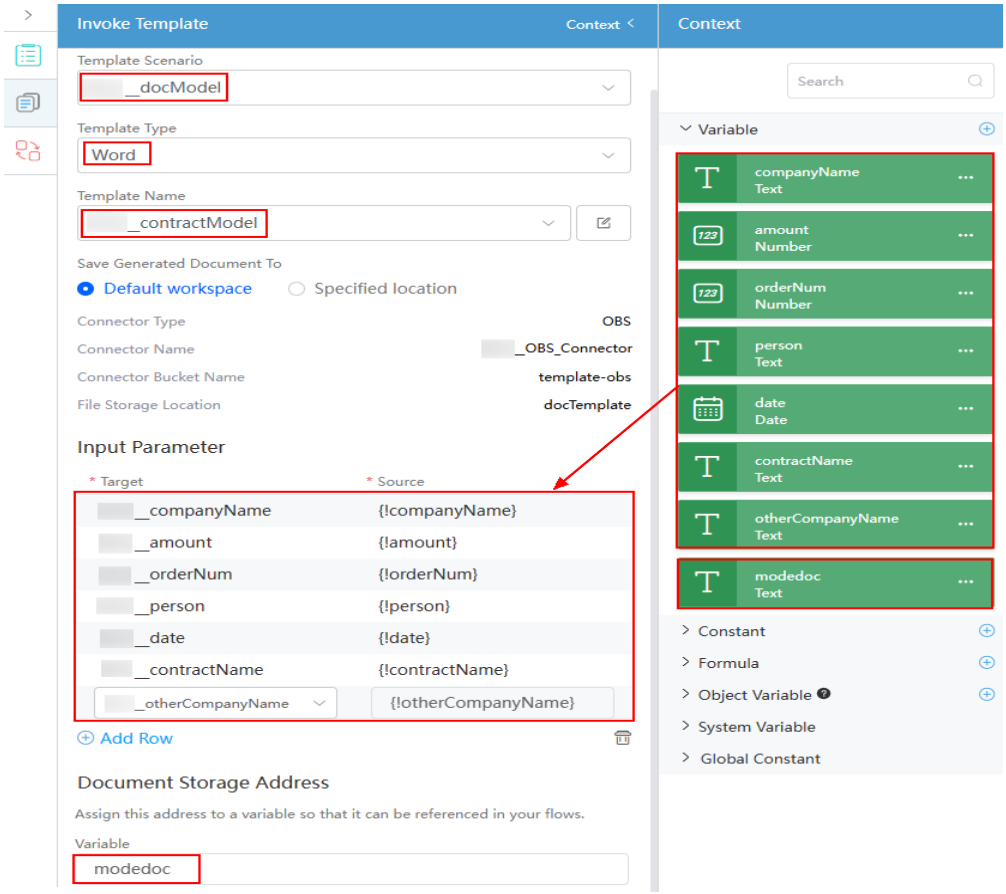


Table 4-11 Parameter description

Parameter	Description	Example
Template Scenario	Select the template scenario associated with the **Invoke Template** diagram element, that is, the template scenario created in **Step 2**.	*Namespace*__docModel
Template Type	Select a type.	Word
Document Template	Select the document template created in **Step 4**.	*Namespace*__contractModel
Connector Type	Automatically determined by the selected document template.	OBS
Connector Name	Automatically determined by the selected document template.	*Namespace*__OBS_Connector
Connector Bucket Name	Automatically determined by the selected document template.	template-obs

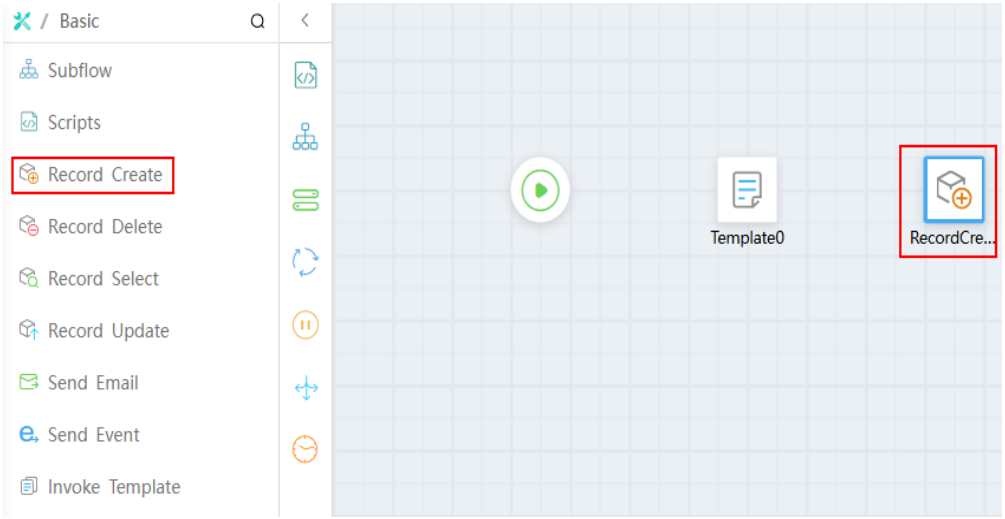
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
File Storage Location	Automatically determined by the selected document template.	docTemplate
Input Parameter	Transfer data to the template and assign the input parameter variables to the corresponding template parameters at a time.	Variables created in **Step 3**
Document Storage Address	Add the output variable **modelDoc** in the document storage address to store the generated document name.	modedoc

Step 5 Add a record creation node.

1. Drag **Record Create** from **Basic** and place it after **Invoke Template**.

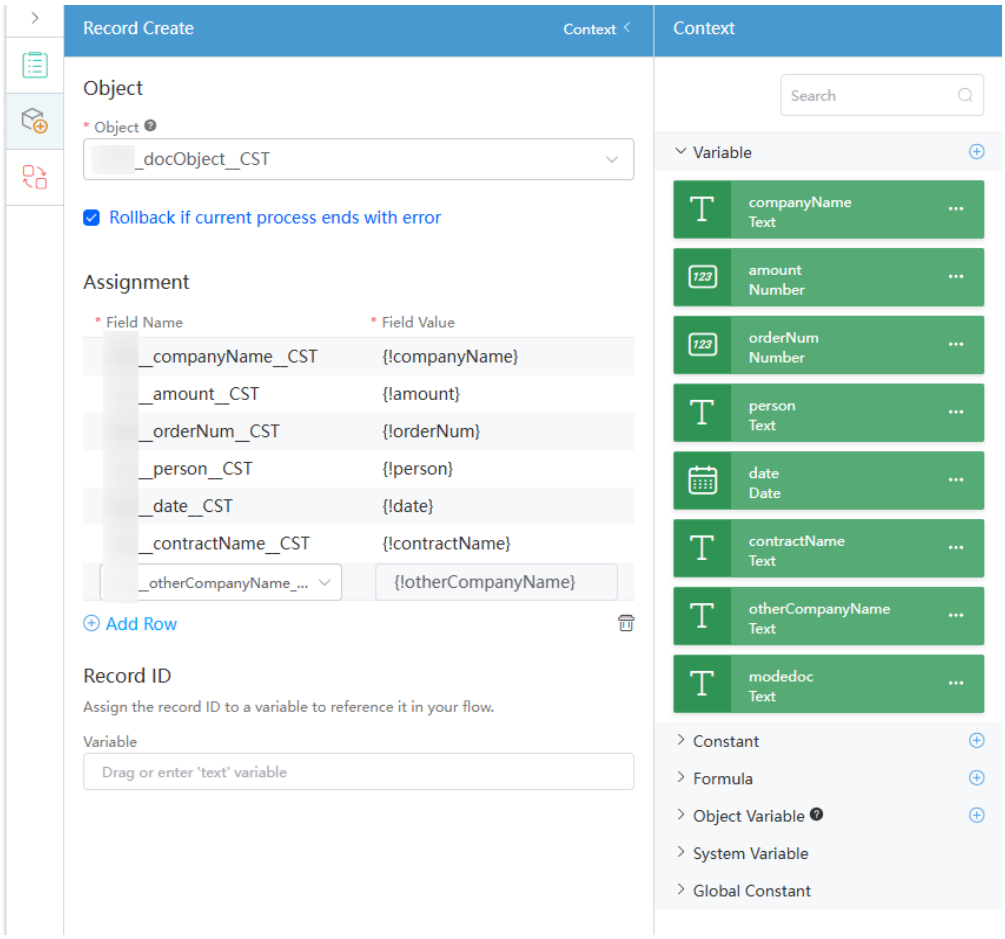
Figure 4-24 Adding Record Create



2. Select the **Record Create** diagram element and click  to save the input and output parameter variables to the object created in **Step 2: Create an Object to Store Template Content and the Template Instance**.

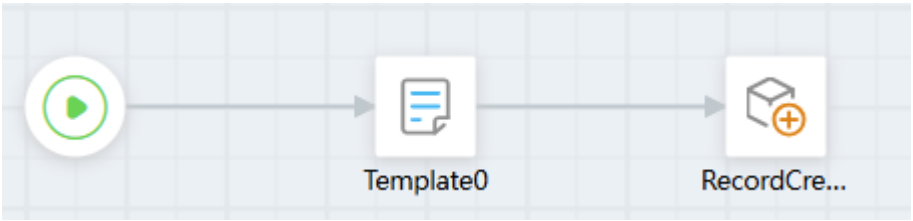
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-25 Configuring Record Create



Step 6 Connect the diagram elements in this order: **Start** to **Invoke Template**, then **Invoke Template** to **Record Create**.

Figure 4-26 Specifying the logical relationship between diagram elements



Step 7 Click  to save the flow.

Step 8 Click  . The flow debugging page is displayed.

Enter the following information in the input parameter text box and click **Run**:

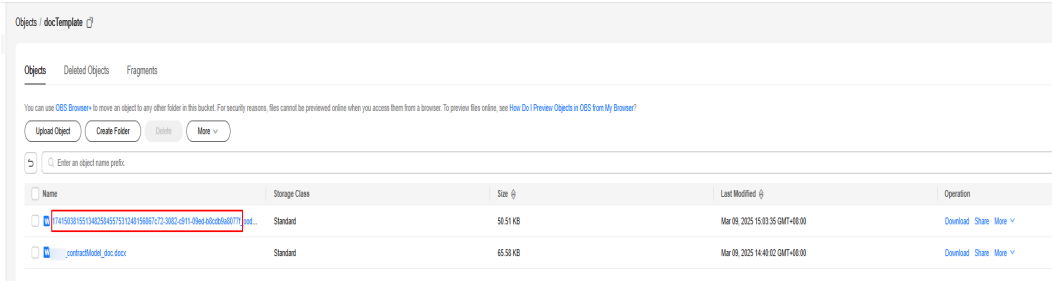
```
{
  "companyName": "Company A",
  "amount": "10",
  "orderNum": "2",
  "person": "Zhang San",
  "date": "2024-11-05",
  "contractName": "New purchase contract",
  "otherCompanyName": "Company B"
}
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The command is successfully executed if the following information is displayed. Log in to the OBS console. The contract in **Figure 4-27** is generated in the path configured in **Step 3**. Download the contract to the local host and view the content, as shown in **Figure 4-29**. You can find that the content in the contract has been replaced.

Figure 4-27 Flow executed successfully

Figure 4-28 Viewing the newly generated contract



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-29 Viewing the contract content

Sales Contract

Article 1: Purpose of the Contract
This Contract is intended to specify the terms and conditions under which companyApurchases goods/services from Party B.

Article 2: Description of Goods/Services
A detailed description of the goods/services, including but not limited to model, specifications, quantity, unit price, etc.
Order Number: 2

Article 3: Price Terms
The total amount for the goods/services is 10RMB.

Article 4: Quality Assurance
Party B guarantees that the provided goods/services meet the agreed quality standards.

Article 5: Liability for Breach of Contract
If one party breaches the terms of this Contract, the breaching party shall compensate the other party for all losses incurred as a result.

Article 6: Conditions for Effectiveness
This Contract shall take effect from the date it is signed and sealed by the authorized representatives of both parties.

Party A: companyA
Party B: companyB

Contract Signatory: Paul
Date: 2024-11-05

Step 9 Click  to activate the flow.

----End

Step 4: Create a Script for Downloading the Document

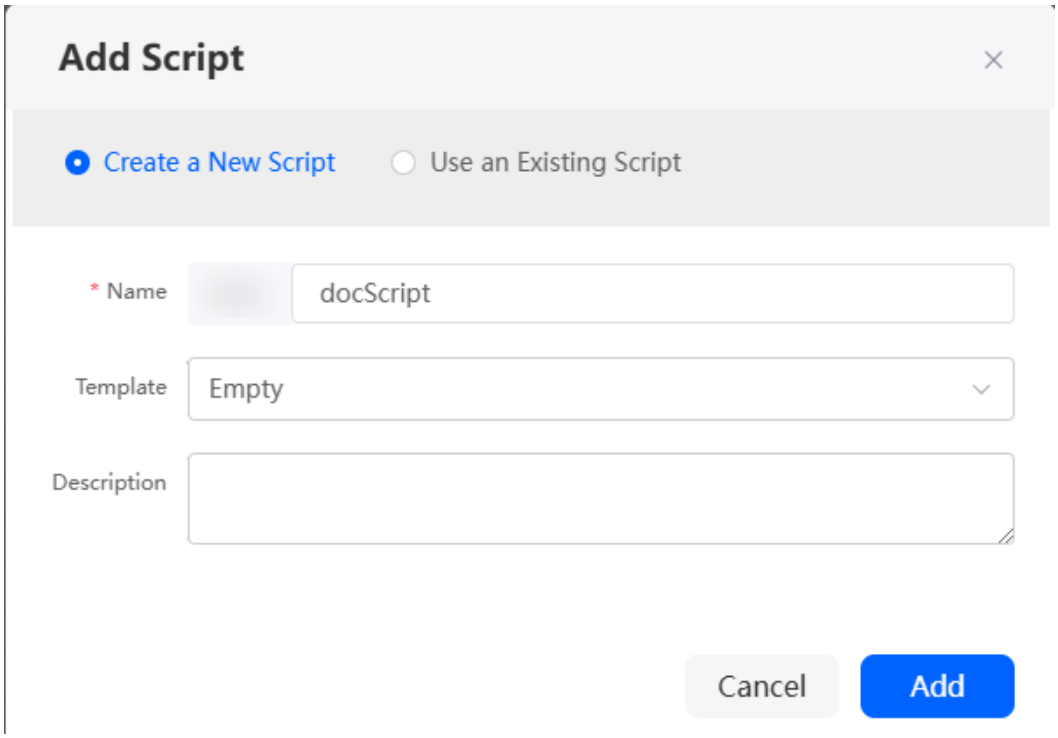
Create a script to download the contract generated in the OBS bucket based on the document name.

Step 1 In the navigation pane, choose **Logic** and click **+** next to **Script**.

Step 2 Create a script, set the name to **docScript**, and click **Add**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-30 Creating docScript



Step 3 In the script editor, enter the sample code.

This sample code is used to download the document. In the example, ***Namespace*__OBS_Connector** is the connector name configured in **Step 3**, and **docTemplate** is the file storage location configured in **Step 3**.

```
import * as context from 'context';//Import the standard library related to the context.
import * as objectstorage from 'objectstorage';


//Define the input parameter structure.
@action.object({ type: "param" })
export class ActionInput {
    @action.param({ type: 'String', required: true, label: 'String' })
    docName: string;
}
//Define the output parameter structure.
@action.object({ type: "param" })
export class ActionOutput {
    @action.param({ type: 'Any' })
    buf: any;
}

@action.object({ type: "method" })
export class CreateWorkOrder {    //Define the API class. The input parameter of the API is ActionInput, and the output parameter is ActionOutput.
    @action.method({ input: 'ActionInput', output: 'ActionOutput' })
    public createWorkOrder(input: ActionInput): ActionOutput {
        let out = new ActionOutput();    //Create an instance of the ActionOutput type as the return value.
        let error = new Error();    //Create an instance of the error type to save the error information when an error occurs.
        try {
            // OBS bucket path, which is the same as that in the template configuration.
            let path = "docTemplate/";
            // Call the connector to download the file. Namespace__OBS_Connector indicates the name of the called connector.
            let obsCli = objectstorage.newClient(objectstorage.StoreType.OBS, "Namespace__OBS_Connector");
            let data = obsCli.getObject(path + input.docName);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
            out.buf = data;
        } catch (error) {
            console.error(error.name, error.message);
            context.setError(error.name, error.message);
        }
        return out;
    }
}
```

Step 4 Click  to save the script.

Step 5 Test the script.

1. Click  in the upper part of the editor to run the script.
2. Set the input parameters, click  in the upper right corner of the test window, and check the returned message.

```
{
  "docName": "173087468358901385574315226000 7e733dd-80ea-19f1-
b7a0-93fdcd20a541_Namespace__contractModel.docx"
}
```

173087468358901385574315226000 7e733dd-80ea-19f1-b7a0-93fdcd20a541_Namespace__contractModel.docx is the document generated in **Step 3: Create a Flow to Generate a Document**. That is, use the script to download the actual contract in the OBS bucket based on the document name.

Figure 4-31 Viewing output parameters

Step 6 After the script test is complete, click  in the upper part of the editor to activate the script.

----End

Step 5: Call the Flow and Script on a Standard Page to Generate and Download the Document

Design a standard page and call the flow and script to generate and download the contract on the frontend page.

Step 1 Creating a standard page

1. In the navigation pane, choose **Page**, and click **+** next to **Standard Page**.
2. Specify the label and name of the standard page and click **Add**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-32 Adding a standard page

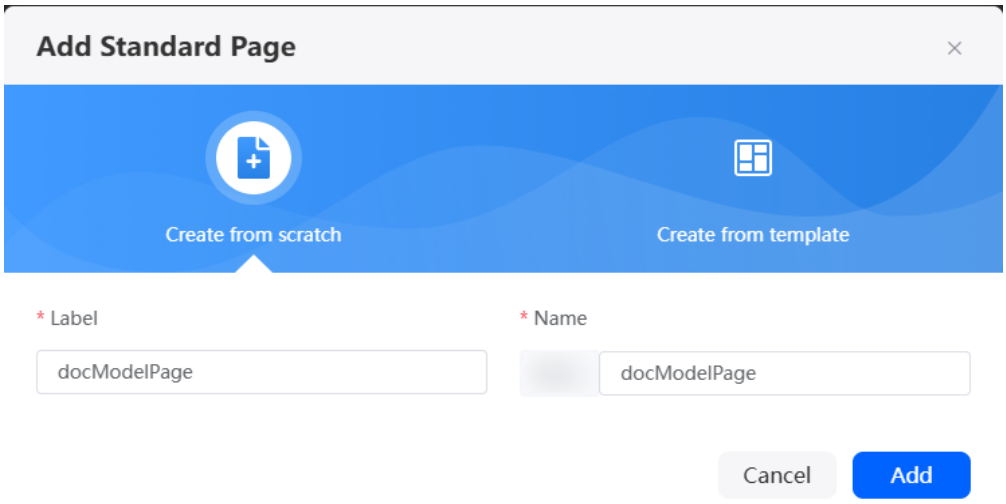


Table 4-12 Parameters for creating a standard page

Parameter	Description	Example
Label	Label of the standard page, which is displayed on the page and can be modified after being created. Value: 1–64 characters.	Standard page of the document template
Name	Name of the standard page. The name is the unique identifier of the standard page in the system and cannot be changed after being created. Naming rules: – Max. 64 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and can contain only letters, digits, and an underscore (_). It cannot end with an underscore (_).	docModePge

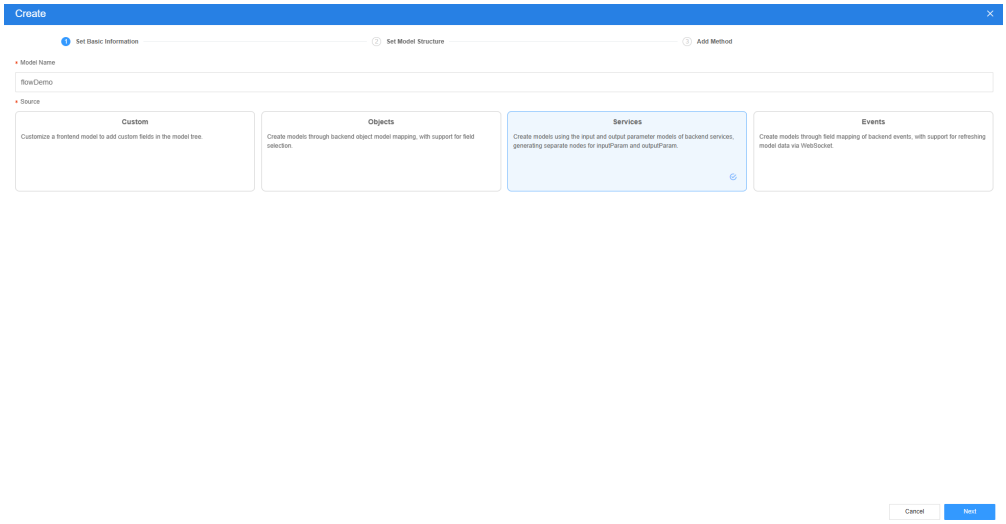
Step 2 Creating an object model

1. At the bottom of the standard page, click **Model View**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

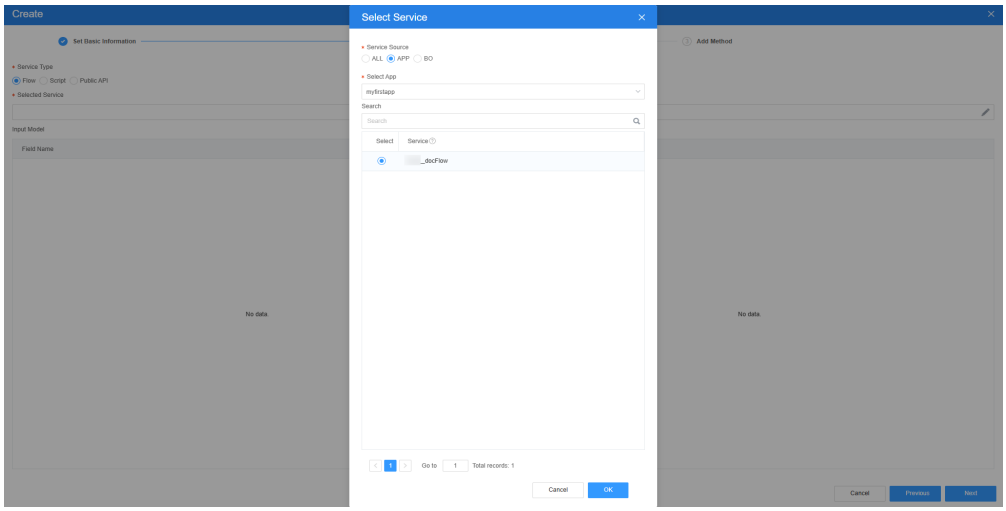
2. Click **New**, specify **Model Name** (for example, **flowDemo**), select **Services** for **Source**, and click **Next**.

Figure 4-33 Adding a flow object model



3. Select the flow created in **Step 3: Create a Flow to Generate a Document**, click **Next**, and then click **OK**.

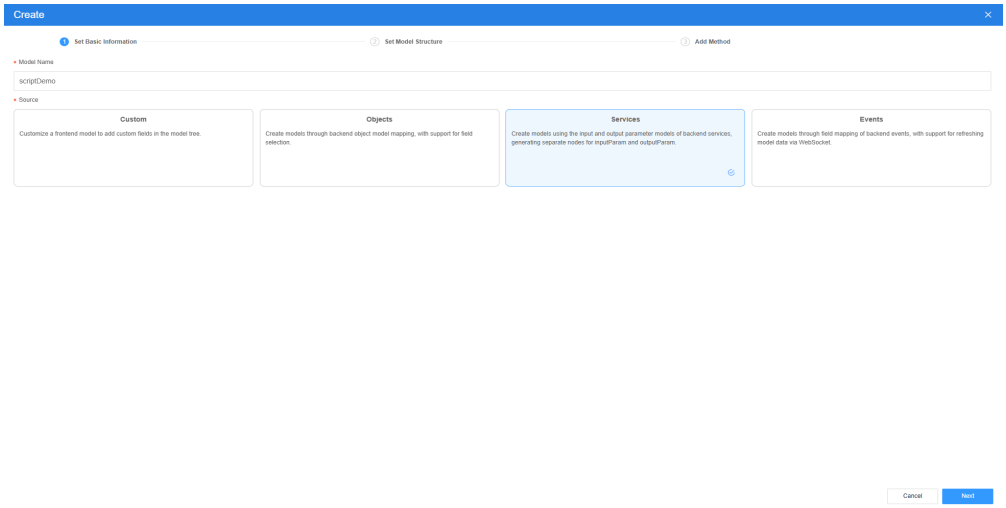
Figure 4-34 Selecting the target flow



4. Click **New**, specify **Model Name** (for example, **scriptDemo**), select **Services** for **Source**, and click **Next**.

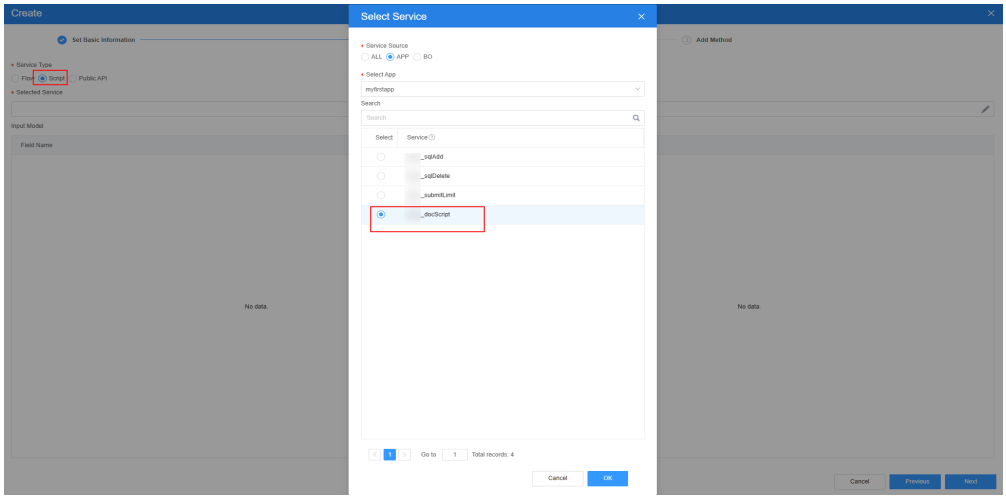
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-35 Adding a script object model



5. Select the script created in **Step 4: Create a Script for Downloading the Document**, click **Next**, and then click **OK**.

Figure 4-36 Selecting the target script

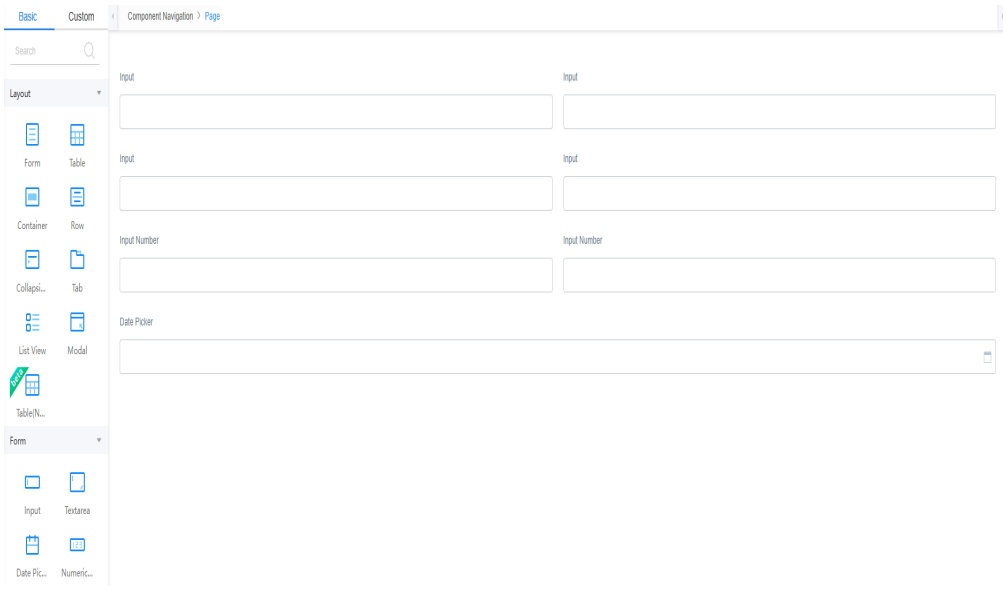


Step 3 Designing a standard page for generating and downloading the contract

1. At the bottom of the standard page, click **Designer View**.
2. Drag four text boxes, two number text boxes, and a date selector under **Basic** to the canvas of the standard page, for example, **Figure 4-37**.

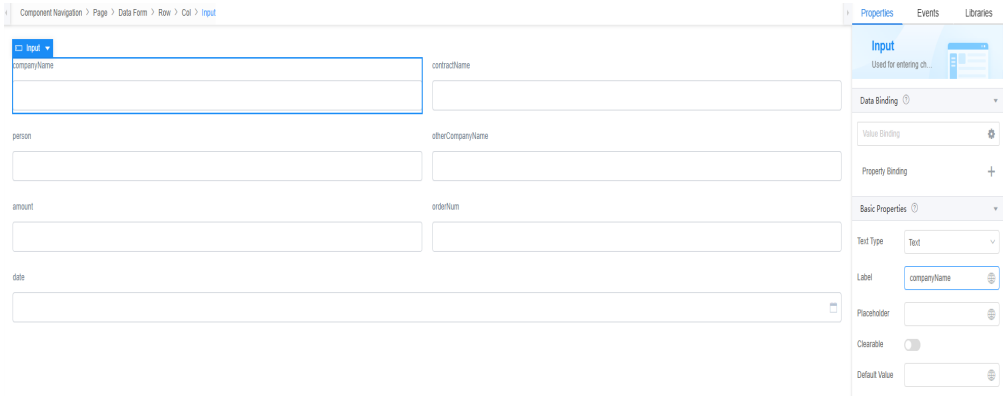
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-37 Dragging widgets to the canvas



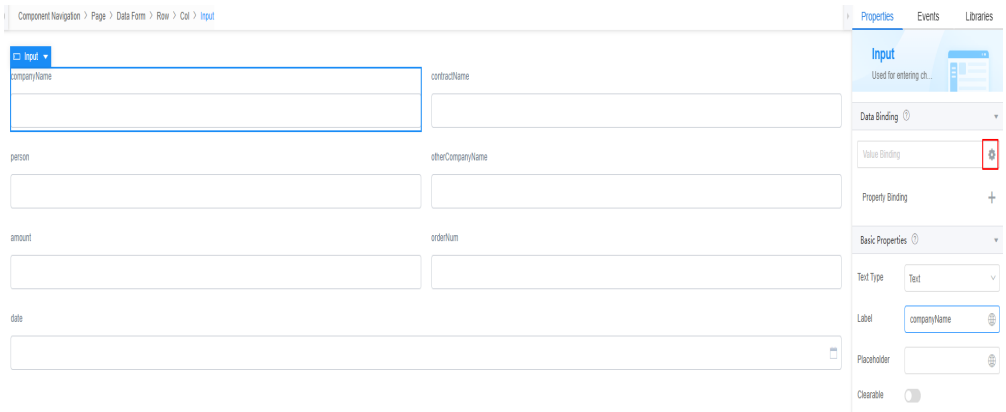
3. Select a widget to modify its label, as shown in **Figure 4-38**.

Figure 4-38 Modifying a widget label



4. Select the company name text box. Choose **Properties** > **Data Binding** and click next to **Value Binding**.

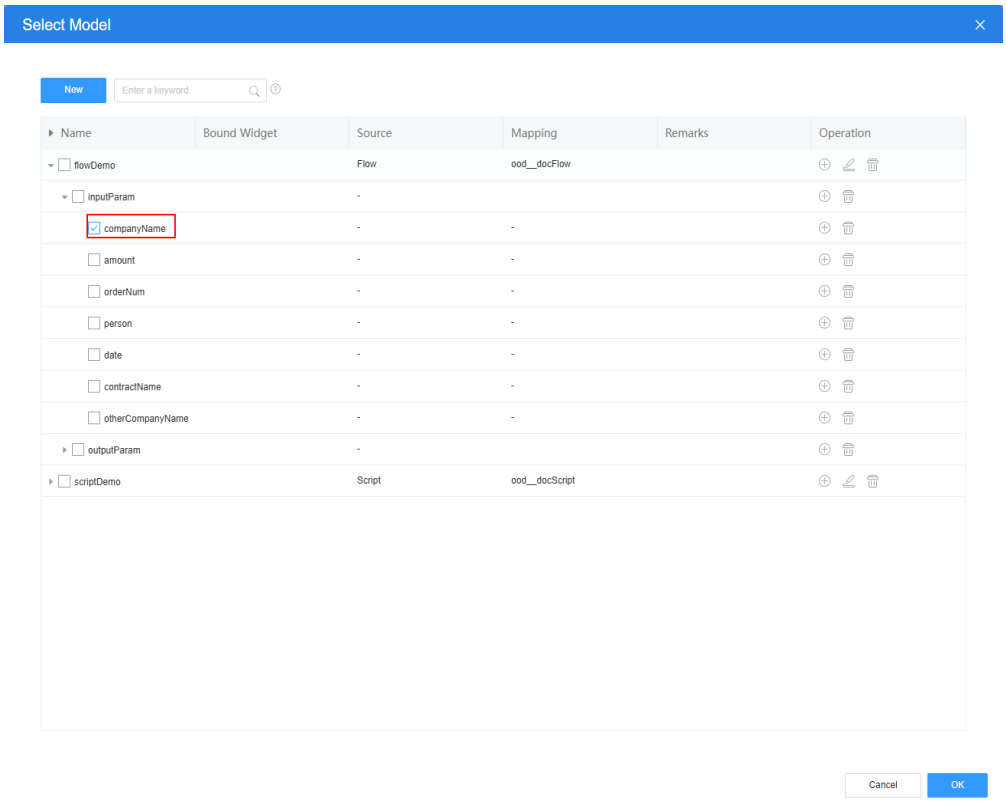
Figure 4-39 Clicking the icon next to Value Binding



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

5. Select **companyName** in **Step 2.2** and click the confirm button.

Figure 4-40 Binding companyName



6. Repeat the preceding operations to bind models to other widgets.

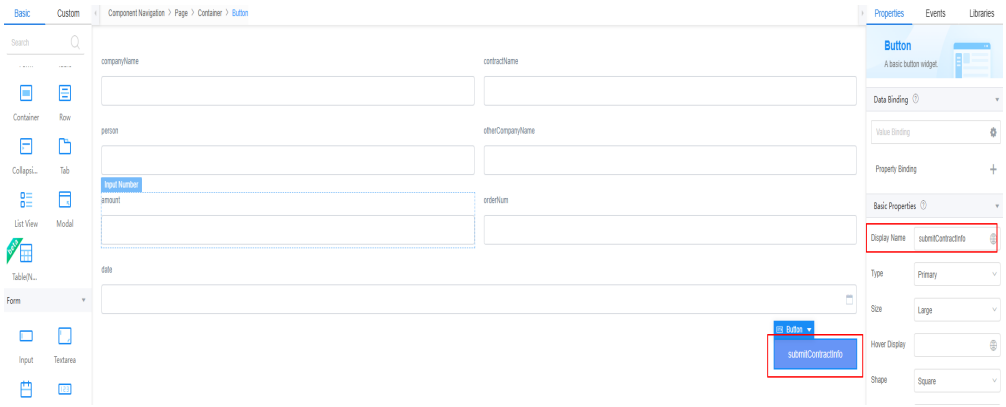
   The contract name is bound to **flowdemo.inputParam.contractName**, the contract signer is bound to **flowdemo.inputParam.person**, the company name of party B is bound to **flowdemo.inputParam.otherCompanyName**, the contract amount is bound to **flowdemo.inputParam.amount**, the number of orders is bound to **flowdemo.inputParam.orderNum**, and the order date is bound to **flowdemo.inputParam.date**.

Step 4 Adding a button widget for the standard page

1. Choose **Basic** > **Common**, drag a button widget to the area below the order date widget, and change the **Display Name** of the button widget to "Contract parameters submission and document generation instance".

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-41 Adding a button widget



2. Select the button widget. On the **Events** tab page, click + next to **on-click**. The page for adding an action is displayed.
3. Under **Custom Action**, enter the following sample code and click **Create**.

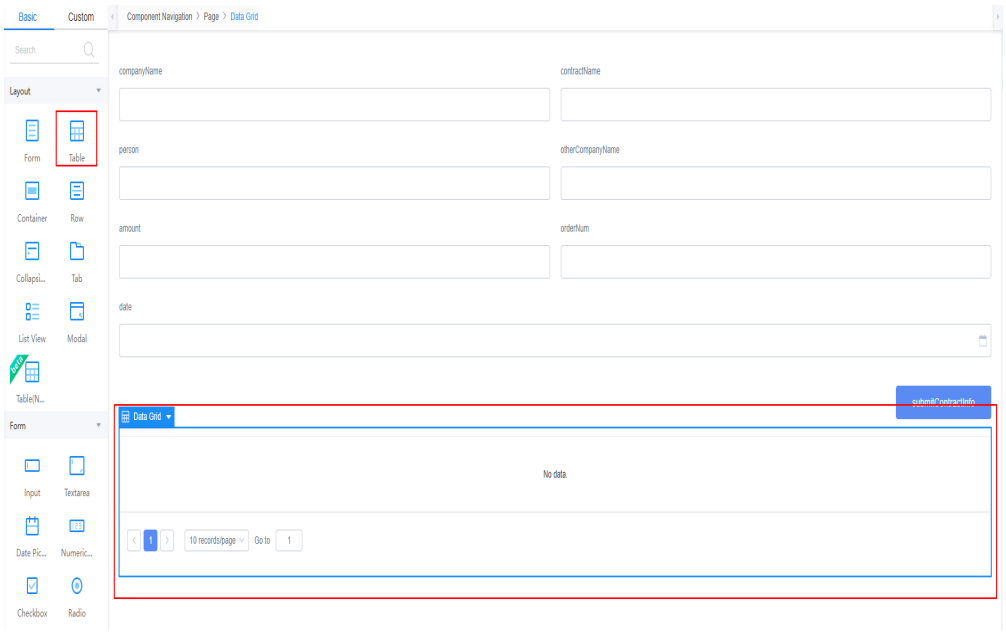
   The sample code is used to generate a contract based on the contract parameters set in the flow model. **flowDemo** indicates the flow model created in **Step 2.2**.

```
$model.ref('flowDemo').run().then(function(data){
    console.log(data);
    context.$message.success('Submitted successfully.');
}).catch(function(error){
    context.$message.error('Submission failed:' + error.resMsg);
});
```

Step 5 Adding a table widget for the standard page

1. Choose **Basic** > **Layout** and drag a table widget below the button widget.

Figure 4-42 Adding a table widget

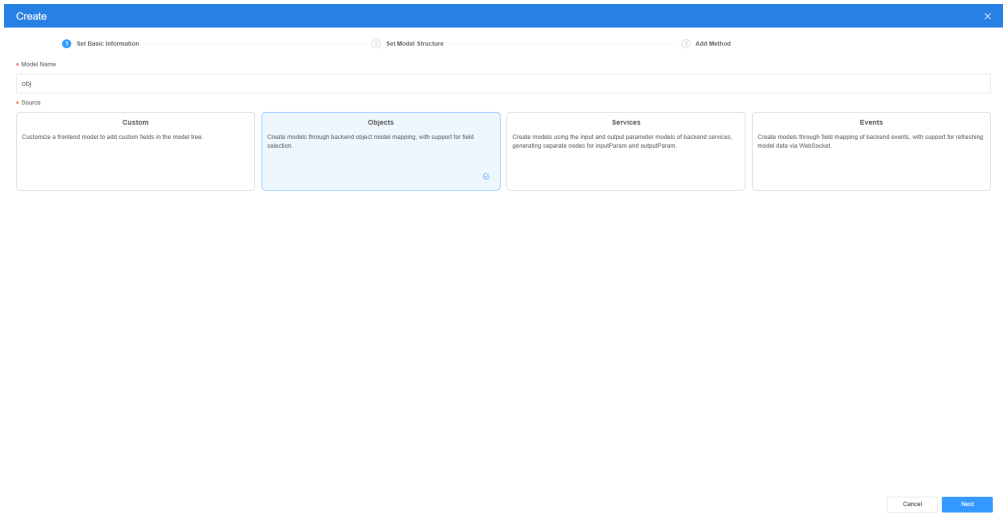


2. Select the table widget, choose **Properties** > **Data Binding**, and click ⚙ next to **Value Binding**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

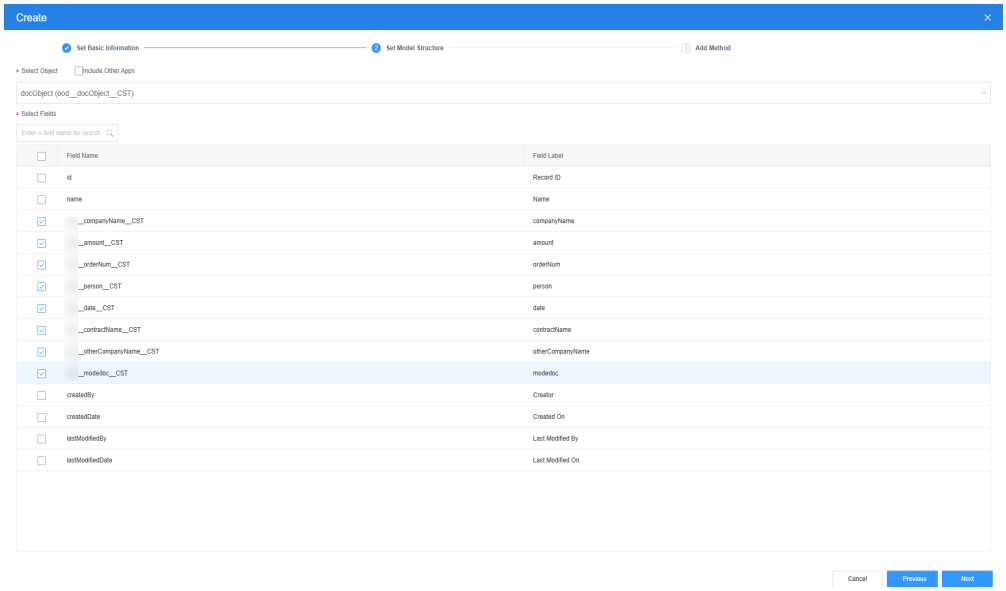
3. Click **New**, specify **Model Name** (for example, **obj**), select **Objects** for **Source**, and click **Next**.

Figure 4-43 Setting basic model information



4. Select the object and fields created in **Step 2: Create an Object to Store Template Content and the Template Instance**, click **Next**, and click **OK**.

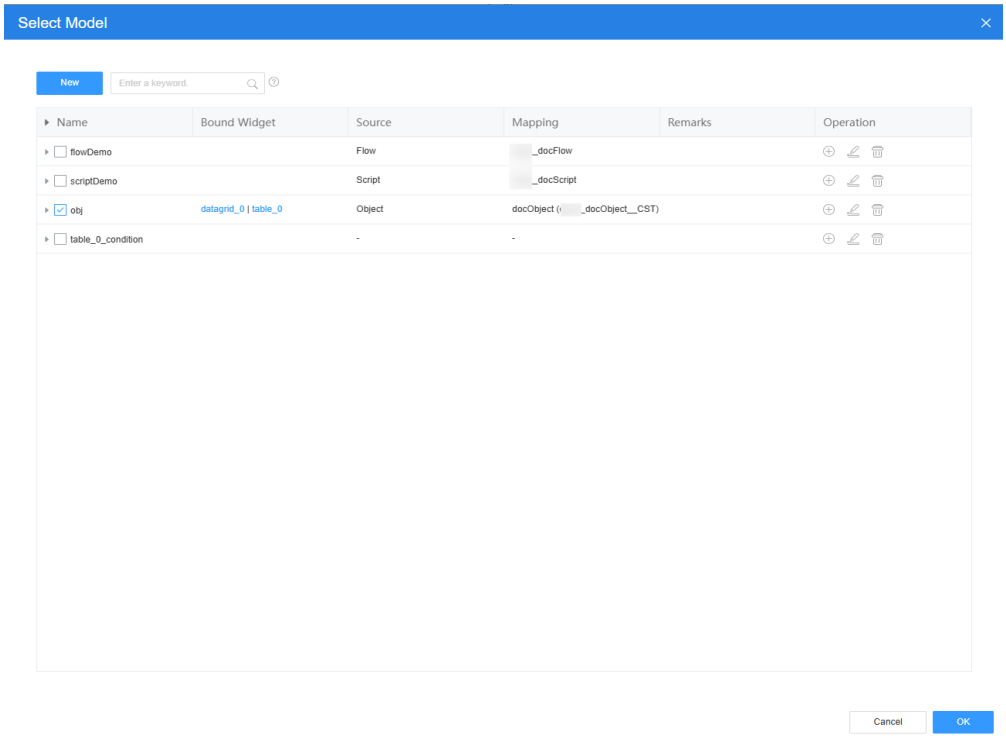
Figure 4-44 Setting the model



5. In the **Select Model** dialog box, select the created model and bind the object model to the table.

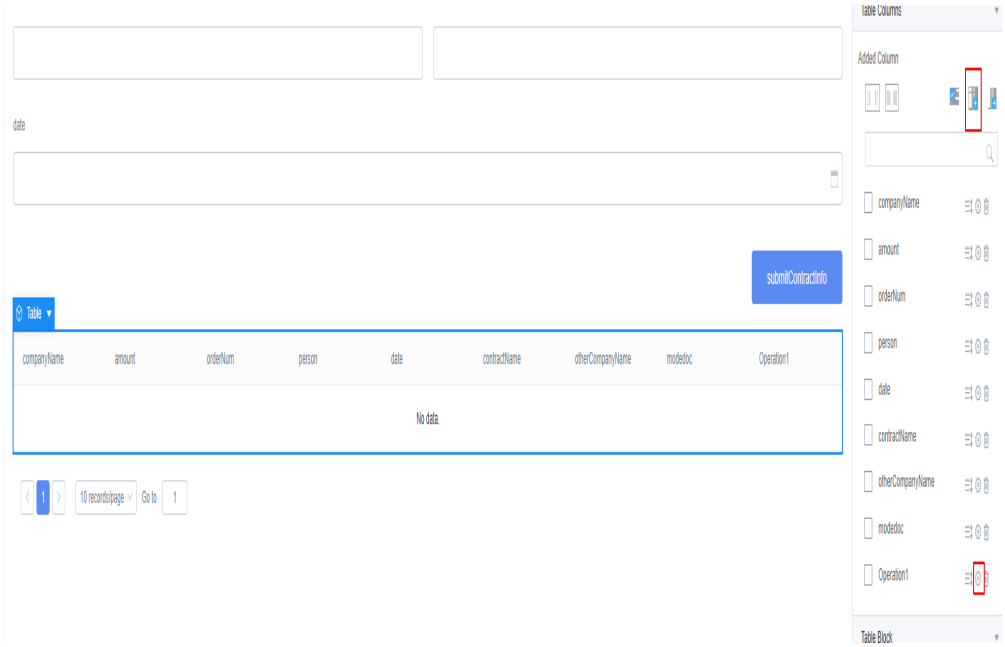
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-45 Binding the model to the table



6. Select the table widget and add an operation column to the table.

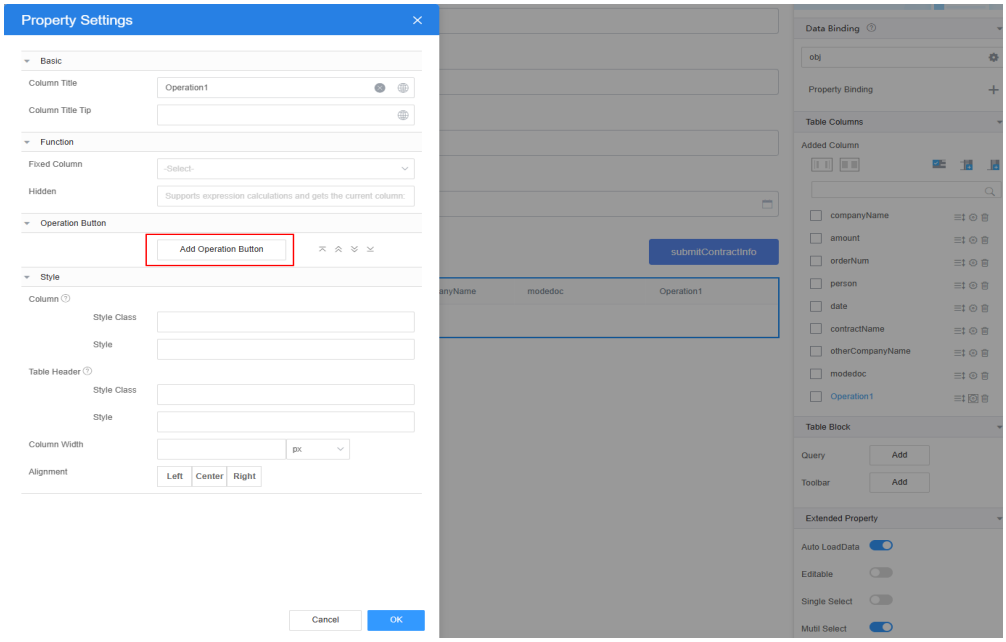
Figure 4-46 Adding an operation column



7. Click next to the new column. Under **Operation Button**, click **Add Operation Button**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

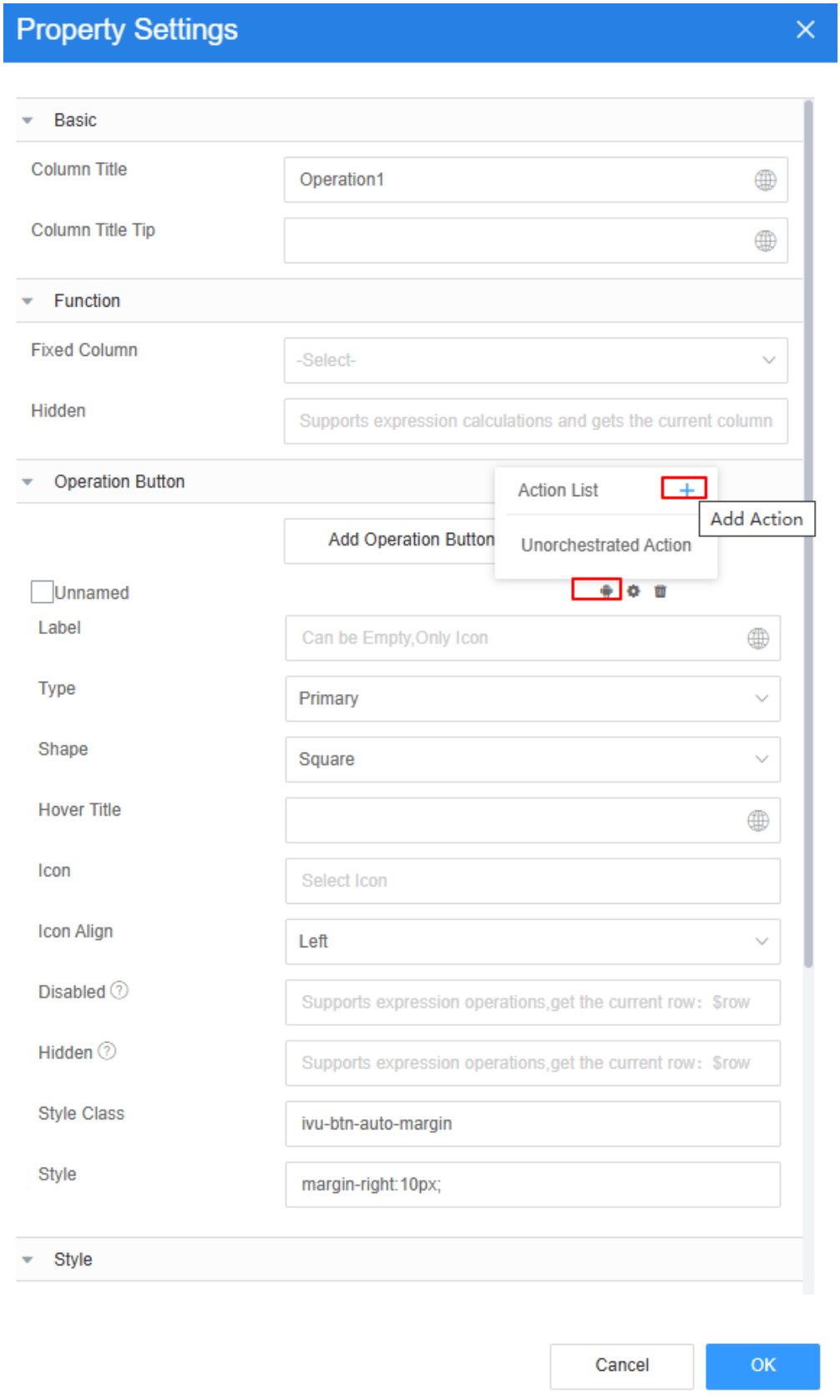
Figure 4-47 Adding an operation button



8. Click and then click next to **Action List** to add an action for the button.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-48 Adding an action for the button



9. Under **Custom Action**, enter the following sample code and click **Create**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

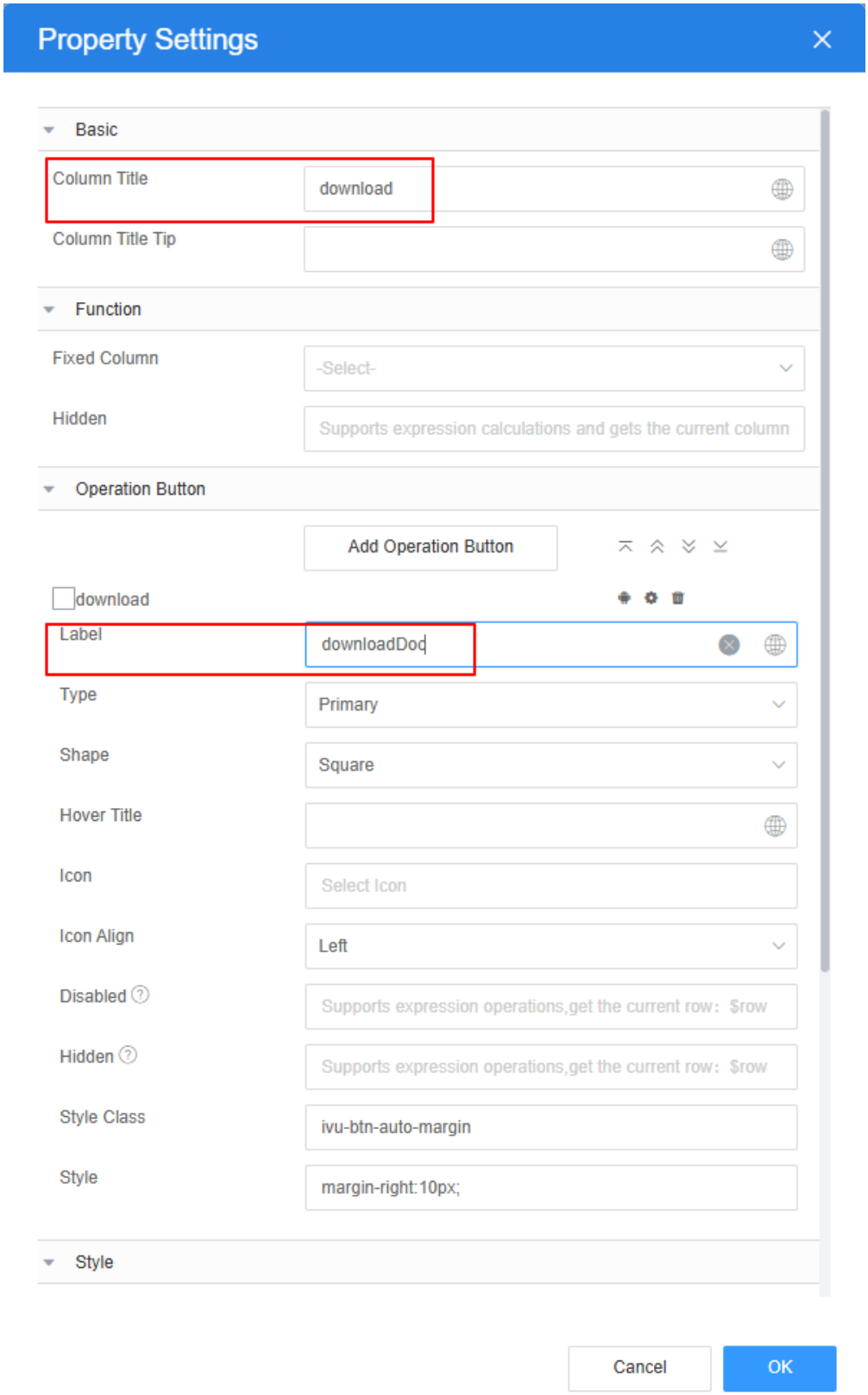
The sample code is used to download the document generated by the flow from OBS based on the field name of the contract template instance. ***Namespace__modedoc__CST*** is the field name of the contract template instance, and **scriptDemo** is the script object model created in **Step 2.4**.

```
// Base64 decoding function
function base64DecodeToBinary(base64String) {
  // Decode the Base64 character string.
  const binaryString = atob(base64String);
  // Convert the decoded string into binary data.
  const len = binaryString.length;
  const bytes = new Uint8Array(len);
  for (let i = 0; i < len; i++) {
    bytes[i] = binaryString.charCodeAt(i);
  }
  // Convert Uint8Array to Blob.
  const blob = new Blob([bytes]);
  return blob;
}
// Obtain the row data in the table. Namespace__modedoc__CST is the field name of the contract template instance in the table.
var rowData = context.$component.current.$attrs.row;
const docName = rowData.Namespace__modedoc__CST;
// Set the input parameters of the script.
let originData = $model.ref('scriptDemo').getData();
originData.inputParam.docName = docName;
$model.ref('scriptDemo').setData(originData);
//Run the script to download the document generated by the flow from OBS.
$model.ref('scriptDemo').run().then(function(data){
    const decodedString = base64DecodeToBinary(data.buf);
    const url = URL.createObjectURL(decodedString);
    let link = document.createElement('a');
    link.href = url;
    // Set the name of the document to be downloaded. Namespace__modedoc__CST is the field name of the contract template instance in the table.
    link.setAttribute('download', rowData.Namespace__modedoc__CST);
    document.body.appendChild(link);
    link.click();
    URL.revokeObjectURL(link.href);
    document.body.removeChild(link);
    context.$message.success('DownLoad successfully.');
}).catch(function(error){
    console.log('error is', error);
    context.$message.error('DownLoad failed:' + error.resMsg);
});
```

10. Set the label of the operation button to **downloadDoc**, and set **Column Title** under **Basic** to **download**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

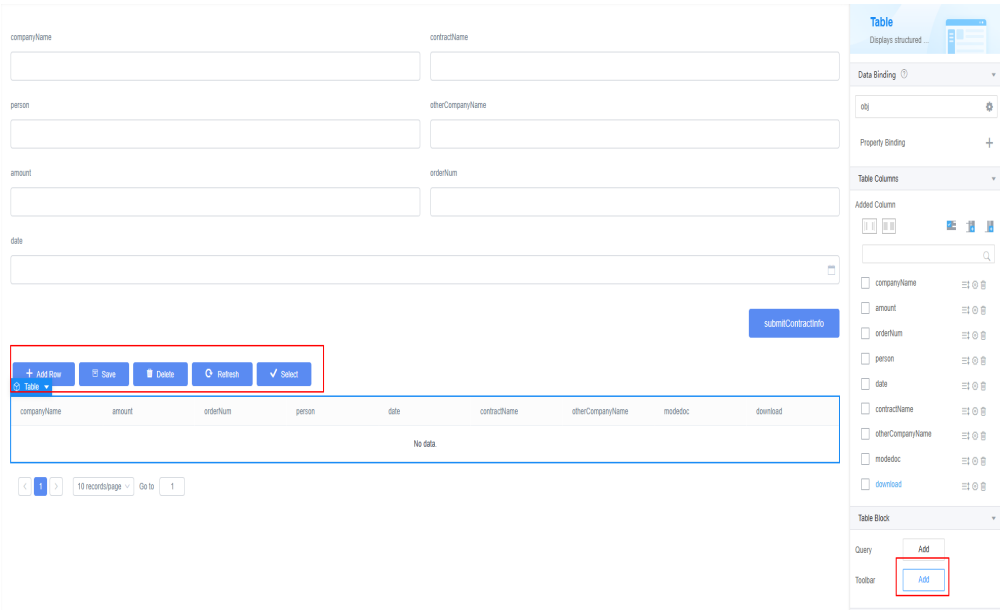
Figure 4-49 Editing the operation column title and operation label



11. Return to the standard page, choose **Properties** > **Table Block**, and click **Add** next to **Toolbar** to add a toolbar to the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-50 Adding a toolbar to the table



Step 6 After the standard page is designed, click  in the upper part of the page to save the standard page.

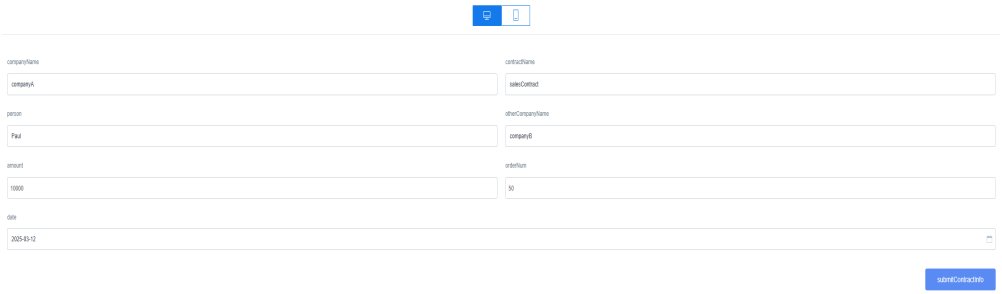
----End

Step 6: Verify the Function of Generating and Downloading the Contract

Step 1 In the upper part of the standard page, click  . The preview page is displayed.

Step 2 Enter the contract content and click the button to submit contract parameters and generate document instance.

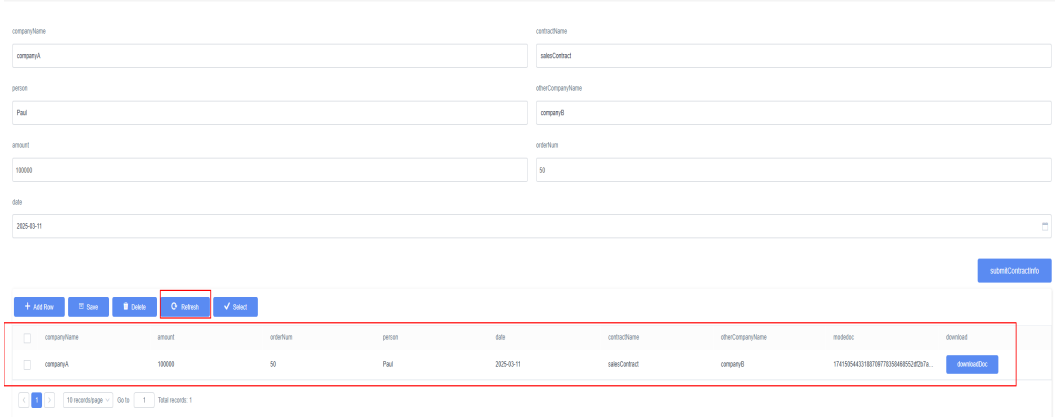
Figure 4-51 Entering the contract content



Step 3 After the submission success message appears, click the refresh button to view the submitted contract data.

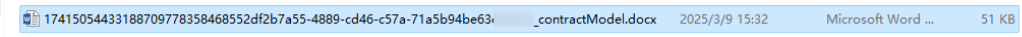
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-52 Viewing the submitted contract data



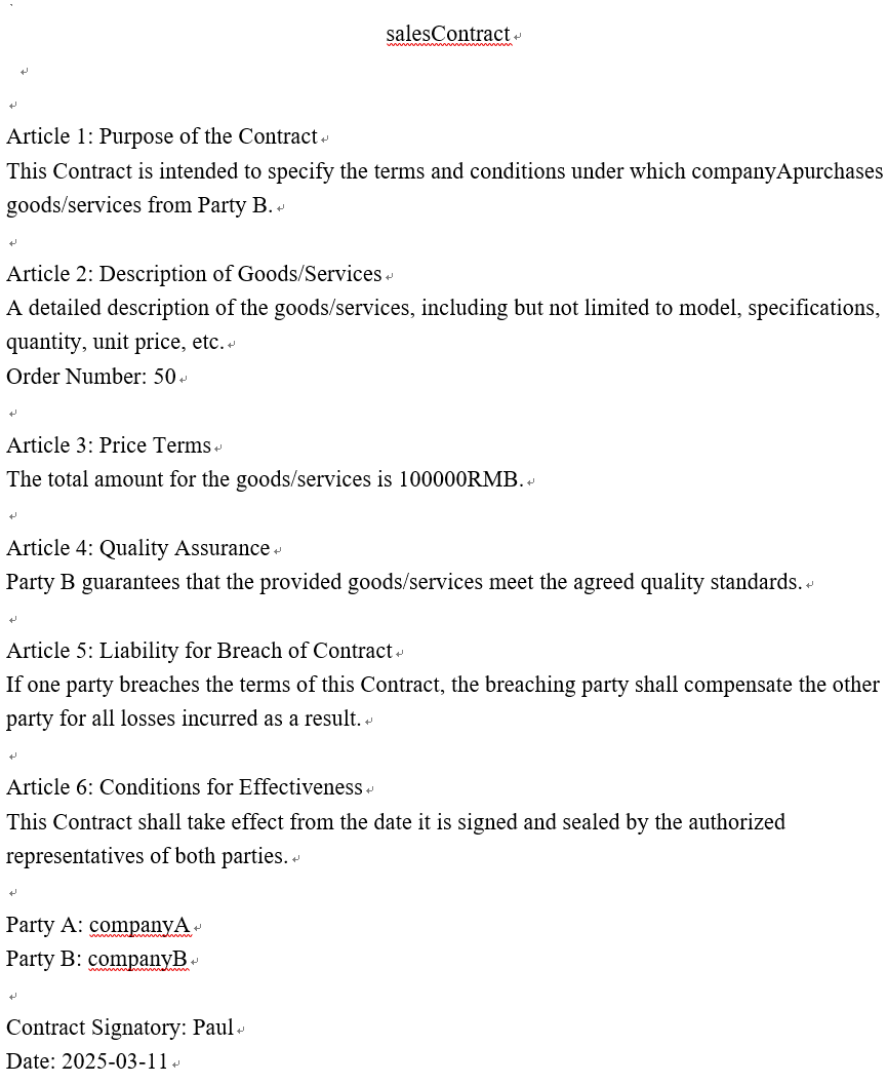
Step 4 Click the download button next to the data to save the contract to your local PC and view the document.

Figure 4-53 Contract downloaded



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-54 Viewing the contract



salesContract

Article 1: Purpose of the Contract
This Contract is intended to specify the terms and conditions under which companyApurchases goods/services from Party B.

Article 2: Description of Goods/Services
A detailed description of the goods/services, including but not limited to model, specifications, quantity, unit price, etc.
Order Number: 50

Article 3: Price Terms
The total amount for the goods/services is 100000RMB.

Article 4: Quality Assurance
Party B guarantees that the provided goods/services meet the agreed quality standards.

Article 5: Liability for Breach of Contract
If one party breaches the terms of this Contract, the breaching party shall compensate the other party for all losses incurred as a result.

Article 6: Conditions for Effectiveness
This Contract shall take effect from the date it is signed and sealed by the authorized representatives of both parties.

Party A: companyA
Party B: companyB

Contract Signatory: Paul
Date: 2025-03-11

----End

4.2 Using Email Templates to Send Email Notifications

Application Scenarios

Huawei Cloud Astro Zero enables quick creation of **Word**, **Excel**, **Email**, and **SMS** templates, enhancing document efficiency and standardization.

This practice uses an **Email** template as an example. Developers can set the email format and framework in advance to save time and effort. For example, in business communication (order confirmations, conference invitations) or daily life (holiday greetings, activity notifications), simply input key information like the recipient, subject, and content to quickly generate a complete email.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Advantages

In a flow, you can call an email template node and configure its input and output parameters to generate a complete email.

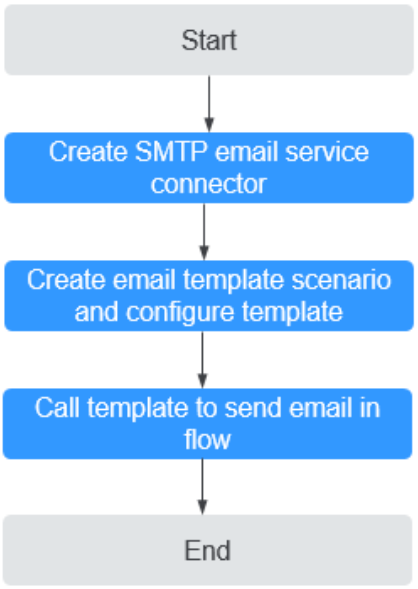
Constraints

You can set multiple email addresses for the recipient, CC, and BCC. Use semicolons (;) to separate email addresses. The recipient, CC, BCC, and subject fields cannot exceed 4,096 characters.

Procedure

Figure 4-55 shows the process of sending an email using an email template.

Figure 4-55 Sending an email



Step 1: Creating an SMTP Email Service Connector

Create an SMTP email service connector in the application to implement the email sending function. SMTP is a protocol that provides reliable and efficient email transmission.

Step 1 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.

4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

Figure 4-56 Creating a blank application

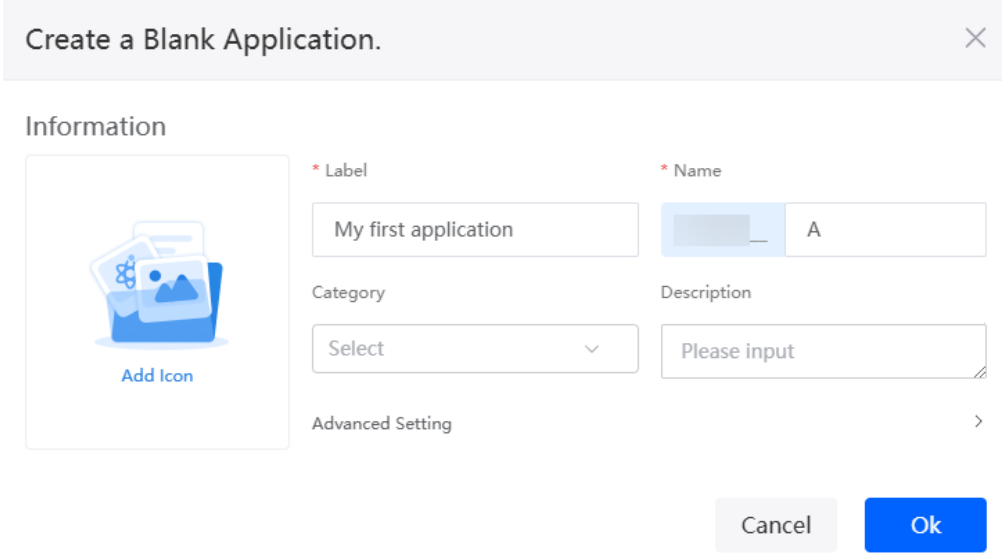


Table 4-13 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application

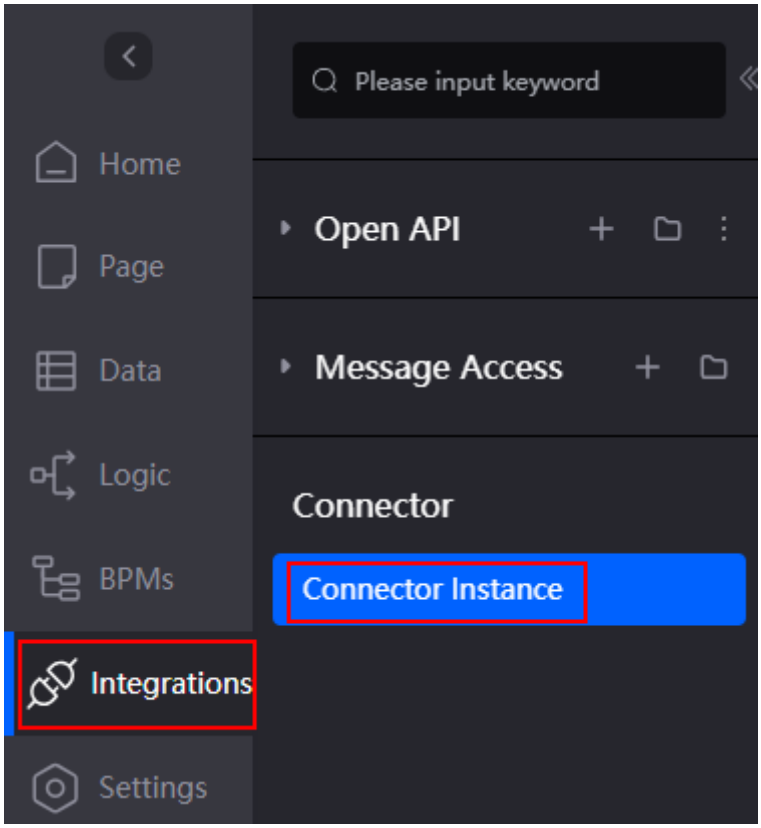
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 2 Create an SMTP email service connector.

1. In the navigation pane of the application designer, choose **Integrations** and choose **Connector** > **Connector Instance**.

Figure 4-57 Clicking Connector Instance



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2. Choose **Type** > **Message** > **Email by SMTP**. On the displayed page, click **+**. The page for creating an SMTP email service connector is displayed.
3. Set the information about the SMTP mailbox service connector and click **Save**.

Figure 4-58 Creating an SMTP email service connector

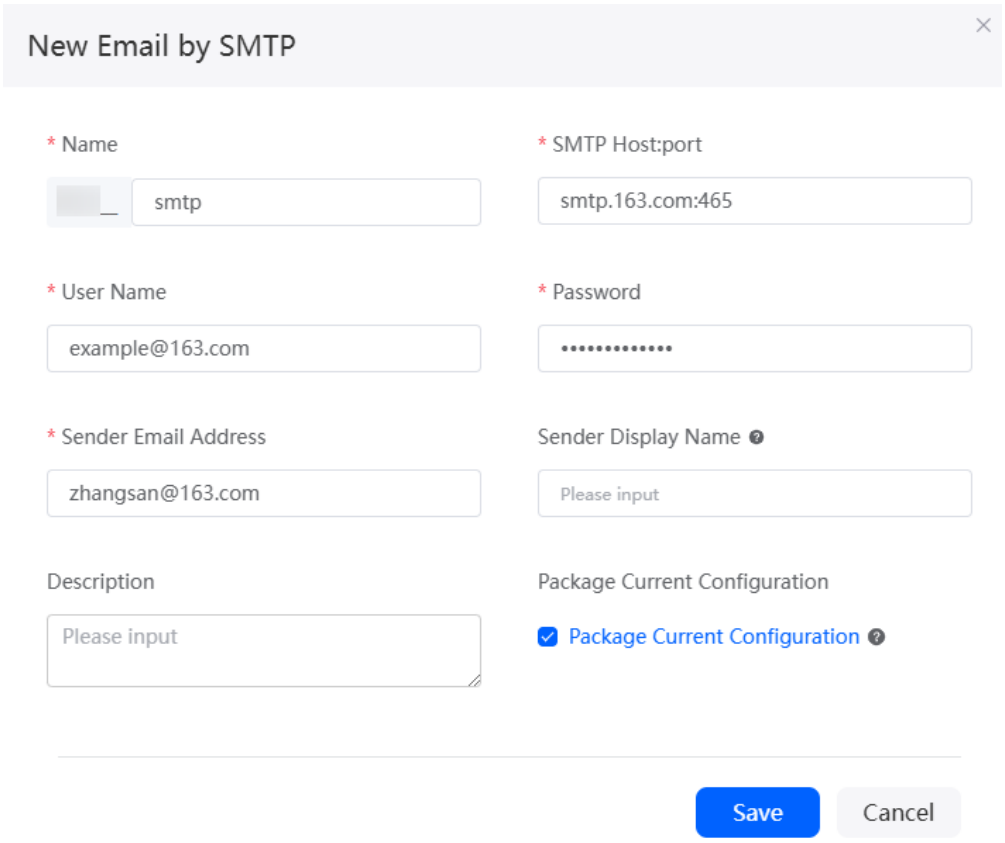


Table 4-14 describes only the parameters used in this practice. For details about other parameters, see **Interconnecting with SMTP to Send Emails**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-14 Parameter description

Parameter	Description	Example
Name	Name of the SMTP email server connector to be created. The name cannot be changed after the connector is created. The naming requirements are as follows: – Max. 64 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	smtp
SMTP Host:port	Server address and port number configured when the SMTP function is enabled for the mailbox. For details about the SMTP server address and port number after the SMTP function is enabled for a common mailbox, see **Table 4-15**.	smtp.163.com:465
User Name	The username of the mailbox. If the username has not been changed, the default value is the email address.	example@163.com
Password	A random character string generated when SMTP is enabled for the mailbox.	-
Sender Email Address	Email address for sending emails. The format is **username@domain name**.	zhangsan@163.com

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-15 Address and port number of the SMTP server for common mailboxes

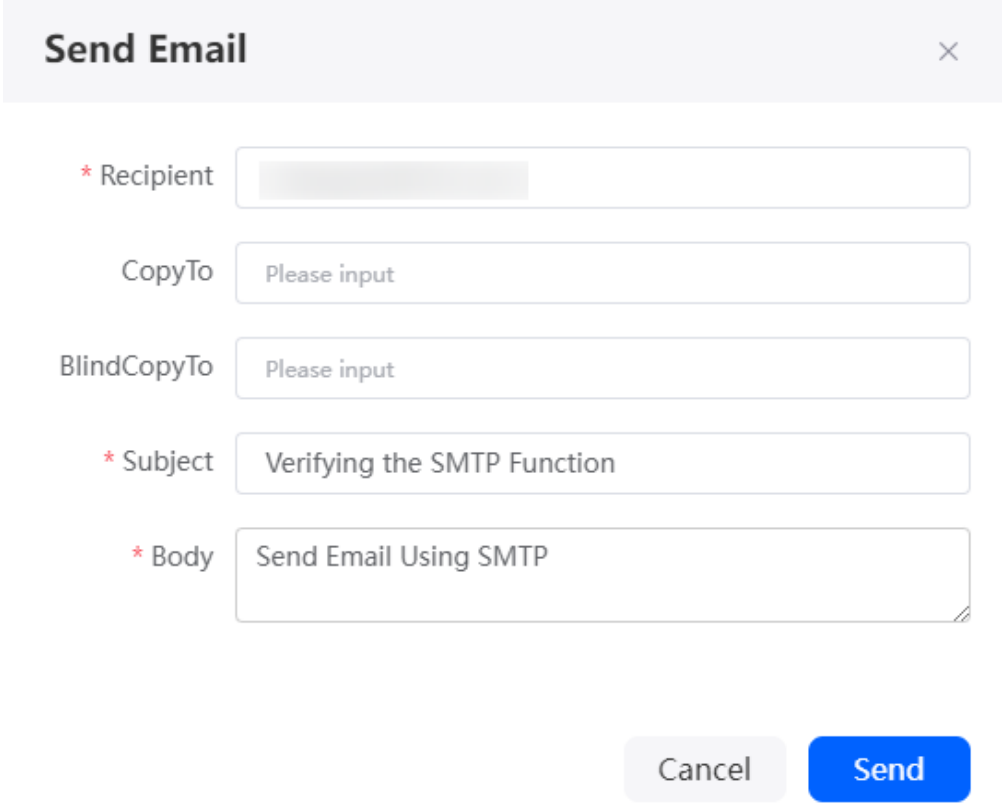
Mailbox	SMTP Server IP Address	Service Port
163 mailbox	smtp.163.com	– If SSL is used, set the port number to 465 or 994. – Set the parameter to 25 for other protocols.
126 mailbox	smtp.126.com	25
Yeah mailbox	smtp.yeah.net	25
QQ enterprise mailbox	– For users in China: smtp.exmail.qq.com – For users outside the Chinese mainland: hwsmtp.exmail.qq.com	The SSL protocol is used, and the port number is 465.
Sina mailbox	smtp.sina.com	25

Step 3 Check whether the email can be sent.

1. On the connector details page, click the send button.
2. Enter the recipient, subject, and email content, and click **Send**.

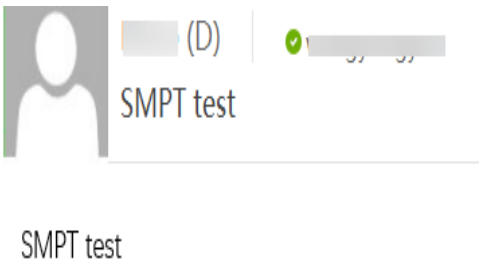
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-59 Sending an email



If the recipient receives an email, the test is successful, as shown in **Figure 4-60**.

Figure 4-60 Receiving an email



----End

Step 2: Creating an Email Template Scenario and Configure the Template

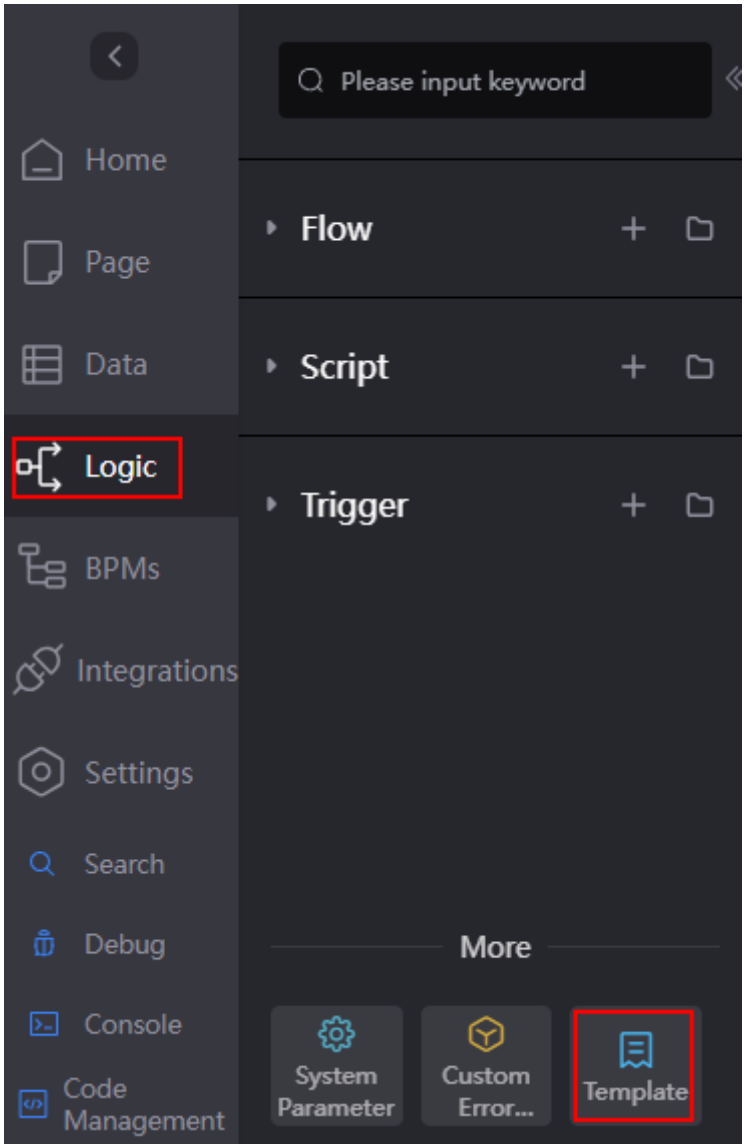
Create a template scenario and add an email template to the scenario. In the template scenario configuration, add the parameters to be replaced in the email template. A template scenario is a collection of service scenarios. Multiple templates can be created in a template scenario. The templates share the data structure.

Step 1 Create an email template scenario.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1. In the navigation pane of the application designer, choose **Logic** > **More** > **Template**.

   Figure 4-61 Clicking Template



2. Click **Add Template Scenario**, set the label and name of the template scenario, and click **Confirm**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-62 Creating a template scenario

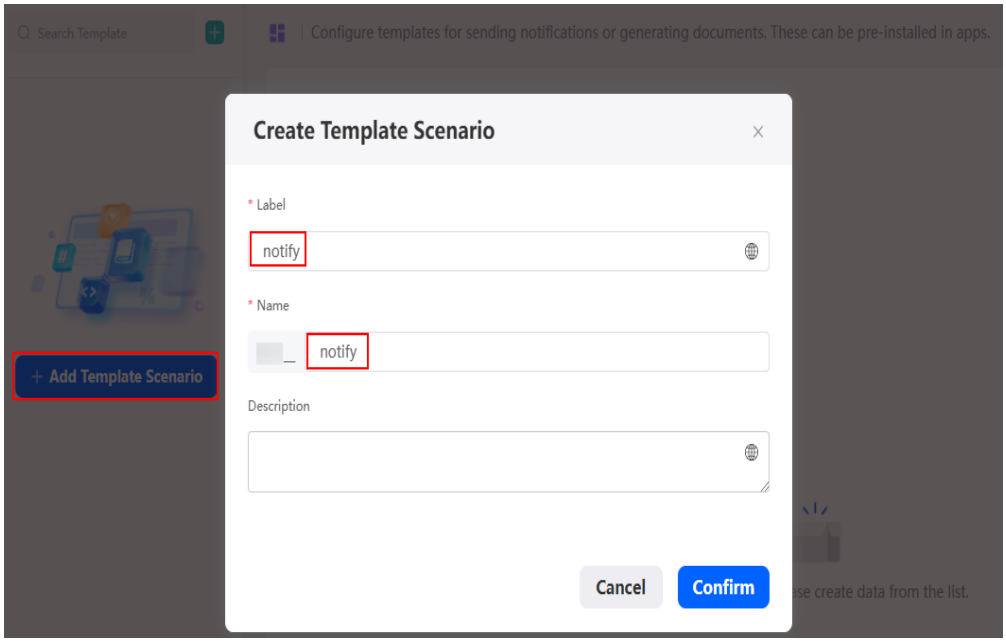


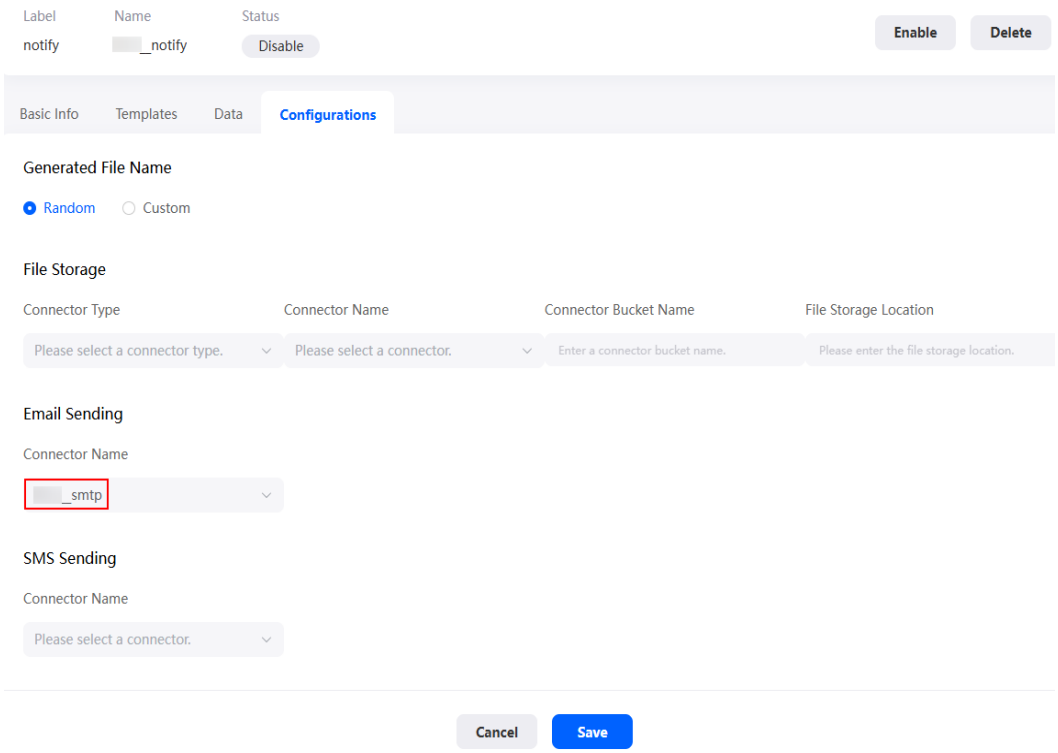
Table 4-16 Parameters for creating a template scenario

Parameter	Description	Example
Label	Name of the new template scenario, which can be changed after the template is created. Value: 1–80 characters.	notify
Name	ID of a new template scenario in the system. The ID cannot be modified after the template is created. The naming requirements are as follows: – Max. 64 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	notify

Step 2 On the **Configurations** tab page, select the connector created in **Step 1: Creating an SMTP Email Service Connector** and click **Save**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

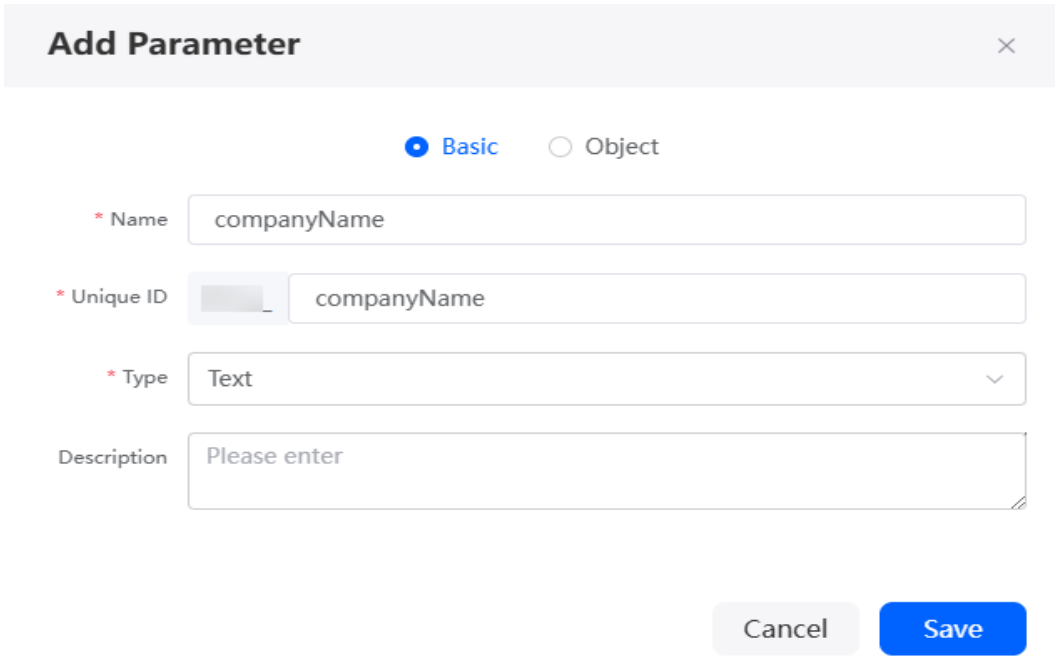
Figure 4-63 Selecting a connector



Step 3 On the **Data** tab page, add the parameters in **Table 4-17** to the notification template.

The template parameters added here correspond to the content to be replaced in the notification template.

Figure 4-64 Adding a company name parameter to a notification template

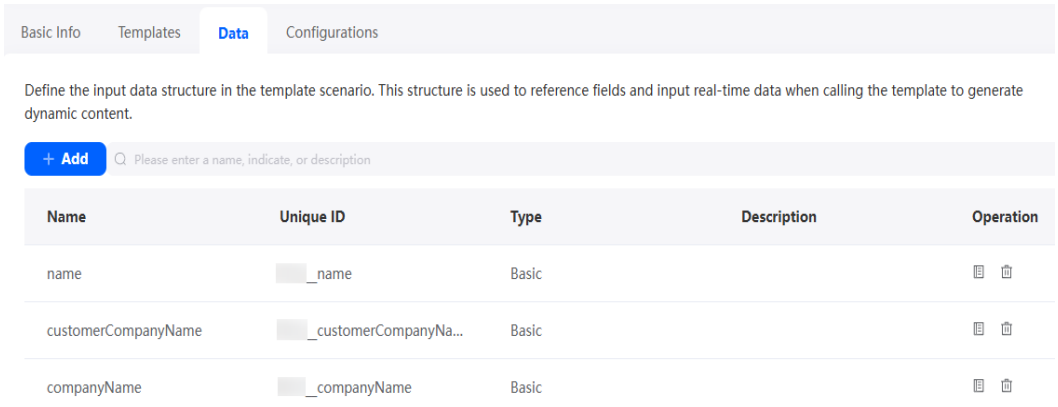


 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-17 Parameters to be added

Name	Unique ID	Type
Company name	companyName	Text
Customer company name	customerCompanyName	Text
Full name	name	Text

Figure 4-65 Viewing added parameters



Step 4 On the **Templates** tab page, click **Add** to create an email template.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-66 Creating an email template

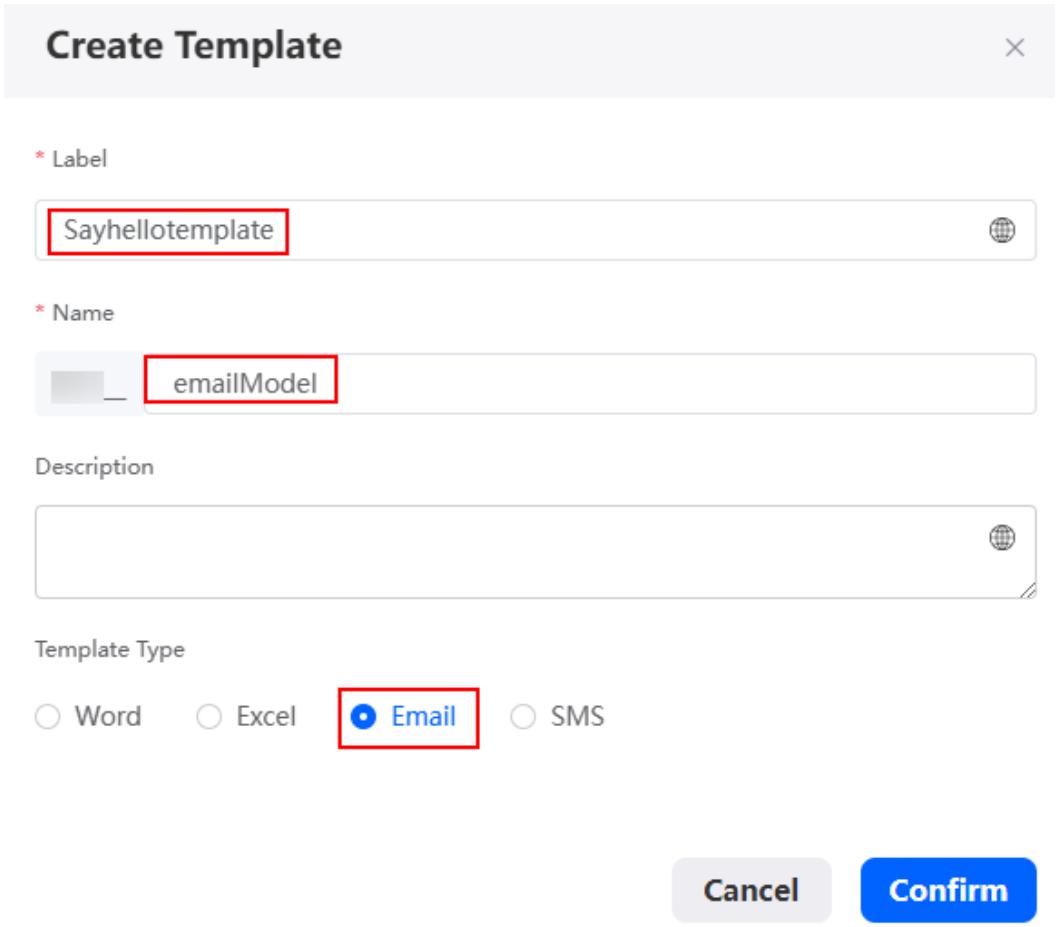


Table 4-18 Parameters for creating an email template

Parameter	Description	Example
Label	Name of the new template, which can be changed after the template is created. Value: 1–80 characters.	Sayhellotemplate

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

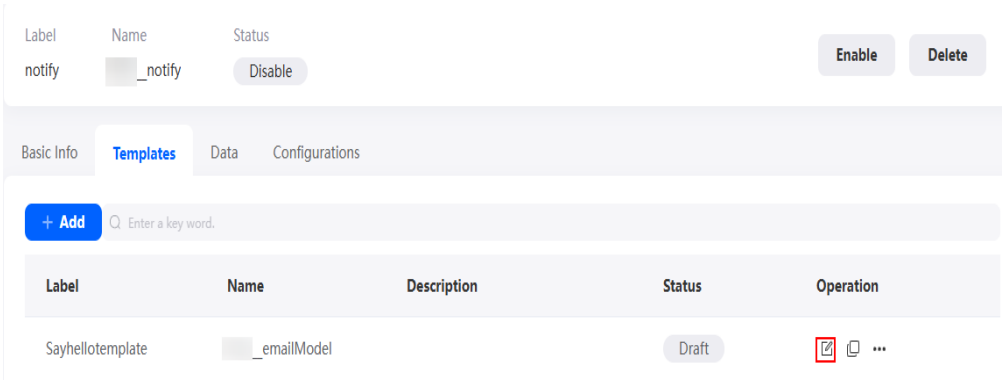
Parameter	Description	Example
Name	ID of a new template in the system. The ID cannot be modified after the template is created. The naming requirements are as follows: • Max. 63 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. • Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	emailModel
Template Type	Type of the template to be created. • **Word**: You can upload a Word template. The recommended template size is less than 10 MB. • **Excel**: You can upload an Excel template. The uploaded Excel file cannot contain more than 10 sheets. Each sheet cannot contain more than 200 columns, 1,000 rows, and maximum 500 characters in each cell. • **Email**: You can compile email templates on the template editing page. • **SMS**: You can compile SMS templates on the template editing page.	Mail

Step 5 Edit the email template.

1. On the **Templates** tab page, click  next to the template created in **Step 4**. The email template page is displayed.

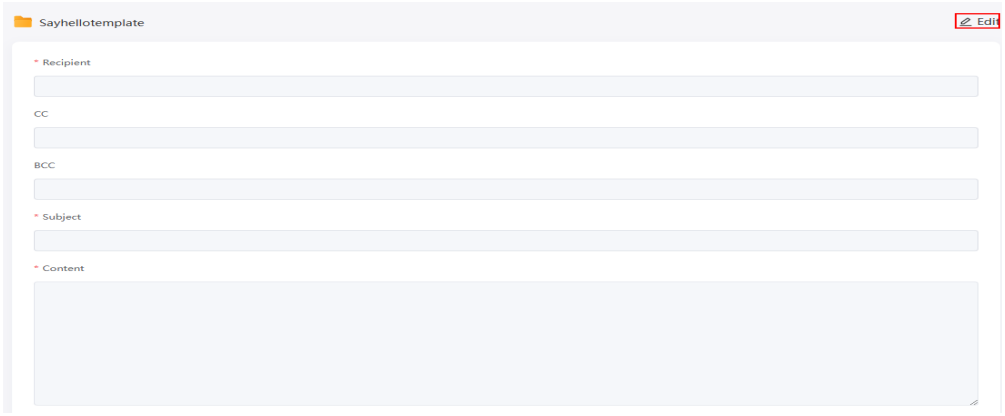
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-67 The template editing page



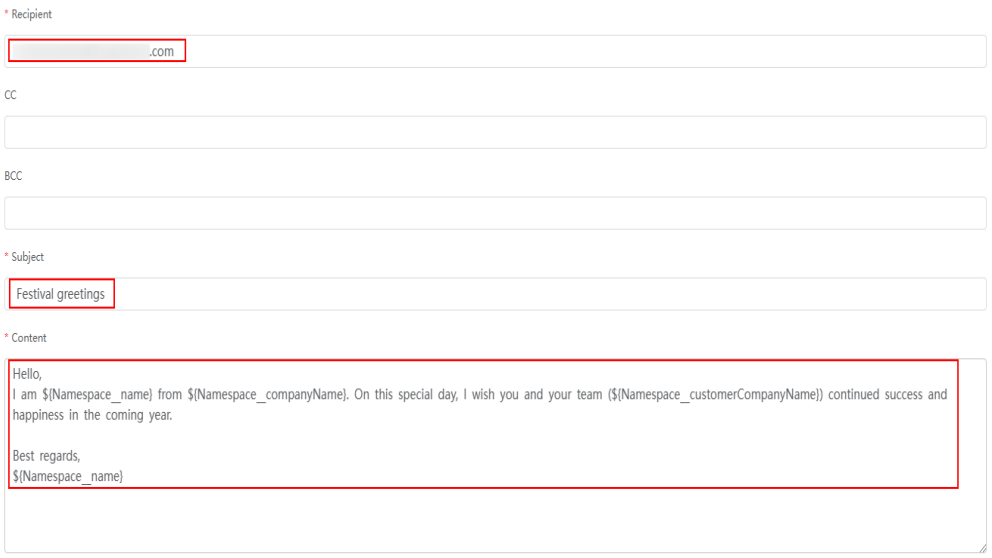
2. Click **Edit** in the upper right corner of the page.

Figure 4-68 Clicking Edit



3. Edit the email content. The variables in the email are the template parameters added in **Step 3**.

Figure 4-69 Editing the email content



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-19 Email content parameters

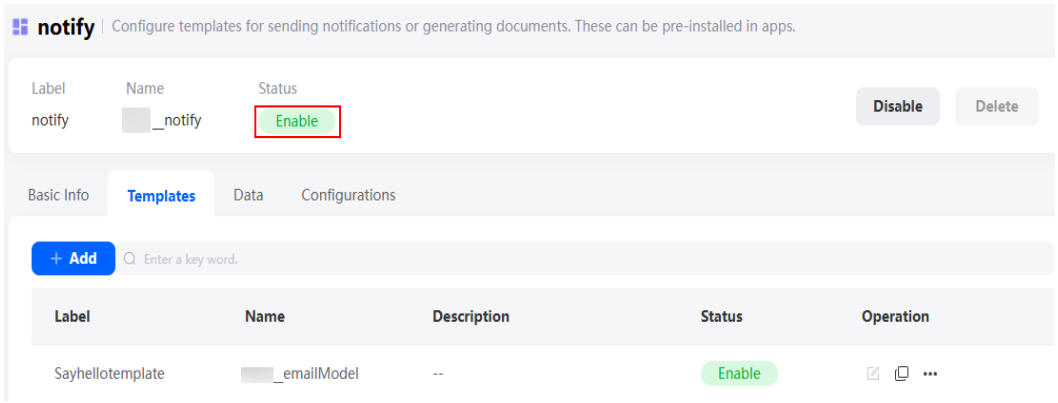
Parameter	Description	Example
Recipient	Recipient's email address (max. 4,096 characters). Use semicolons (;) to separate multiple addresses.	***.com
Subject	Email subject (max. 4,096 characters).	Festival greetings
Content	Email content (max. 4,096 characters). Keep email content concise, clear, and logical to ensure accurate information delivery.	Hello, I am *${Namespace__name}* from *${Namespace__companyName}*. On this special day, I wish you and your team (*${Namespace__customerCompanyName}*) continued success and happiness in the coming year. Best regards, ${Namespace__name}

4. After the content is edited, click the save button and then click **Enable** to enable the email template.

Step 6 Return to the email template scenario and click the enable button to enable the template scenario.

If the value of **Status** of the template scenario changes to **Enable**, the template scenario is enabled.

Figure 4-70 Viewing the template creation status



----End

Step 3: Calling the Template to Send Emails in a Flow

Create a flow and add the **Invoke Template** diagram element to send emails.

Step 1 In the navigation pane, choose **Logic**, and click **+** next to **Flow**.

Step 2 Specify the label and name and click **Add**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-71 Creating a flow

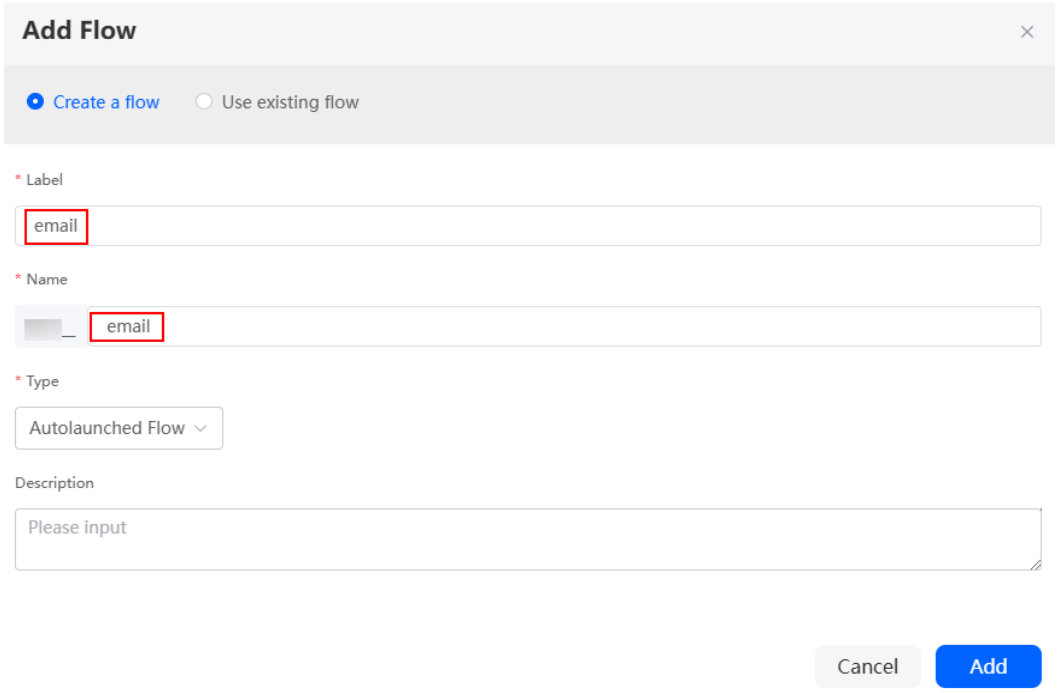


Table 4-20 Parameters for creating a flow

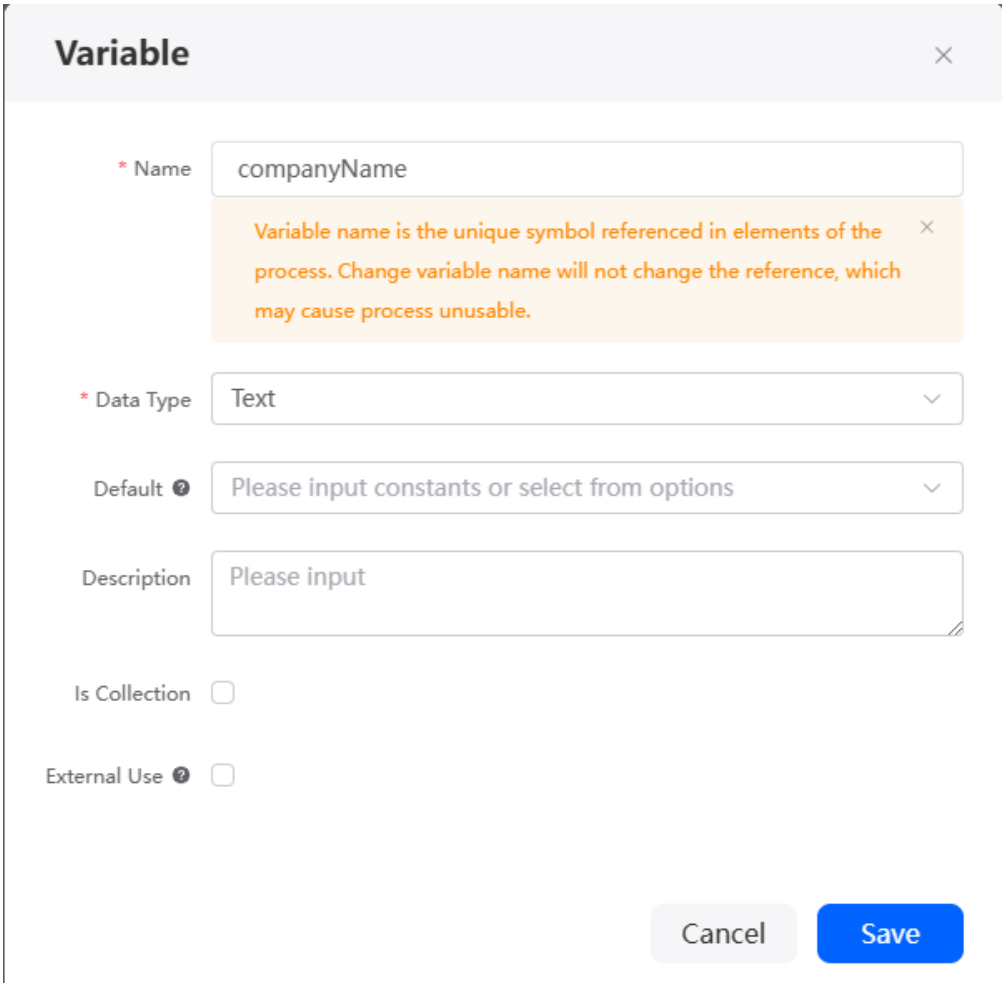
Parameter	Description	Example
Label	Flow label, which is displayed on the page and can be modified after being created. Value: 1–64 characters.	email
Name	Unique ID of a flow in the system, which cannot be modified after being created. The naming requirements are as follows: • Max. 64 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. • Start with a letter and can contain only letters, digits, and an underscore (_). It cannot end with an underscore (_).	email

Step 3 Create a global context variable.

1. On the flow design page, select the start node and click .

2. Click **Context** and click next **Variable** to create the variable **variable0**.
3. Click next to **variable0** and select **Set**.
4. Set **Name** to **companyName** and click **Save**.

Figure 4-72 Creating the companyName variable



5. Repeat the preceding operations to create variables in **Table 4-21**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-73 Viewing created variables

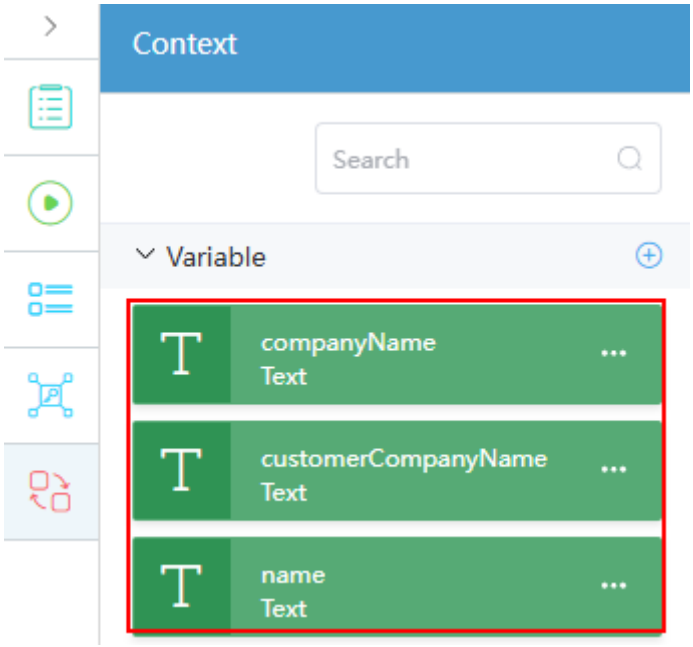


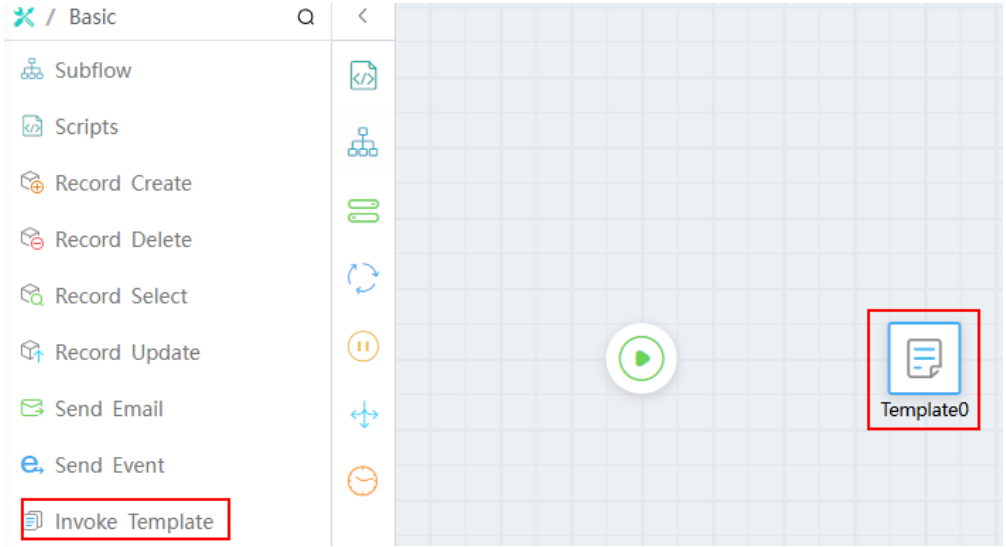
Table 4-21 Variables to be created

Name	Type
companyName (created)	Text
customerCompanyName	Text
name	Text

Step 4 Add the **Invoke Template** diagram element.

1. Under **Basic**, drag the **Invoke Template** diagram element to the end of the start diagram element.

Figure 4-74 Dragging the Invoke Template diagram element to the canvas



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2. Select **Template0**, click , and set the diagram element.

Figure 4-75 Setting the Invoke Template diagram element

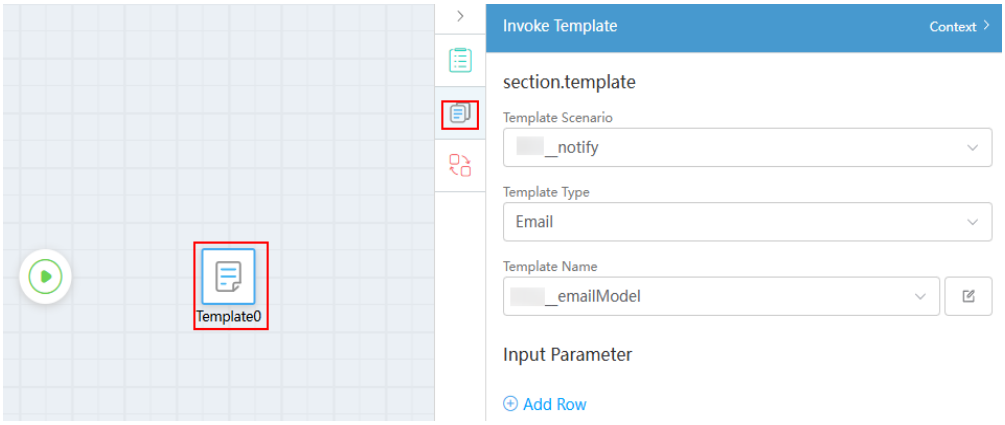


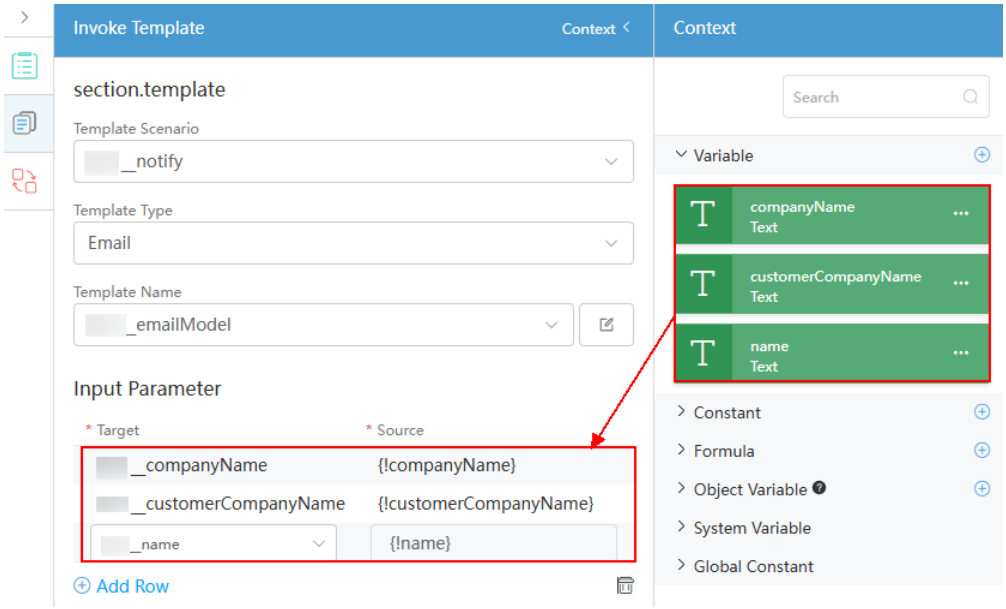
Table 4-22 Parameter description

Parameter	Description	Example
Template Scenario	Select the template scenario associated with the **Invoke Template** diagram element, that is, the template scenario created in **Step 1**.	*Namespace*__notify
Template Type	Select a type.	Email
Template Name	Select the email template created in **Step 4**.	*Namespace*__emailModel

3. In the **Input Parameter** area, click **Add Row** and drag the variables created in **Step 3** to the input parameters of the flow as the input parameters of the flow.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-76 Setting the input parameters of the flow



Step 5 Specify the logical relationship of the diagram elements: from **Start** to **Invoke Template**.

Figure 4-77 Specifying the logical relationship between diagram elements

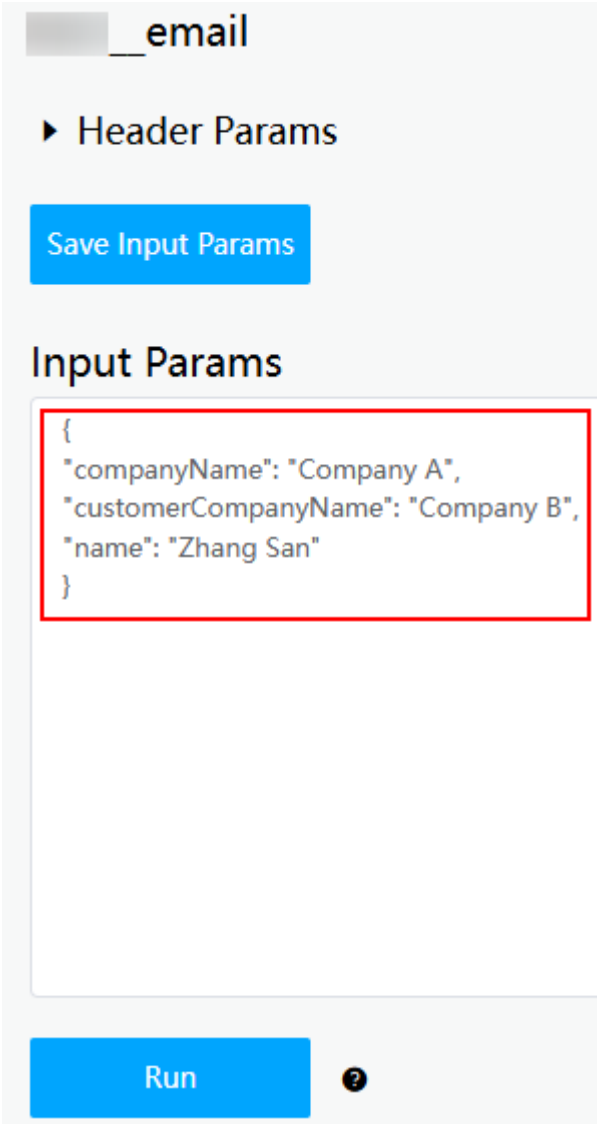
Step 6 Click to save the flow.

Step 7 Click . The flow debugging page is displayed.

Enter the following information in the input parameter text box and click **Run**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-78 Debugging the flow

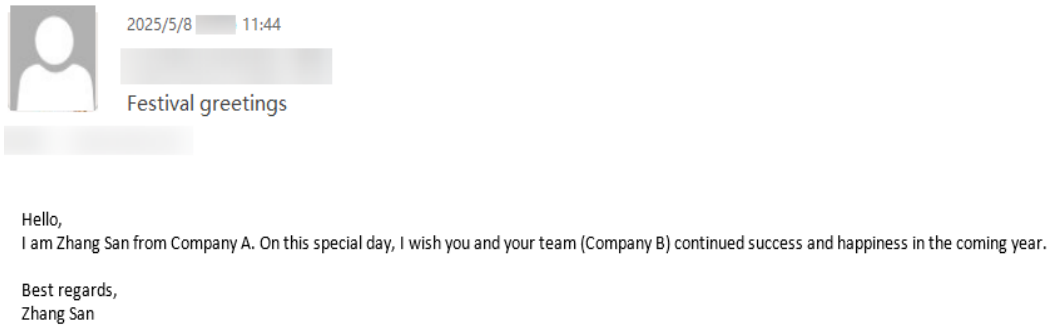



```
{
   "companyName": "Company A",
   "customerCompanyName": "Company B",
   "name": "Zhang San"
}
```

After the execution is successful, the email recipient set in **Step 5** can receive the greeting email, as shown in **Figure 4-79**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 4-79 Email



Step 8 Click to activate the flow.

----End

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

5 BPMs

5.1 Creating a Business Trip Approval Workflow Using Templates

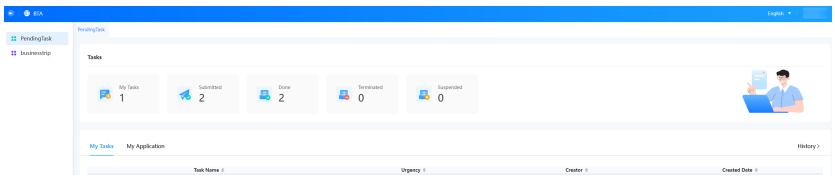
Overview

Huawei Cloud Astro Zero has developed its own Business Process Management (BPM), following the BPMN 2.0 industry standards. BPMs in Huawei Cloud Astro Zero is a graphical process orchestration engine designed for building service processes involving user interactions, such as approval and ticket dispatch processes. This practice walks you through creating a business trip approval workflow to familiarize you with Huawei Cloud Astro Zero's BPMs.

Business trips are a common business scenario in enterprises, involving processes like trip applications, approvals, itinerary planning, and expense reimbursement. This example uses a simple business trip scenario to illustrate how to use BPMs. Here, an employee submits a trip application for approval. The supervisor then either approves or rejects it. The business trip approval application includes the following functions:

- Create a business trip e-flow based on a template.
- Send an email.

Figure 5-1 Final effect of the business trip approval application

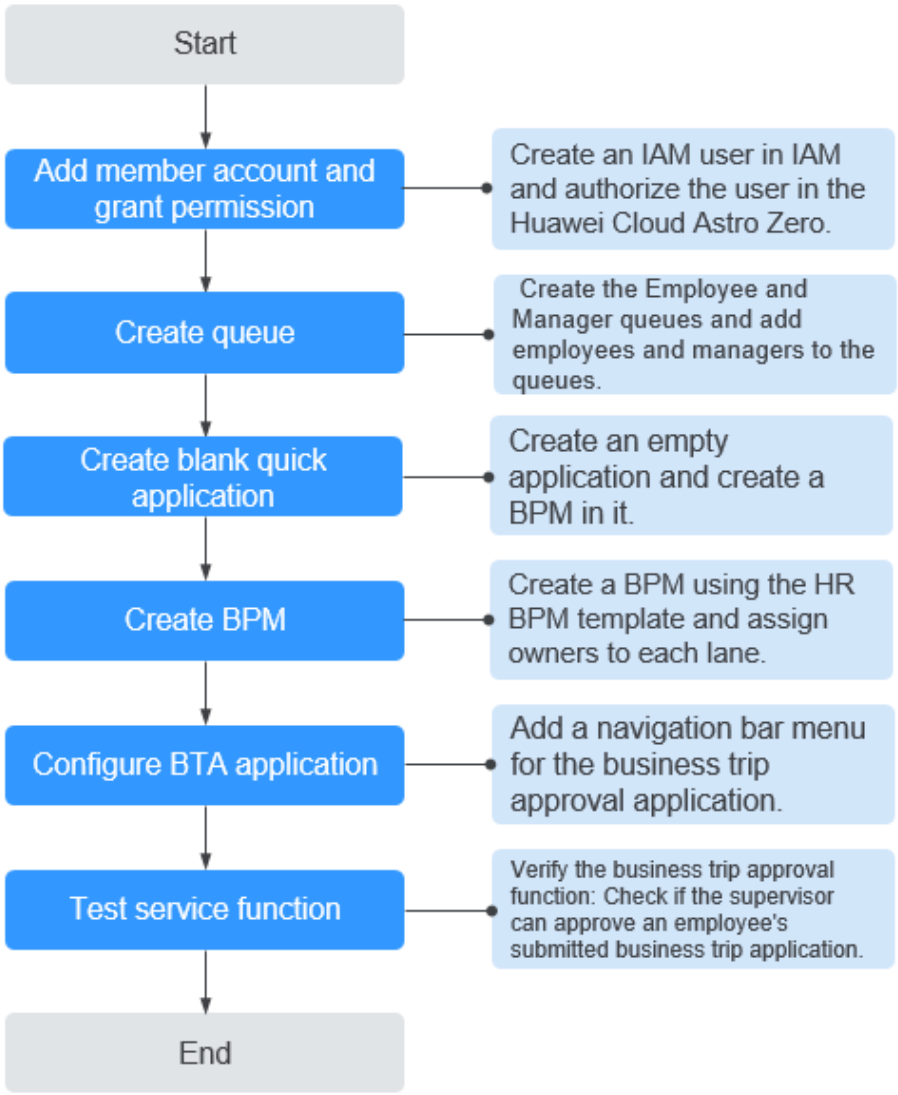


Procedure

Figure 5-2 shows the process of developing a business trip approval application.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-2 Business trip approval application development process



1. **Step 1: Add a Portal User and Grant Permissions**

   The application in this example involves portal users with supervisor and employee roles. Before creating the business trip approval application, add employees and supervisors to Huawei Cloud Astro Zero.

2. **Step 2: Create a Queue**

   In Huawei Cloud Astro Zero, a work queue records a set of members with the same permissions and task objects. In this example, two queues are created to separate tasks processed by different roles.

3. **Step 3: Create a Blank Application**

   Creating an application is the first step for developing a project in the development environment and is also the entry for building end-to-end software applications.

4. **Step 4: Create a BPM**

   Create a BPM using the HR BPM template and assign owners to each lane.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

5. **Step 5: Configure the BTA Application**

   In the application configuration, define the navigation menu bar of the business trip approval application.

6. **Step 6: Debug Service Functions**

   Check whether the business trip approval workflow is executed as expected. That is, the employee submits a business trip application, and the supervisor approves, rejects, and re-fills the application.

Step 1: Add a Portal User and Grant Permissions

The application in this example involves portal users with supervisor and employee roles. Before creating the business trip approval application, add employees and supervisors to Huawei Cloud Astro Zero.

Step 1 Create two member accounts (supervisor and employee) in IAM.

1. Log in to the **Huawei Cloud official website**. On the right of the top navigation menu, click **Console** to go to the management console.
2. Click  and select the region and project where the service instance is located.
3. Click  , search for the Identity and Access Management service in the search box, and click the search result to go to the IAM console.

   You can also choose **Management & Governance** > **Identity and Access Management** to go to the IAM console.
4. In the user area, click the button to create users for supervisor and an employee, respectively.

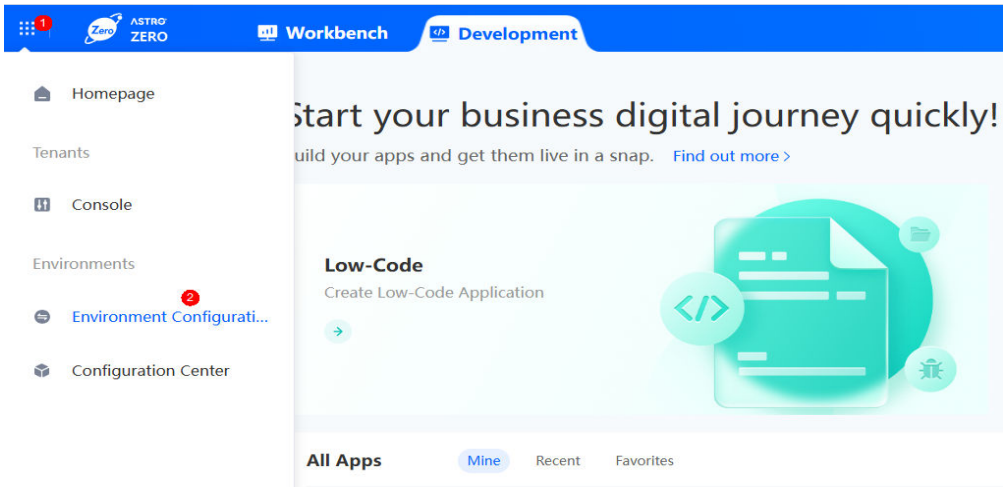
   For details about how to create an IAM user, see **Creating an IAM User**. In this example, the supervisor is Helen and the employee is Mike.

Step 2 Add a sub-account to Huawei Cloud Astro Zero and authorize the sub-account.

1. Log in to the using a Huawei account.
2. On the homepage, click **Access Homepage**.
3. In the upper left corner of the page, click  and choose **Environments** > **Environment Configuration**.

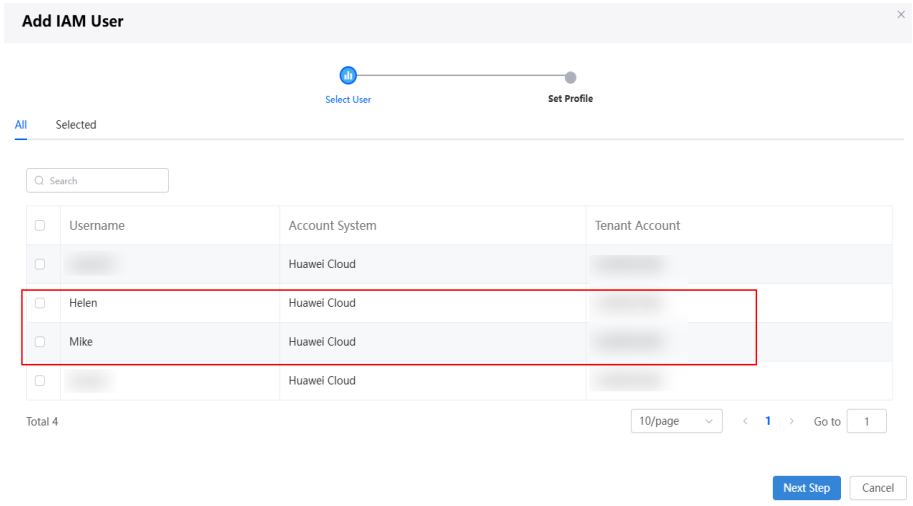
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-3 Entering the Huawei Cloud Astro Zero environment configuration page



4. In the navigation pane, choose **User Security** > **Users** and click **Add IAM User**.
5. In the user list, select the sub-account created in **Step 1** and click the next button.

Figure 5-4 Selecting the sub-account to be added

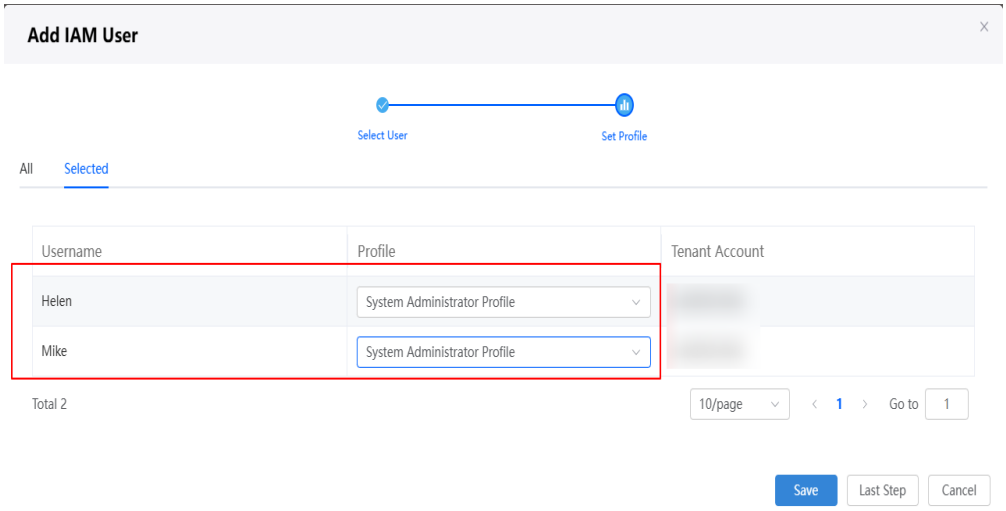


6. Assign the **System Administrator Profile** profile to the supervisor and employee, and click the save button.

   System Administrator Profile grants users full permissions. In the actual business trip approval application, you only need to assign the Portal User Profile or Anonymous User Profile profile to portal users. To verify the functions of the business trip approval application, assign the **System Administrator Profile** profile to portal users (employee Mike and supervisor Helen).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-5 Granting permissions to the sub-account



----End

Step 2: Create a Queue

In Huawei Cloud Astro Zero, a work queue records a set of members with the same permissions and task objects. In this example, the two queues in **Table 5-1** are used to separate tasks processed by different roles.

Table 5-1 Queues

Name	Description
Employee	Employee queue. The added queue member is a common employee, for example, Mike.
Manager	Supervisor queue (responsible for level-1 approval). The added queue member is a supervisor, for example, Helen.

Step 1 Log in to Huawei Cloud Astro Zero using a Huawei account and configure the environment.

Step 2 Choose **Maintenance** from the main menu.

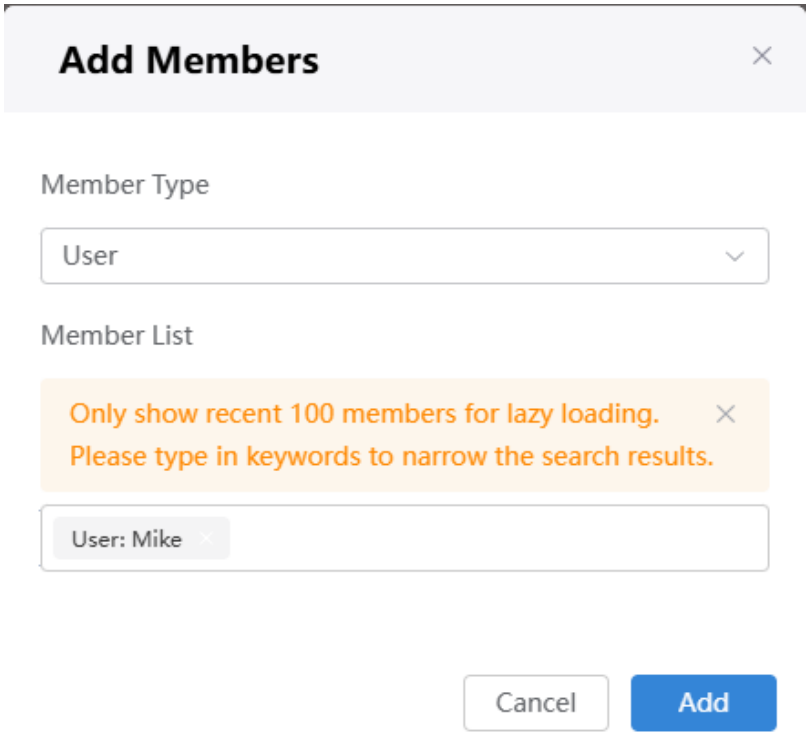
Step 3 In the navigation pane, choose **Global Elements** > **Queues**.

Step 4 On the displayed page, click **New** to create the Employee queue in **Table 5-1**.

1. In the basic information about the new queue, set the queue label and name, and retain the default values for other parameters.
   - Label: Set the label of the new queue to **Employee** in this example.
   - Name: Set the name of the new queue to **Employee** in this example.
2. In the queue member area, click **Add** to add Mike and the current tenant account as members of the **Employee** work queue.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-6 Adding Mike as a member

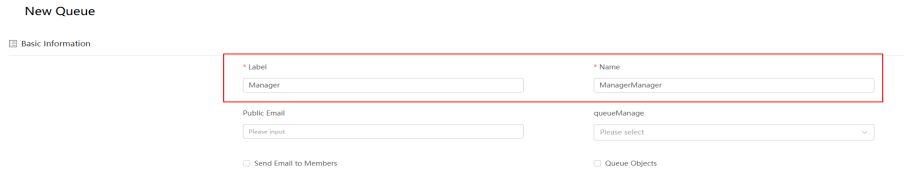


3. Click **Save**. The queue details page is displayed.

   On the **Employee** queue details page, you can view its information. If no member is added when you create a queue, click the add button in the member information area to add members.

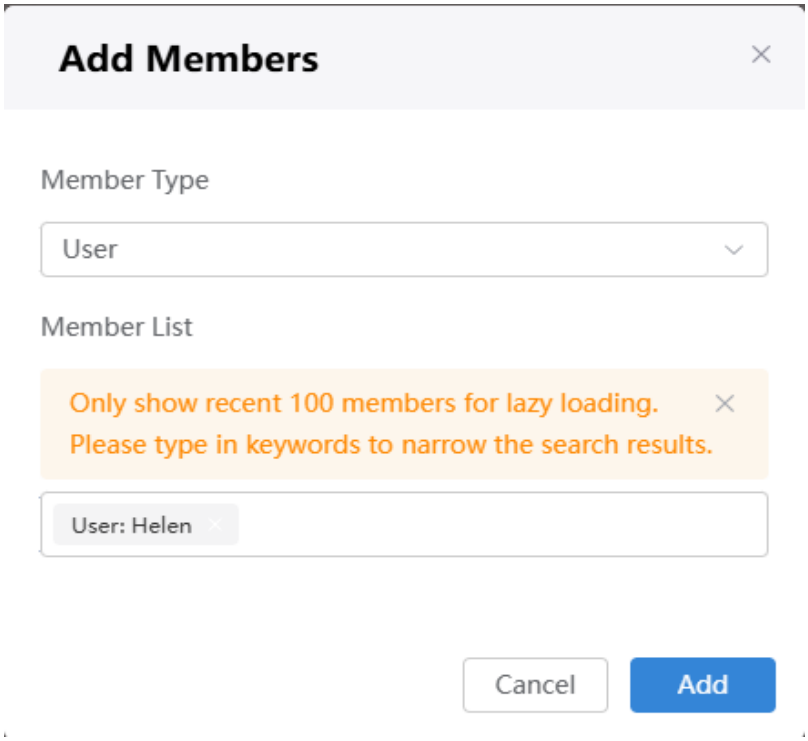
Step 5 Create a **Manager** queue by referring to **Step 4**.

Figure 5-7 Setting the label and name



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-8 Adding Helen and tenant accounts to the queue



----End

Step 3: Create a Blank Application

Creating an application is the first step for developing a project in the Huawei Cloud Astro Zero development environment and is also the entry for building end-to-end software applications.

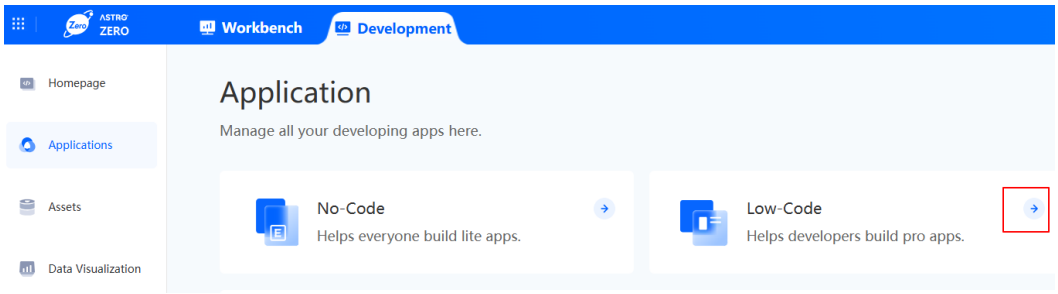
Step 1 Log in to the using a Huawei account.

Step 2 On the homepage, click **Access Homepage**. The application development page is displayed.

Step 3 In the navigation pane, choose **Applications**.

Step 4 Click → next to **Low-Code**. The page for creating a blank application is displayed.

Figure 5-9 Accessing the page for creating an application

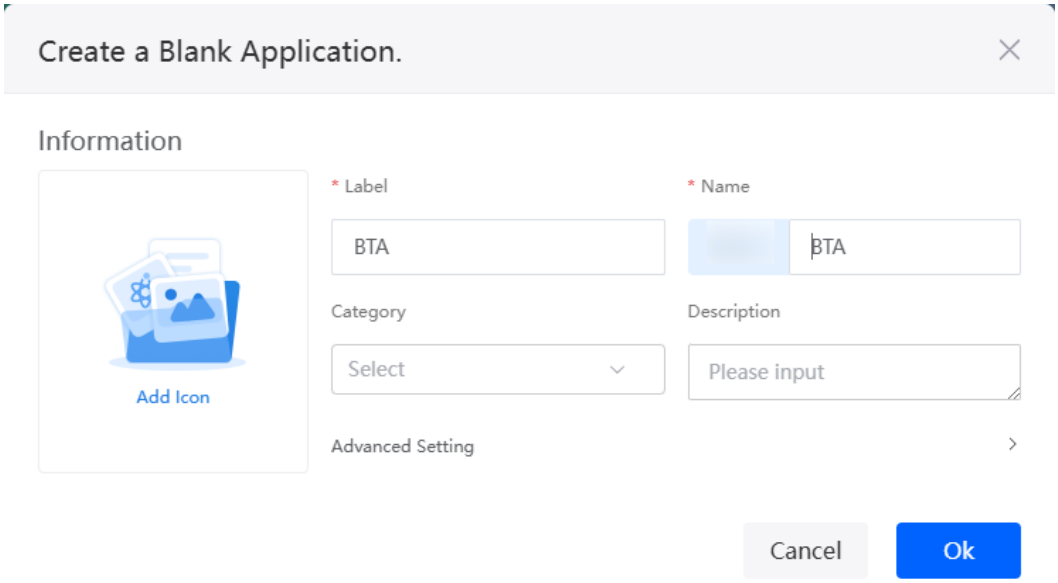


Step 5 On the page that is displayed, select **Standard Applications** and click **Confirm**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 6 Set the application label and name to **BTA**.

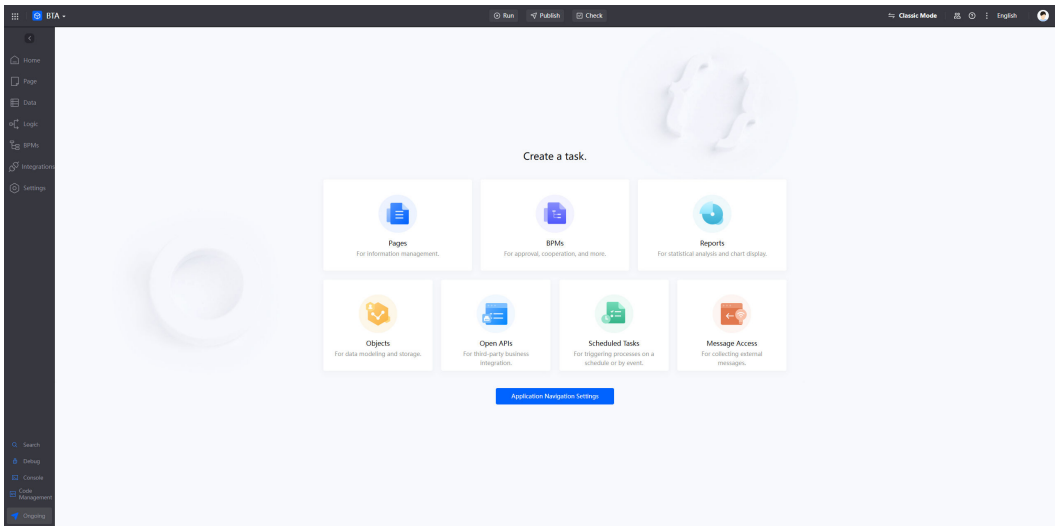
Figure 5-10 Setting the label and name



As shown in **Figure 5-10**, the content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified.

Step 7 Click the confirm button to access the application designer.

Figure 5-11 Application designer



----End

Step 4: Create a BPM

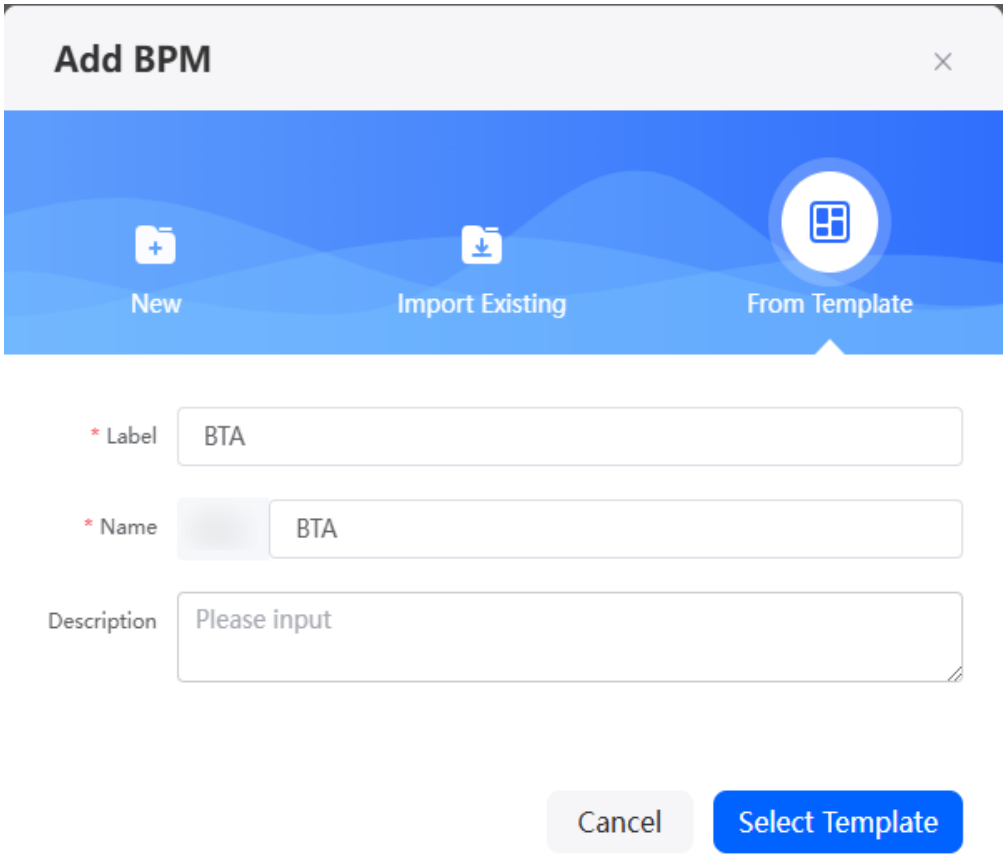
Create a BPM using the HR BPM template and assign owners to each lane.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 1 Create a BPM.

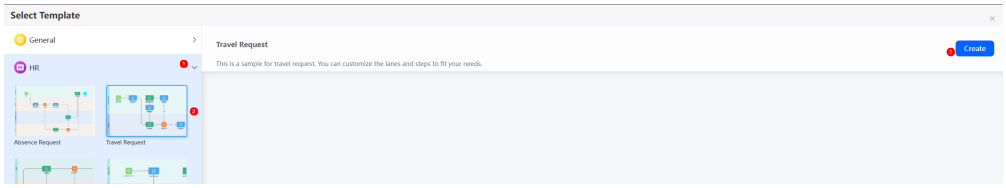
1. On the **Home** page of the application designer, click **BPMs**.
2. On the displayed dialog box, click **From Template**, and specify the label, name, and description.

Figure 5-12 Adding a BPM



3. Click **Select Template**, choose **HR** > **Travel Request**, and click **Create**.

Figure 5-13 Selecting a business trip application template



After you click **Create**, the page for editing the business trip application workflow is displayed. Before setting the workflow, you can understand the functions of each node in the business trip application workflow by referring to **Table 5-2**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-14 Development page

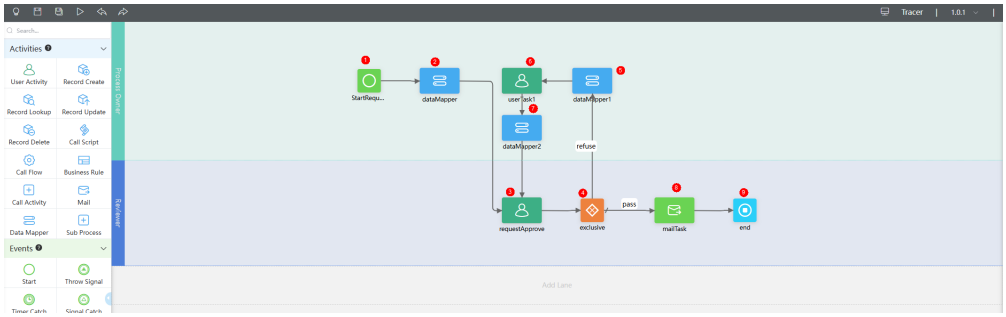


Table 5-2 Functions of each node in the business trip application workflow

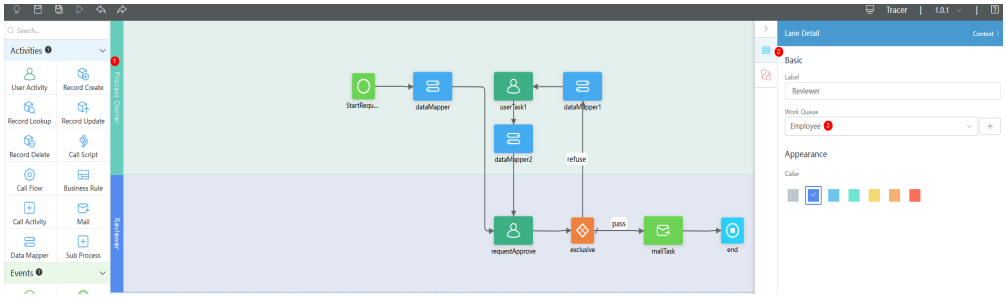
No.	Node	Description
1	Request submission	Start node, which calls a standard form for business trip applicants to submit business trip applications.
2	Data mapping	Maps the request fields in the standard business trip application form to the object.
3	Request approval	User task. Renders the fields in the object to the business trip application standard form and determine the approver.
4	Approval gateway	The approver approves or rejects the application.
5	Data mapping	After the application is rejected, the fields in the standard form are mapped back to the business trip application standard form.
6	Reapplication	Update the content in the business trip application form and submit the application again.
7	Data mapping	Map the field values in the business trip application form to the approval from and re-initiate the approval application.
8	Sending emails	If the application is approved, send an email to notify the applicant of the result.
9	End	End node, which is used to end the entire workflow.

Step 2 Set the workflow and configure the handler of each lane.

1. On the development page, click the **Process Owner** lane and set the queue to **Employee**.

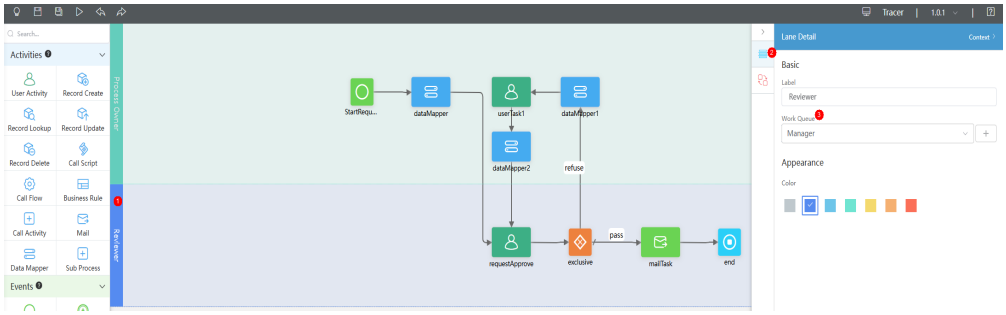
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-15 Configuring an employee as the applicant



2. Click the **Reviewer** lane and set the queue to **Manager**.

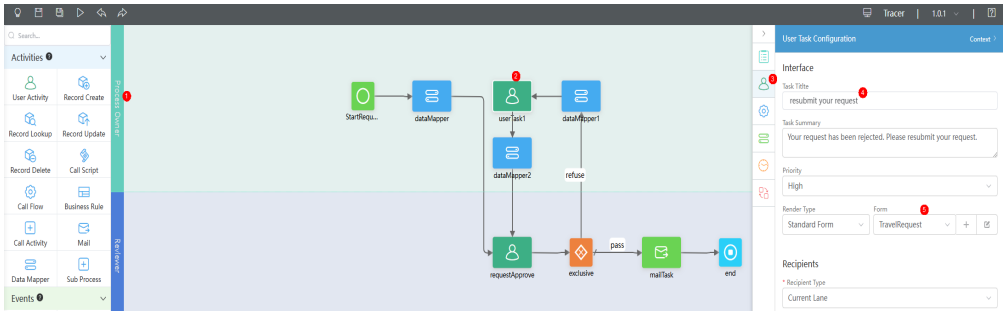
Figure 5-16 Configuring the manager as the approver



3. Click the **Refill the request** user task element on the **Process Owner** lane and set the task title to "To be resubmitted".

   After this operation is performed, you can view the status of the task to be resubmitted in **Pending Task**.

Figure 5-17 Setting the title to "To be resubmitted"

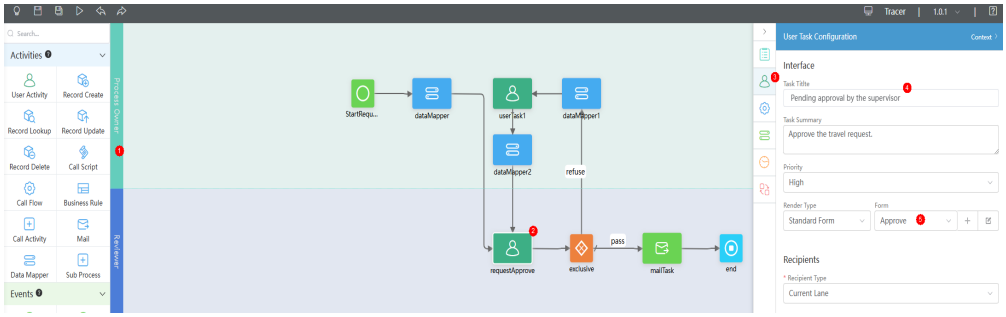


4. Click the user task element "Apply for approval" on the **Reviewer** lane and set the task title to "Waiting for supervisor approval".

   After this operation is performed, you can view the status of the task to be approved by the supervisor in **Pending Task**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-18 Setting the title to "Waiting for supervisor approval"



5. Click  to save the BPM.
6. Click  to activate the BPM.

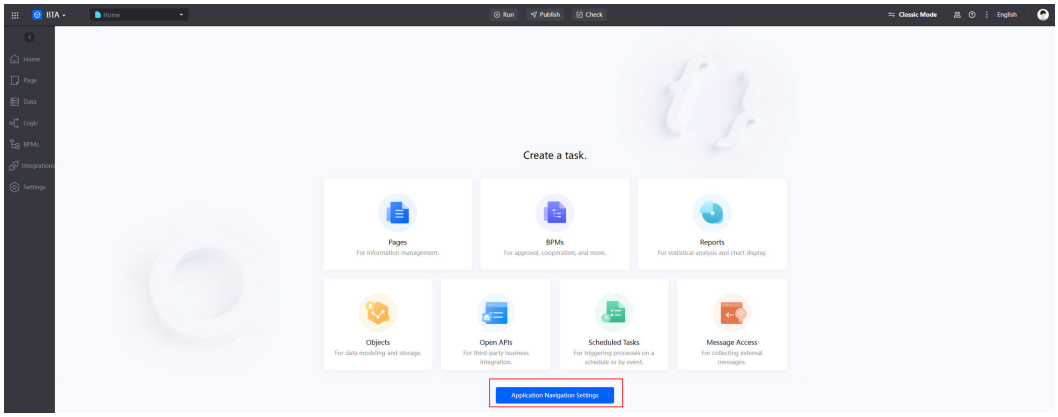
----End

Step 5: Configure the BTA Application

In the application configuration, define the navigation menu bar of the business trip approval application.

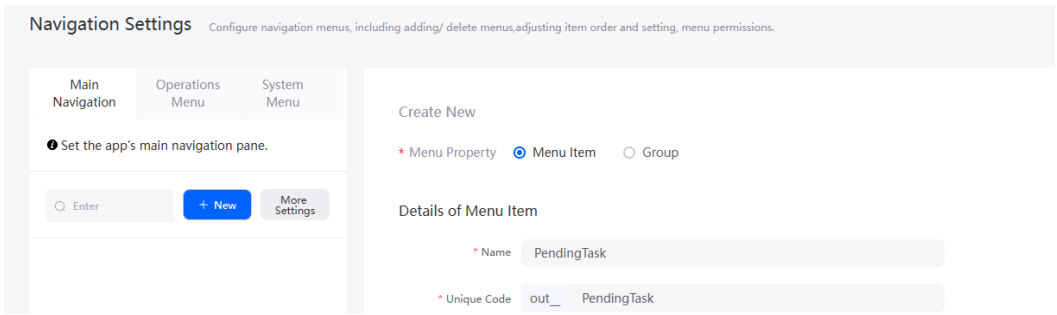
Step 1 On the **Home** page of the new BTA application designer, click **Application Navigation Settings** at the bottom.

Figure 5-19 Accessing the application configuration page



Step 2 On the **Main Navigation** tab page, click **Home**, set the menu name to **PendingTask**, and click **Save**.

Figure 5-20 Editing a tab



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 3 On the **Main Navigation** tab page, click **New**, create a menu item, for example, I want to go on a business trip menu item, and click **Save**.

Figure 5-21 Adding a tab

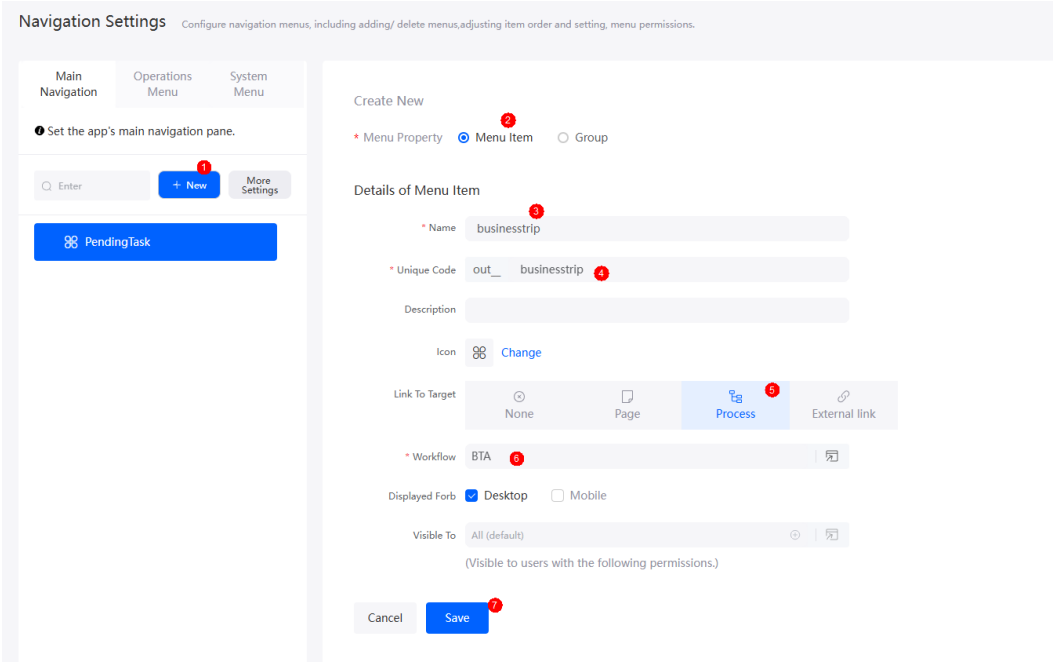
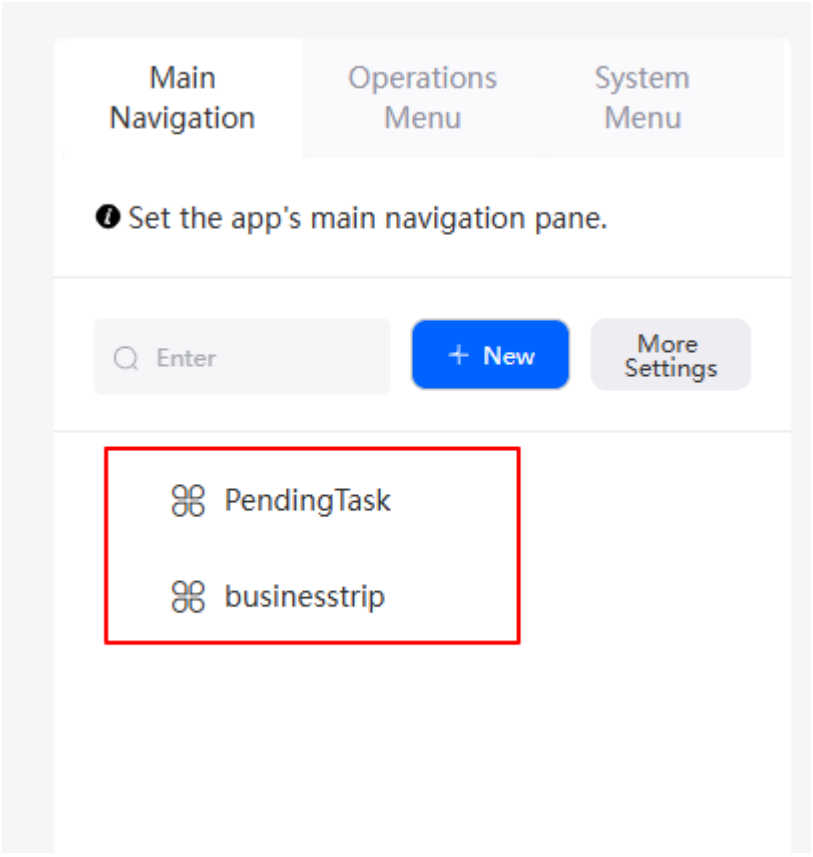


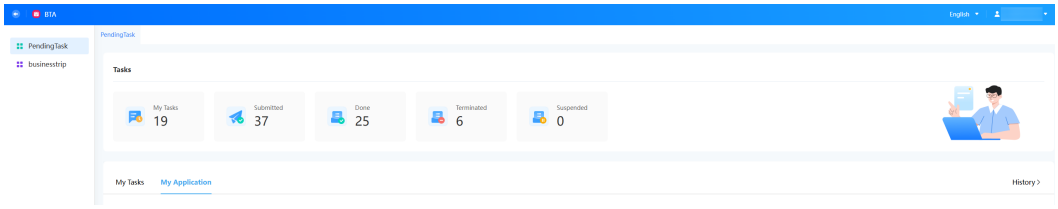
Figure 5-22 Final effect



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 4 On the main menu of the new BTA application designer, click **Run** > **Run Now** to preview the business trip approval application.

Figure 5-23 Final effect of the business trip approval application



The business trip approval application is developed.

Question: How do I display the navigation bar menu at the top of the business trip approval application?

On the **Appearance Settings** tab of the application configuration page, you can adjust the menu style and application icon to display the navigation bar menu at the top.

Figure 5-24 Modifying the menu style and application icon

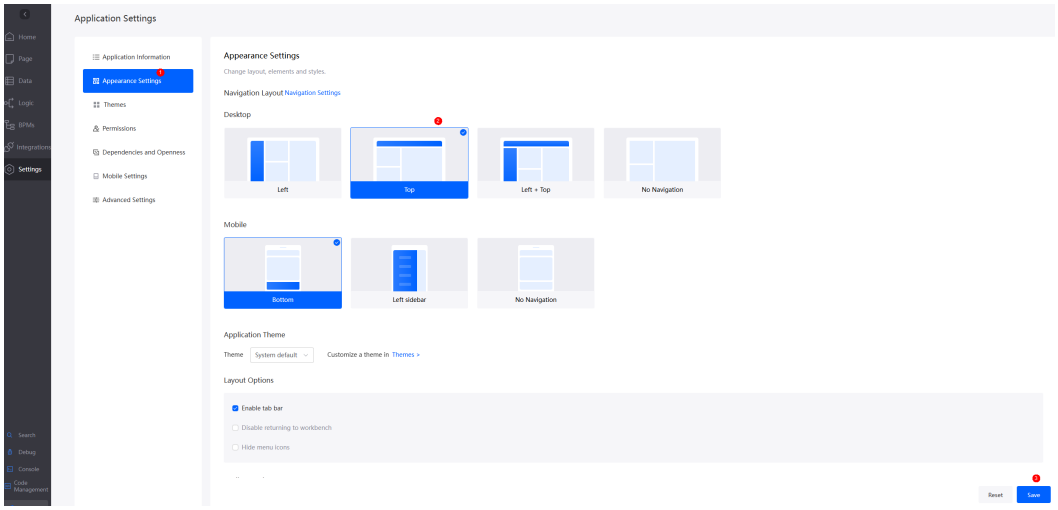
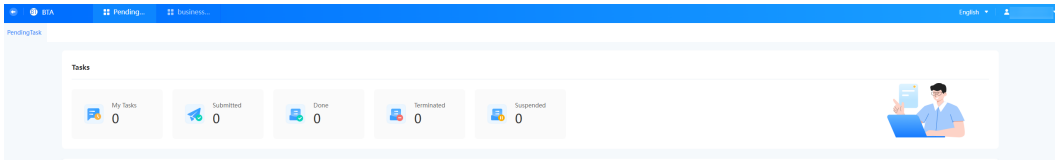


Figure 5-25 Effect after setting



----End

Step 6: Debug Service Functions

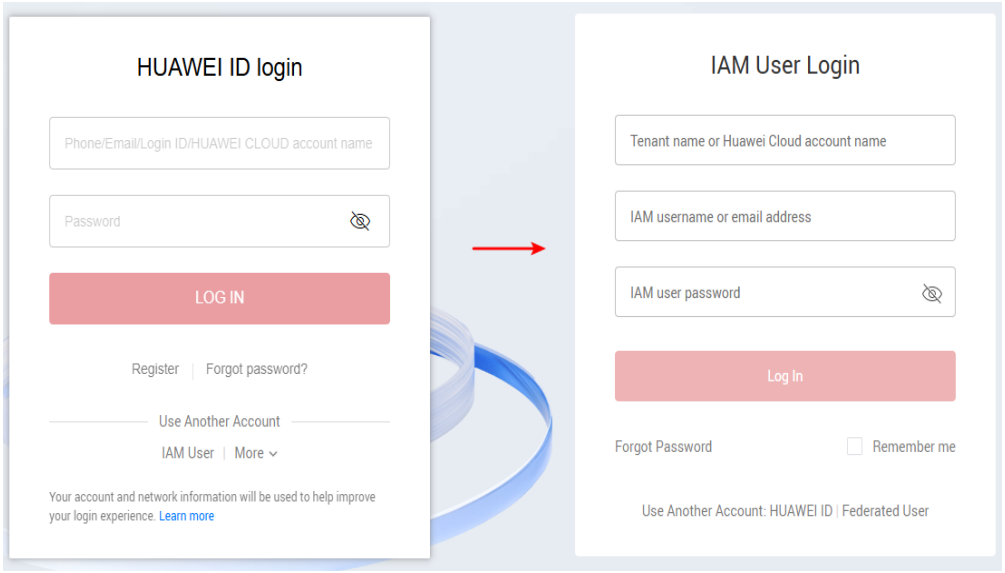
Business trip approval test process: An employee submits a business trip application. The supervisor approves the application, rejects the application, and the employee re-fills in the application.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 1 Fill out a business trip application as an employee.

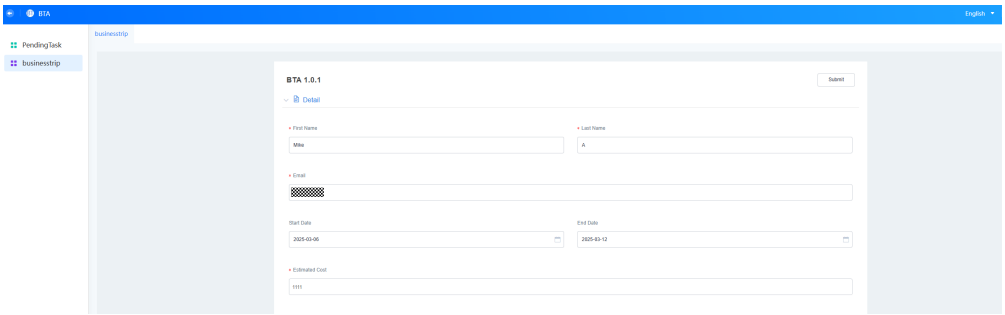
1. Log in to the **Huawei Cloud Astro Zero console** as an employee (Mike in this example).

   Figure 5-26 Logging in to Huawei Cloud Astro Zero



2. On the homepage, click **Access Homepage**.
3. In the navigation pane, choose **Applications**.
4. In the low-code application list, click **Edit** next to the created BTA application to access the BTA application designer.
5. On the main menu, choose **Run** > **Run Now**. The application preview page is displayed.
6. Click **businesstrip**, enter travel information, and click **Submit**.

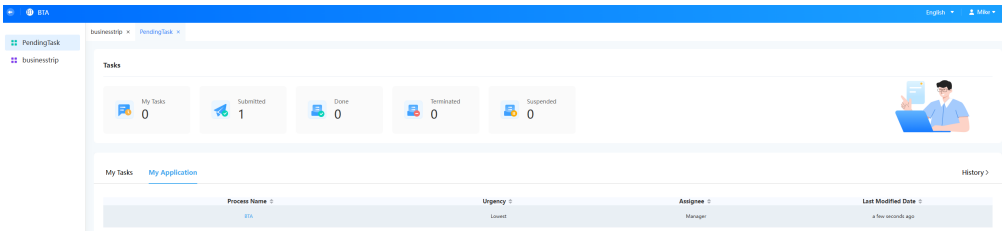
   Figure 5-27 Filling out a business trip application as an employee



7. Choose **Pending Task** > **My Application**. You can view the submitted business trip application.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

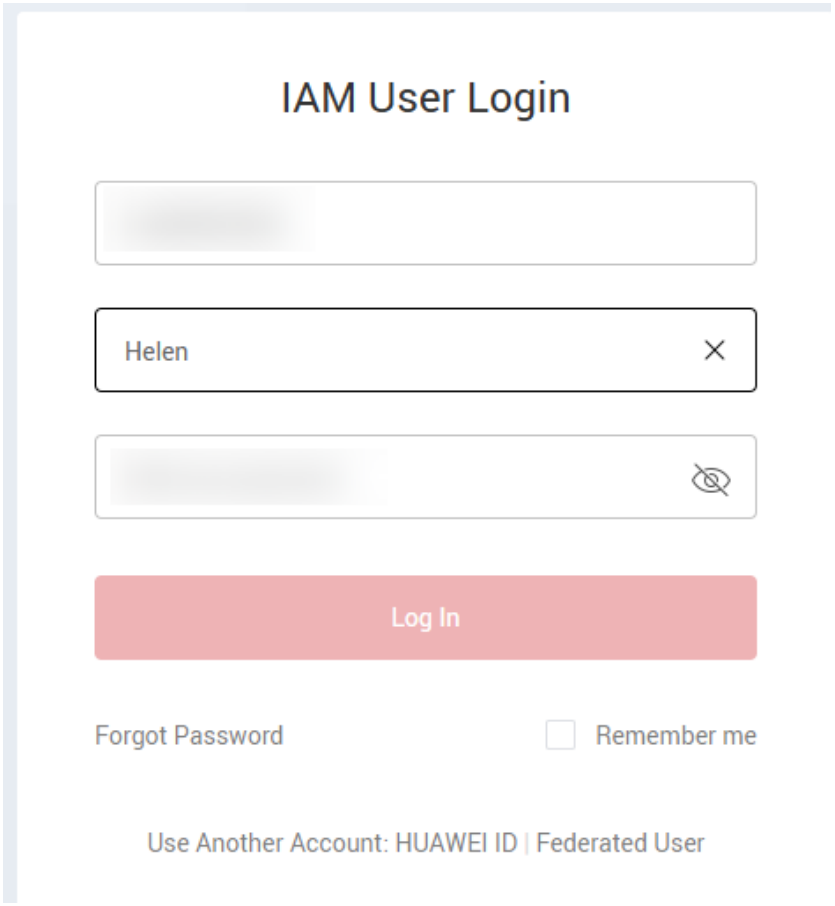
Figure 5-28 Viewing my applications



Step 2 Approve the business trip application as a supervisor.

1. Log in to the **Huawei Cloud Astro Zero console** as a supervisor (Helen in this example).

Figure 5-29 Logging in to Huawei Cloud Astro Zero



2. On the homepage, click **Access Homepage**.
3. In the navigation pane, choose **Applications**.
4. In the low-code application list, click **Edit** next to the created BTA application to access the BTA application designer.
5. On the main menu, choose **Run** > **Run Now**. The application preview page is displayed.
6. Click **Pending Task** and click **Pending approval by the supervisor**. The supervisor approval page is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-30 Pending Task

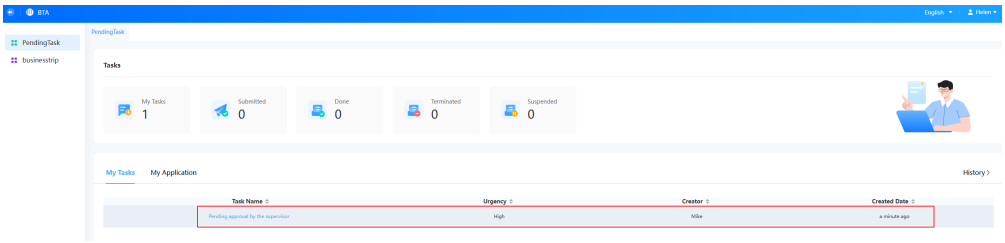
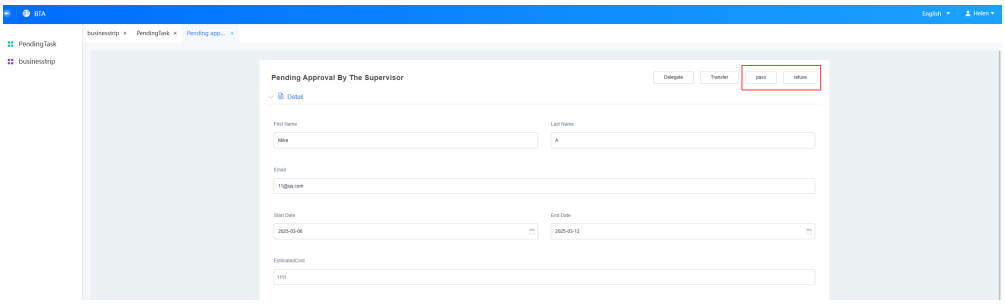


Figure 5-31 Supervisor approval page



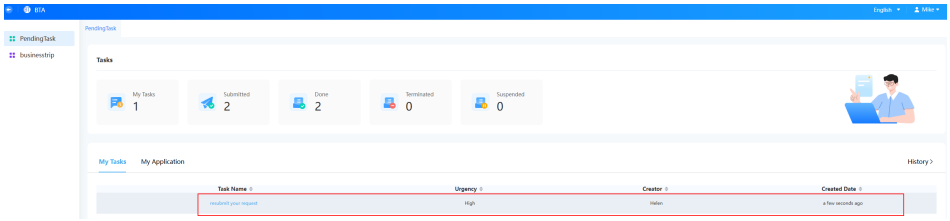
7. Approve or reject the application.
   - If the supervisor approves the application, an approval email is sent to the employee's email address, as shown in **Figure 5-32**. The email address is specified in **Figure 5-27**.

     Figure 5-32 Approval email

   - If the supervisor rejects the application, the employee can view the application record (such as **Figure 5-33**) in **My Tasks**.

     Figure 5-33 Viewing rejected records

You have completed the development and functional testing of the business trip approval application. Through the operations in this section, you are now familiar with the quick applications and BPMs.

----End

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

5.2 Implementing the Priority Approval Function with User Activities in BPMs

Application Scenarios

During enterprise operations, companies often face tasks that require quick processing or special attention. For example, urgent projects need swift approval to start promptly, and important customer requests demand immediate responses. In special cases, senior management may require priority approval rights.

To address these needs, you can set up priority approval for user activities in BPMs. A user activity is a task completed by users within a service process. When the engine processes a task at this node, it creates a task item assigned to a specific user or group.

This practice explains how to configure approval and termination actions for a user task in a BPM to enable priority approval during countersigning.

Advantages

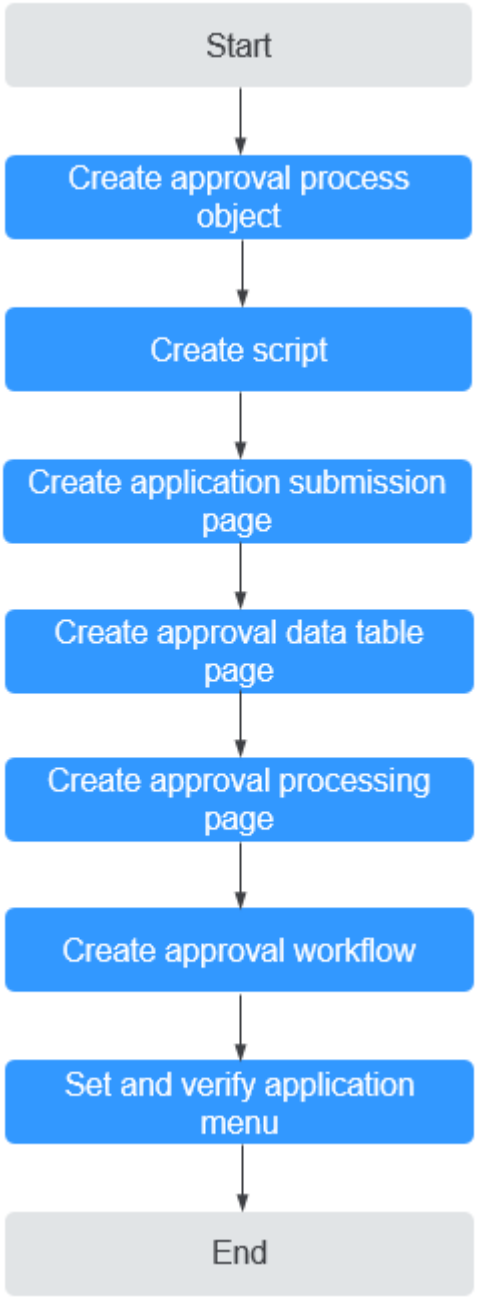
Huawei Cloud Astro Zero BPMs provide built-in task assignment rules, including sequential approval, sign-off, and countersign. Additionally, complex task assignment rules can be implemented through user activities to handle complex interactions.

Procedure

In a BPM, you can configure user activities to implement the priority approval function during countersigning, as shown in **Figure 5-34**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-34 Implementation process of the preferential approval function



Step 1: Create an Approval Process Object

Create an approval process object to store the application data submitted by users.

Step 1 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

Figure 5-35 Creating a blank application

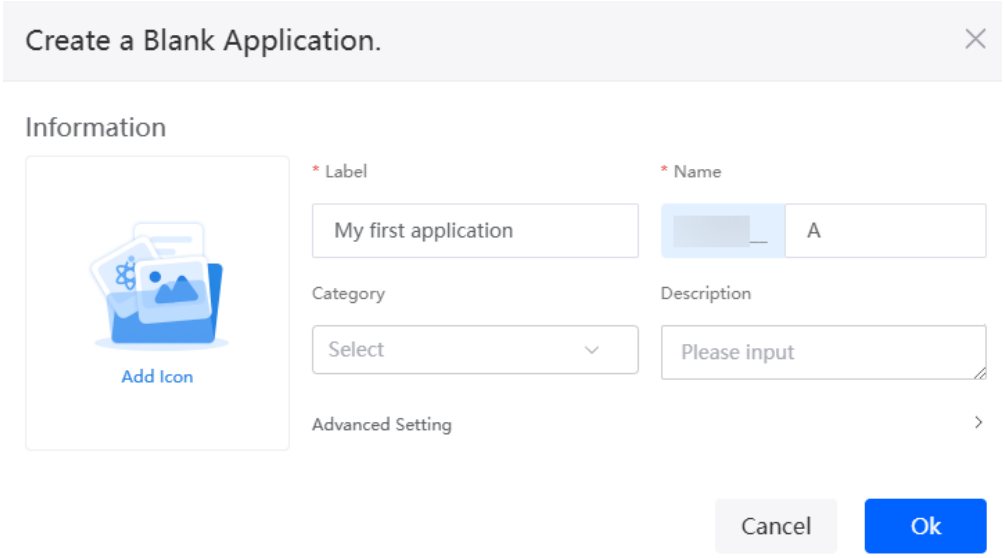


Table 5-3 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 2 In the navigation pane, choose **Data**, and click **+** next to **Object**.

Step 3 Complete the configuration and click the confirm button.

Figure 5-36 Creating processDemo

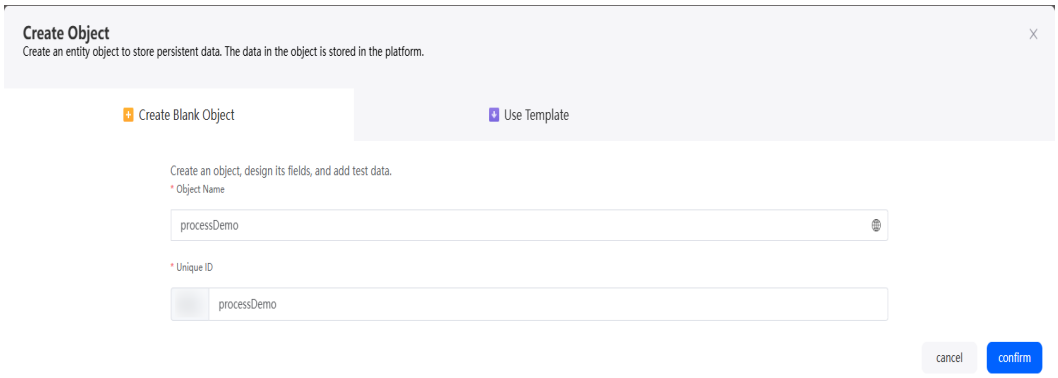


Table 5-4 Parameters for creating processDemo

Parameter	Description	Example
Object Name	Name of the new object, which can be changed after the object is created. Value: 1–80 characters.	processDemo

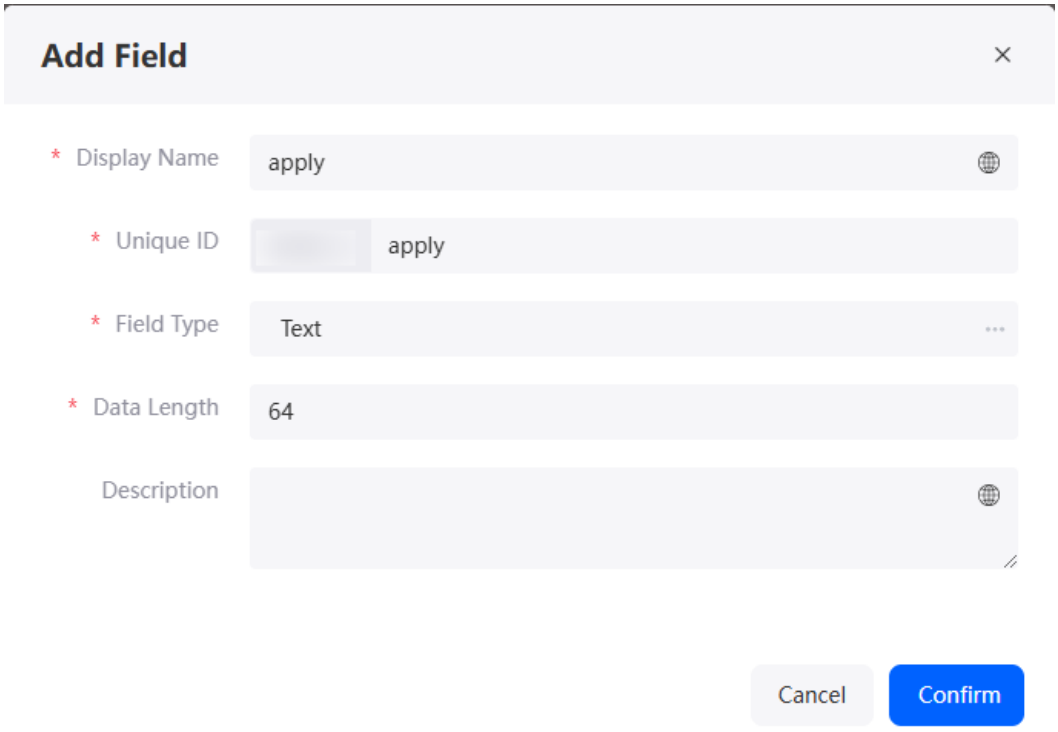
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Unique ID	ID of a new object in the system, which cannot be modified after being created. Naming rules: • Max. 63 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. • Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	processDemo

Step 4 Click  to go to the object details page.

Step 5 On the **Fields** tab page, click **Add** and add the **apply** field for the object.

Figure 5-37 Adding the apply field



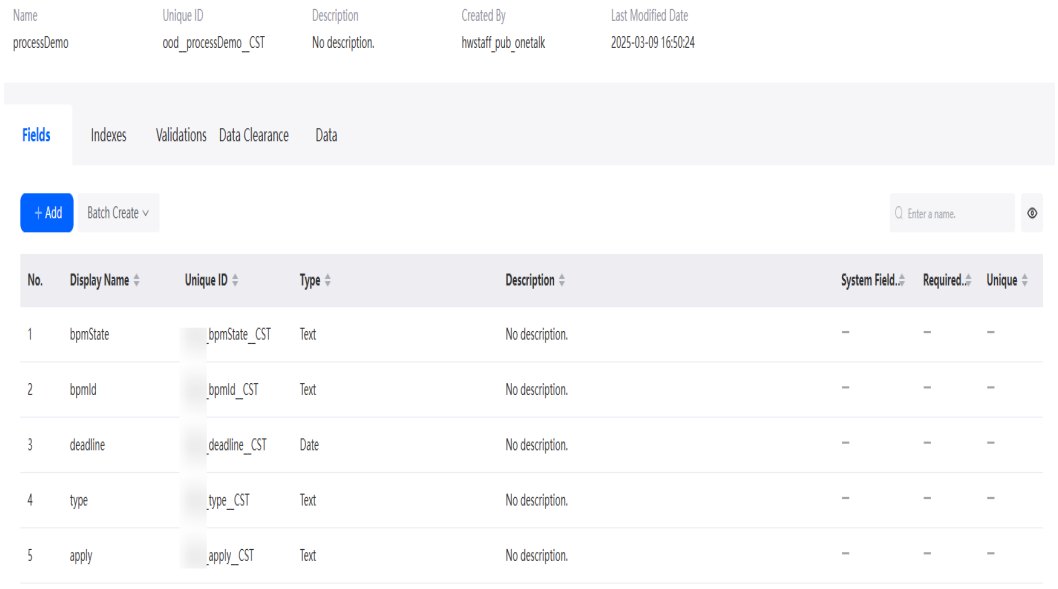
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 5-5 Parameters for adding the apply field

Parameter	Description	Example
Display Name	Name of the new field, which can be changed after the field is created. Value: 1–63 characters.	Application title
Unique ID	ID of a new field in the system. The value cannot be changed after the field is created. Naming rules: • Max. 63 characters, including the prefix namespace. • Start with a letter and use only letters, digits, and an underscore (_). Do not end with an underscore (_).	apply
Field Type	Click ***. On the page that is displayed, select the type of the new field based on the parameter description.	Text
Data Length	Length of a field that can be entered.	64

Step 6 On the **Fields** tab page, click **Add** again to add fields in **Table 5-6**.

Figure 5-38 Fields contained in the processDemo object



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 5-6 Fields to be added

Display Name	Unique ID	Field Type
Application type	type	Text
Application time	deadline	Date
BPM ID	bpmId	Text
BPM status	bpmState	Text

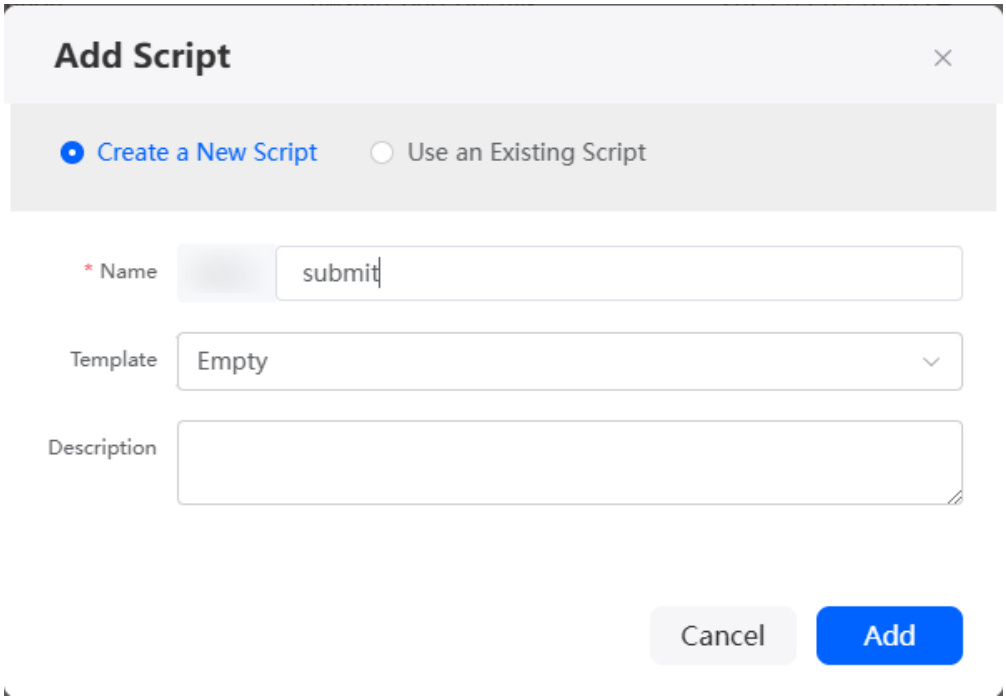
----End

Step 2: Create a Script

Step 1 Create the "submit" script to start the BPM.

1. In the navigation pane, choose **Logic** and click **+** next to **Script**.
2. Create a script, set the name to **submit**, and click **Add**.

Figure 5-39 Creating submit



3. In the script editor, enter the sample code.

```
import * as bpm from 'bp';

@action.object({ type: "param" })
export class ActionInput {
    @action.param({ type: 'Object', required: true, label: 'tripData' })
    applicationData: object;
}

@action.object({ type: "param" })
export class ActionOutput {
    @action.param({ type: 'String', required: true, label: 'interviewID' })
    interviewID: string;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
}

@action.object({ type: "method" })
export class SubmitTrip {
    @action.method({ input: 'ActionInput', output: 'ActionOutput' })
    public run(input: ActionInput): ActionOutput {
        let out = new ActionOutput();
        out.interviewID = bpm.newClientV1().start('Namespace__bpm1', { data:
input.applicationData }).interviewId;

        return out;
    }
}
```

Namespace__bpm1 indicates the name of the BPM to be created, that is, the value set in **Step 6: Create an Approval Workflow**.

4. Click  to save the script. After the script is saved, click  to activate it.

Step 2 Create the **createApplication** script to submit the approval process data and write the data to the object created in **Step 1: Create an Approval Process Object**.

1. In the navigation pane, choose **Logic** and click **+** next to **Script**.
2. Create a script, set the name to **createApplication**, and click **Add**.
3. In the script editor, enter the sample code.

```
import * as db from 'db';

@useObject (['Namespace__processDemo__CST'])

@action.object({ type: "param" })
export class ActionInput {
    @action.param({ type: 'object', required: true, label: 'applicationData' })
    applicationData: object;
    @action.param({ type: 'string', required: true, label: 'interviewID' })
    interviewID: string;
}
@action.object({ type: "param" })
export class ActionOutput {
    @action.param({ type: 'string', required: true, label: 'id' })
    id: string;
}
@action.object({ type: "method" })
export class CreateBusiTrip {
    @action.method({ input: 'ActionInput', output: 'ActionOutput' })
    public run(input: ActionInput): ActionOutput {
        const out = new ActionOutput();
        let engine = db.object('Namespace__processDemo__CST');
        input.applicationData['Namespace__bpmId__CST'] = input.interviewID;
        out.id = engine.insert(input.applicationData);
        return out;
    }
}
```

Namespace__processDemo__CST is the object created in **Step 1: Create an Approval Process Object**, and ***Namespace__bpmId__CST*** is the field added to the object.

4. Click  to save the script. After the script is saved, click  to activate it.

Step 3 Create the **queryApplication** script to query approval process data based on **bpmId**.

1. In the navigation pane, choose **Logic** and click **+** next to **Script**.
2. Create a script, set the name to **queryApplication**, and click **Add**.
3. In the script editor, enter the sample code.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
import * as bpm from 'bp';
import * as db from 'db';
@useObject (['Namespace__processDemo__CST'])
@action.object({ type: "param" })
export class ActionInput {
    @action.param({ type: 'String', required: true, label: 'interviewID' })
    interviewID: string;
}
@action.object({ type: "param" })
export class ActionOutput {
    @action.param({ type: 'Object', required: true })
    detail: object;
}
@action.object({ type: "method" })
export class QueryMyApproDetail {
    @action.method({ input: 'ActionInput', output: 'ActionOutput' })
    public run(input: ActionInput): ActionOutput {
        let out = new ActionOutput();
        out.detail = {};
        let instanceId = input.interviewID;
        const bp = bpm.newInstanceClient();
        let appliObj = db.object('Namespace__processDemo__CST');
        let conditions = [{
            "field": "Namespace__bpmId__CST",
            "operator": db.Operator.eq,
            "value": input.interviewID
        }];
        let applications = appliObj.queryByCondition({
            "conjunction": "AND",
            "conditions": conditions
        });
        if (applications.length > 0) {
            input.interviewID;
            out.detail = {
                Namespace__apply__CST: applications[0].Namespace__apply__CST,
                Namespace__deadline__CST: applications[0].Namespace__deadline__CST,
                Namespace__type__CST: applications[0].Namespace__type__CST,
                Namespace__bpmId__CST: input.interviewID
            };
        }
        return out;
    }
}
```

Namespace__processDemo__CST is an object created in **Step 1: Create an Approval Process Object** and ***Namespace__bpmId__CST***, ***Namespace__apply__CST***, ***Namespace__deadline__CST***, and ***Namespace__type__CST*** are fields added to the object.

4. Click  to save the script. After the script is saved, click  to activate it.

Step 4 Create the **approve** script for processing the approval task.

1. In the navigation pane, choose **Logic** and click **+** next to **Script**.
2. Create a script, set the name to **approve**, and click **Add**.
3. In the script editor, enter the sample code.

```
import * as bpm from 'bp';
import * as context from 'context';
import * as db from 'db';
@action.object({ type: "param" })
export class ActionInput {
    @action.param({ type: 'String', required: true, label: 'state' })
    state: string;

    @action.param({ type: 'String', required: true, label: 'interviewID' })
    interviewID: string;
}
@action.object({ type: "method" })
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
export class ApproveTrip {
    @action.method({ input: 'ActionInput', output: 'ActionOutput' })
    public run(input: ActionInput) {
        let instanceId = input.interviewID;
        let taskClient = bpm.newTaskClient();
        let res = taskClient.queryByCondition({
            condition: {
                conjunction: db.Conjunction.AND,
                conditions: [
                    {
                        field: "processInsID",
                        operator: db.Operator.eq,
                        value: instanceId
                    },
                    {
                        field: "assigneeID",
                        operator: db.Operator.eq,
                        value: context.getUserId()
                    }
                ]
            },
        });
        if (res.length) {
            const taskId = res[0].id;
            var count;
            let vars = taskClient.getVars(taskId, ["pass"]);
            if (vars["pass"]) {
                count = vars["pass"];
            } else {
                count = 0;
            }
            if (input.state == "PASS") {
                bpm.newTaskClient().complete(taskId, '', {
                    pass: count + 1
                })

            } else {
                bpm.newTaskClient().complete(taskId, '', {})
            }
        }
    }
}
```

4. Click  to save the script. After the script is saved, click  to activate it.

Step 5 Create the **getTask** script to obtain the task ID in the approval process.

1. In the navigation pane, choose **Logic** and click **+** next to **Script**.
2. Create a script, set the name to **getTask**, and click **Add**.
3. In the script editor, enter the sample code.

```
import * as db from 'db';
import * as bp from 'bp';
import * as context from 'context';

@action.object({ type: "param" })
export class ActionInput {
    @action.param({ type: 'String', required: true, label: 'interviewID' })
    interviewID: string;
}
export class ActionOutput {
    @action.param({ type: 'String', required: true, label: 'taskId' })
    taskId: string;
}

@action.object({ type: "method" })
export class ApproveTrip {
    @action.method({ input: 'ActionInput', output: 'ActionOutput' })
    public run(input: ActionInput): ActionOutput {
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
        let out = new ActionOutput();
        let instanceId = input.interviewID;

        let taskClient = bp.newTaskClient()
        let res = taskClient.queryByCondition({
            condition: {
                conjunction: db.Conjunction.AND,
                conditions: [
                    {
                        field: "processInsID",
                        operator: db.Operator.eq,
                        value: instanceId
                    },
                    {
                        field: "assigneeID",
                        operator: db.Operator.eq,
                        value: context.getUserId()
                    }
                ]
            },
        });
        if (res.length > 0) {
            let taskId = res[0]['id'];
            out.taskId = taskId;
            console.log(taskId)
            return out;
        }

        return out;
    }
}
```

4. Click  to save the script. After the script is saved, click  to activate it.

Step 6 Create the **getUserName** script for obtaining the user name.

1. In the navigation pane, choose **Logic** and click **+** next to **Script**.
2. Create a script, set the name to **getTask**, and click **Add**.
3. In the script editor, enter the sample code.

```
import * as objectstorage from 'objectstorage';
import * as buffer from 'buffer';
import * as excel from 'excel';
import * as context from 'context';
import * as db from 'db';//Import the standard library related to the object.


//Define the input parameter structure.
@action.object({ type: "param" })
export class ActionInput {
}
//Define the output parameter structure. The output parameter contains one parameter, that is, the
record ID of workOrder.
@action.object({ type: "param" })
export class ActionOutput {
    @action.param({ type: 'string', required: false, label: 'string' })
    res: string;
}
//Use the data object namespace__processDemo__CST.
@useObject(['User'])
@action.object({ type: "method" })
export class CreateWorkOrder {    //Define the API class. The input parameter of the API is
ActionInput, and the output parameter is ActionOutput.
    @action.method({ input: 'ActionInput', output: 'ActionOutput' })
    public createWorkOrder(input: ActionInput): ActionOutput {
        let out = new ActionOutput();
        try {
            let s = db.object('User');    //Query the user table.
            let condition = {
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
            "conjunction": "AND",
            "conditions": [{
                "field": "id",
                "operator": "eq",
                "value": context.getUserId()
            }]
        };
        let userRecod = s.queryByCondition(condition);
        if (userRecod) {
            out.res = userRecod[0].usrName;
            return out;
        }
    } catch (error) {
        console.error(error.name, error.message);
    }
    return out;
  }
}
```

4. Click  to save the script. After the script is saved, click  to activate it.

----End

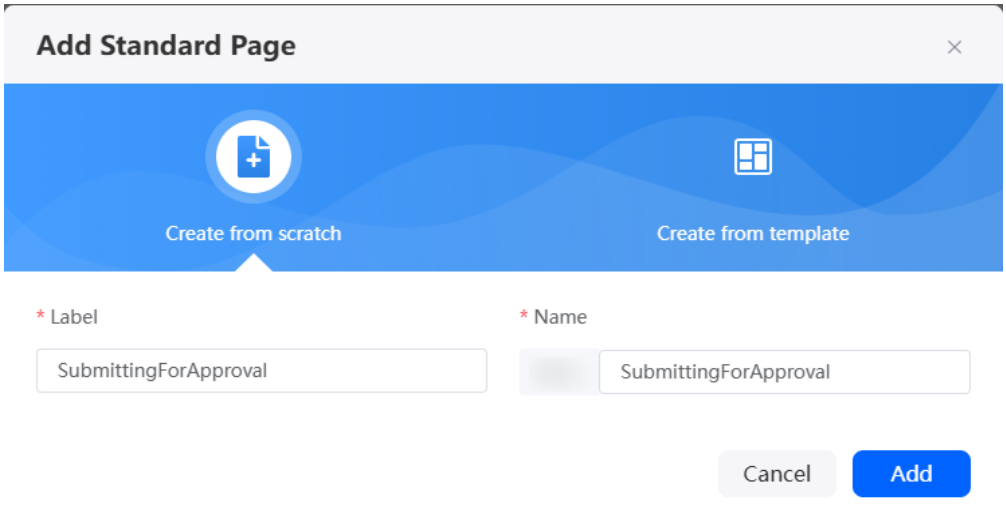
Step 3: Create an Application Submission Page

Create a standard page for submitting applications, including fields for application title, type, and time.

Step 1 Create a blank standard page.

1. In the navigation pane, choose **Page**, and click **+** next to **Standard Page**.
2. Enter a label and name and click **Add** to create a standard page.

Figure 5-40 Creating a standard page for submitting an application

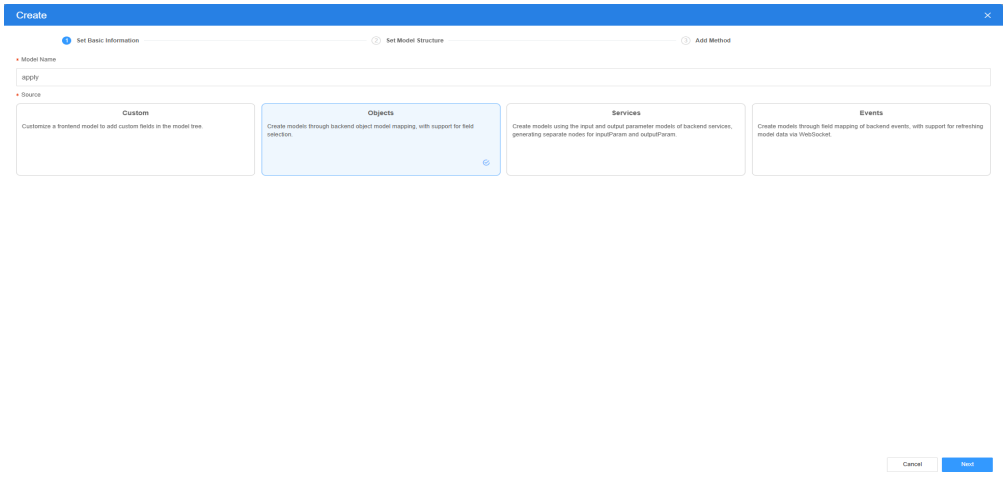


Step 2 Create an object model **apply**.

1. At the bottom of the standard page, click **Model View**.
2. Click **New**, specify **Model Name** (for example, **apply**), select **Objects** for **Source**, and click **Next**.

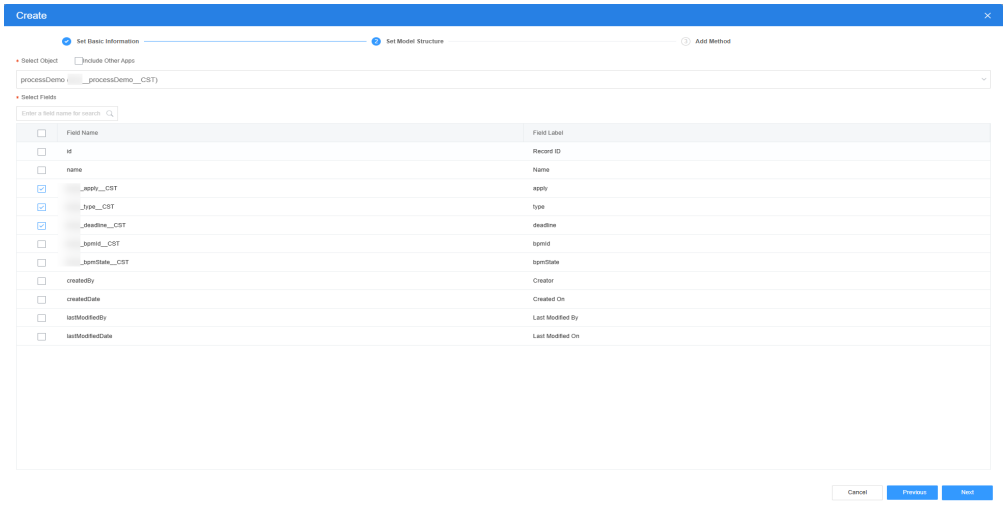
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-41 Creating an object model



3. Select the object created in **Step 1: Create an Approval Process Object**, select the **apply**, **type**, and **deadline** fields, click **Next**, and click **OK**. The model is created.

Figure 5-42 Selecting objects and fields

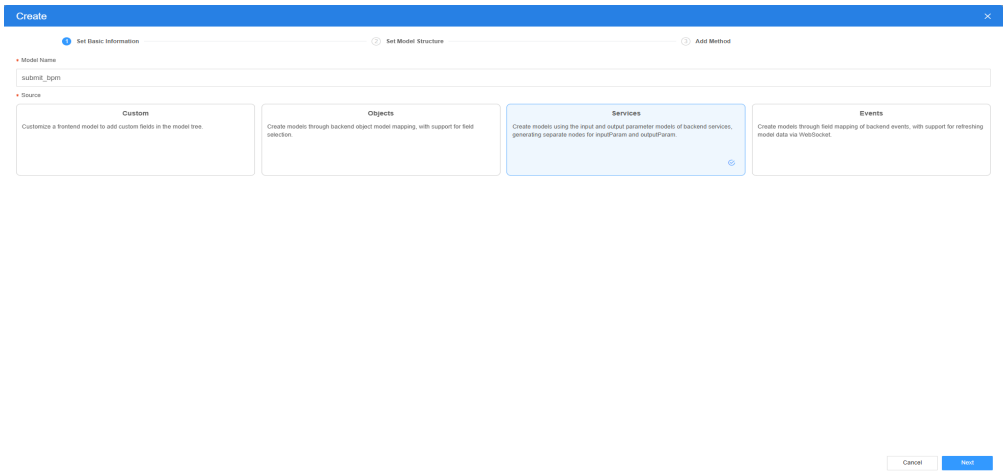


Step 3 Create a service model **submit_bpm**.

1. On the **Model View** page, click **New**, enter the model name (for example, **submit_bpm**), select **Services** for **Source**, and click **Next**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-43 Creating a service model



2. Select the script created in **Step 1** and click the confirm button.

Figure 5-44 Selecting the submit script

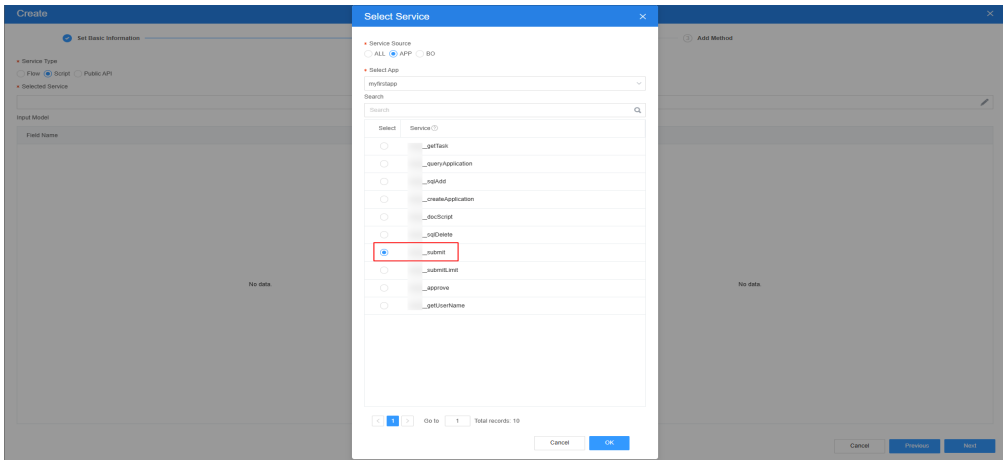
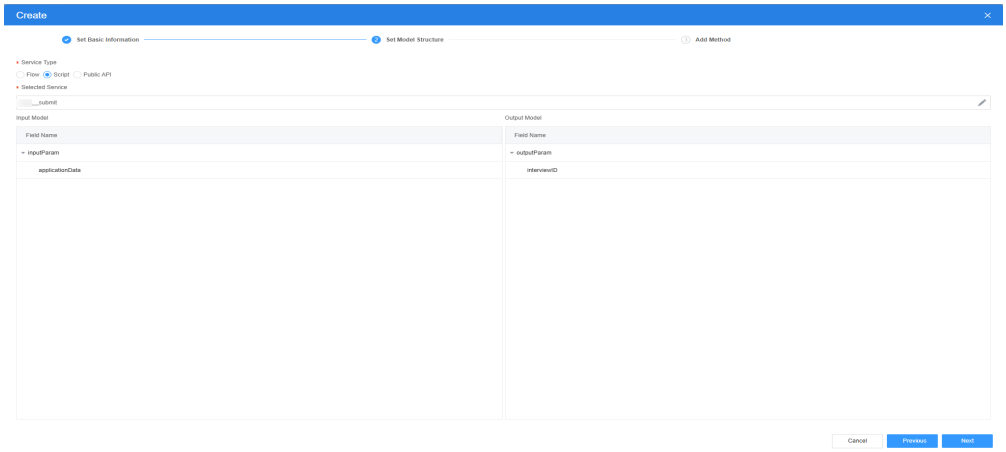


Figure 5-45 Effect after script selection



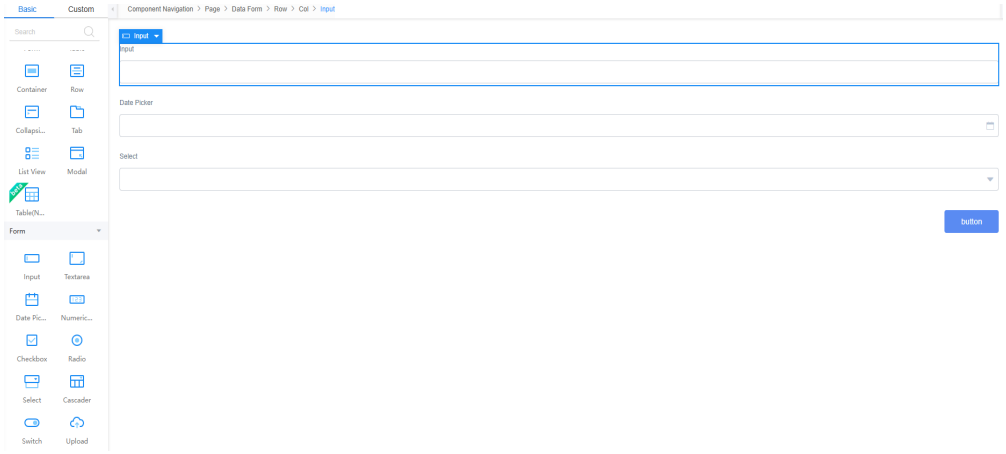
3. Click **Next** and then click **OK**.

Step 4 Return to the design view page, drag the text box, date selection box, and drop-down list box widget, and bind the corresponding object model.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

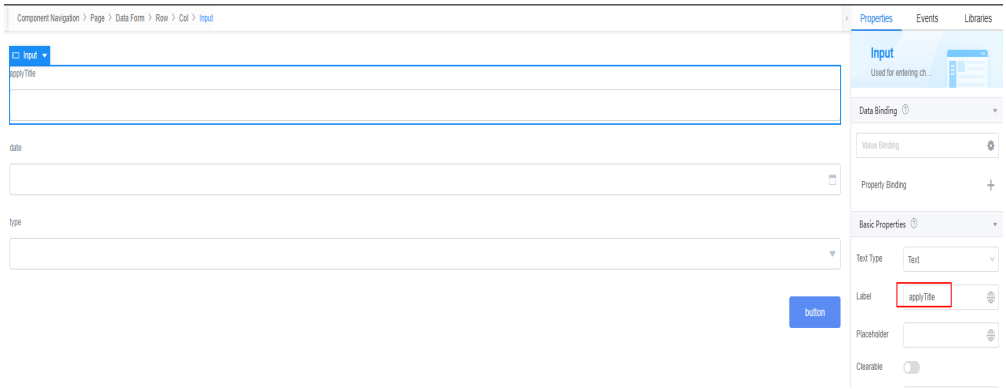
1. At the bottom of the standard page, click **Designer View**.
2. In the **Basic** area, drag the text box, date selection box, drop-down list, and button widgets to the canvas and arrange them, as shown in **Figure 5-46**.

Figure 5-46 Layout



3. Select the text box, date selection box, and drop-down list, and change their labels to the application title, time, and type, respectively.

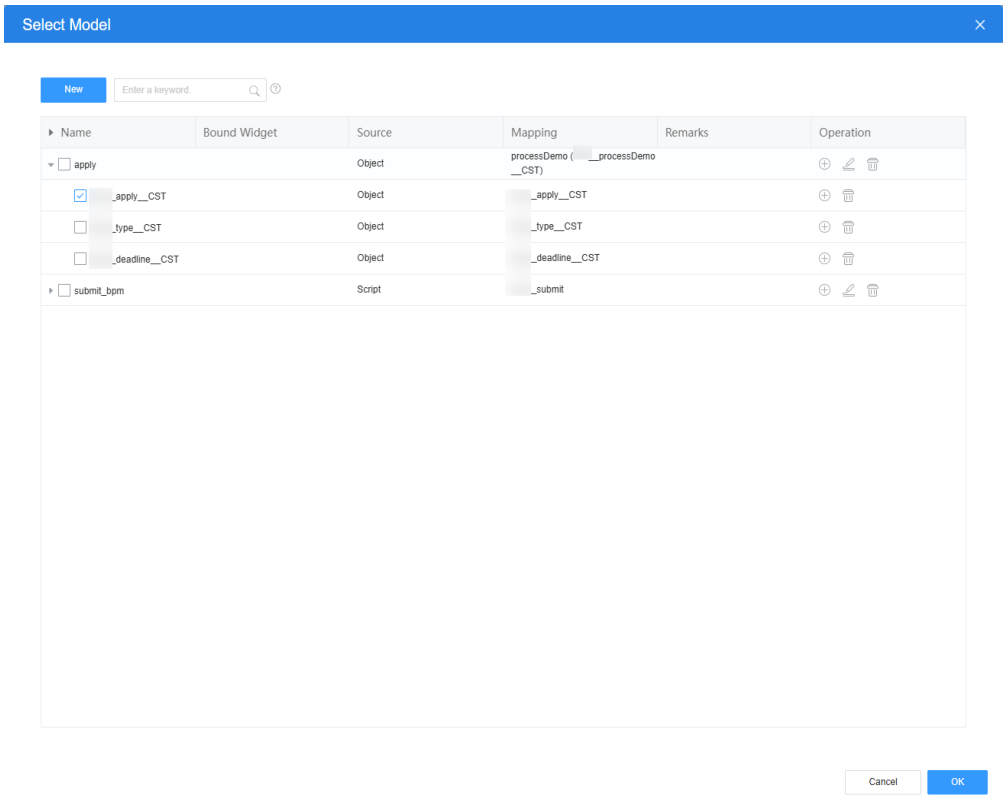
Figure 5-47 Modifying the widget labels



4. Select the input widget. In the **Properties** > **Data Binding** area, click next to **Value Binding** to bind the ***Namespace*__apply__CST** field of the object model in **Step 2** to the widget.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-48 Binding the **apply** field in the object model to the widget



5. Repeat the previous steps to bind the ***Namespace*__deadline__CST** and ***Namespace*__type__CST** fields of the object model in **Step 2** to the date selection box and drop-down list widget, respectively.

Figure 5-49 Binding the deadline field in the object model to the widget

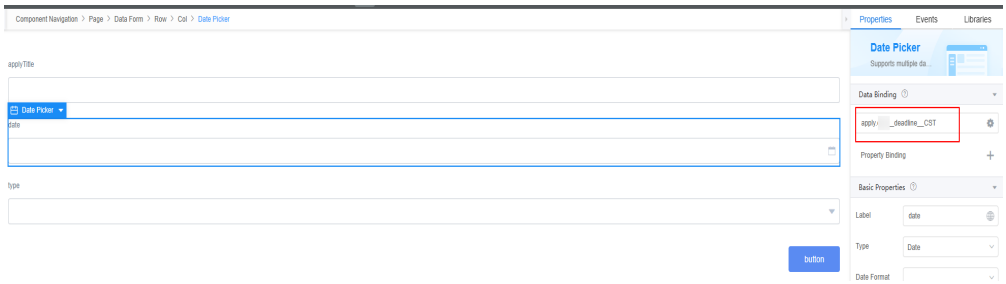
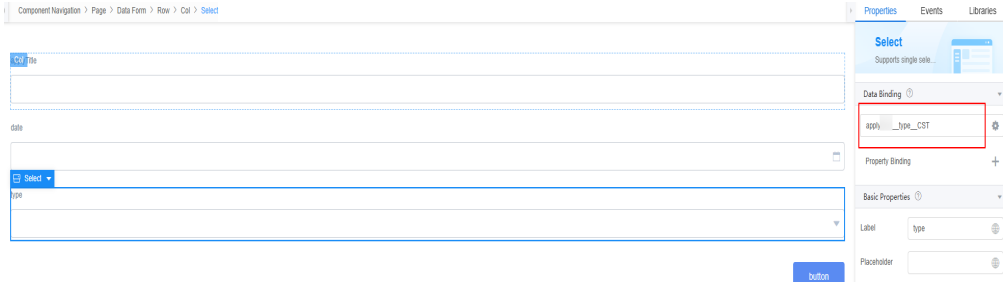


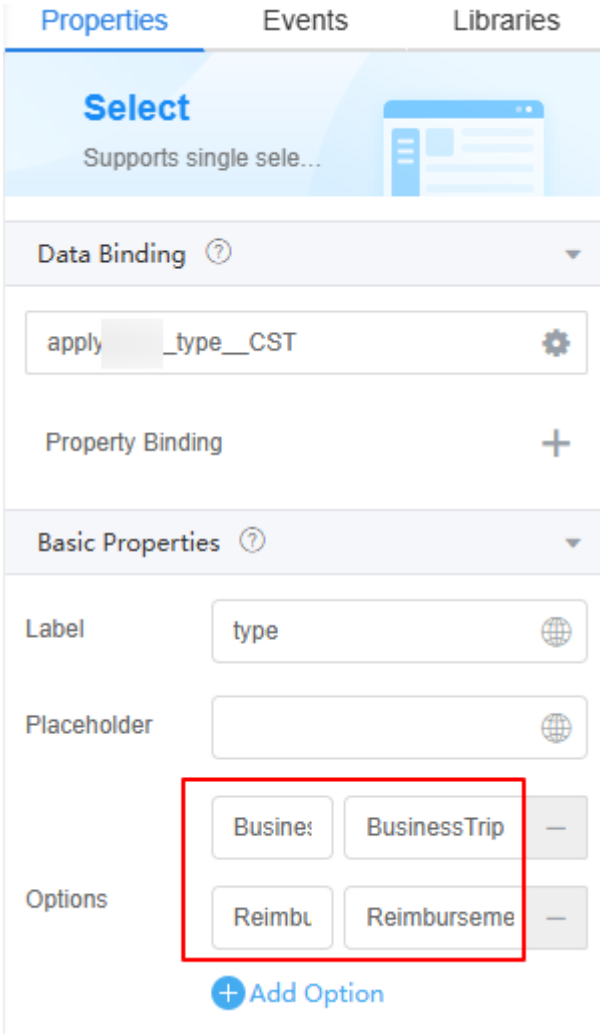
Figure 5-50 Binding the type field in the object model to the widget



6. Select the drop-down box widget. In the **Properties** > **Basic Properties** area, set **Options** to **BusinessTrip** and **Reimbursement**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-51 Options



7. Select the button widget and set the display name to "Submit Application".
8. On the **Events** tab page, click **+** next to **on-click** and add the following event for the button widget.

```
var apply = $model.ref("apply").getData();
var submit = $model.ref("submit_bpm").getData();
submit.inputParam.applicationData = apply;
$model.ref("submit_bpm").run().then(function(data){
    context.$message.success('Submitted successfully.');
    closeCurrentTab();
    })
.catch(function (err) {
    console.log('error is', error);
    context.$message.error('Submission failed:' + error.resMsg);
});

function closeCurrentTab() {
    if (parent.bingo) {
        var removeTab = parent.bingo.removeTab;
        var getCurrentTab = parent.bingo.getCurrentTab;
        if (removeTab && getCurrentTab) {
            return removeTab(getCurrentTab());
        }
    }
    if (window.parent != window) {
        window.parent.close();
    } else {
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
        window.close();
    }
}
```

Step 5 Click in the upper part of the page to save the page settings.

Step 6 After the settings are saved, click in the upper part of the page to preview.

----End

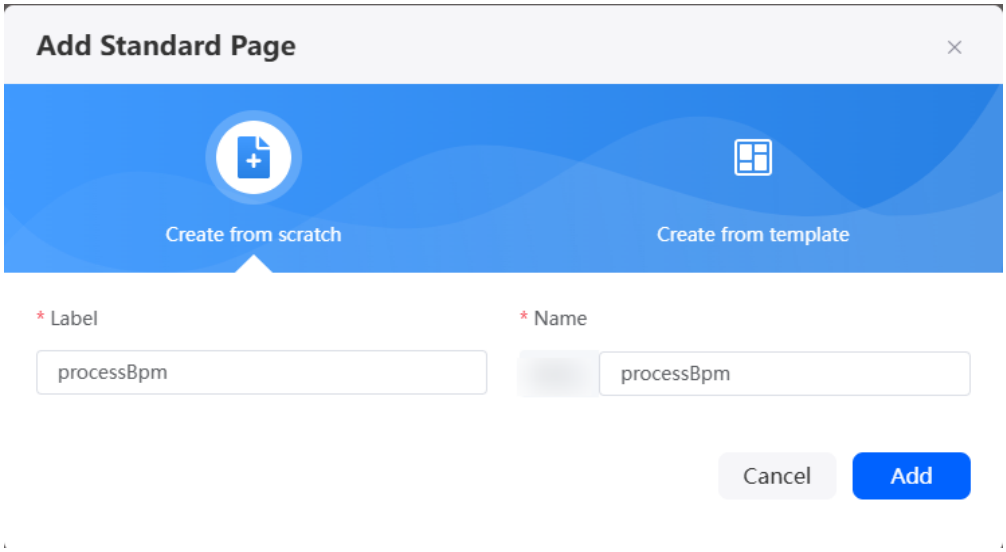
Step 4: Create an Approval Data Table

Create a standard page to display submitted approval data and add an "approve" button in the operation column. Clicking this button navigates to the approval page.

Step 1 Create a blank standard page.

1. In the navigation pane, choose **Page**, and click **+** next to **Standard Page**.
2. Enter a label and name (such as **processBpm**) and click **Add**.

Figure 5-52 Creating a standard page for approving the data table

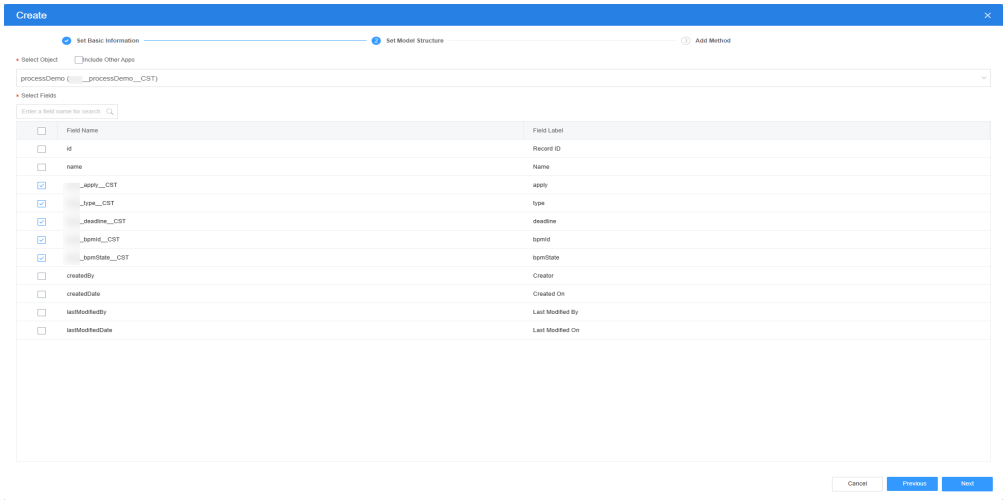


Step 2 Create an object model **processBpm**.

1. At the bottom of the standard page, click **Model View**.
2. Click **New**, specify **Model Name** (for example, **processBpm**), select **Objects** for **Source**, and click **Next**.
3. Select the object created in **Step 1: Create an Approval Process Object**, select the **apply**, **type**, and **deadline**, **bpmId**, and **bpmState** fields, click **Next**, and click **OK**. The model is created.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

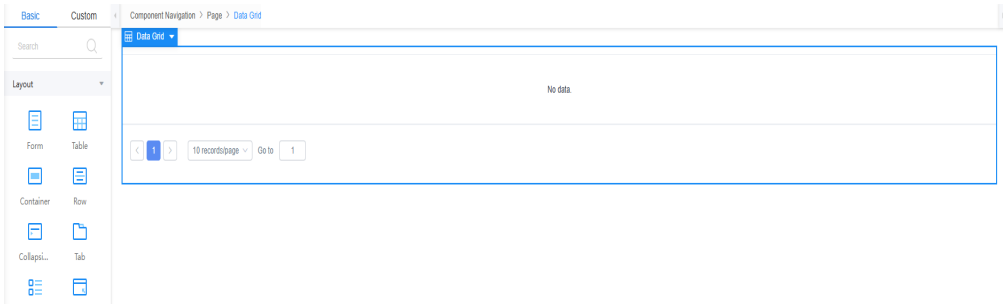
Figure 5-53 Selecting objects and fields



Step 3 Return to the design view page, drag a table widget to the canvas, and bind an object model to the widget.

1. In the **Basic** area, drag a table widget to the canvas.

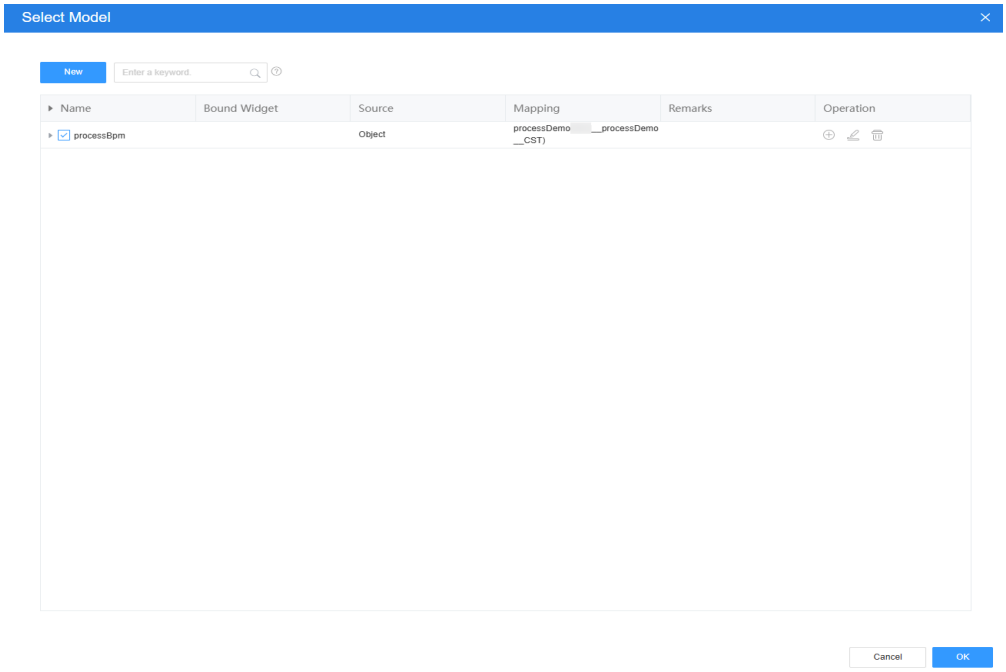
Figure 5-54 Layout



2. Select the table widget, choose **Properties** > **Data Binding**, click  next to **Value Binding**, and bind the object model **processBpm** in **Step 2** to the widget.

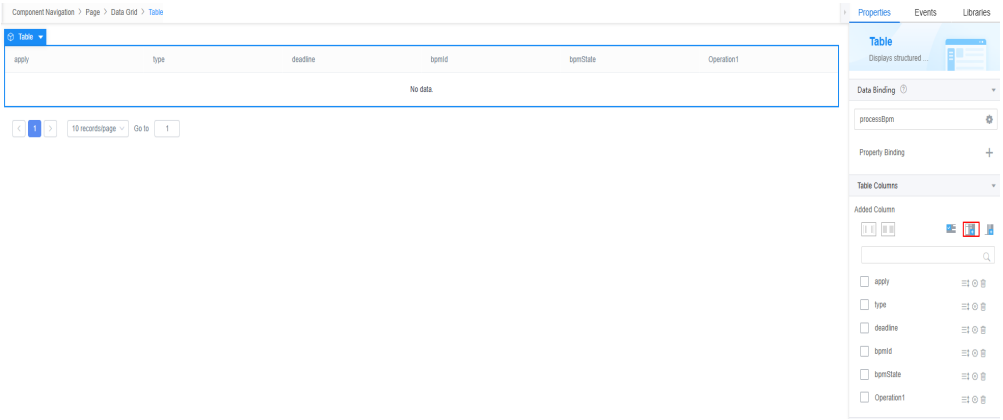
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-55 Binding the object model **processBpm** to the widget



3. Select the table widget, choose **Properties** > **Table Columns**, and click  to add operation columns to the table.

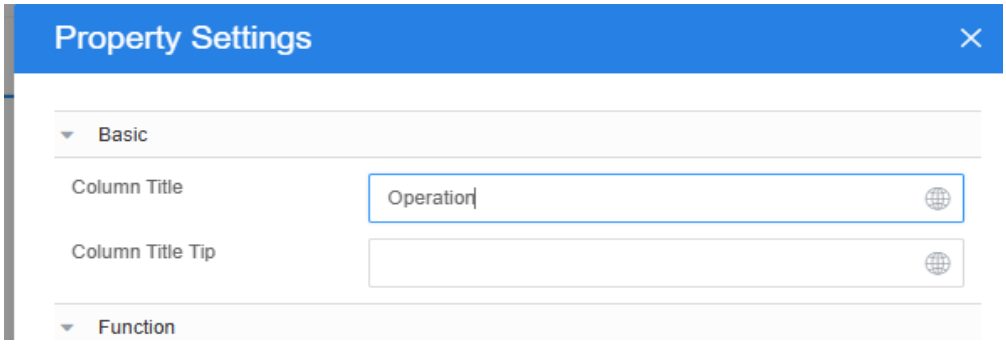
Figure 5-56 Adding operation columns



4. Click  next to the **Operation1** column and change the column title to **Operation**.

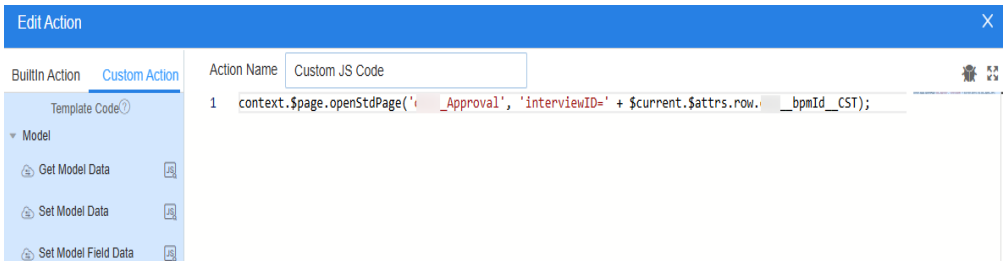
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-57 Changing the column title to Operation



5. In the **Operation Button** area, click **Add Operation Button** and change **Label** to **approve**.
6. Click next to the approval button and click **+** next to the action list.

Figure 5-58 Adding a custom action for the approval button



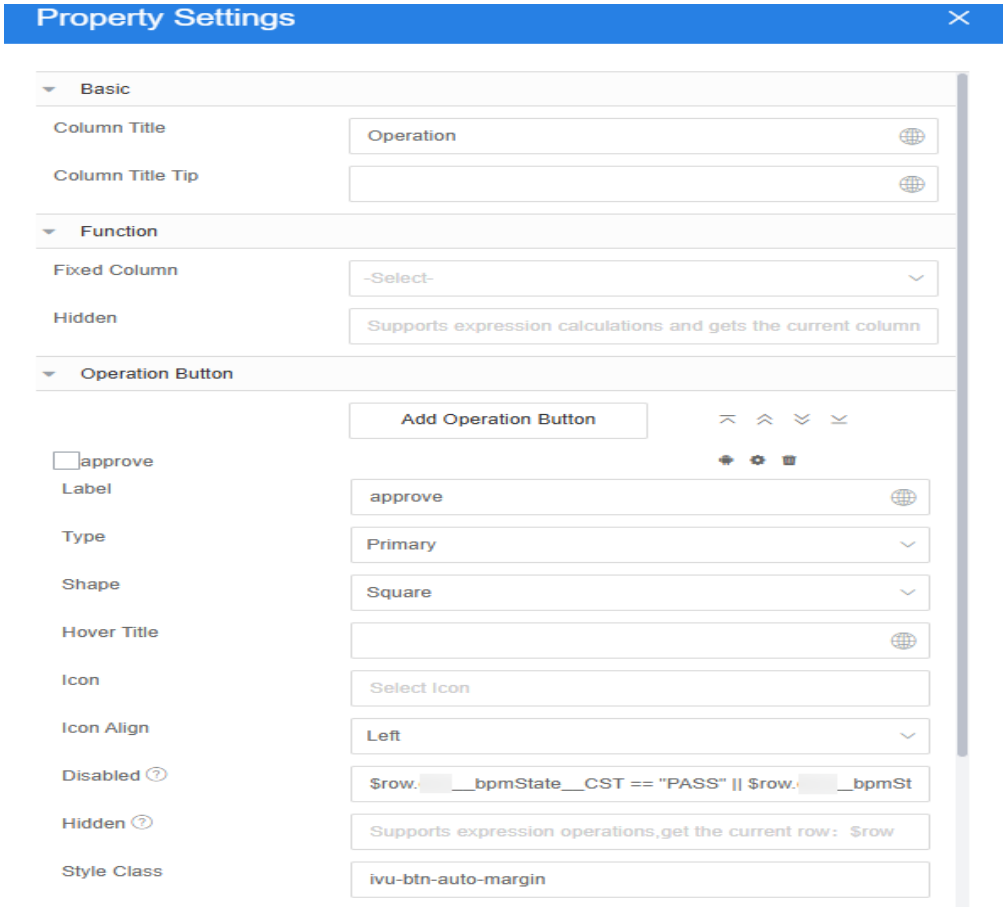

```
context.$page.openStdPage('Namespace__Approval', 'interviewID=' +
$current.$attrs.row.Namespace__bpmId__CST);
```

***Namespace*__Approval** is the standard page created in **Step 5: Create an Approval Processing Page**, and ***Namespace*__bpmId__CST** is the field added in **Step 1: Create an Approval Process Object**.

7. In the **Disabled** text box, enter the following code to disable the approval button after the application is approved.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-59 Disabling the button




```
$row.Namespace__bpmState__CST == "PASS" || $row. Namespace__bpmState__CST == "FAIL"
```

***Namespace*__bpmState__CST** is the object field added in **Step 6**.

Step 4 Click in the upper part of the page to save the page settings.

----End

Step 5: Create an Approval Processing Page

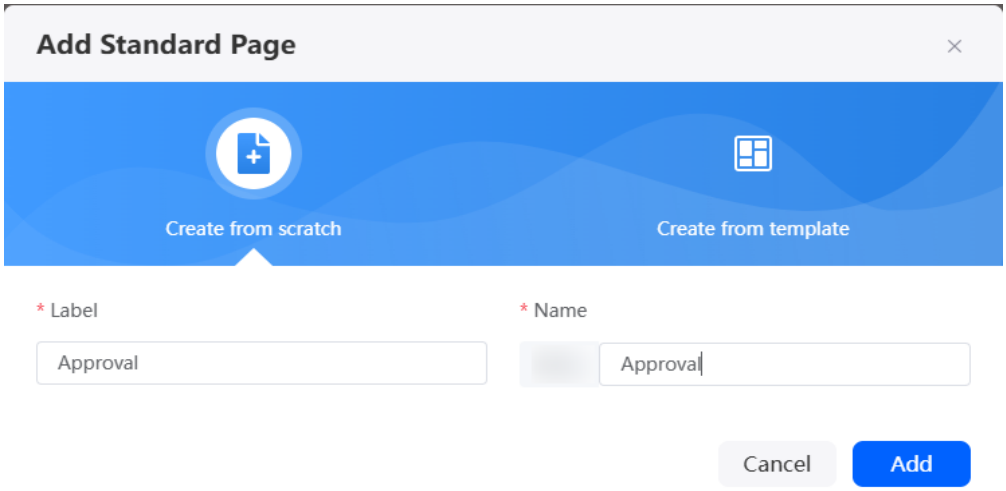
Create a standard page for approving applications. The page contains three fields: application title, application type, and application time. The page also contains four buttons: approve, reject, approved by the manager, and rejected by the manager.

Step 1 Create a blank standard page.

1. In the navigation pane, choose **Page**, and click **+** next to **Standard Page**.
2. Enter a label and name (such as **Approval**) and click **Add**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

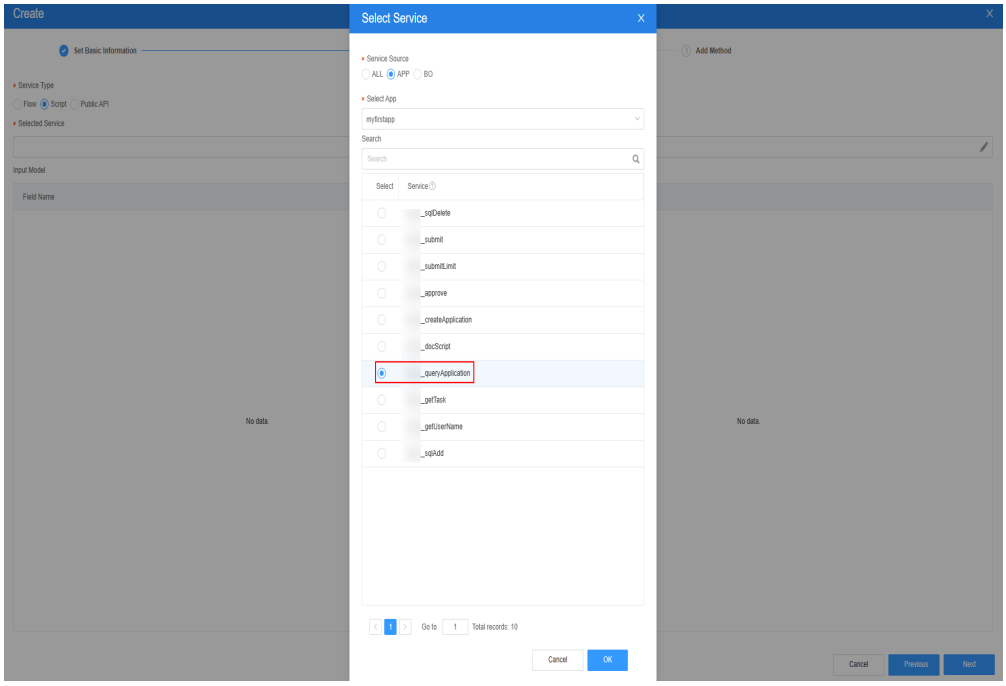
Figure 5-60 Creating a standard page for approving the data table



Step 2 Create the script model **queryApplication** for querying the approval data corresponding to **bpmId**.

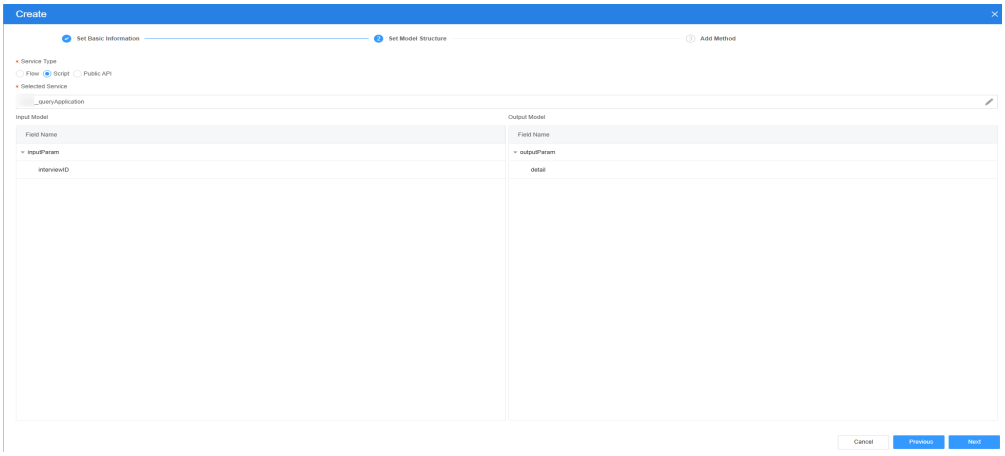
1. At the bottom of the standard page, click **Model View**.
2. Click **New**, specify **Model Name** (for example, **queryApplication**), select **Services** for **Source**, and click **Next**.
3. Select the script created in **Step 3**, click **Next**, and then click **OK**.

Figure 5-61 Selecting the required script



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

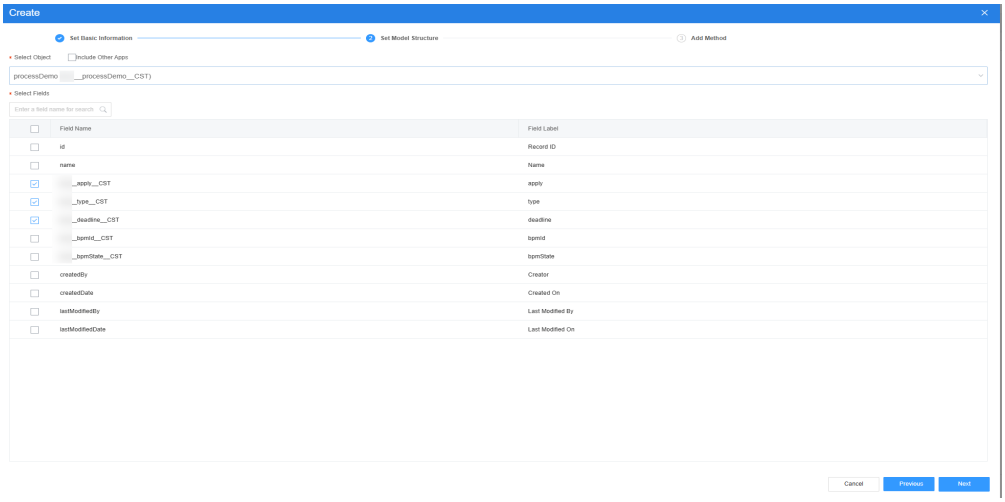
Figure 5-62 Effect after script selection



Step 3 Create an object model **process**.

1. On the **Model View** page, click **New**, enter the model name (for example, **process**), select **Objects** for **Source**, and click **Next**.
2. Select the object created in **Step 1: Create an Approval Process Object**, select the **apply**, **type**, and **deadline** fields, click **Next**, and click **OK**. The model is created.

Figure 5-63 Selecting objects and fields



Step 4 Create the script model **approve** for submitting the approval result.

1. On the **Model View** page, click **New**, enter the model name (for example, **approve**), select **Services** for **Source**, and click **Next**.
2. Select the script created in **Step 4**, click **Next**, and then click **OK**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-64 Selecting the required script

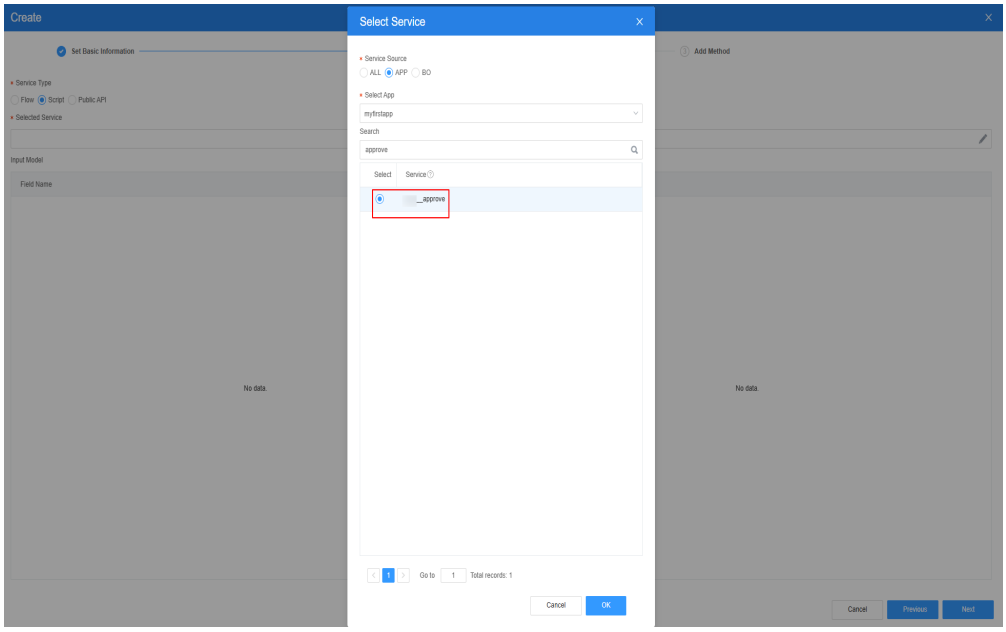
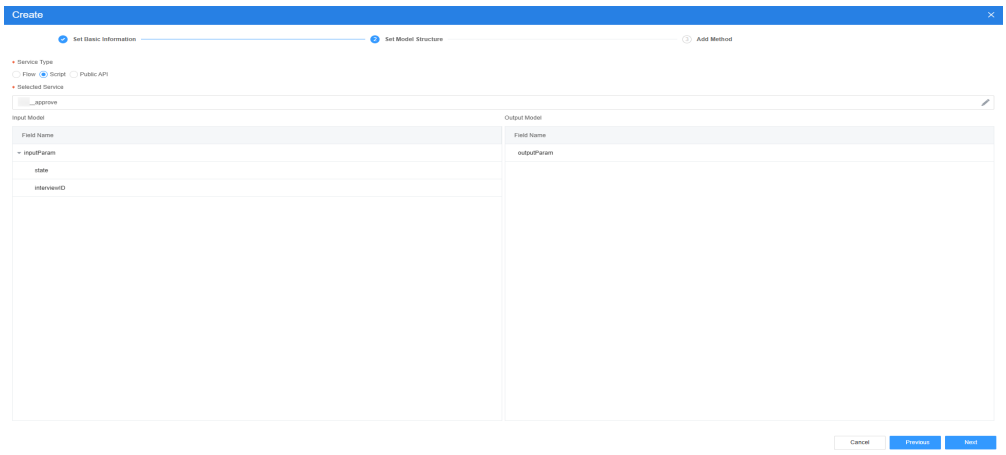


Figure 5-65 Effect



Step 5 Create the script model **getTask**, which is used to automatically complete the node after a leader approves the application. The node ignores the comments of others and submits the approval result.

1. On the **Model View** page, click **New**, enter the model name (for example, **getTask**), select **Services** for **Source**, and click **Next**.
2. Select the script created in **Step 5**, click **Next**, and then click **OK**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-66 Selecting the required script

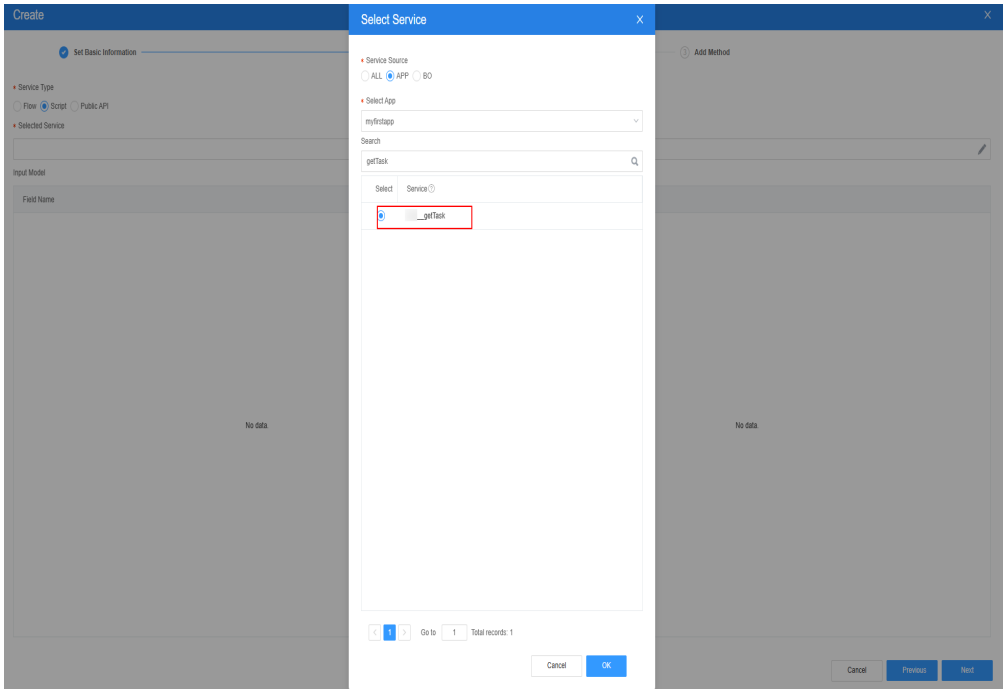
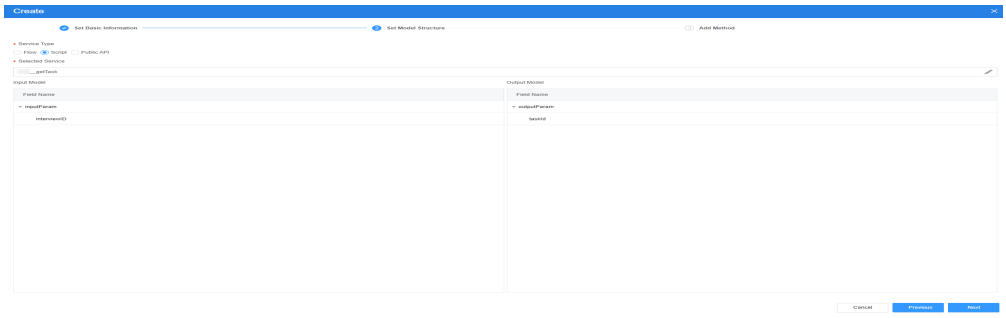


Figure 5-67 Effect



Step 6 Create the script model **getUserName** to obtain usernames.

1. On the **Model View** page, click **New**, enter the model name (for example, **getUserName**), select **Services** for **Source**, and click **Next**.
2. Select the script created in **Step 6**, click **Next**, and then click **OK**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-68 Selecting the required script

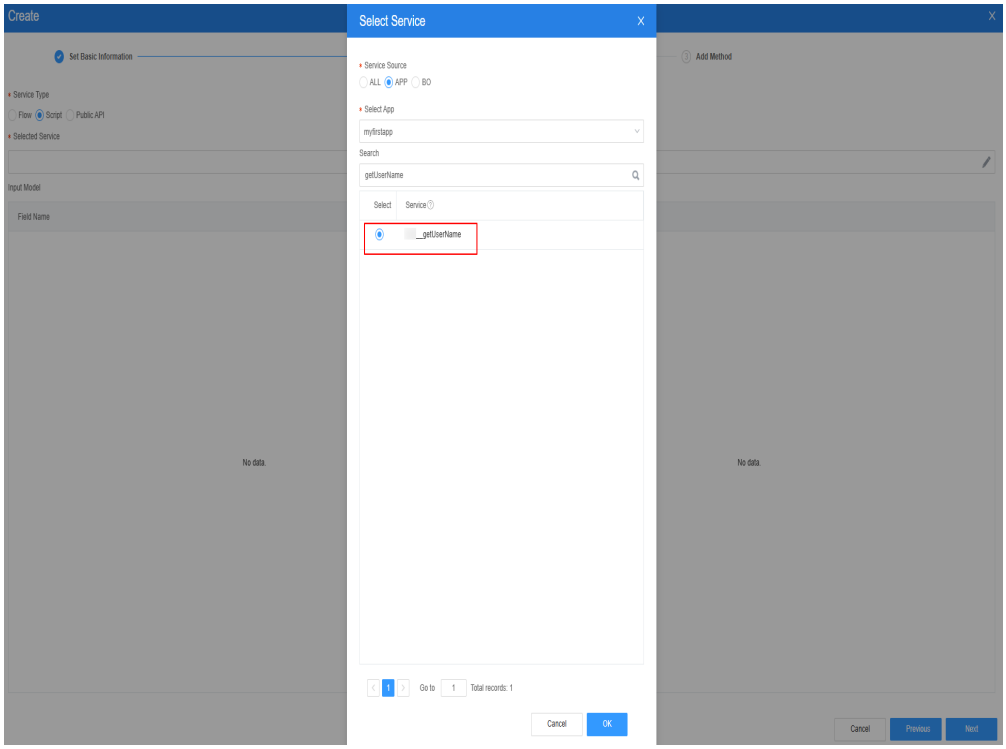
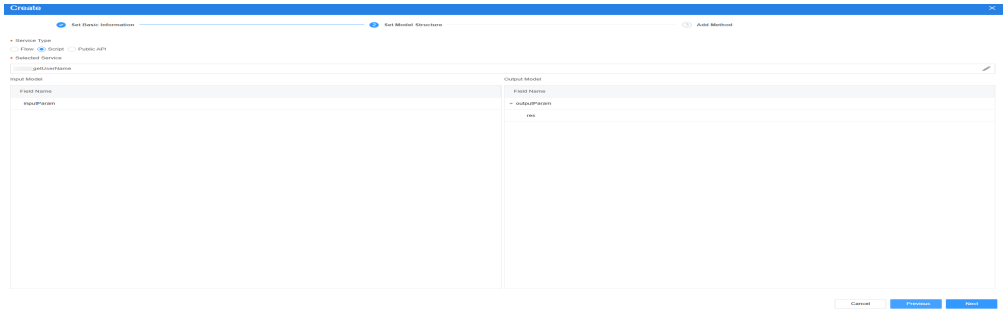


Figure 5-69 Effect

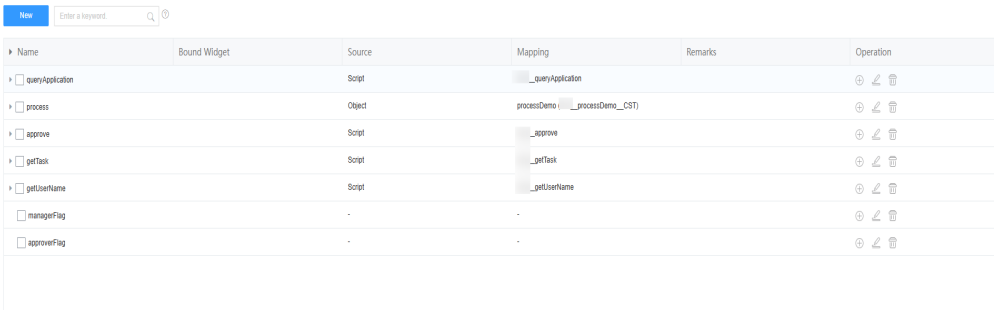


Step 7 The customized models **managerFlag** and **approverFlag** are added to control the button display.

1. On the **Model View** page, click **New**, enter the model name (for example, **managerFlag**), select **Custom** for **Source**, and click **Next**.
2. Click **Next** and then click **OK**.
3. Repeat the preceding operations to create the custom model **approverFlag**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

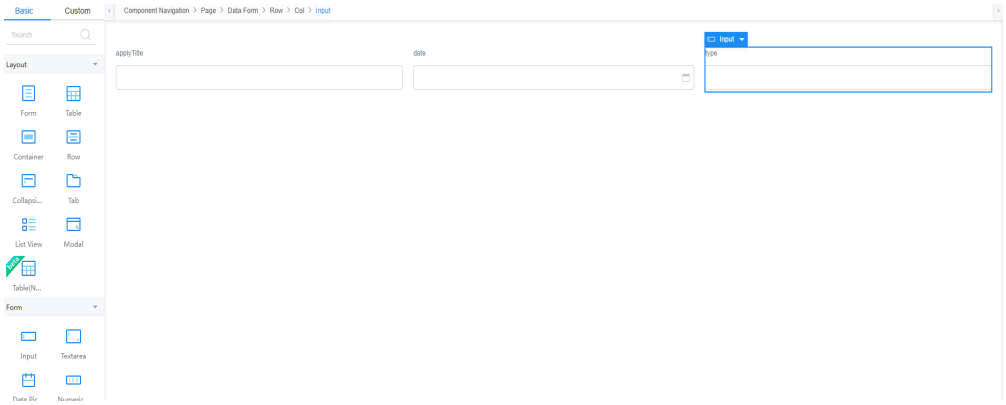
Figure 5-70 Checking the created model



Step 8 Return to the design view page, drag a widget, and bind an object model to the widget.

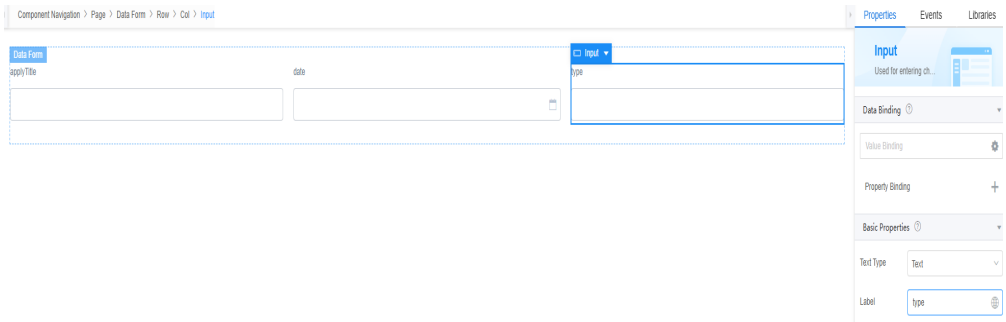
1. In the **Basic** area, drag two text box widgets and one date selection box widget to the canvas.

Figure 5-71 Layout of the approval processing standard page



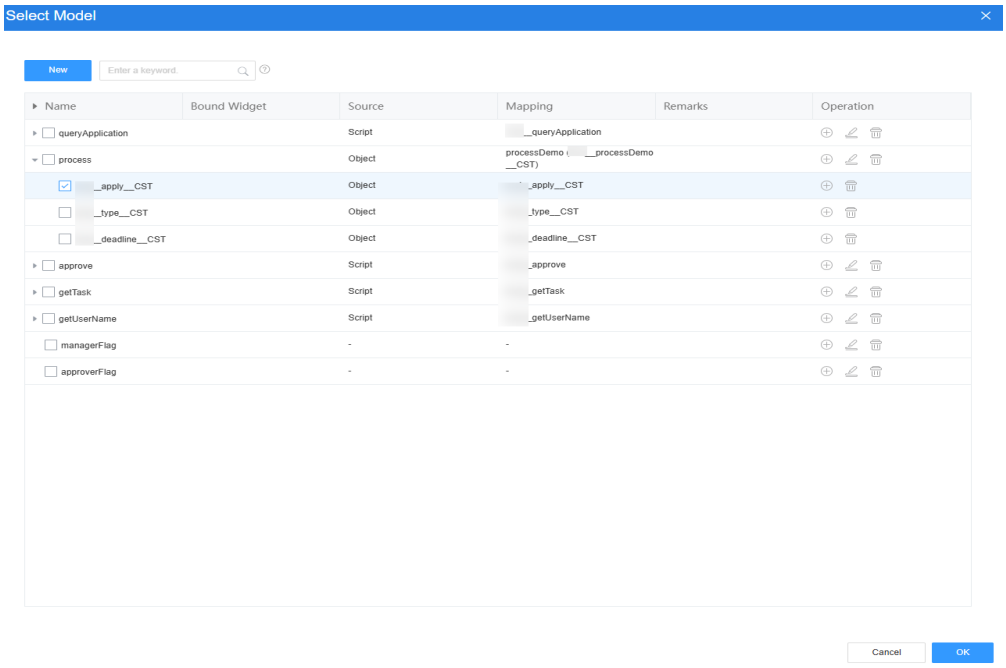
2. Select widgets and change their label to the application title, application time, and application type.

Figure 5-72 Modifying the widget label



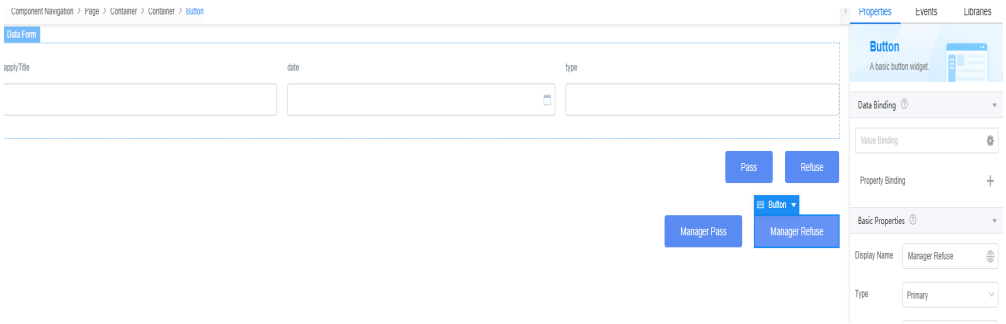
3. Select the first input widget. In the **Properties** > **Data Binding** area, click next to **Value Binding** to bind the ***Namespace*__apply__CST** field of the object model in **Step 3** to the widget.

Figure 5-73 Binding the **apply** field in the object model to the widget



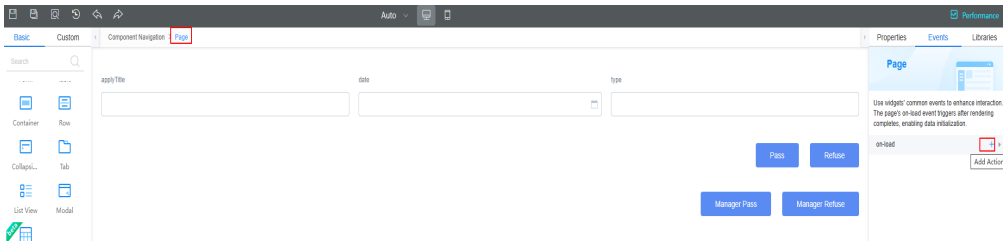
4. Use the same method to bind the ***Namespace*__deadline_CST** field to the date selection box widget and bind the ***Namespace*__type_CST** field to the second input widget.
5. Drag four button widgets to the canvas and change the button names to **Pass**, **Refuse**, **Manager Pass**, and **Manager Refuse**.

Figure 5-74 Modifying the button name



6. Select the page widget. On the **Events** tab page, click **+** next to **on-load**.

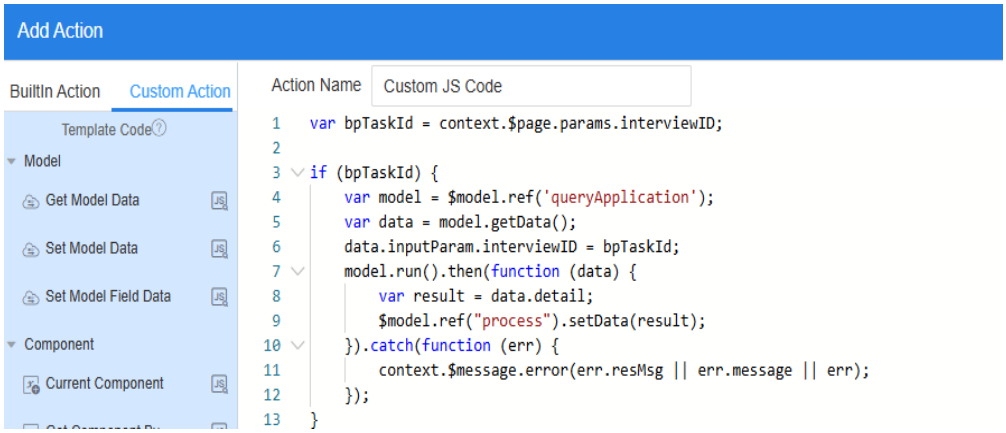
Figure 5-75 Adding an event for the page



7. In the custom action, enter the following code:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-76 Entering an event code




```
var bpTaskId = context.$page.params.interviewID;

if (bpTaskId) {
    var model = $model.ref('queryApplication');
    var data = model.getData();
    data.inputParam.interviewID = bpTaskId;
    model.run().then(function (data) {
        var result = data.detail;
        $model.ref("process").setData(result);
    }).catch(function (err) {
        context.$message.error(err.resMsg || err.message || err);
    });
}
//Approver list
var approver = ["user_name1","user_name2"]
//Manager list
var manager = ["user_name3"]
$model.ref('getUserName').run().then(function(data){
        if (!approver.includes(data['res'])) {
        $model.ref("approverFlag").setData(true);
    }
    if (!manager.includes(data['res'])) {
        $model.ref("managerFlag").setData(true);
    } else {
        //If the approver is a manager, the approver button is also hidden.
       $model.ref("approverFlag").setData(true);
    }
}).catch(function(error){
    console.log('error is', error);
    context.$message.error('Submission failed:' + error.resMsg);
});
```

queryApplication indicates the script model created in **Step 2**, **user_name1** and **user_name2** indicate the approver names, and **user_name3** indicates the manager's approver name.

8. Select the **Pass** button. On the **Events** tab page, click **+** next to **on-click** to add a custom action for the button.

```
var data = $model.ref('process').getData();
var approve = $model.ref('approve').getData();
approve.inputParam.interviewID = data['Namespace__bpmId__CST'];
approve.inputParam.state = 'PASS';
$model.ref('approve').run().then(function(data){
    context.$message.success('Submitted successfully.');
    closeCurrentTab();
}).catch(function(error){
    console.log('error is', error);
    context.$message.error('Submission failed:' + error.resMsg);
});
```

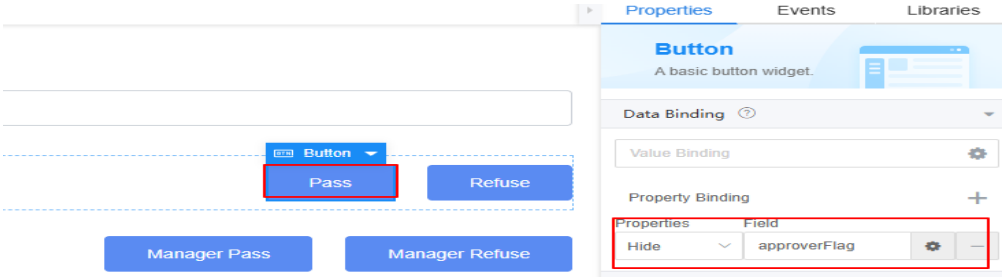
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
function closeCurrentTab() {
    if (parent.bingo) {
        var removeTab = parent.bingo.removeTab;
        var getCurrentTab = parent.bingo.getCurrentTab;
        if (removeTab && getCurrentTab) {
            return removeTab(getCurrentTab());
        }
    }
    if (window.parent != window) {
        window.parent.close();
    } else {
        window.close();
    }
}
```

process is the object model created in **Step 3**, and **approve** is the script model created in **Step 4**. ***Namespace*__bpmId__CST** is the field added in **Step 1: Create an Approval Process Object**.

9. Select the **Pass** button again. In the **Properties** area, click **+** next to **Value Binding** to bind the **approverFlag** model defined in **Step 7** to the widget.

   Figure 5-77 Binding the approverFlag model to the button widget



10. Select the **Refuse** button. On the **Events** tab page, click **+** next to **on-click** to add a custom action for the button.

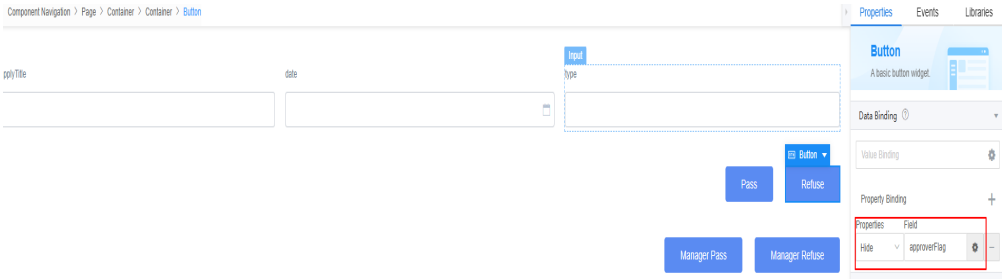
```
var data = $model.ref('process').getData();
var approve = $model.ref('approve').getData();
approve.inputParam.interviewID = data['Namespace__bpmId__CST'];
approve.inputParam.state = 'FAIL';
$model.ref('approve').run().then(function(data){
    context.$message.success('Submitted successfully.');
    closeCurrentTab();
}).catch(function(error){
    console.log('error is', error);
    context.$message.error('Submission failed:' + error.resMsg);
});
function closeCurrentTab() {
    if (parent.bingo) {
        var removeTab = parent.bingo.removeTab;
        var getCurrentTab = parent.bingo.getCurrentTab;
        if (removeTab && getCurrentTab) {
            return removeTab(getCurrentTab());
        }
    }
    if (window.parent != window) {
        window.parent.close();
    } else {
        window.close();
    }
}
```

process is the object model created in **Step 3**, and **approve** is the script model created in **Step 4**. ***Namespace*__bpmId__CST** is the field added in **Step 1: Create an Approval Process Object**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

11. Select the **Refuse** button again. In the **Properties** area, click **+** next to **Value Binding** to bind the **approverFlag** model defined in **Step 7** to the widget.

    Figure 5-78 Binding the approverFlag model to the Refuse button widget

12. Select the **Manager Pass** button. On the **Events** tab page, click **+** next to **on-click** to add a custom action for the button.

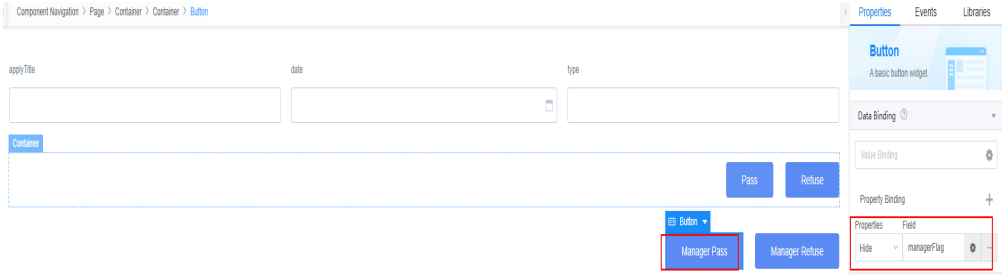
```
var dataTask = $model.ref('getTask').getData();
dataTask.inputParam.interviewID= context.$page.params.interviewID;
$model.ref('getTask').run().then(function(data){
    context.$message.success('Submitted successfully.');
    console.log(data);
    var taskId = data.taskId;
    var _url = `/u-route/baas/bp/v2.0/runtime/tasks/${taskId}`;
    var _inputParam = {action:'Manager approval', variables:{pass:2}};
    var _method = 'PUT';
    var _header = {'Content-Type':'application/json'};
    context.service(_url).run(_inputParam,_method,_header).then(function(response){
        console.log("success");
    });
    closeCurrentTab();
}).catch(function(error){
    console.log('error is', error);
    context.$message.error('Submission failed:' + error.resMsg);
});

function closeCurrentTab() {
    if (parent.bingo) {
        var removeTab = parent.bingo.removeTab;
        var getCurrentTab = parent.bingo.getCurrentTab;
        if (removeTab && getCurrentTab) {
            return removeTab(getCurrentTab());
        }
    }
    if (window.parent != window) {
        window.parent.close();
    } else {
        window.close();
    }
}
```

    getTask indicates the script model created in **Step 5**.

13. Select the **Manager Pass** button again. In the **Properties** area, click **+** next to **Value Binding** to bind the custom model **managerFlag** created in **Step 7** to the widget.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-79 Selecting the managerFlag model for the Manager Pass button



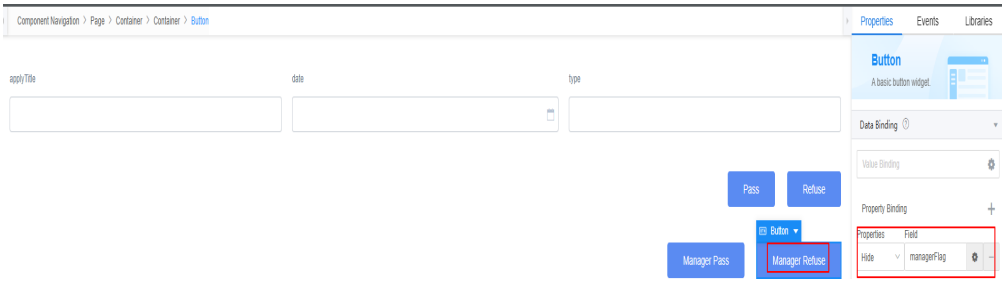
14. Select the **Manager Refuse** button. On the **Events** tab page, click **+** next to **on-click** to add a custom action for the button.

```
var dataTask = $model.ref('getTask').getData();
dataTask.inputParam.interviewID= context.$page.params.interviewID;
$model.ref('getTask').run().then(function(data){
    context.$message.success('Submitted successfully.');
    console.log(data);
    var taskId = data.taskId;
    var _url = `/u-route/baas/bp/v2.0/runtime/tasks/${taskId}`;
    var _inputParam = {action:'Manager approval', variables:{pass:0}};
    var _method = 'PUT';
    var _header = {'Content-Type':'application/json'};
    context.service(_url).run(_inputParam,_method,_header).then(function(response){
        console.log("success");
    });
    closeCurrentTab();
}).catch(function(error){
    console.log('error is', error);
    context.$message.error('Submission failed:' + error.resMsg);
});

function closeCurrentTab() {
    if (parent.bingo) {
        var removeTab = parent.bingo.removeTab;
        var getCurrentTab = parent.bingo.getCurrentTab;
        if (removeTab && getCurrentTab) {
            return removeTab(getCurrentTab());
        }
    }
    if (window.parent != window) {
        window.parent.close();
    } else {
        window.close();
    }
}
```

getTask indicates the script model created in **Step 5**.

15. Select the **Manager Refuse** button again. In the **Properties** area, click **+** next to **Value Binding** to bind the custom model **managerFlag** created in **Step 7** to the widget.

Figure 5-80 Selecting the managerFlag model for the Manager Refuse button



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 9 Click in the upper part of the page to save the page settings.

----End

Step 6: Create an Approval Workflow

Create an approval workflow and configure the user activity's approval and termination actions to implement priority approval.

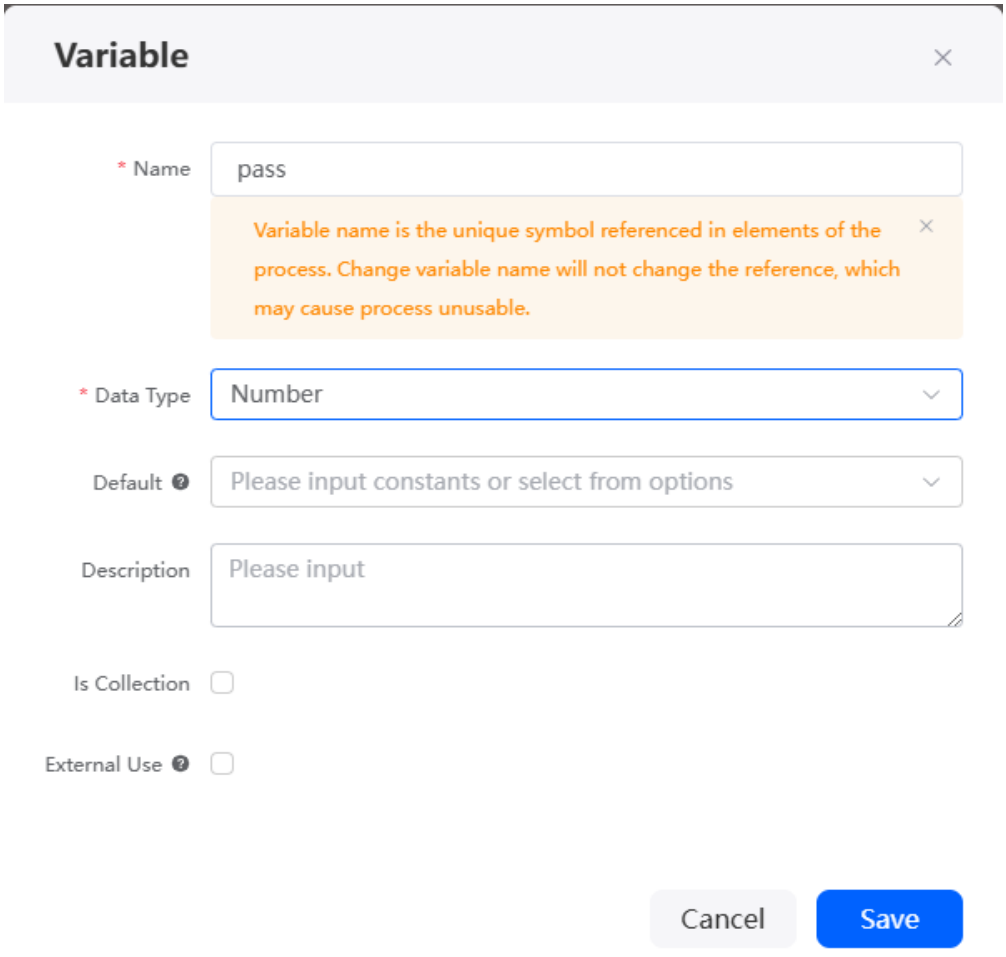
Step 1 In the navigation pane of the application designer, choose **BPMs** and click **+** next to **BPM**.

Step 2 Enter a label and name (for example, **bpm1**) and click **Add**.

Step 3 Create variables and object variables.

1. Click . The **Context** page is displayed.
2. Click **+** next to **Variable** to create **variable0**.
3. Click next to **Variable**, click the button to set the name to **pass** and the data type to **Number**.

Figure 5-81 Creating a variable pass



4. In **Context**, click **+** next to **Object Variable** to create an object variable.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-82 Creating an object variable data

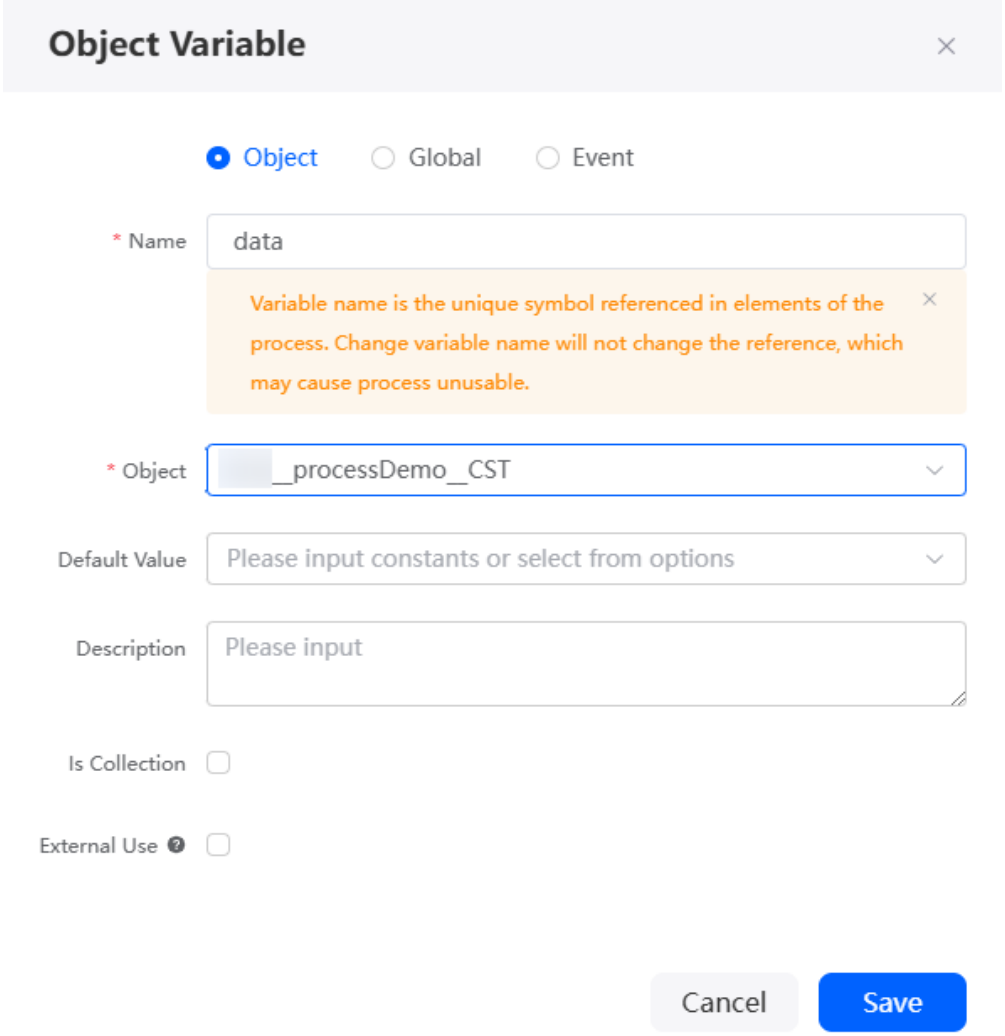
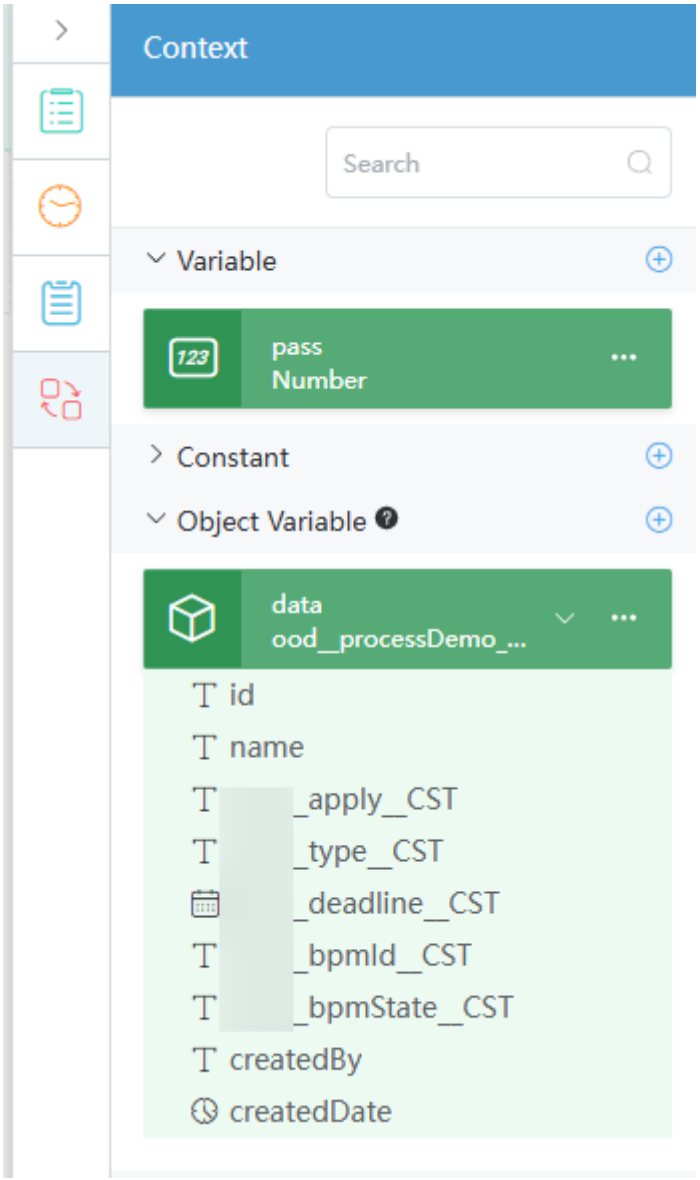


Table 5-7 Parameters for creating an object variable data

Parameter	Description	Example
Name	Name of the new variable. The name is the unique identifier of the variable referenced in a flow. Changing this name does not change its reference in the diagram element, but may cause the flow to be unavailable. Value: 1–80 characters.	data
Object	Select an object from the drop-down list. In this example, select the object created in **Step 1: Create an Approval Process Object**.	***Namespace*__processDems__CST**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-83 Viewing the created variables

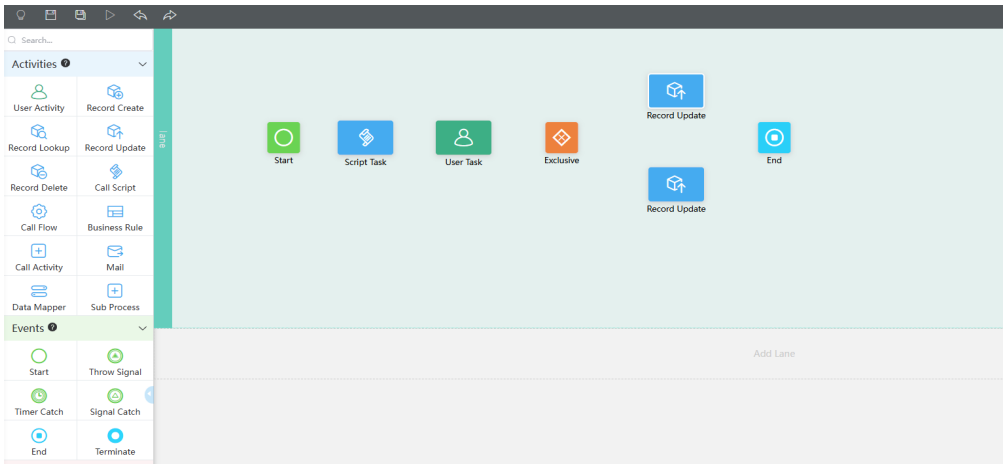


Step 4 Create an approval workflow.

1. Add required diagram elements to the workflow, as shown in **Figure 5-84**.

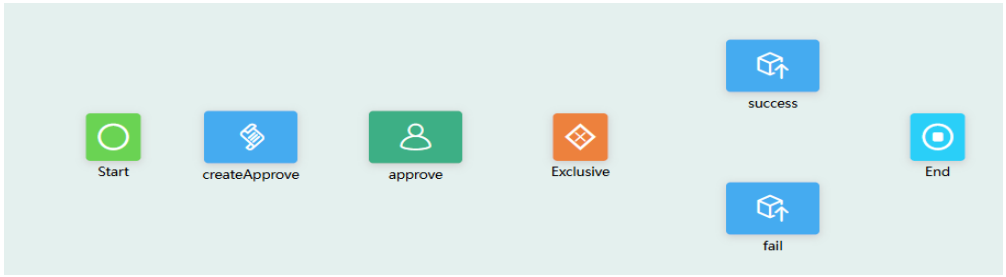
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-84 Layout



2. Select the script, user activity, and record update diagram elements, click , and change the diagram element labels to **createApprove**, **approve**, **success**, and **fail**.

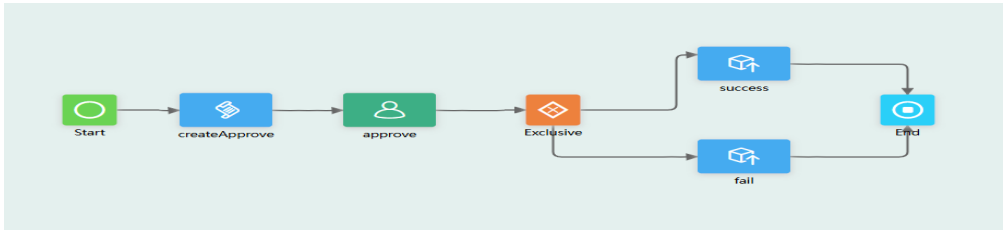
Figure 5-85 Modifying the diagram element labels



3. Connect diagram elements and specify their logical relationships. See **Figure 5-86**.

   Based on the logic implementation, drag and configure all diagram elements and connect them in the correct sequence. When executing a workflow, the system processes these elements in order to complete the entire workflow.

Figure 5-86 Specifying logical relationships

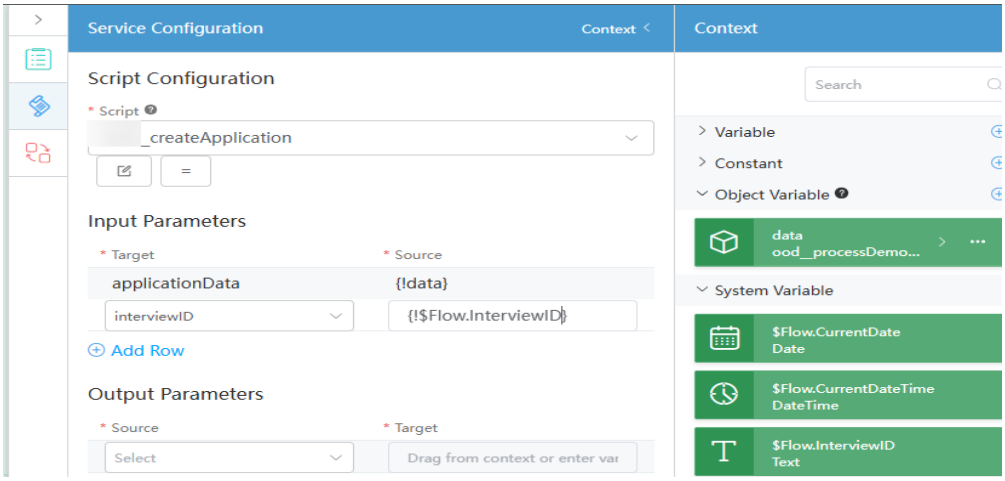


Step 5 Configure diagram elements.

1. Select the script diagram element, click , and set this diagram element based on **Figure 5-87**.

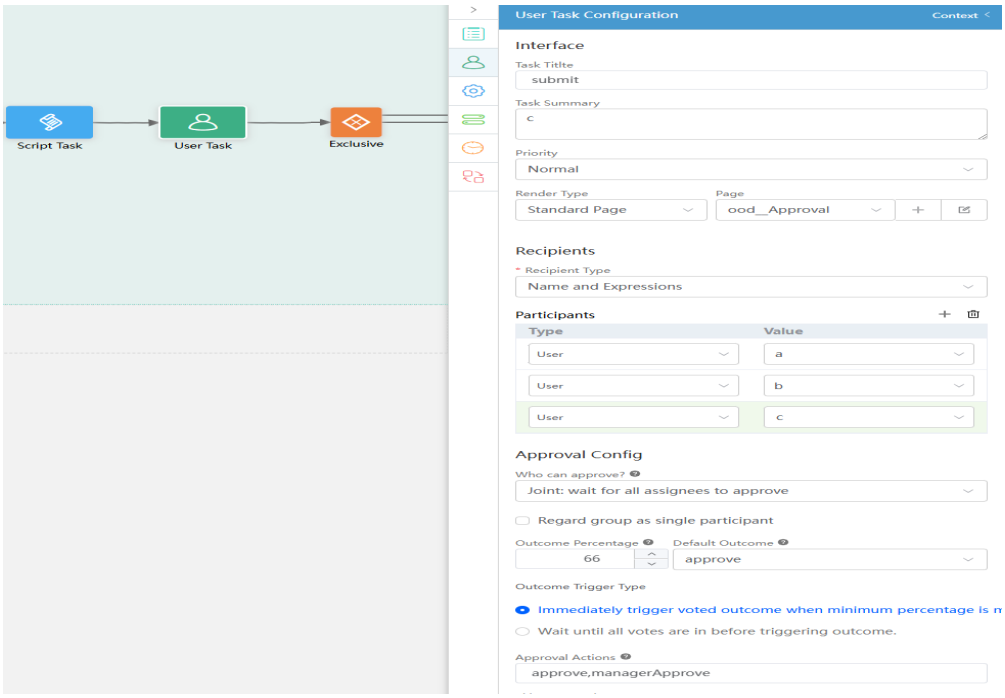
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-87 Setting the script diagram element



2. Select the user activity diagram element, click , and set this diagram element based on **Figure 5-88**.

Figure 5-88 Setting the user activity diagram element



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-89 Approval configuration

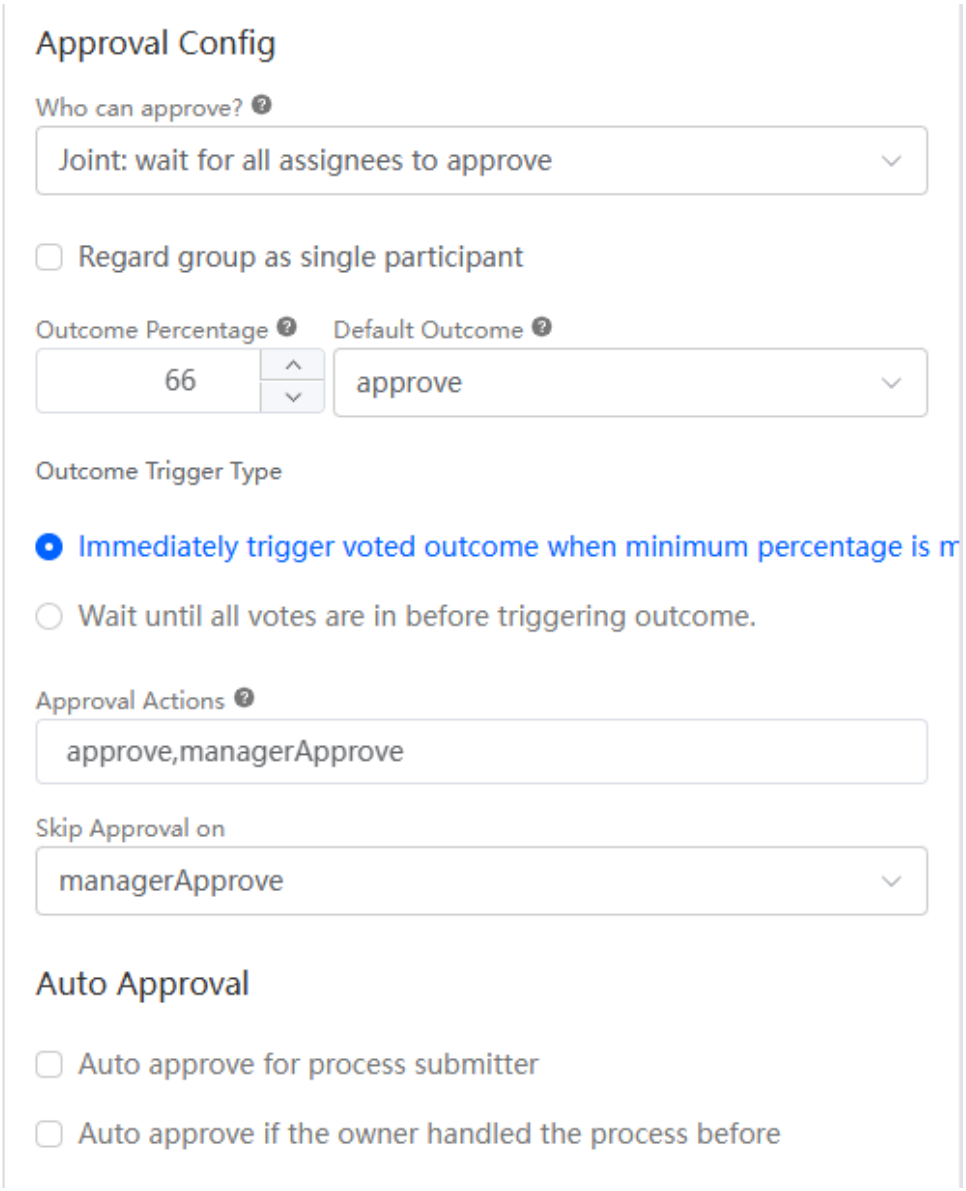


Table 5-8 Parameters for configuring a user activity

Parameter	Description	Example
Interface > Task Title	Title displayed on the task page.	**submit**
Interface > Render Type	Page on which the user processes a task. This parameter can be set to **Standard Page**, **Standard Form**, or **Advance Page**.	**Standard Page**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

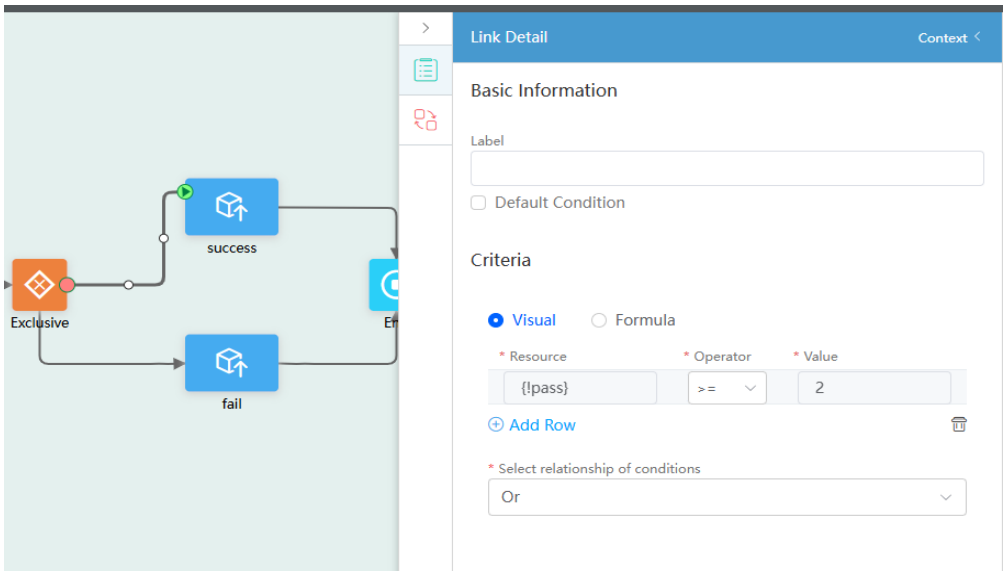
Parameter	Description	Example
Interface > Page	Select a page based on **Render Type**.	Select the ***Namespace*__Approval** page created in **Step 5: Create an Approval Processing Page**.
Recipients > Recipient Type	Type of the user who processes a task.	**Name and Expressions**
Recipients > Participants > Type	This parameter is displayed only when **Recipient Type** is set to **Name and Expressions**. The participant type can be a user, portal user, group, or expression.	**User**
Recipients > Participants > Value	Select the user who receives the task, excluding portal users.	**user_name1** **user_name2** **user_name3**
Approval Config > Who can approve	Select an approver type. – **Joint: wait for all assignees to approve** If any one recipient completes the approval, the task process advances to the next task. – **Single: one of selected members to approve** The task process advances to the next task only if the recipient's user group meets the approval conditions you set.	**Single: one of selected members to approve**
Approval Config > Outcome Percentage	The value of this parameter indicates the percentage. If the percentage is reached, the approval result with the highest vote will overwrite the value of **$BP.TaskOutcome**.	**66**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Approval Config > Default Outcome	Default approval result when the approval percentage (number of approved employees/ total number of valid approvers) does not reach the voting result threshold. This parameter is displayed only when the approval type is **Single: one of selected members to approve**.	**approve**
Approval Config > Approval Actions	Set specific candidate actions as the default approval results.	**approve,managerApprove**
Approval Config > Skip Approval on	When one or more specified actions exist, the task is terminated immediately.	managerApprove

3. Select the connection between the **Exclusive** gateway and **success** diagram element and set the parameters as follows:

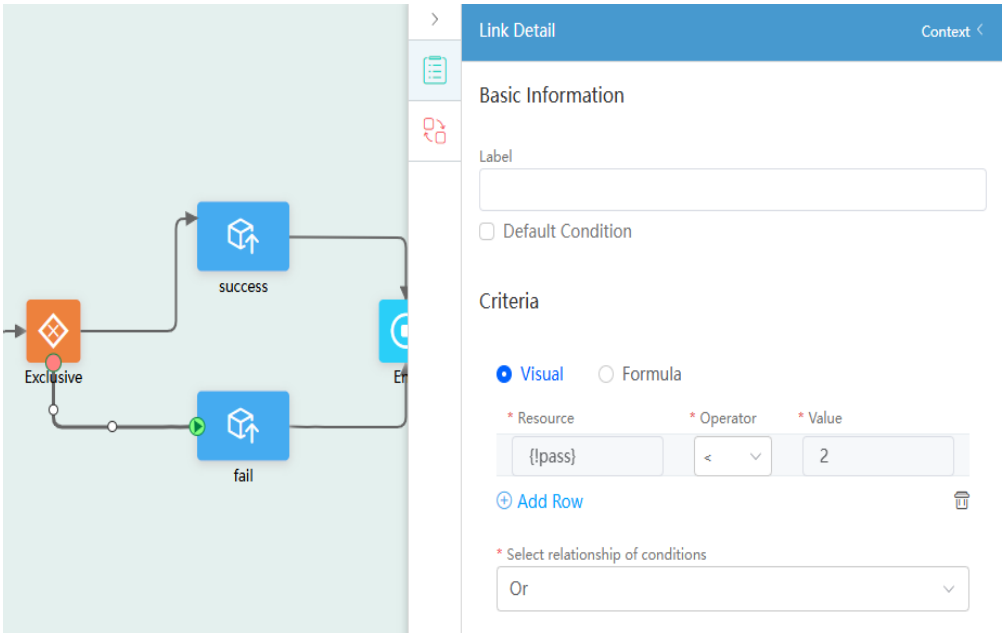
Figure 5-90 Setting the connection line between the Exclusive gateway and success diagram element



4. Select the connection between the **Exclusive** gateway and **fail** diagram element and set the parameters as follows:

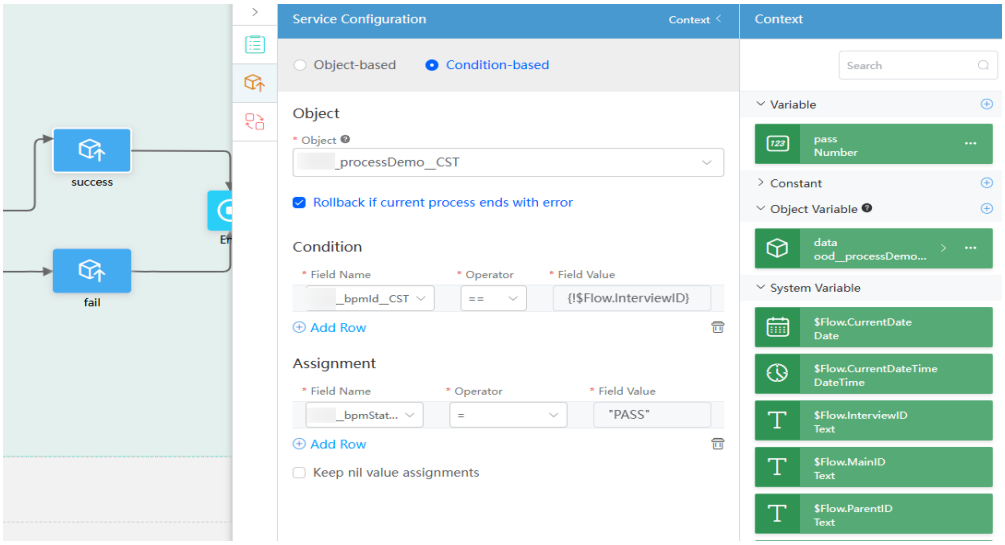
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-91 Setting the connection line between the Exclusive gateway and fail diagram element



5. Select the **success** diagram element and set it based on **Figure 5-92**.

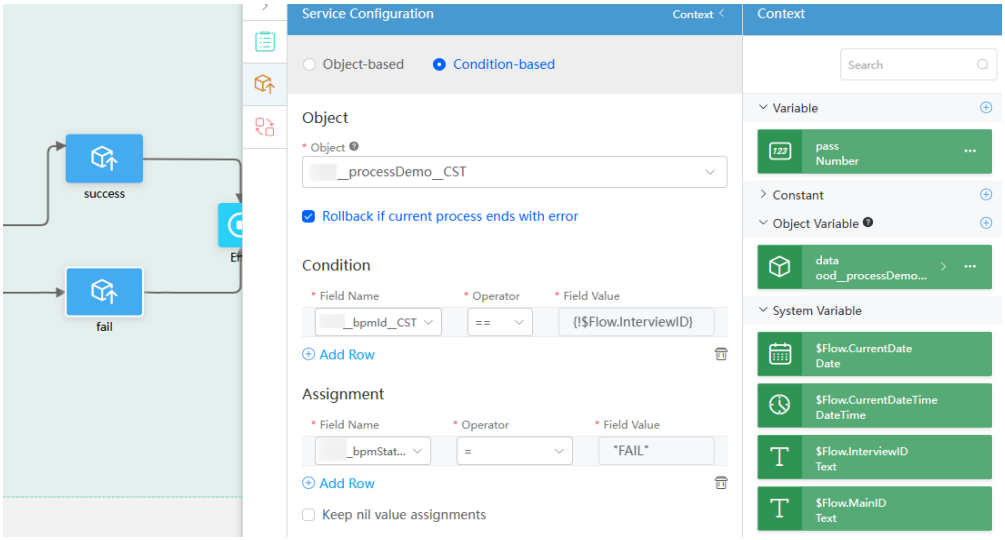
Figure 5-92 Setting the success diagram element



6. Select the **fail** diagram element and set **Figure 5-93**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 5-93 Setting the fail diagram element



Step 6 Click  in the upper part of the page to save the workflow.

Step 7 Click  to activate the workflow.

----End

Step 7: Configure and Verify the Application Menus

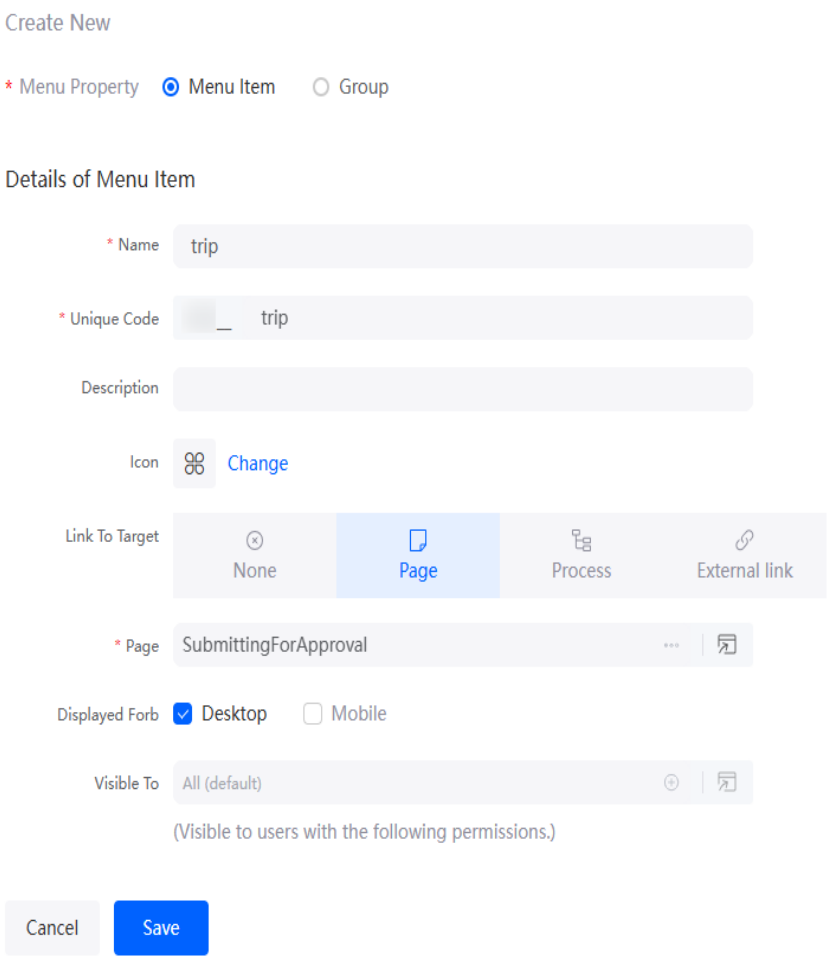
Add the **trip** and **PendingTask** menus for the application and check whether the workflow is executed based on the preset process.

Step 1 Set the application navigation bar.

1. On the **Home** page, click **Application Navigation Settings**.
2. On the main navigation settings tab page, click **New** to create a **trip** menu.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

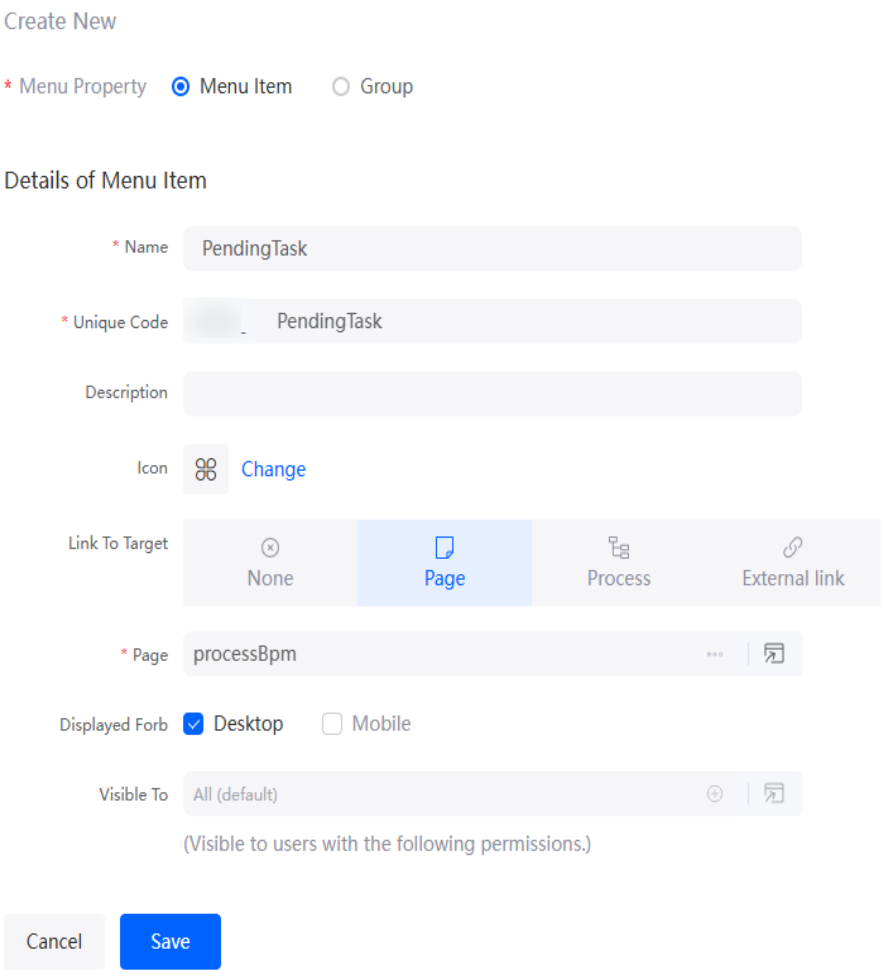
Figure 5-94 Adding the trip menu



3. Repeat the preceding steps to add a **PendingTask** menu.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

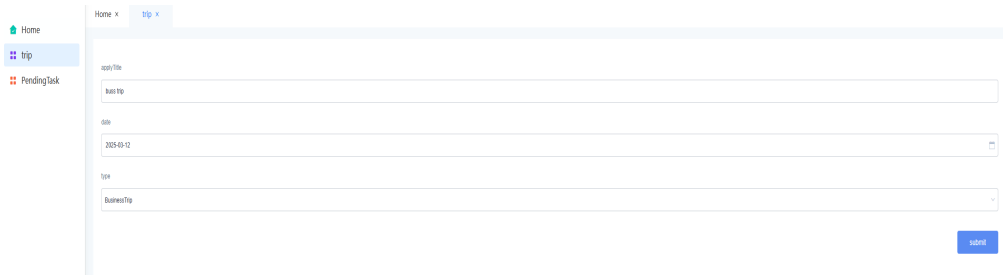
Figure 5-95 Adding the PendingTask menu



Step 2 An employee submits a business trip application.

1. On the main menu of the application, choose **Run** > **Run Now**. The application preview page is displayed.
2. Choose **trip** to submit an application.

Figure 5-96 Submitting an application



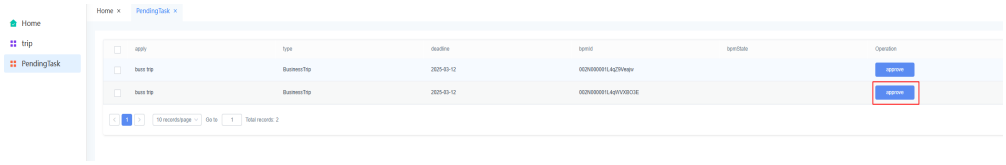
After the submission is successful, the message **Submitted successfully** is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 3 The supervisor approves the application.

1. On the main menu of the application, choose **Run** > **Run Now**. The application preview page is displayed.
2. Choose **PendingTask**. On the displayed page, click **approve** next to the data to be approved.

   Figure 5-97 Clicking approve

3. Approve the application.

   The application submitted in this practice needs to be approved by **user_name1**, **user_name2**, and **user_name3** configured in **Step 5.2**. **user_name3**, as the manager, has approval priority. Once **user_name3** approves the application, **user_name1** and **user_name2** are no longer required to approve it.

   Assume Tom submits a trip application. Normally, **user_name1** approves it first, followed by **user_name2**, and finally by **user_name3** (the manager). Once **user_name3** approves, the entire approval process ends. If **user_name3** (the manager) approves the application after Tom submits it, the approval process ends. **user_name1** and **user_name2** are not required to approve it, as **user_name3** has priority.

   Figure 5-98 Normal approval page

   Figure 5-99 Manager approval page

----End

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

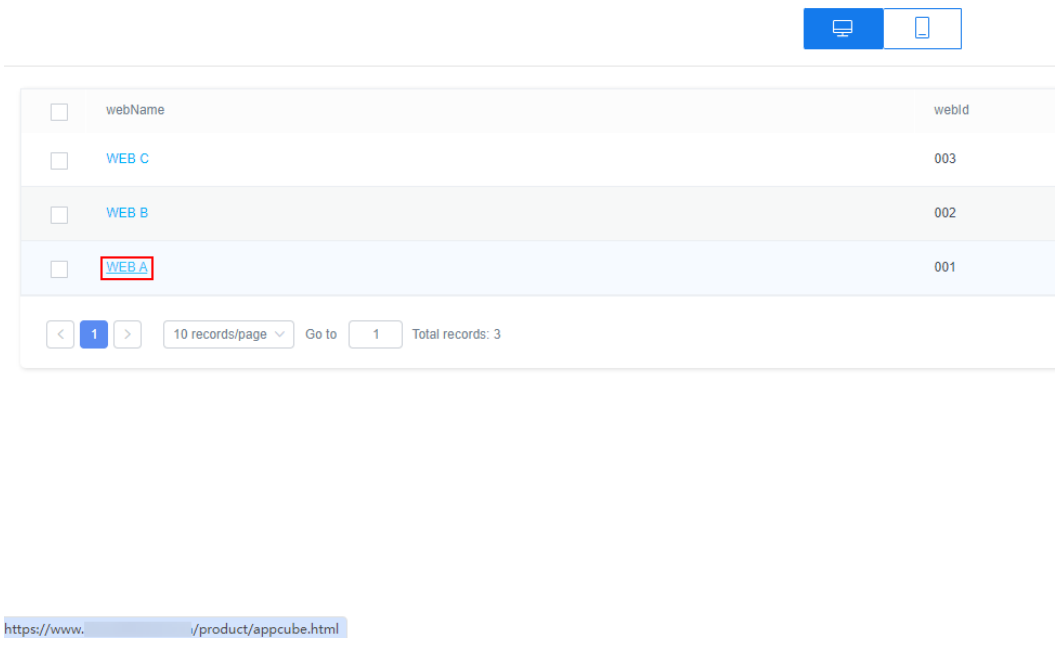
6 Standard Pages

6.1 Adding Links to Data in Standard Page Tables

Expected Results

On a standard page, hyperlinks can be added to data in tables to improve user experience and facilitate data interaction. For example, in the **webName** column of a table, you can move the cursor to **WEB A** to view the corresponding link address in the lower left corner of the page. You can click the link address to see the page.

Figure 6-1 Implementation results



Implementation Methods

Step 1 Create a low-code application.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

Figure 6-2 Creating a blank application

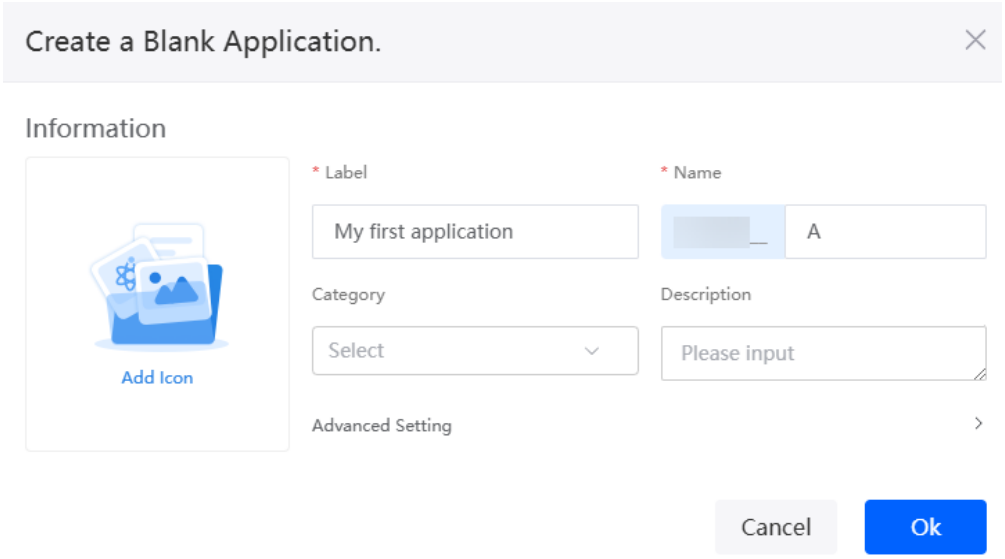


Table 6-1 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 2 Create an object named **websiteList** and add fields to the object.

1. In the navigation pane, choose **Data**, and click **+** next to **Object**.
2. Set **Object Name** and **Unique ID** of the object to **websiteList** and click the confirm button.

Figure 6-3 Creating an object named websiteList

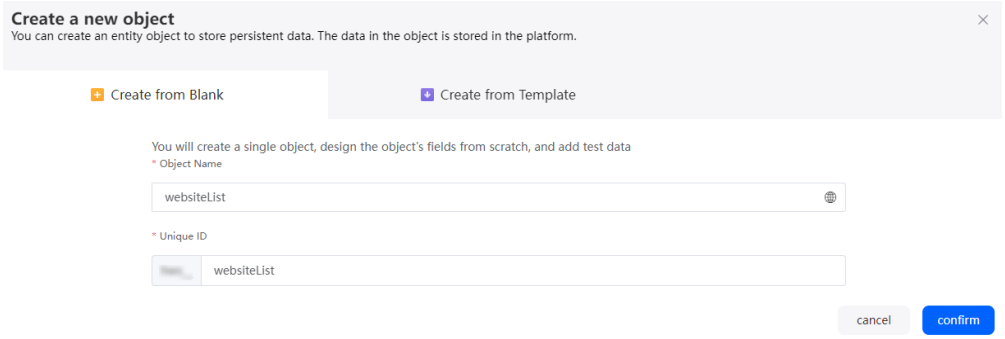


Table 6-2 Parameters for the created object websiteList

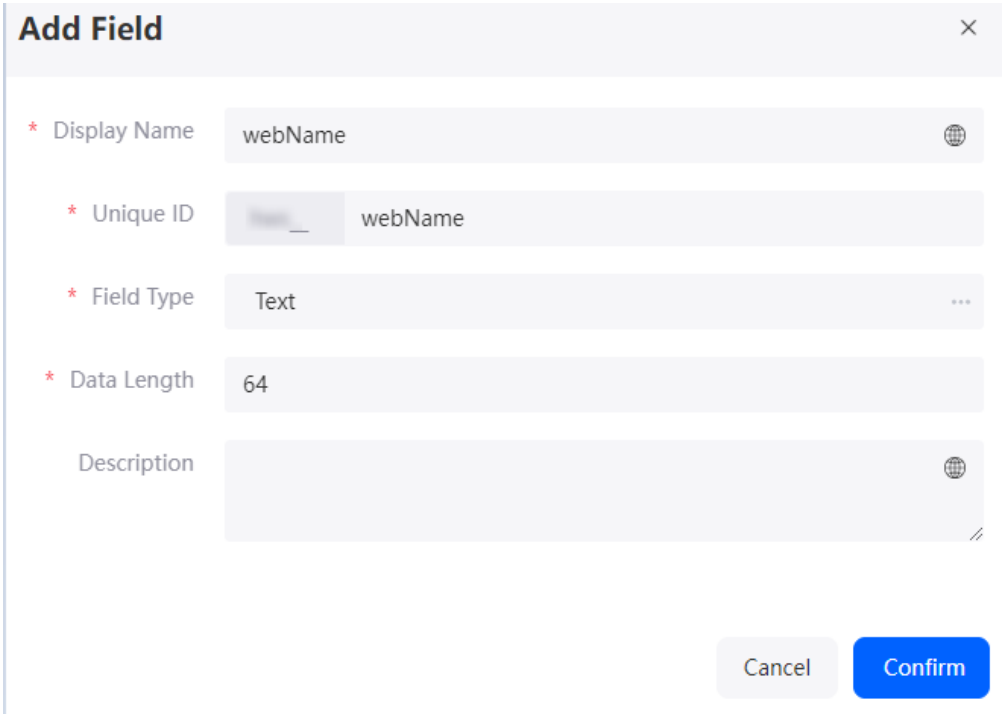
Parameter	Description	Example
Object Name	Name of the new object, which can be changed after the object is created. Value: 1–80 characters.	websiteList

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Unique ID	ID of a new object in the system, which cannot be modified after being created. Naming rules: • Max. 63 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. • Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	websiteList

3. Click  to go to the object details page.
4. On the **Fields** tab page, click **Add** and add the **webName** field to the object.

Figure 6-4 Adding the webName field



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-3 Parameters for adding the webName field

Parameter	Description	Example
Display Name	Name of the new field, which can be changed after the field is created. Value: 1–63 characters.	webName
Unique ID	ID of a new field in the system. The value cannot be changed after the field is created. Naming rules: – Max. 63 characters, including the prefix namespace. – Start with a letter and use only letters, digits, and an underscore (_). Do not end with an underscore (_).	webName
Field Type	Click ***. On the page that is displayed, select the type of the new field based on the parameter description.	Text

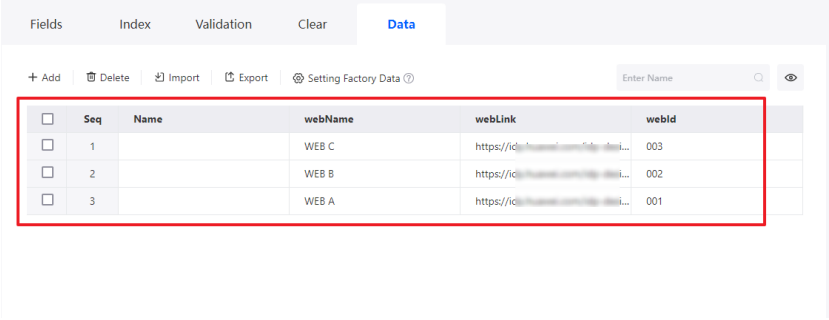
5. Repeat the preceding operations to add the **webLink** and **webId** fields to the object.

Table 6-4 Adding the webLink and webId fields

Display Name	Unique ID	Field Type
webLink	webLink	Text area
webId	webId	Text

6. On the **Data** tab, click **Add** to add data in **Figure 6-5** to the object.

Figure 6-5 Adding data to an object

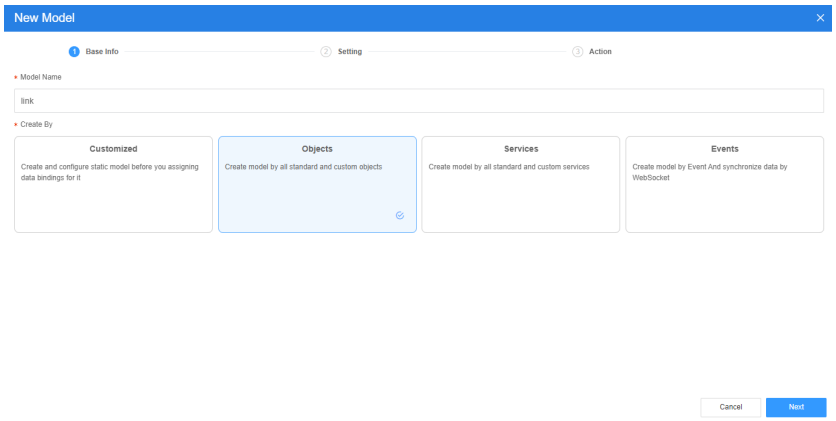


Step 3 Create an object model.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

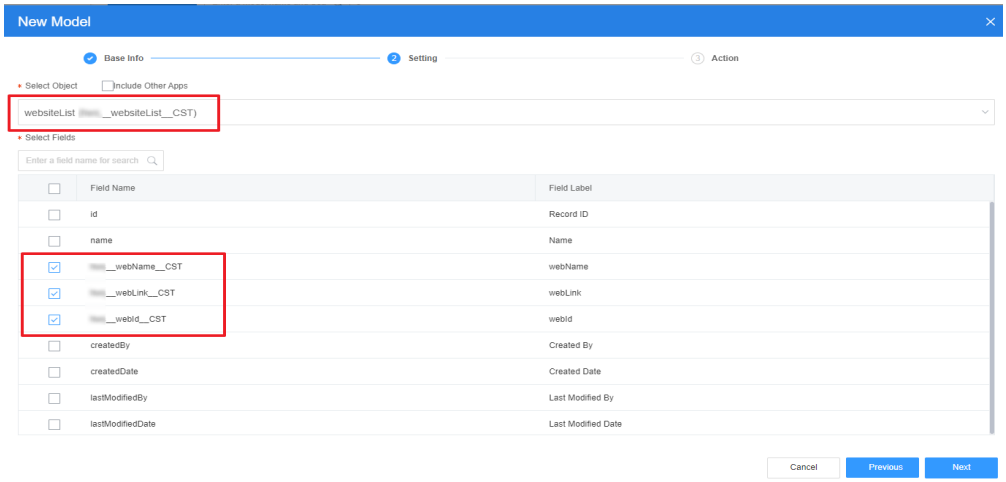
1. In the navigation pane, choose **Page**, and click **+** next to **Standard Page**.
2. Enter the label and name of the page and click **Add** to create a standard page.
3. At the bottom of the standard page, click **Model View**.
4. Click **New**, specify **Model Name** (for example, **link**), select **Objects** for **Source**, and click **Next**.

Figure 6-6 Creating a model



5. Select the object created and the fields added in **Step 3**, click **Next**, and click **OK**.

Figure 6-7 Selecting objects and fields

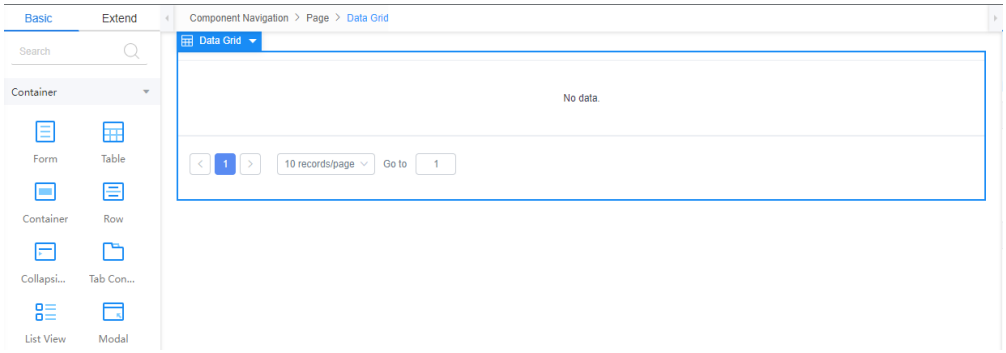


Step 4 Return to the **Designer View** page and bind the model with the widgets.

1. Drag a table widget to the standard page.

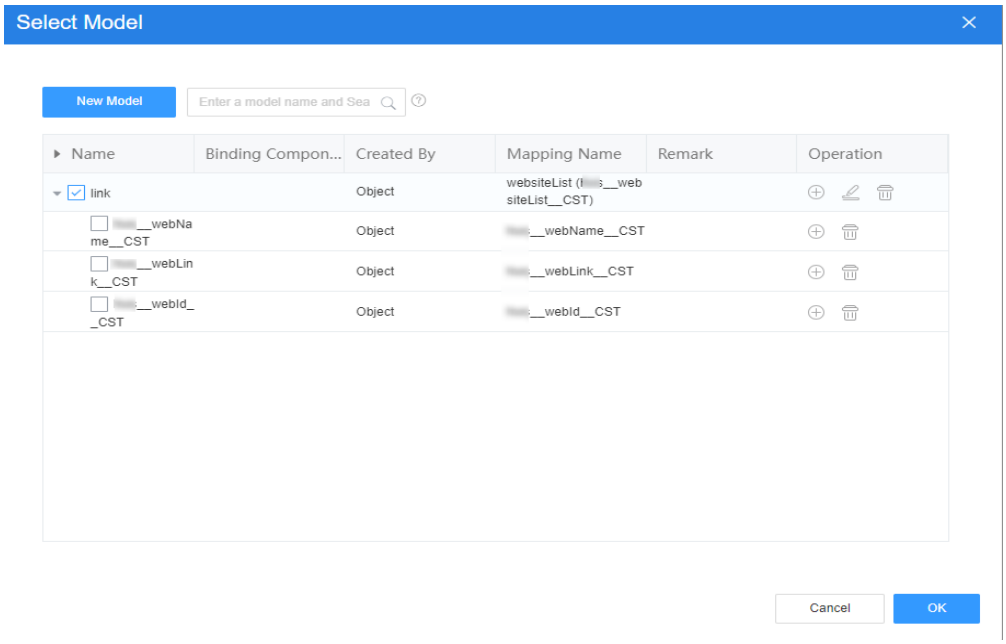
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-8 Dragging a table widget



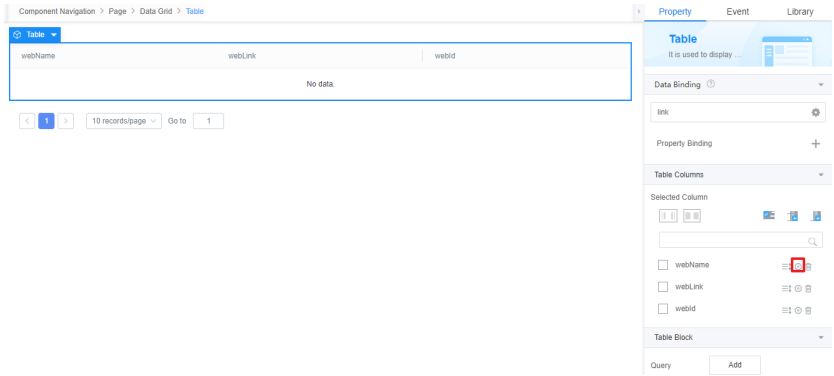
2. Select the table widget, choose **Properties** > **Data Binding** and click next to **Value Binding**.
3. Select the model created in **Step 3** and click the confirm button.

Figure 6-9 Selecting the model



Step 5 In **Table Columns**, click next to the **webName** column.

Figure 6-10 Selecting the required column

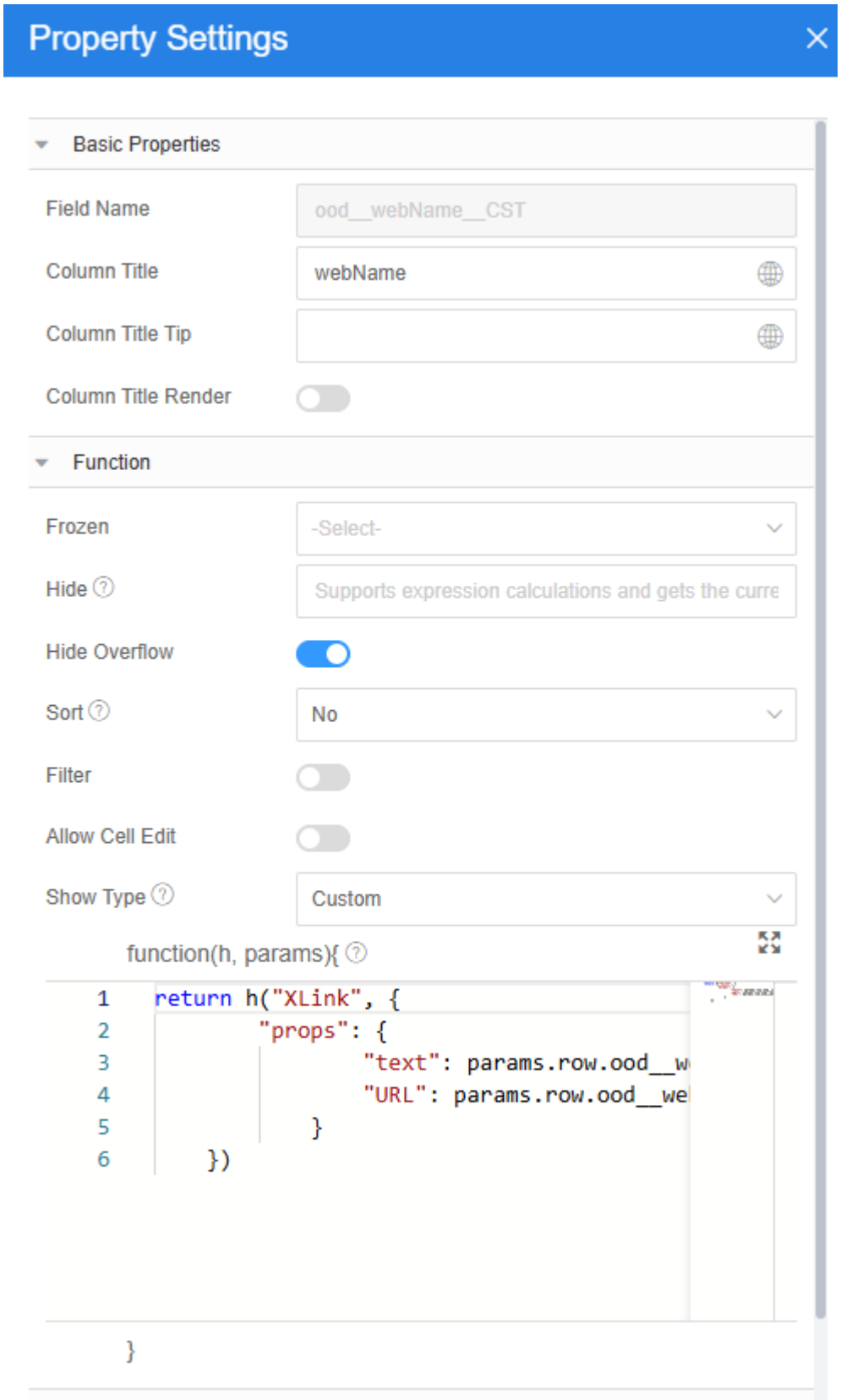


 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 6 In **Property Settings** > **Function**, add the URL information for the **webName** column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-11 Setting column properties



Set **Show Type** to **Custom** and enter the following information.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
return h("XLink", {
      "props": {
            "text": params.row.Namespace__webName__CST,
            "URL": params.row.Namespace__webLink__CST
        }
  })
```

Step 7 Click in the upper part of the page to save the page settings.

Step 8 After the settings are saved, click in the upper part of the page to preview.

----End

6.2 Adding Calculating Capabilities, Such as Summation, to Standard Page Tables

Expected Results

On a standard page, calculation capabilities such as summation and multiplication can be added to tables to improve data processing efficiency. For example, set the value of the **productCost** column in a table to *productNum* x *productPrice* + *cost*.

Figure 6-12 Implementation results

Procedure

Step 1 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-13 Creating a blank application

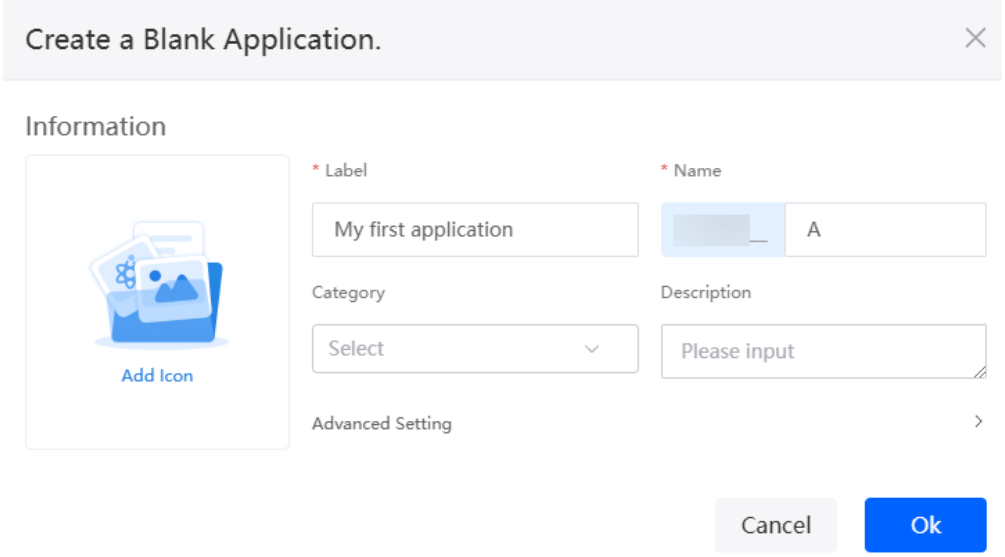


Table 6-5 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 2 Create an object named **ProductList** and add fields and data to the object.

1. In the navigation pane, choose **Data**, and click **+** next to **Object**.
2. Set **Object Name** and **Unique ID** of the object to **ProductList** and click the confirm button.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-14 Creating the object ProductList

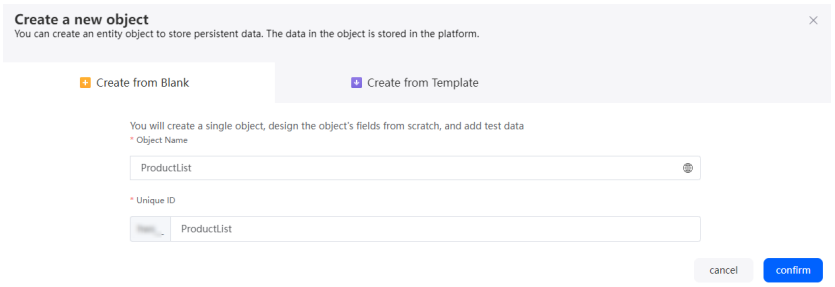


Table 6-6 Parameters for the created object ProductList

Parameter	Description	Example
Object Name	Name of the new object, which can be changed after the object is created. Value: 1–80 characters.	ProductList
Unique ID	ID of a new object in the system, which cannot be modified after being created. Naming rules: • Max. 63 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. • Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	ProductList

3. Click  to go to the object details page.
4. On the **Fields** tab page, click **Add** and add the **productName** field to the object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-15 Adding the productName field

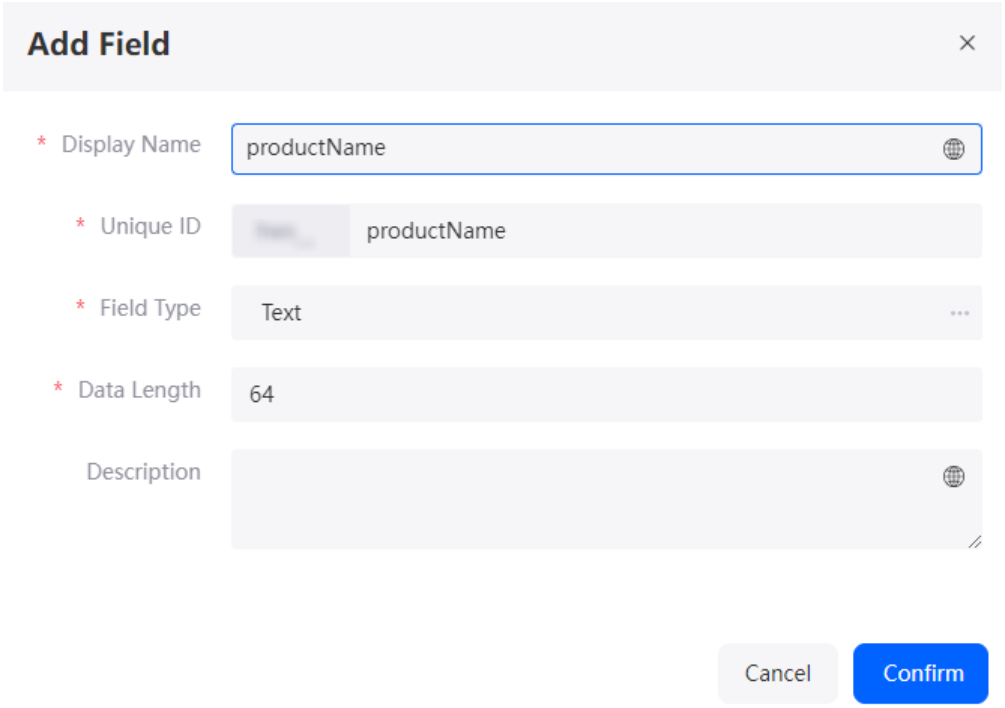


Table 6-7 Parameters for adding the cost field

Parameter	Description	Example
Display Name	Name of the new field, which can be changed after the field is created. Value: 1–63 characters.	Product name
Unique ID	ID of a new field in the system. The value cannot be changed after the field is created. Naming rules: – Max. 63 characters, including the prefix namespace. – Start with a letter and use only letters, digits, and an underscore (_). Do not end with an underscore (_).	productName
Field Type	Click ***. On the page that is displayed, select the type of the new field based on the parameter description.	Text

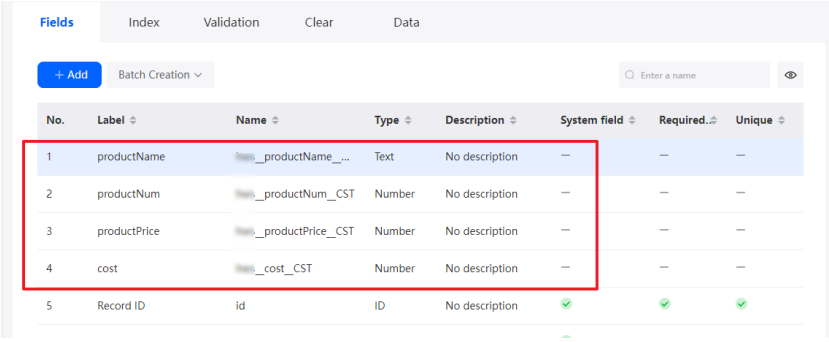
5. Repeat the preceding operations to add the fields in **Table 6-8** to the object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-8 Adding other fields

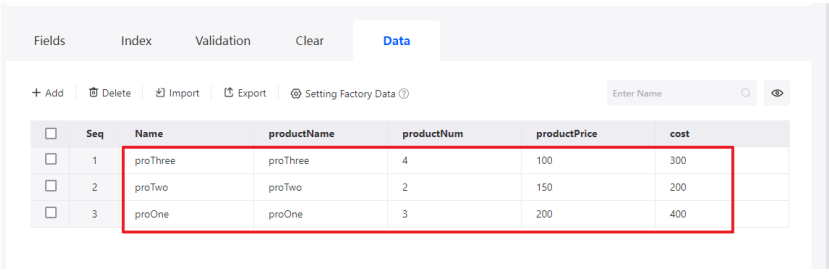
Display Name	Unique ID	Field Type
Product quantity	productNum	Number
Product price	productPrice	Number
Other costs	cost	Number

Figure 6-16 Viewing the added fields



6. On the **Data** tab, click **Add** to add data in **Figure 6-17** to the object.

Figure 6-17 Adding data to an object

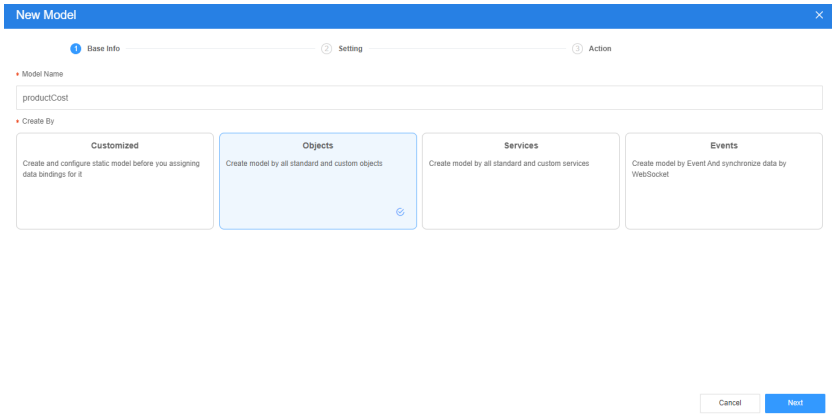


Step 3 Create an object model.

1. In the navigation pane, choose **Page**, and click **+** next to **Standard Page**.
2. Enter the label and name of the page and click **Add** to create a standard page.
3. At the bottom of the standard page, click **Model View**.
4. Click **New**, specify **Model Name** (for example, **productCost**), select **Objects** for **Source**, and click **Next**.

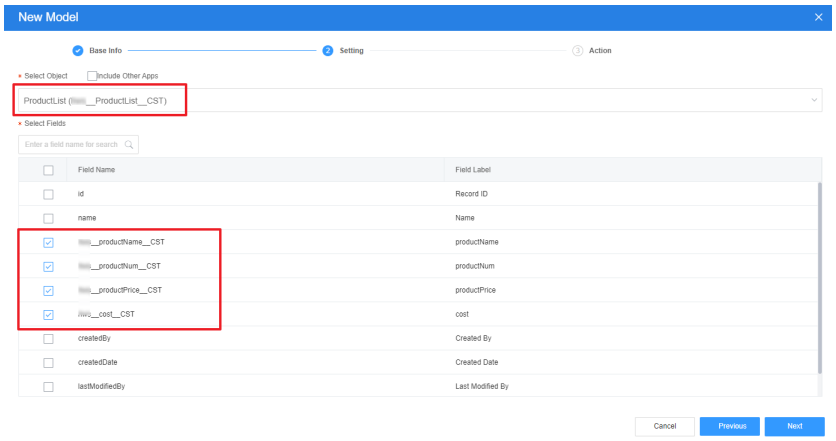
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-18 Creating a model



5. Select the object created and the fields added in **Step 3**, click **Next**, and click **OK**.

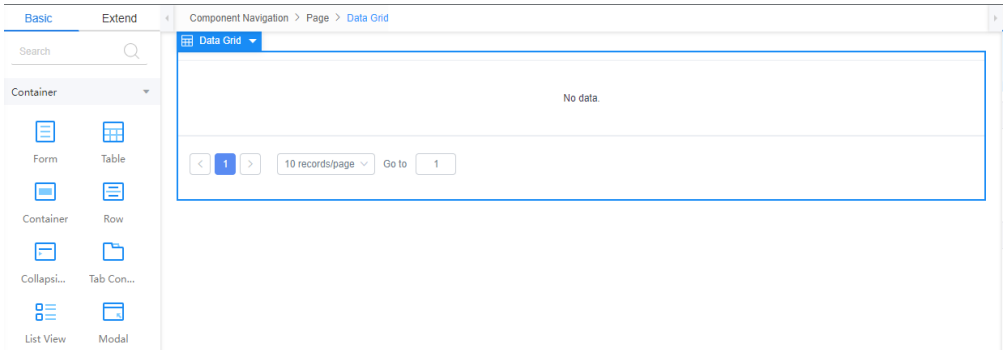
Figure 6-19 Selecting objects and fields



Step 4 Return to the **Designer View** page and bind the model with the widgets.

1. Drag a table widget to the standard page.

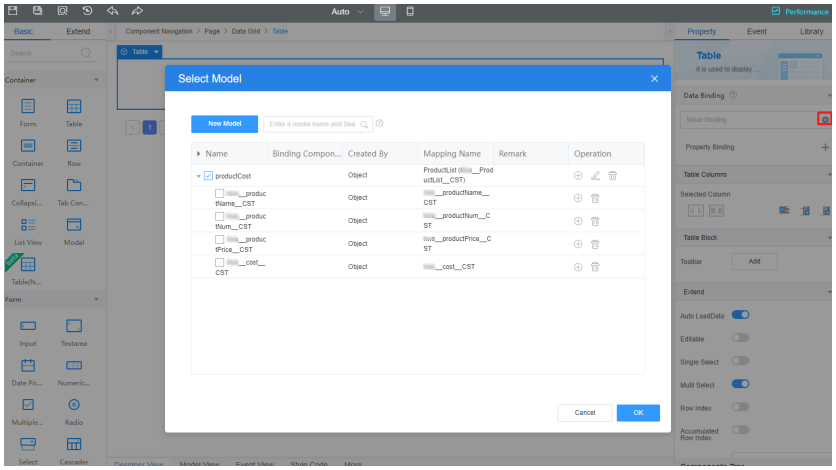
Figure 6-20 Dragging a table widget



2. Select the table widget, choose **Properties** > **Data Binding** and click next to **Value Binding**.
3. Select the model created in **Step 3** and click the confirm button.

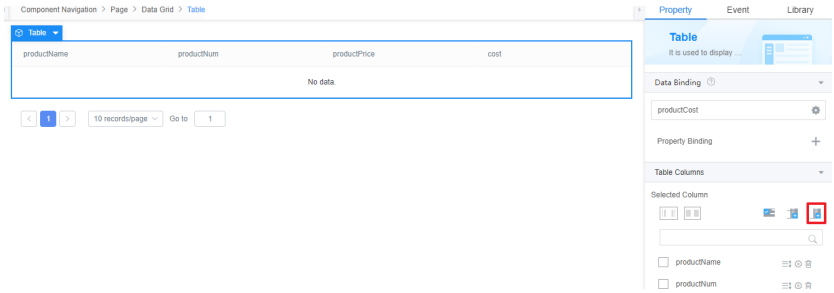
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-21 Selecting the model



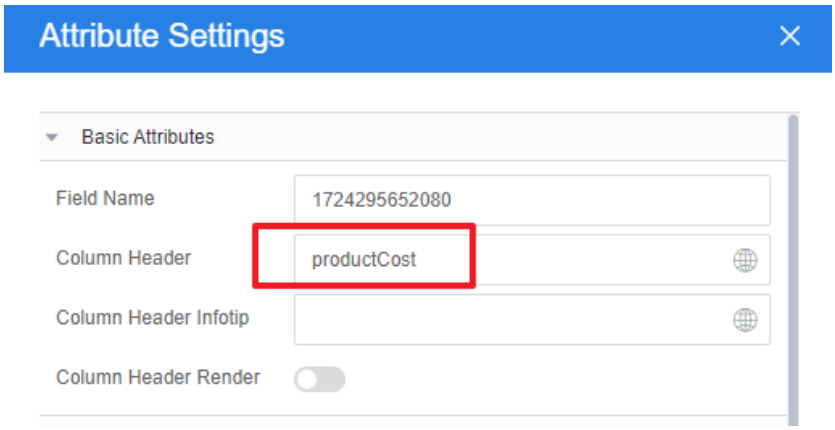
Step 5 Select a table, choose **Properties** > **Table Columns** > **Added Column**, and click  to add a blank column.

Figure 6-22 Adding a blank column



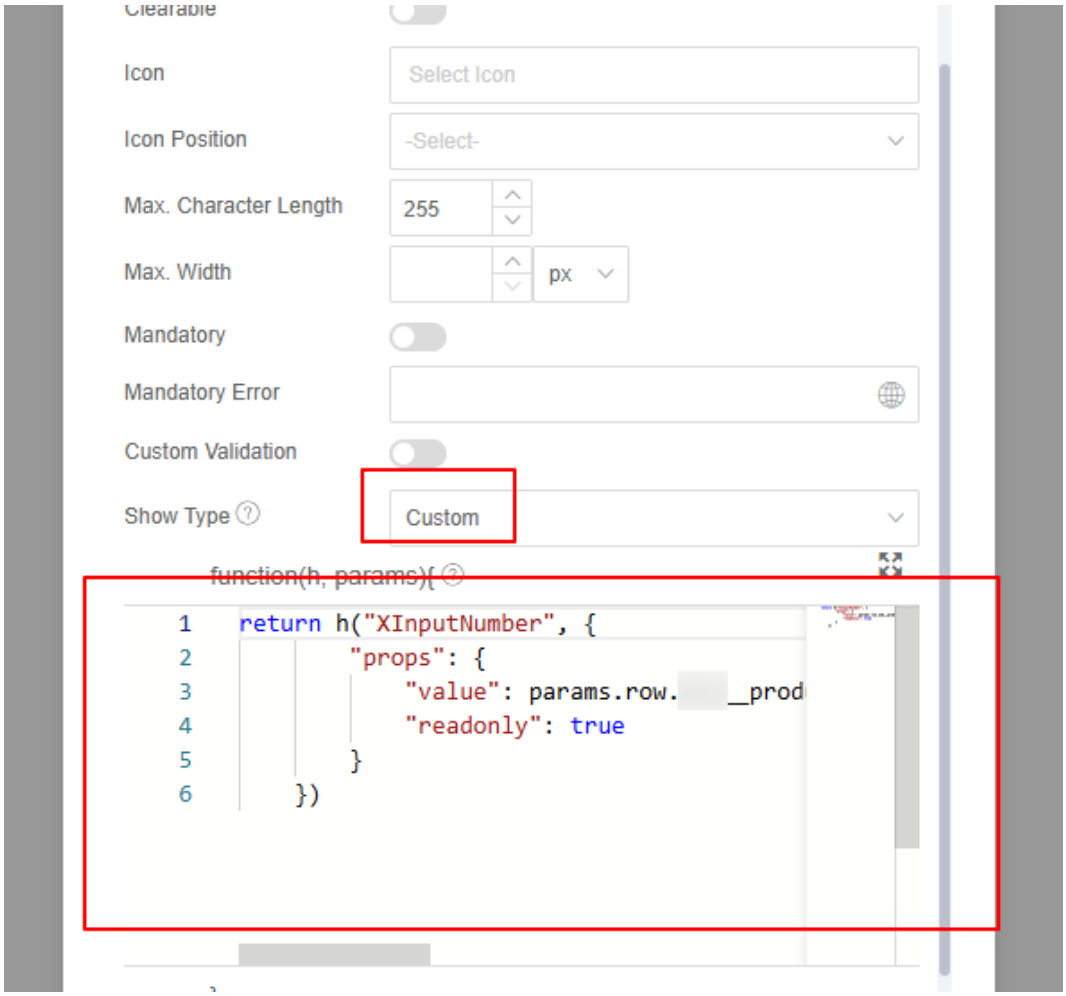
Step 6 Click  next to the new blank column, change the value of **Column Title** to **productCost**, set **Show Type** to **Custom**, and enter the custom code.

Figure 6-23 Changing the column title to productCost



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-24 Setting Show Type to Custom and entering the custom code



The custom code is as follows:

```
return h("XInputNumber", {
        "props": {
            "value": params.row.Namespace__productPrice__CST * params.row.Namespace__productNum__CST
+ params.row.Namespace__cost__CST,
            "readonly": true
        }
    })
```

Step 7 Click in the upper part of the page to save the page settings.

Step 8 After the settings are saved, click in the upper part of the page to preview.

----End

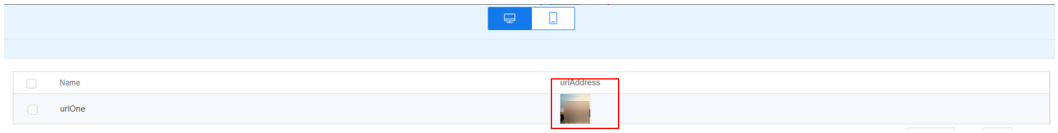
6.3 Displaying Images in Standard Page Tables

Expected Results

On a standard page, you can customize the display type of a column to display images in a table. Displaying images in a table makes information more intuitive and easier to understand.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-25 Implementation results



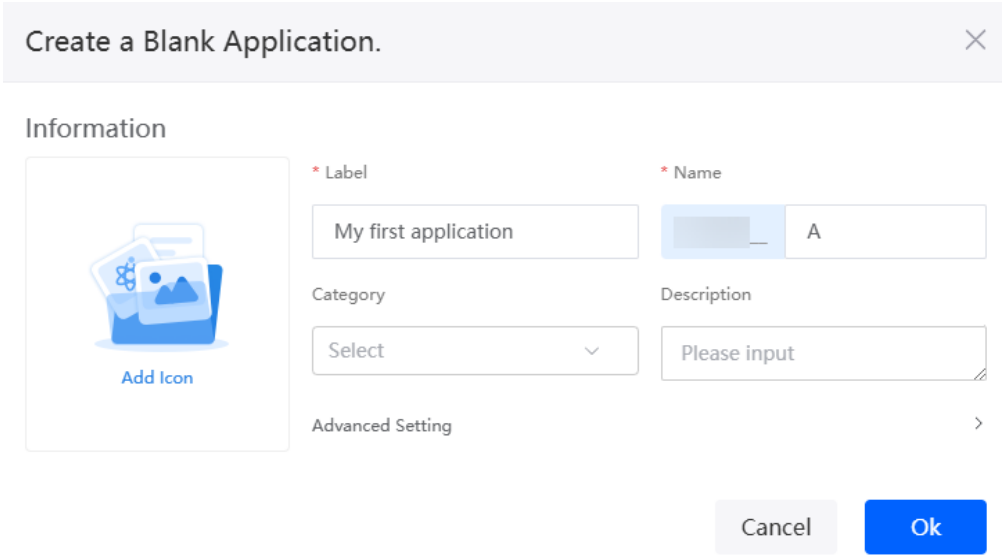
Implementation Methods

Step 1 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

Figure 6-26 Creating a blank application



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

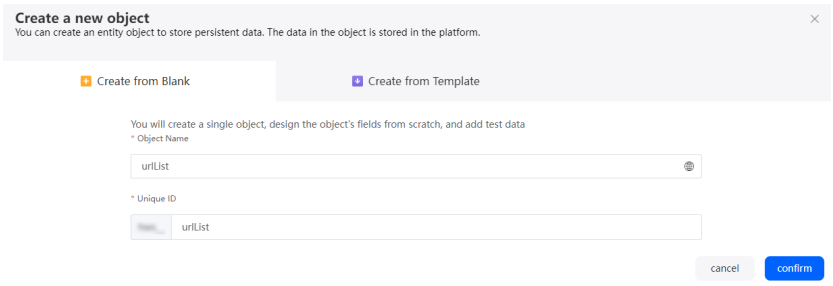
Table 6-9 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 2 Create an object named **urlList** and add fields to the object.

1. In the navigation pane, choose **Data**, and click **+** next to **Object**.
2. Set **Object Name** and **Unique ID** of the object to **urlList** and click the confirm button.

Figure 6-27 Creating the object urlList



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-10 Parameters for the created object urlList

Parameter	Description	Example
Object Name	Name of the new object, which can be changed after the object is created. Value: 1–80 characters.	urlList
Unique ID	ID of a new object in the system, which cannot be modified after being created. Naming rules: • Max. 63 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. • Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	urlList

3. Click  to go to the object details page.
4. On the **Fields** tab page, click **Add** and add the **urlAddress** field to the object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-28 Adding the urlAddress field

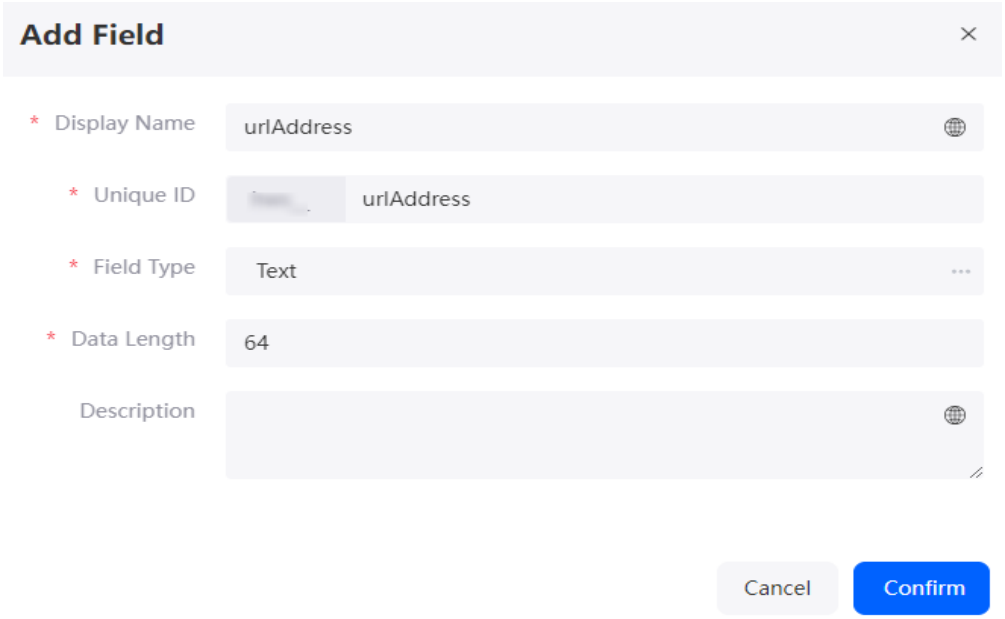


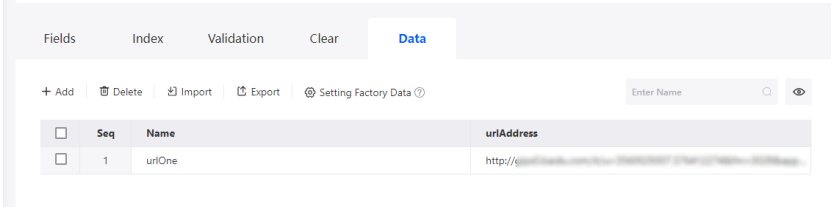
Table 6-11 Parameters for adding the urlAddress field

Parameter	Description	Example
Display Name	Name of the new field, which can be changed after the field is created. Value: 1–63 characters.	urlAddress
Unique ID	ID of a new field in the system. The value cannot be changed after the field is created. Naming rules: – Max. 63 characters, including the prefix namespace. – Start with a letter and use only letters, digits, and an underscore (_). Do not end with an underscore (_).	urlAddress
Field Type	Click ***. On the page that is displayed, select the type of the new field based on the parameter description.	Text area

5. On the **Data** tab, click **Add** to add data to the object.
   Set **urlAddress** to the path for storing images.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

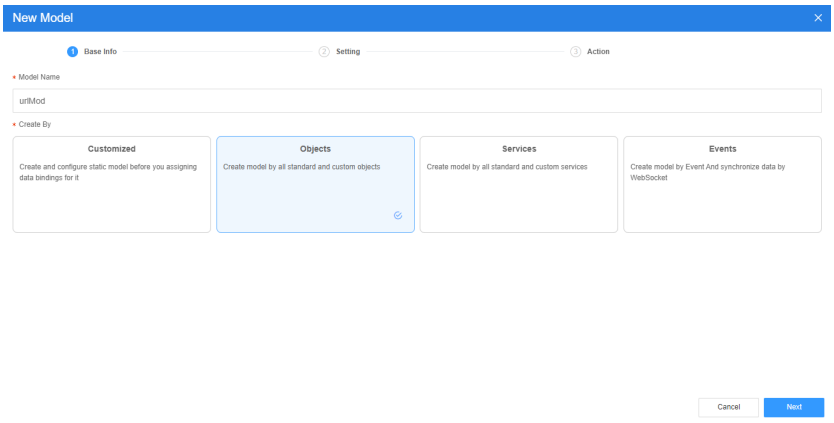
Figure 6-29 Adding data to an object



Step 3 Create an object model.

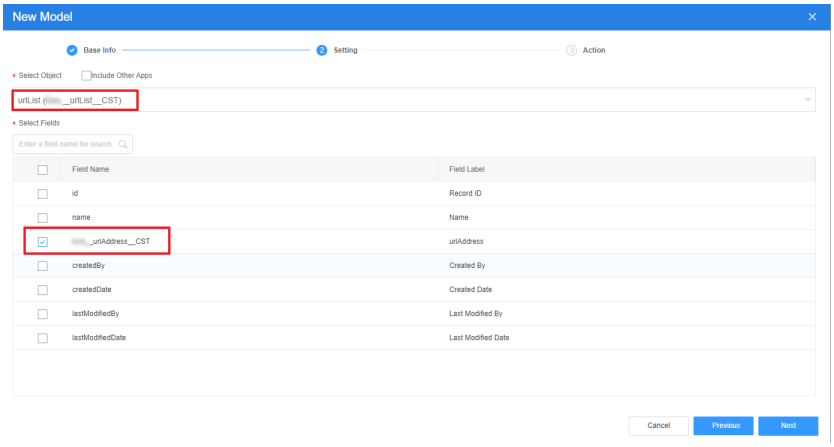
1. In the navigation pane, choose **Page**, and click **+** next to **Standard Page**.
2. At the bottom of the standard page, click **Model View**.
3. Click **New**, specify **Model Name** (for example, **urlMod**), select **Objects** for **Source**, and click **Next**.

Figure 6-30 Creating a model



4. Select the object created and the fields added in **Step 3**, and click **Next**.

Figure 6-31 Selecting objects and fields



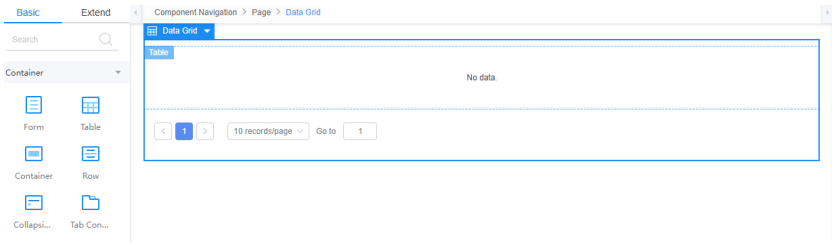
5. Click **OK**.

Step 4 Return to the **Designer View** page and bind the model with the widgets.

1. At the bottom of the standard page, click **Designer View**.
2. Drag a table widget to the standard page.

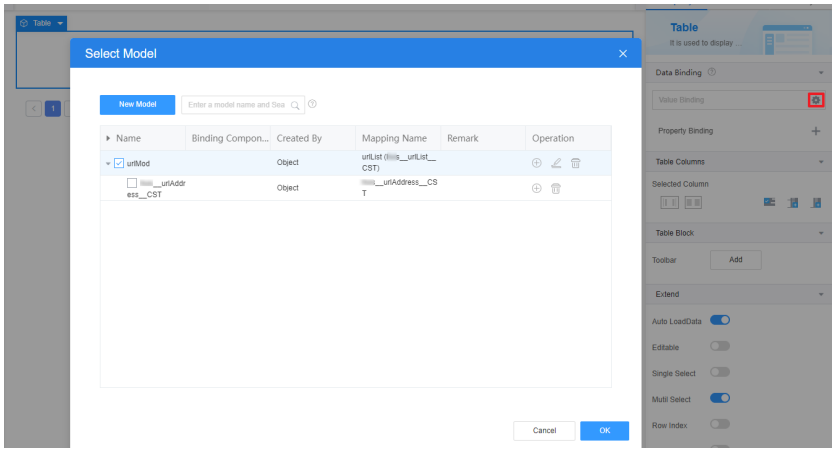
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-32 Dragging a table widget



3. Select the table widget, choose **Properties** > **Data Binding** and click next to **Value Binding**.
4. Select the model created in **Step 3** and click the confirm button.

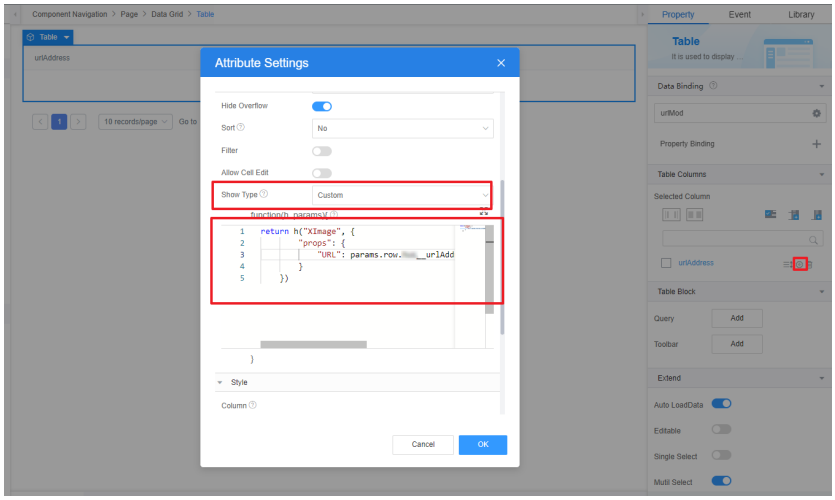
Figure 6-33 Selecting the model



Step 5 Select a table, choose **Properties** > **Table Columns** > **Added Column**, and click next to the **urlAddress** column.

Step 6 Set the properties, and then click **OK** to return to the standard page.

Figure 6-34 Setting properties



Set **Show Type** to **Custom** and enter the following information.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
return h("XImage", {
      "props": {
         "URL": params.row.Namespace__urlAddress__CST, width: 50, height: 50
      }
   })
```

Step 7 Click in the upper part of the page to save the page settings.

Step 8 After the settings are saved, click in the upper part of the page to view the configuration effect.

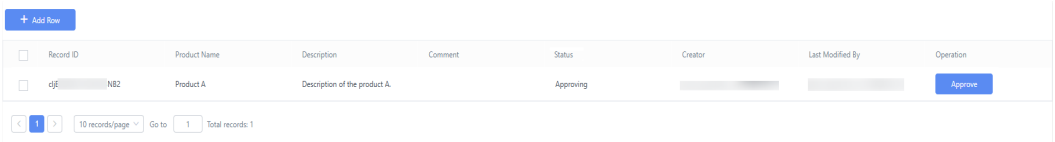
----End

6.4 Reusing a Submitted Form Page on an Approval Page

Expected Results

You can create pages for submitting and approving data in a table for employee trips, leave requests, or goods movement. You can also approve expense requests. If the pages have similar content, you can use the same functions on one page. This saves time and makes things easier.

Figure 6-35 Results



Procedure

Step 1 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-36 Creating a blank application

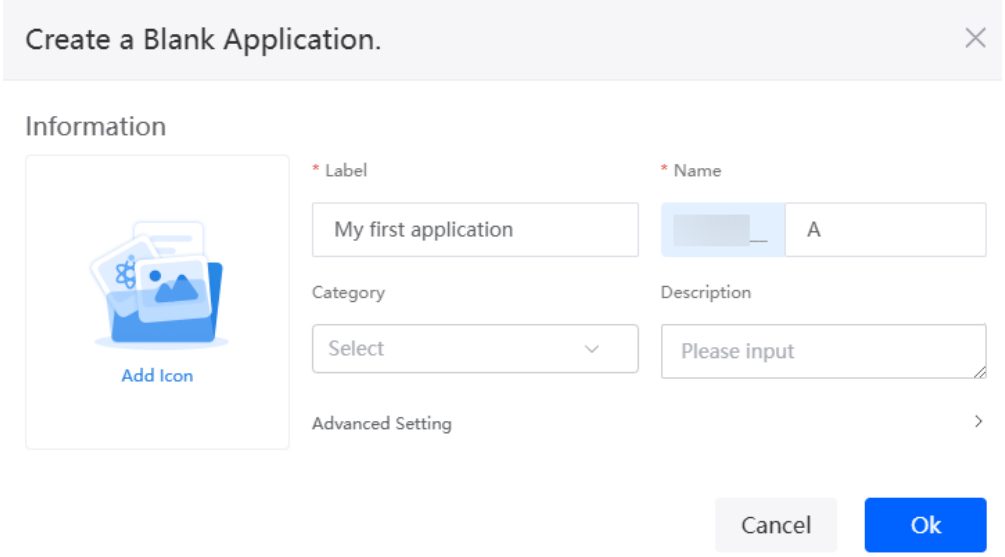


Table 6-12 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 2 Create an object named **productList** and add fields to the object.

1. In the navigation pane, choose **Data**, and click **+** next to **Object**.
2. Set the object name and unique ID to **productList**, and click the confirm button. The object details page is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-37 Creating the object productList

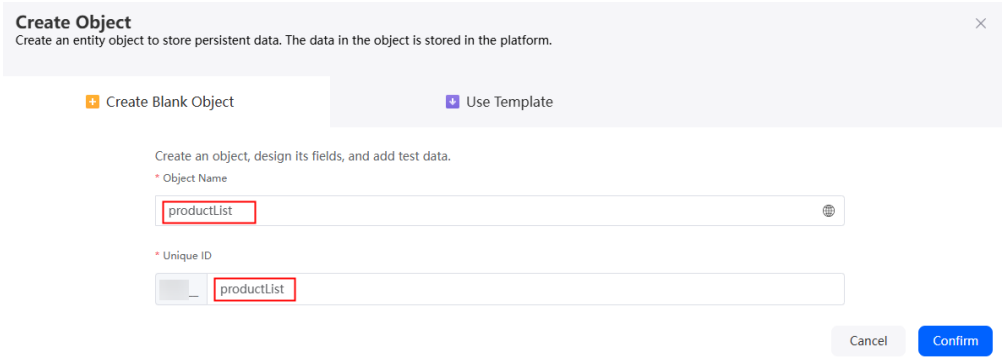


Table 6-13 Parameters for creating productList

Parameter	Description	Example
Object Name	Name of the new object, which can be changed after the object is created. Value: 1–80 characters.	productList
Unique ID	ID of a new object in the system, which cannot be modified after being created. Naming rules: • Max. 63 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. • Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	productList

3. Click  to go to the object details page.
4. On the **Fields** tab, click **Add** and add the **productName** field to the object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-38 Adding the productName field

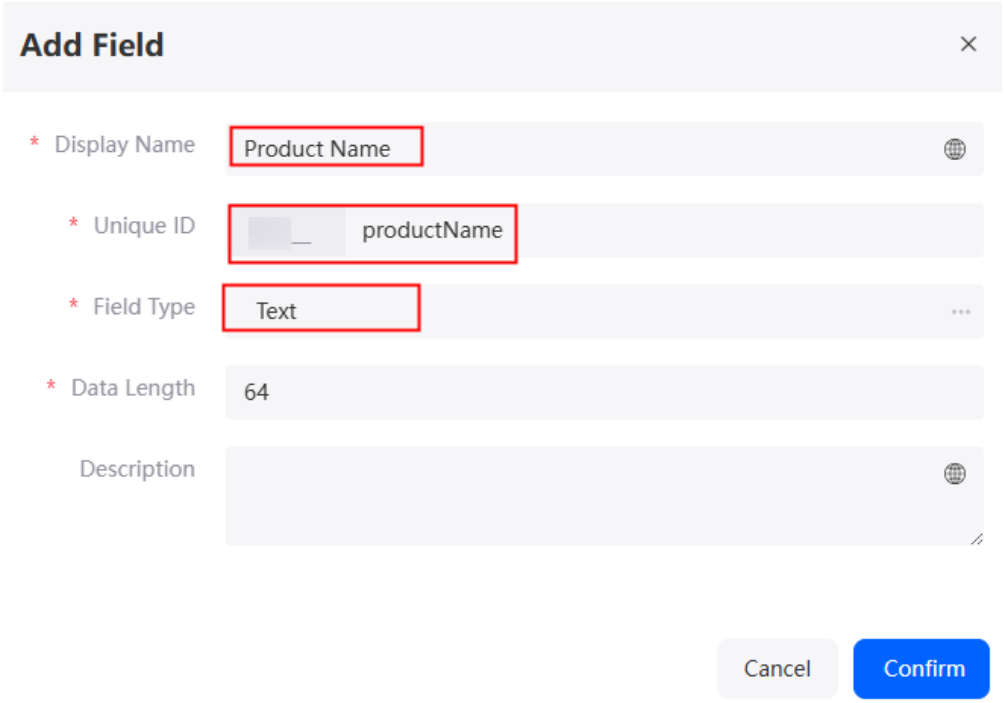


Table 6-14 Parameters for adding the productName field

Parameter	Description	Example
Display Name	Name of the new field, which can be changed after the field is created. Value: 1–63 characters.	Product name
Unique ID	ID of a new field in the system. The value cannot be changed after the field is created. Naming rules: – Max. 63 characters, including the prefix namespace. – Start with a letter and can contain only letters, digits, and an underscore (_). Do not end with an underscore (_).	productName
Field Type	Click ***. On the display page, select the type of the new field based on the parameter description.	Text

5. Repeat the preceding operations to add the fields in **Table 6-15** to the object.

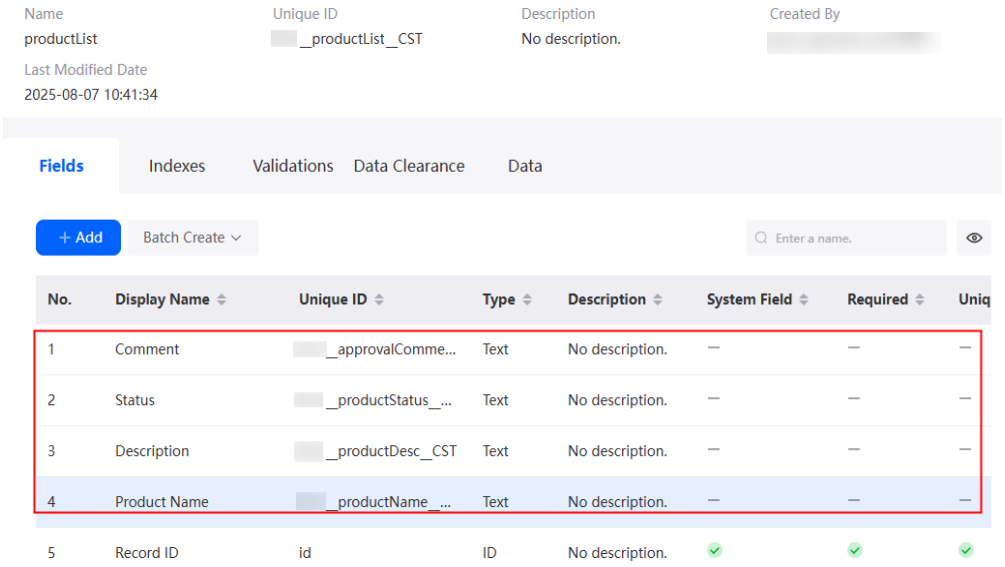
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-15 Adding fields to the object

Display Name	Unique ID	Type	Data Length
Description	productDesc	Text	255
Comment	approvalComments	Text	255
Status	productStatus	Text	255

After you finish the operations, you can see the new custom fields on the **Fields** tab of the object.

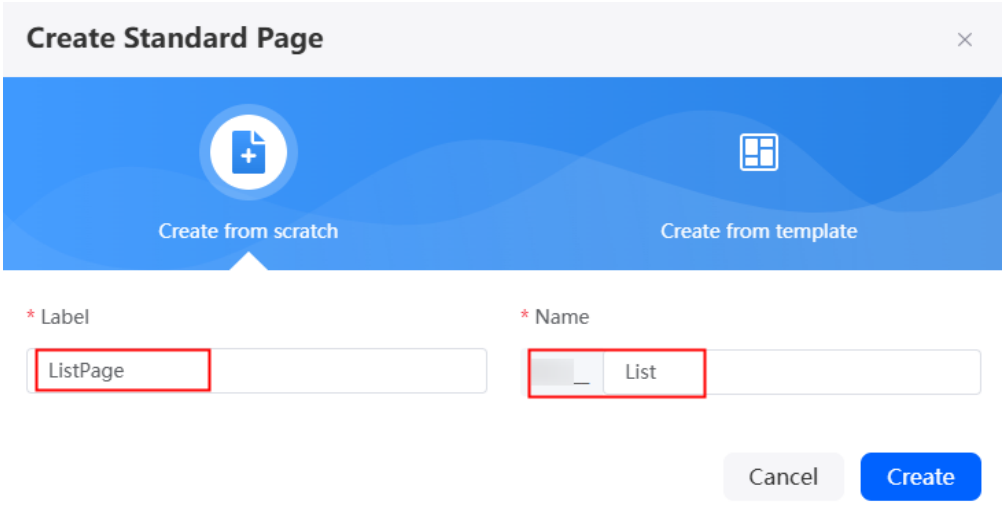
Figure 6-39 Viewing the added fields



Step 3 Create a list page and create an object model.

1. In the navigation pane, choose **Page**, and click **+** next to **Standard Page**.

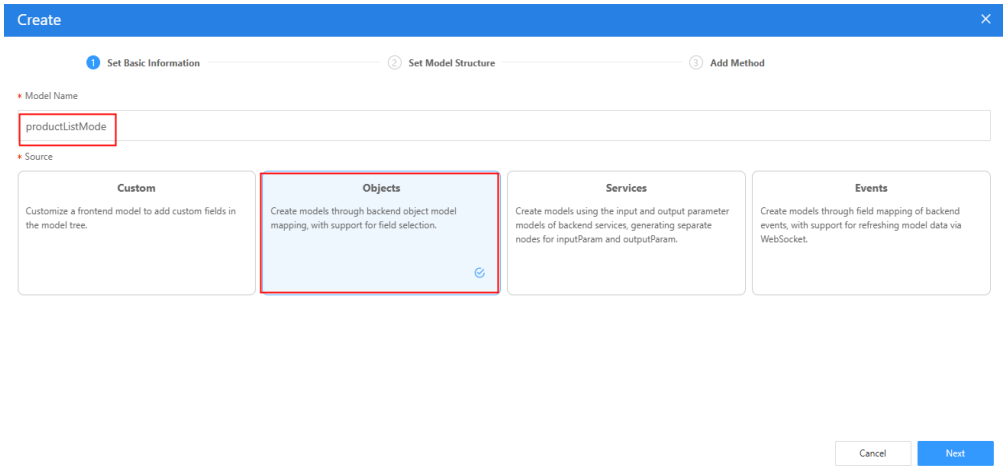
Figure 6-40 Creating a list page



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

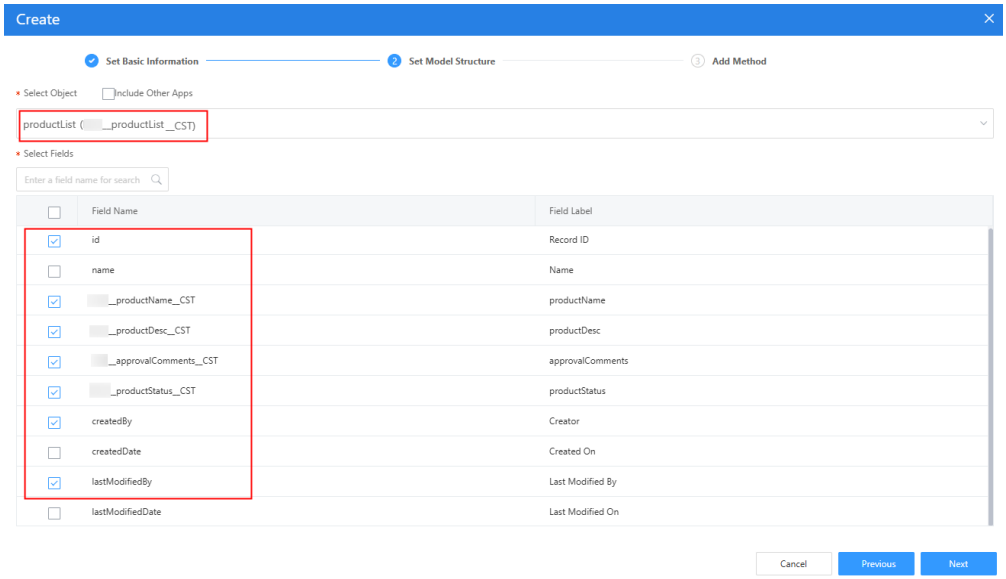
2. At the bottom of the standard page, click **Model View**.
3. Click **New**, specify **Model Name** (for example, **productListMode**), select **Objects** for **Source**, and click **Next**.

Figure 6-41 Creating a model



4. Select the object created and the fields added in **Step 2**, and click **Next**.

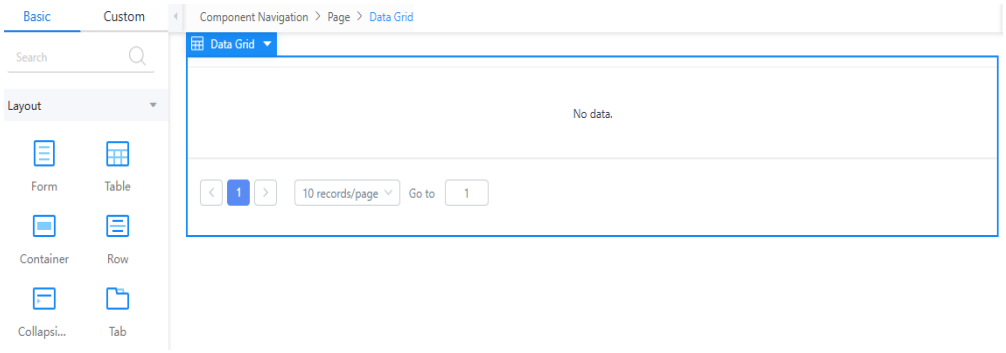
Figure 6-42 Selecting objects and fields



5. Click **OK**.
6. Return to the design view page, click **Designer View** at the bottom of the standard page.
7. Drag a table widget to the standard page.

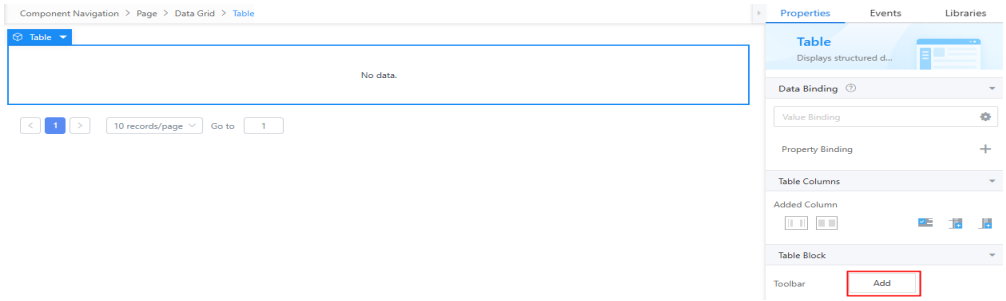
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-43 Dragging a table widget



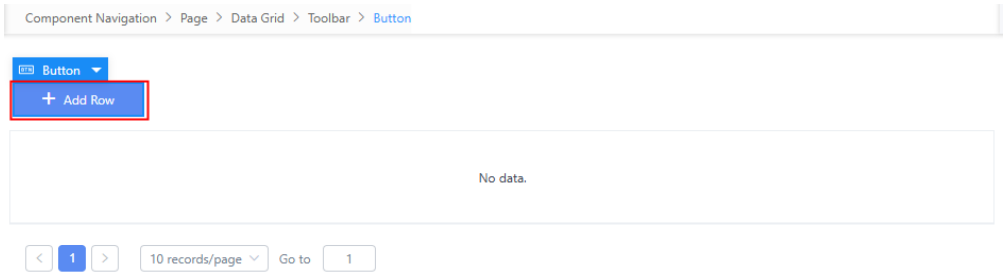
8. Select the table widget, choose **Properties** > **Table Block** > **Toolbar**, and click **Add** to add a toolbar to the table.

Figure 6-44 Adding a toolbar



9. Delete unnecessary buttons from the toolbar and retain only the **Add Row** button.

Figure 6-45 Deleting unnecessary buttons



10. Select the table widget, choose **Properties** > **Data Binding** and click next to **Value Binding**.
11. Select the model created in **Step 3.3** and click the confirm button to create a table list.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-46 Selecting objects and fields

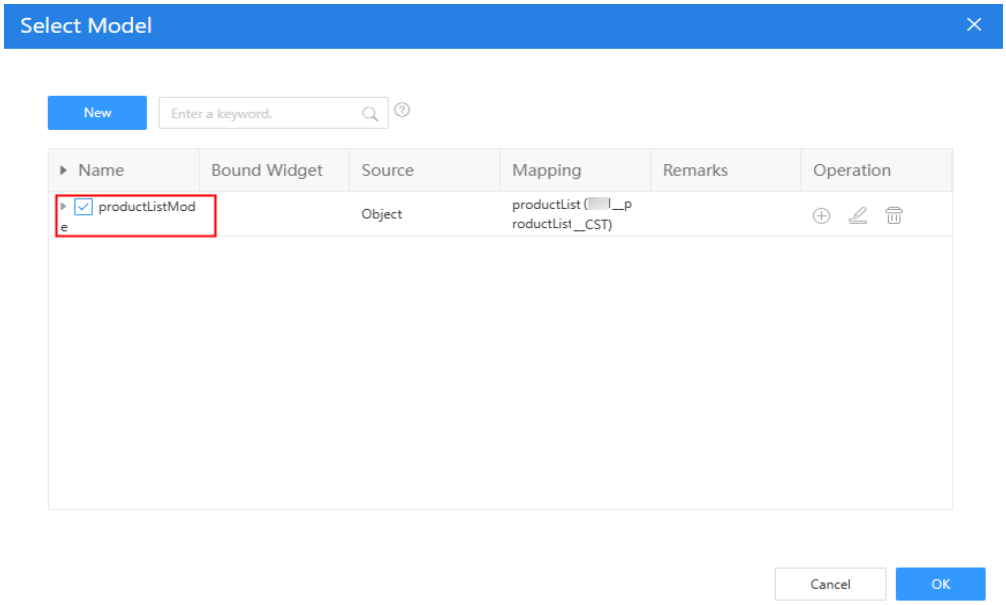
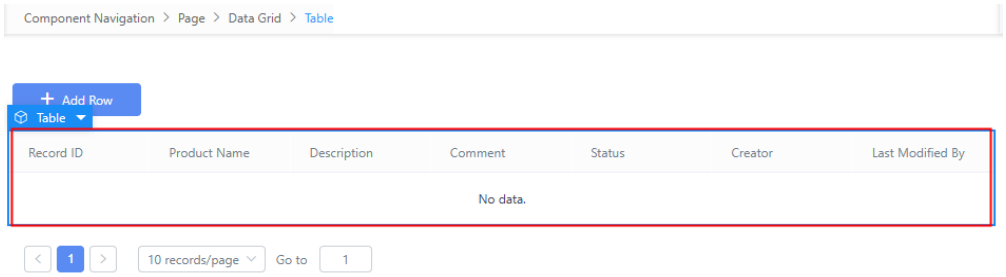


Figure 6-47 Table list

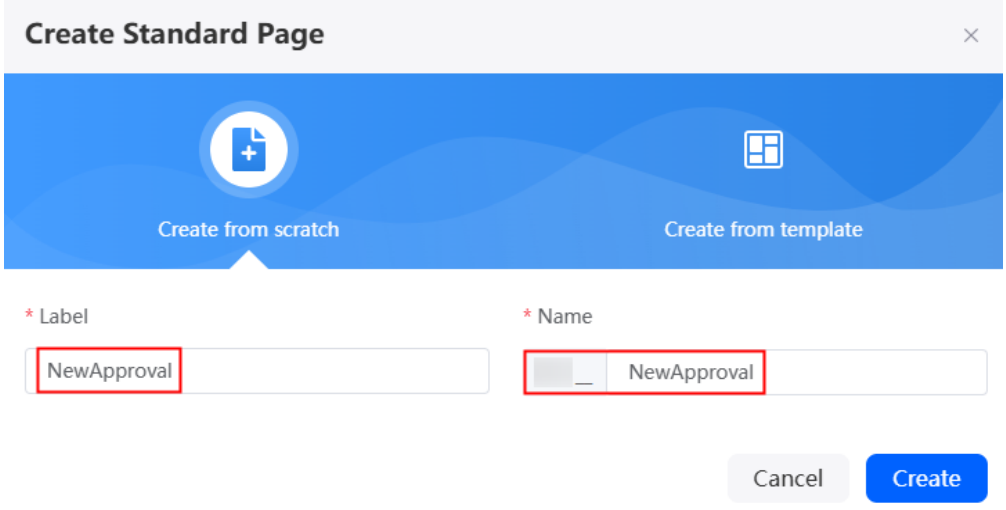


12. Click in the upper part of the page to save the page settings.

Step 4 Create a page for creation and approval, then add models to it.

1. In the navigation pane, choose **Page**, and click **+** next to **Standard Page**.

Figure 6-48 Creating a page



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2. Drag a form widget to the standard page, bind the data object, and add operation buttons.

Figure 6-49 Dragging a form widget

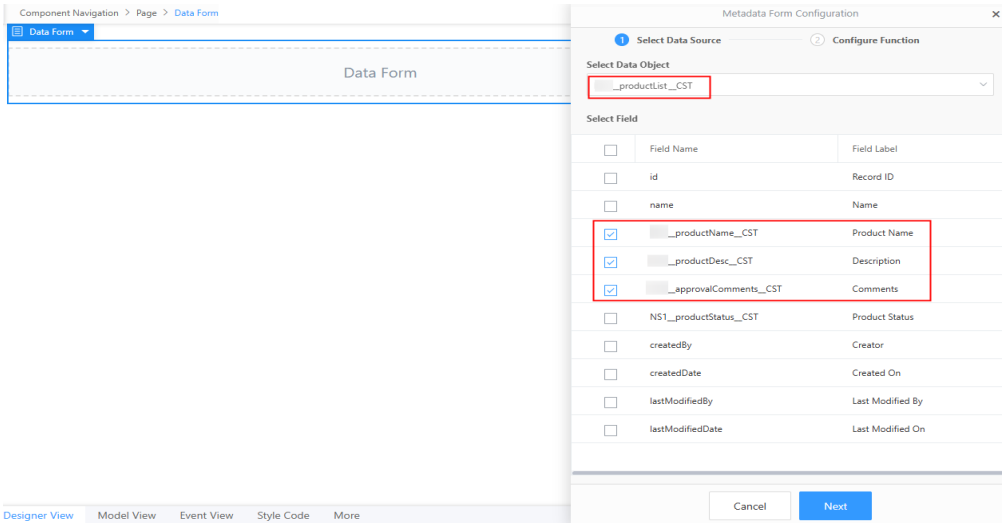
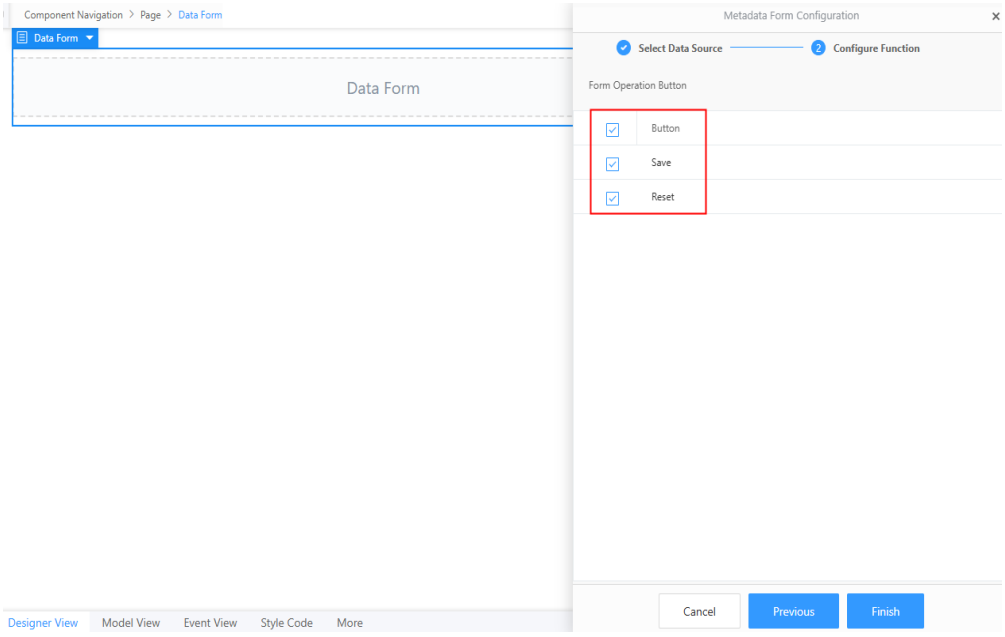
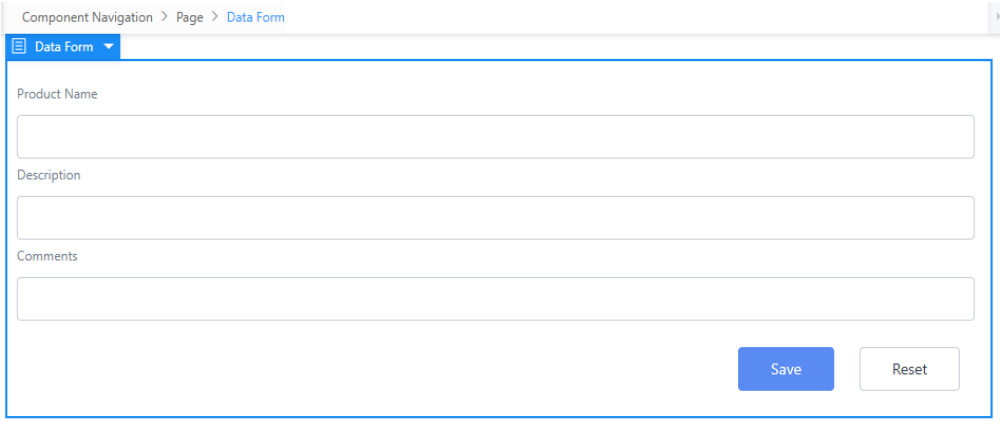


Figure 6-50 Adding operation buttons



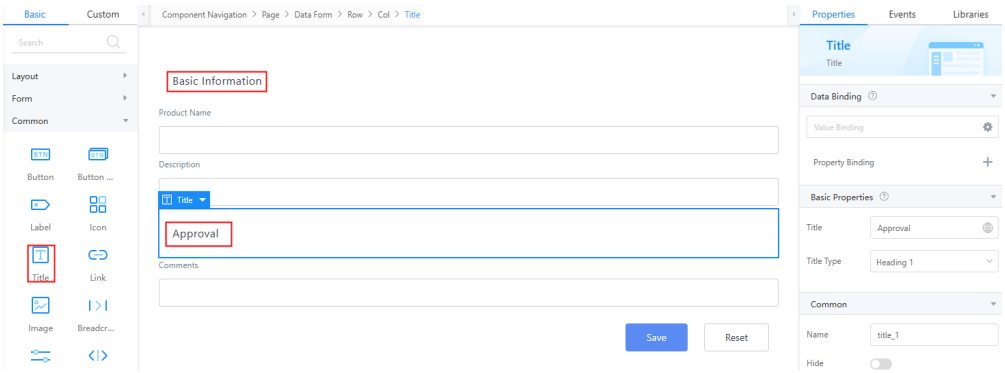
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-51 Effect



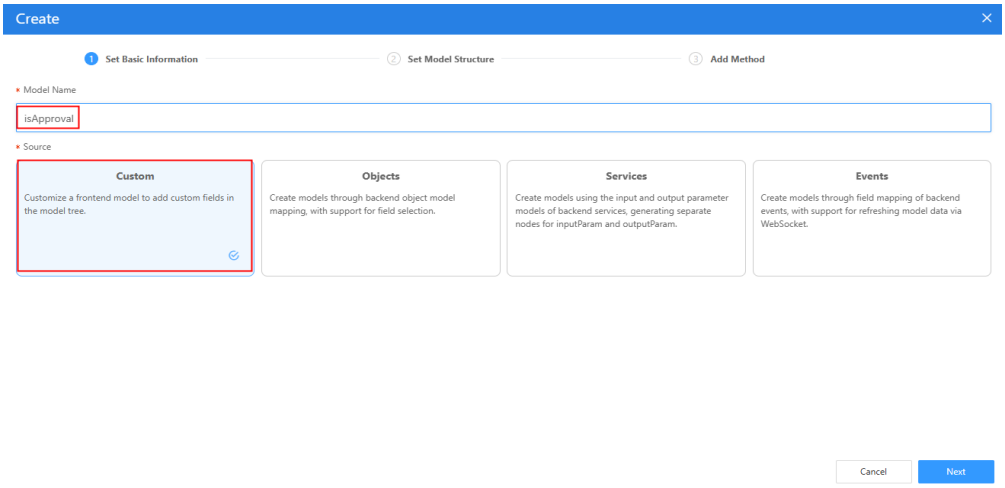
3. Drag two title widgets to the page, and change the title content to **Basic Information** and **Approval**. Figure 6-52 shows the layout effect.

Figure 6-52 Adding title widgets and adjusting the layout



4. At the bottom of the standard page, click **Model View**.
5. Click **New**, specify **Model Name** (for example, **isApproval**), select **Custom** for **Source**, and click **Next**.

Figure 6-53 Adding the isApproval model

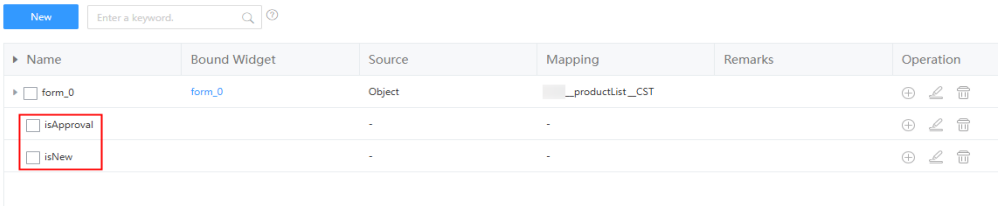


6. Retain the default values, click **Next**, and click **OK**. The model is created.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

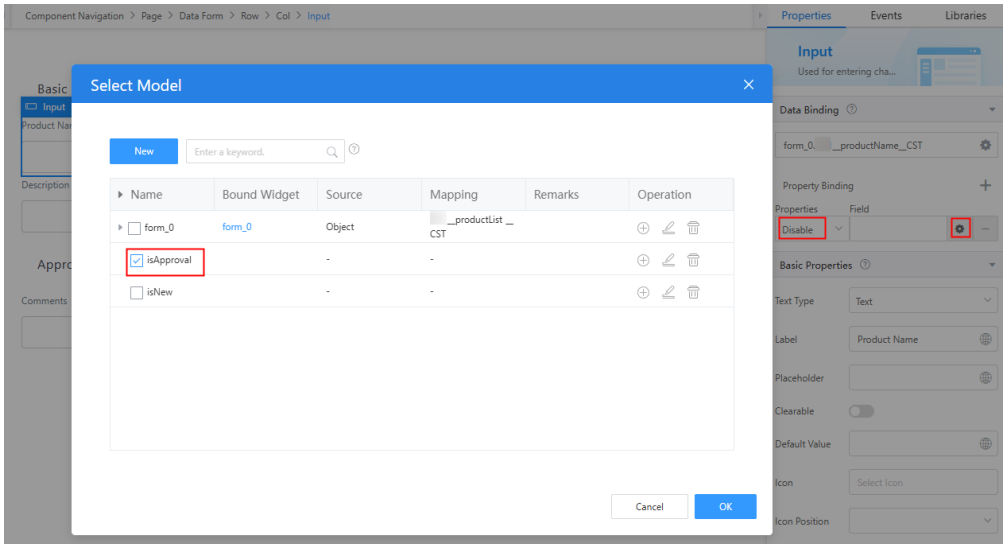
7. Repeat **Step 4.5** to **Step 4.6** to create the **isNew** model. The following figure shows the final effect.

Figure 6-54 Checking the created models



8. At the bottom of the standard page, click **Designer View**. Select the "Product Name" input box widget. Choose **Properties** > **Data Binding**, and click **+** next to **Property Binding**. Select **Disable** and click  to bind the **isApproval** model to the field.

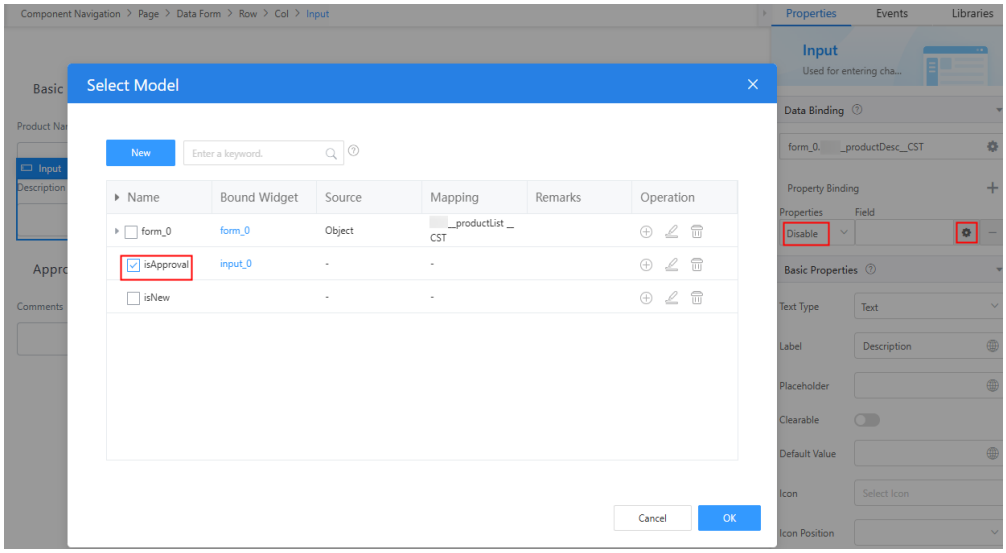
Figure 6-55 Binding the isApproval model to the product name input box widget



9. Select the "Description" input box widget. Choose **Properties** > **Data Binding**, and click **+** next to **Property Binding**. Select **Disable** and click  to bind the **isApproval** model to the widget.

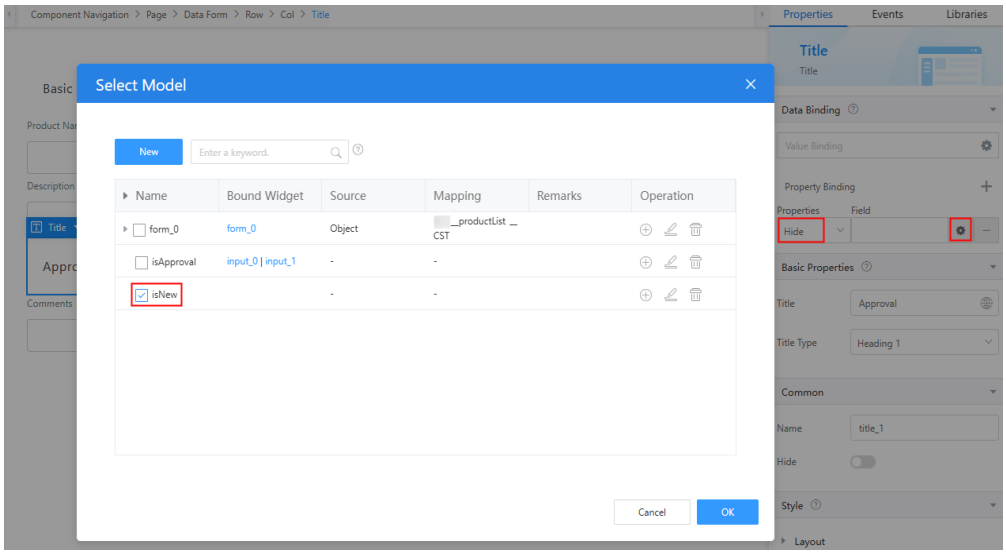
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-56 Binding the isApproval model to the description input box widget



10. Select the "Approval" title widget. Choose **Properties** > **Data Binding**, and click **+** next to **Property Binding**. Select **Hide** and click  to bind the **isNew** model to the field.

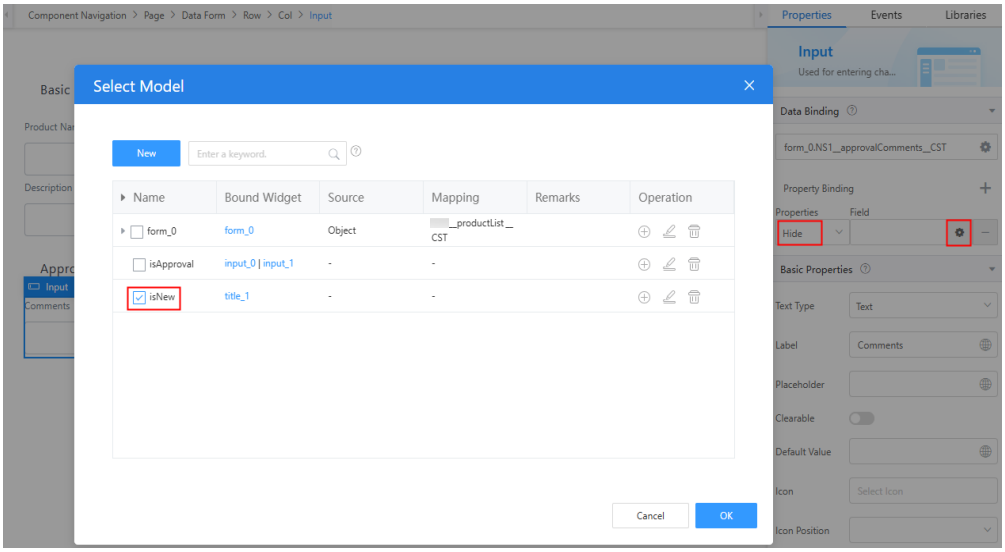
Figure 6-57 Binding the isNew model to the approval title widget



11. Similarly, select the "Comments" input box widget and bind the **isNew** model to the widget according to the operations in **Step 4.10**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-58 Binding the isNew model to the approval comment input box widget



12. Select the page widget, click the **Events** tab, and click **+** next to **on-load** to add an event for the page widget.

Figure 6-59 Adding an event for the page widget

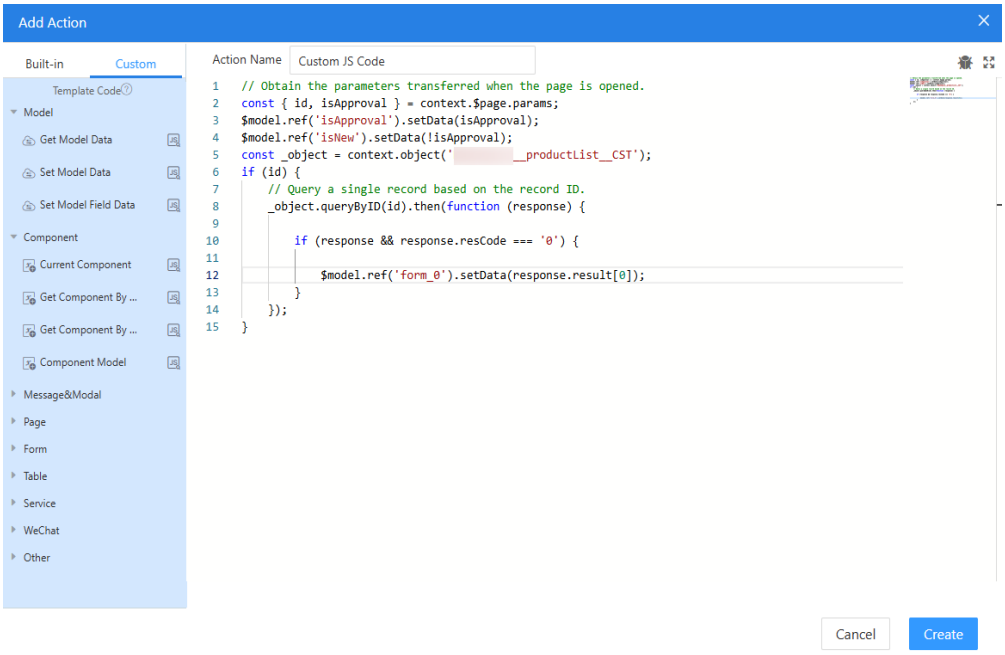
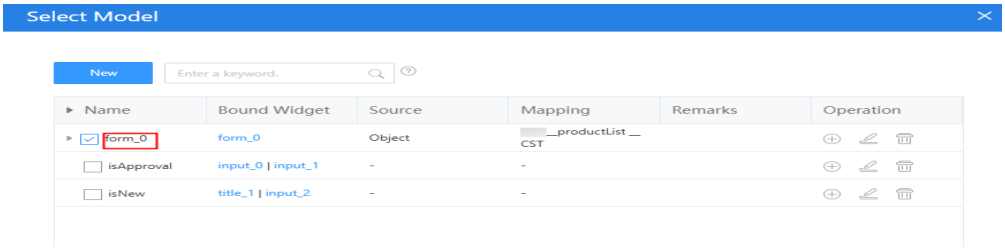


Figure 6-60 Viewing the name of the model bound to the form



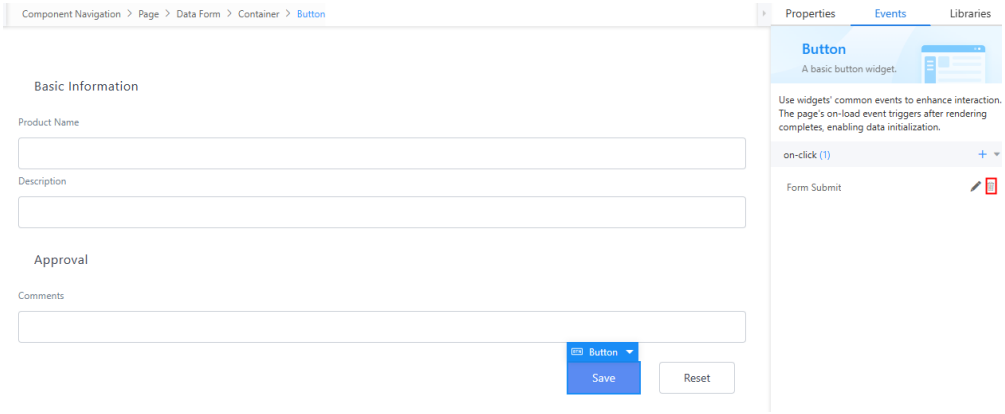
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Enter the following example in the custom action to obtain external data. *isApproval* is the model created in **Step 4.5**, *Namespace__productList__CST* is the object created in **Step 2**, and *form_0* is the model bound to the form in **Step 4.2**, as shown in **Figure 6-60**.

```
// Obtain the parameters transferred when the page is opened.
const { id, isApproval } = context.$page.params;
$model.ref('isApproval').setData(isApproval);
$model.ref('isNew').setData(!isApproval);
const _object = context.object('Namespace__productList__CST');
if (id) {
    // Query a single record based on the record ID.
    _object.queryByID(id).then(function (response) {

        if (response && response.resCode === '0') {

            $model.ref('form_0').setData(response.result[0]);
        }
    });
}
```

13. Select the **Save** button. On the **Events** tab page, click next to **Form Submit** to delete the current form submission method, and click **+** next to **on-click** to add a custom action for the button.

Figure 6-61 Deleting the form submission method



In the custom action, enter the following example to process new or updated data. *isNew* is the model created in **Step 4.7**, *Namespace__productList__CST* is the object created in **Step 2.2**, and *Namespace__productStatus__CST* and *Namespace__approvalComments__CST* are the object fields added in **Step 2.5**.

```
// Obtain the model data.

const isNew = $model.ref('isNew').getData();

// Obtain the form data.

const  _form = context.$component.form;

const data = _form.getFormData();

// Obtain the object.

const _object = context.object('Namespace__productList__CST');

// Create

if (isNew) {
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    data.Namespace__productStatus__CST = 'Approval in progress';
    // Insert data. Batch operations are supported.
    _object.insert([data]).then(function (response) {
        if (response && response.resCode === '0') {
            // Success message.
            context.$message.success('Created.');
            // Close the pop-up page. This function only works on pop-up pages.
            context.$dialog.popin();
        }
    });
} else {
    const id = context.$page.params.id;
    const params = {
        Namespace__productStatus__CST: 'Approved',
        Namespace__approvalComments__CST: data.Namespace__approvalComments__CST
    }
    // Update data based on the record ID.
    _object.updateByID(id, params).then(function (response) {
        if (response && response.resCode === '0') {
            // Success message.
            context.$message.success('Approved.');
            // Close the pop-up page. This function only works on pop-up pages.
            context.$dialog.popin();
        }
    });
}
```

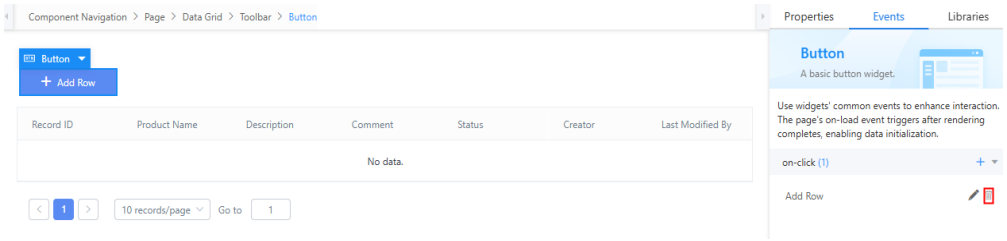
14. Click  in the upper part of the page to save the settings.

Step 5 Return to the list page and add an event.

1. Select the **Add Row** button widget. On the **Events** tab page, click  next to **Add Row** to delete the new row, and click **+** next to **on-click** to add a custom action for the button.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-62 Deleting a preset event



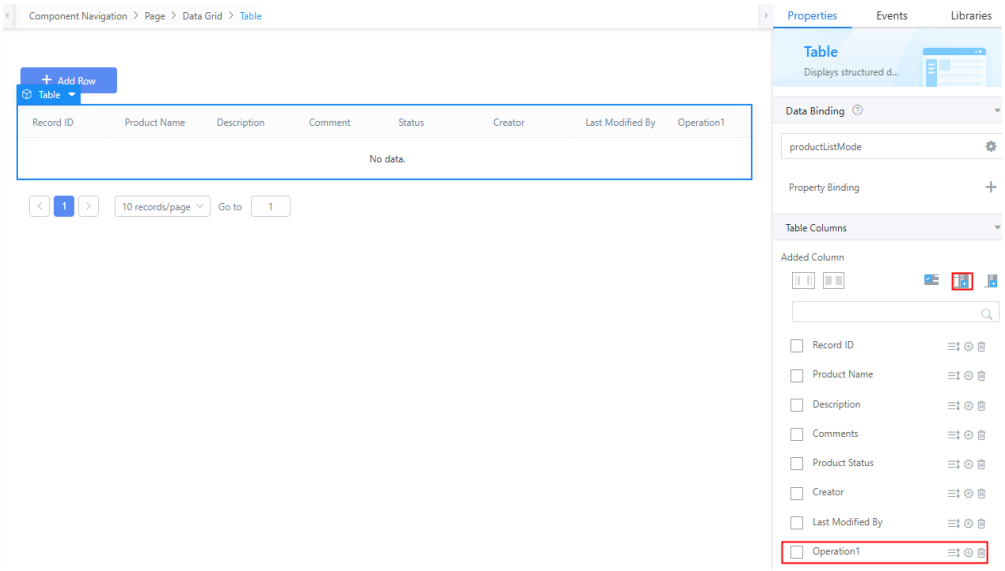
In the custom action, enter the following example to display a standard page. *Namespace__NewApproval* is the standard page created in **Step 4**.

```
// Display the standard page.
context.$dialog.popup({
    title: 'New',
    page: 'Namespace__NewApproval',
    width: 600,
    height: 400,
    footerHide: true,
    showCancel: true,
    okText: 'ok',
    params: {},
    onClose: function () {
        // Wait until the database operation is complete.
        setTimeout(() => {
            // Current table.
            const _table = context.$component.table;
            // Refresh the table.
            _table.doQuery();
        }, 100);

    }
});
```

2. Select the table widget, choose **Properties** > **Table Columns**, and click to add an operation column to the table.

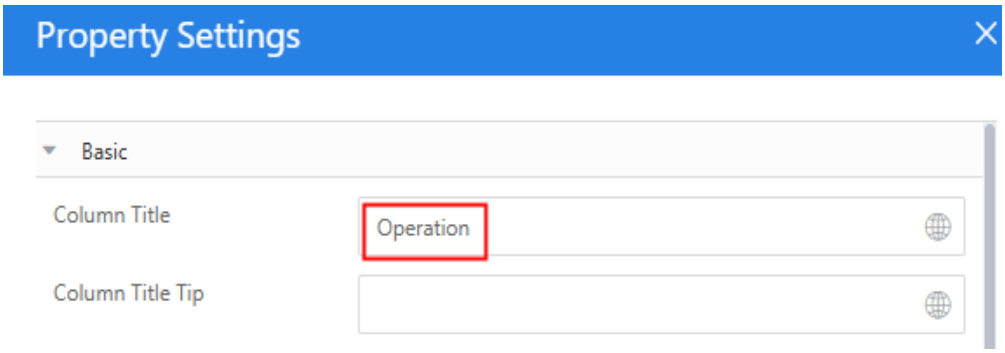
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-63 Adding an operation column



3. Click next to the new operation column **Operation1**. The **Property Settings** dialog box is displayed.
4. In the **Basic** area, change the value of **Column Title** to **Operation**.

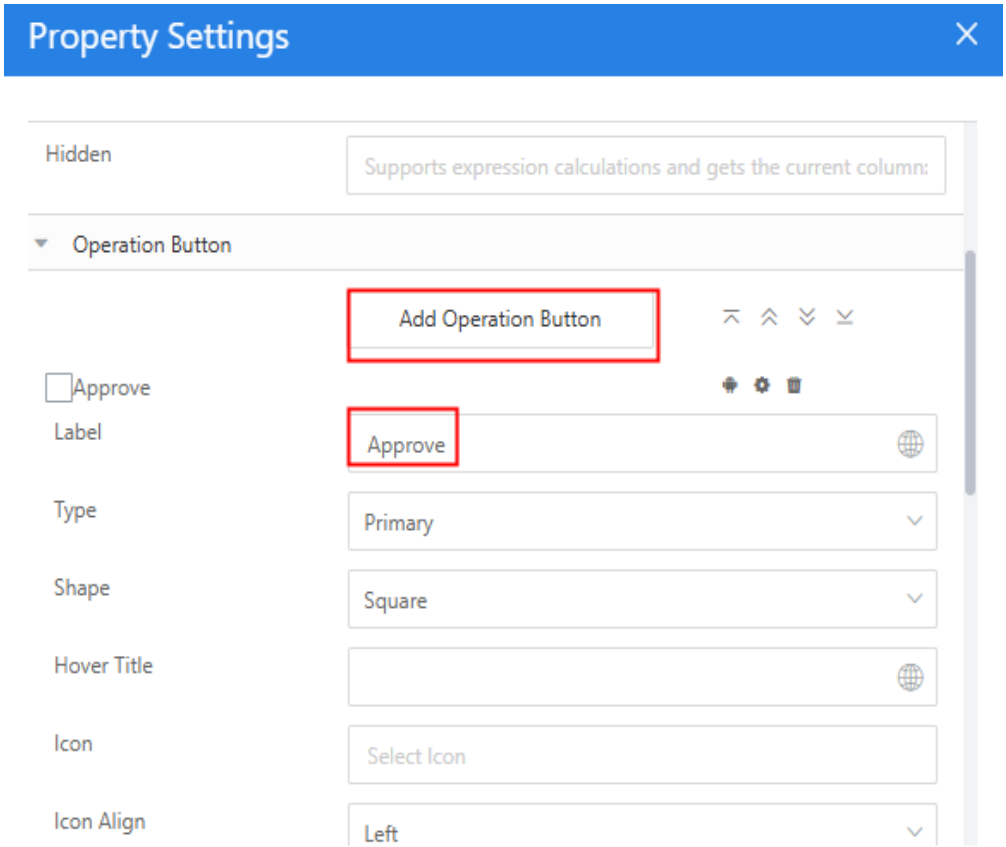
Figure 6-64 Modifying the column title



5. In the **Operation Button** area, click **Add Operation Button** to add an operation button. Set **Label** to **Approve**.

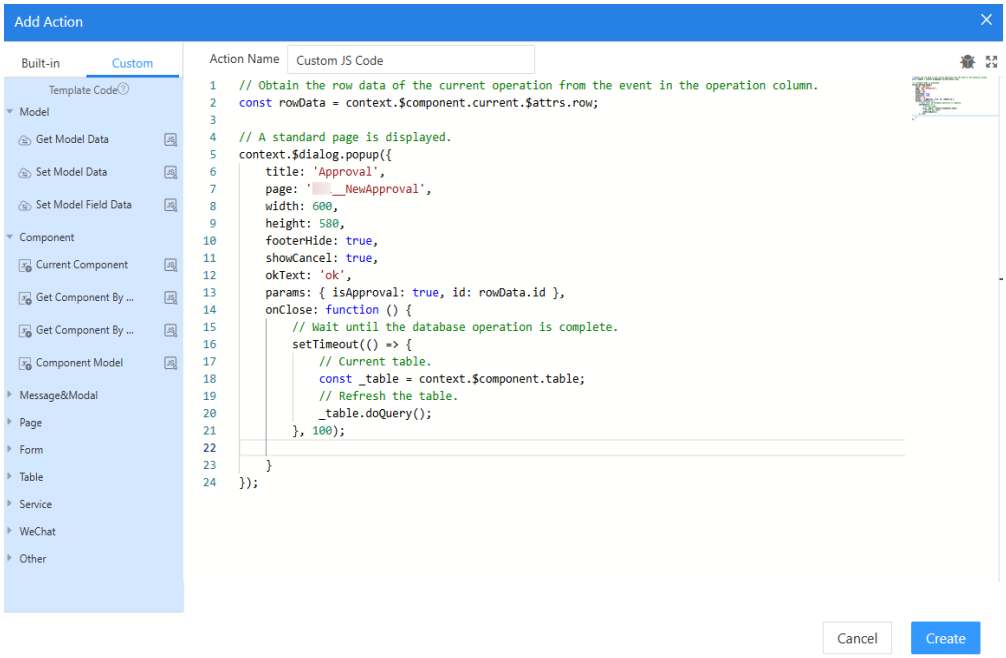
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-65 Adding an Approve button



6. Click next to **Approve** and click **+** next to **Action List** to add a custom action for the operation button.

Figure 6-66 Adding a custom action for the operation button



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

In the custom action, enter the following event to obtain the row data. *Namespace__NewApproval* is the standard page created in **Step 4.1**.

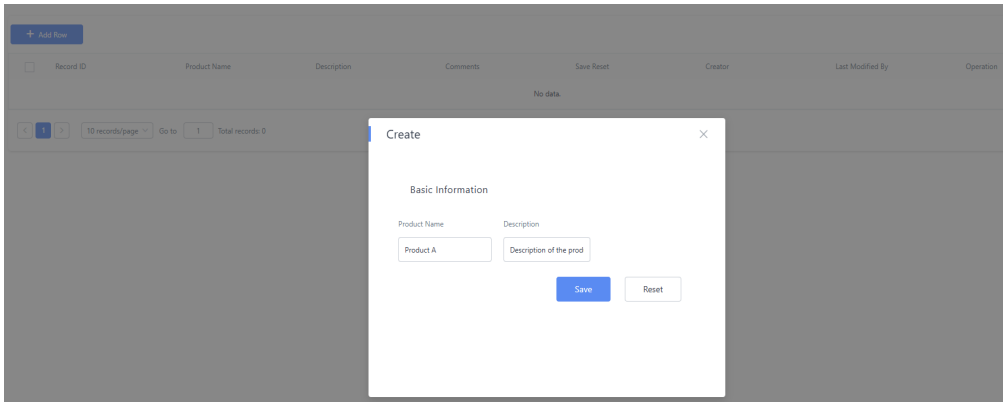
```
// Obtain the row data of the current operation from the event in the operation column.
const rowData = context.$component.current.$attrs.row;

// A standard page is displayed.
context.$dialog.popup({
    title: 'Approval',
    page: 'Namespace__NewApproval',
    width: 600,
    height: 580,
    footerHide: true,
    showCancel: true,
    okText: 'ok',
    params: { isApproval: true, id: rowData.id },
    onClose: function () {
        // Wait until the database operation is complete.
        setTimeout(() => {
            // Current table.
            const _table = context.$component.table;
            // Refresh the table.
            _table.doQuery();
        }, 100);

    }
});
```

7. Click  in the upper part of the page to save the settings.

Step 6 Verify the effect.

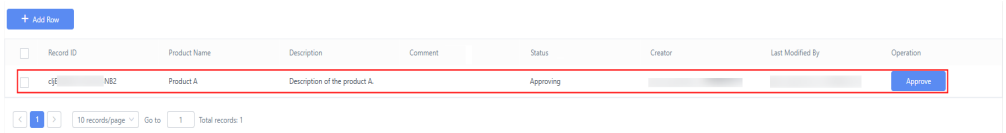
1. On the list page, click  in the upper part of the page to go to the preview page.
2. Click **Add Row**, enter the product name and product description, and click **Save** to add a data record.

Figure 6-67 Adding a row of data



After the data is saved, you can view the added data in the list, and the product status is **Approving**.

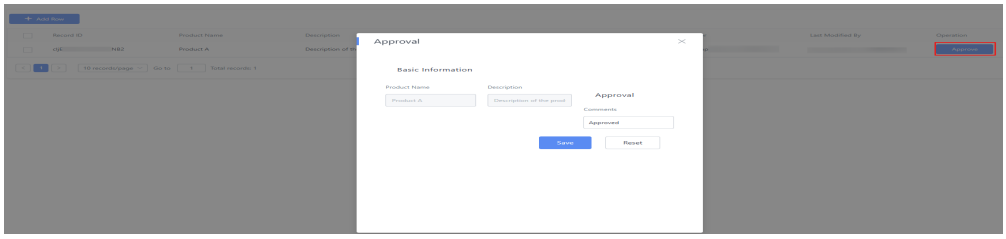
Figure 6-68 Viewing the added data



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

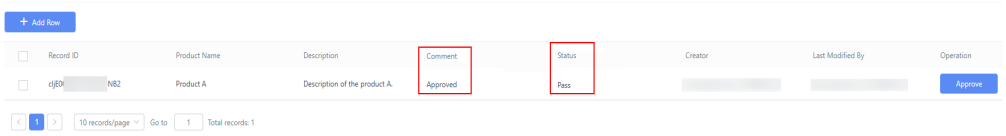
3. Click **Approve** next to the new data, add approval comments, and then click **Save**.

Figure 6-69 Approval completed



After the data is saved, you can view that the comment is approved and the product status is **Pass**.

Figure 6-70 Viewing the approval comment



----End

6.5 Customizing Standard Page Widget Packages

6.5.1 Overview of Customizing Standard Page Widgets

Application Scenarios

You can customize standard page widgets according to this practice. This lets you add features to the platform and build applications faster and simpler.

Using **imgButton** as the example, this practice shows which configuration items to change, then walks through publishing the widget and adding it to a page.

You can create your own button and text widgets using **imgButton**.

Advantages

When preset widgets fall short, customize ones to add flexible features and boost low-code service capabilities.

Constraints

The file structure and internal configuration items of the widget must be set according to the platform standards.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6.5.2 Process of Customizing Standard Page Widgets

Huawei Cloud Astro Zero provides you with a custom widget template package **workCards.zip**. This section shows how to customize an **imgbutton** widget from it. **Table 6-16** shows the process.

Table 6-16 Process of customizing standard page widgets

No.	Step	Description
1	**Step 1: Obtain the Widget Template Package**	1. Obtain **workCards.zip**. 2. Learn the structure and definition specifications of **workCards.zip**.
2	**Step 2: Customize the imgButton Widget**	1. Unzip **workCards.zip** and rename the resulting folder to **imgButton**. 2. Customize the widget. 3. Compress the content in the **imgButton** folder to generate the custom widget package **dist\imgButton-***.zip**.
3	**Step 3: Upload the Custom Widget Package**	Upload the custom widget to Huawei Cloud Astro Zero.
4	**Step 4: Verify the Custom Widget**	Create a standard page and drag the custom widget to the page. The widget is displayed properly. Add the display and hide events for the custom widget and check whether the effect meets the expectation.

6.5.3 Procedure of Customizing Standard Page Widgets

Step 1: Obtain the Widget Template Package

Obtain and use the **workCards.zip** template package to customize widgets. Learn the structure and definition specifications of the template package first.

Step 1 Click **here** to obtain **workCards.zip**.

Step 2 Unzip the **workCards.zip** package and learn its structure and definition specifications.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-71 Structure of workCards.zip

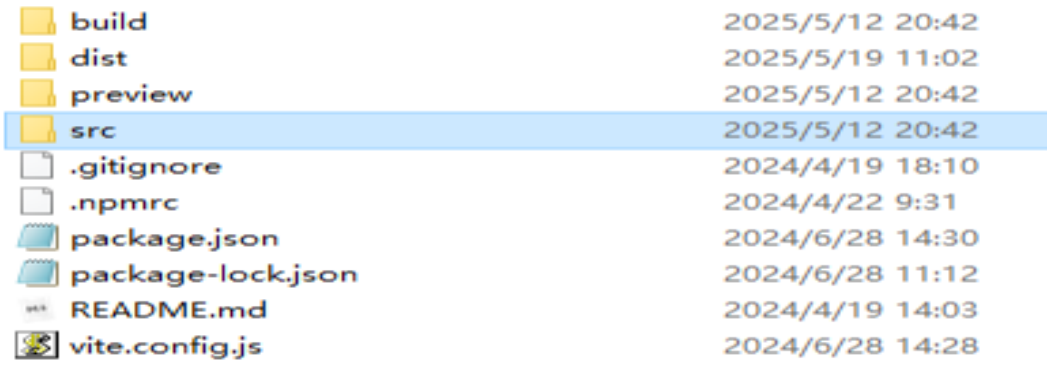


Table 6-17 Directories and files in the workCards.zip template package

Directory and File	Description
build	Stores the widget packaging script.
dist	Stores widget packaging files. The files are generated in this directory after being packaged.
preview	Widget preview folder.
src	Main folder. The widget content is stored in this folder. In this practice, you only need to modify the content in the **src\components\workCards** directory. For details, see **Table 6-18**.
.gitignore	Ignored files of Git.
.npmrc	Sets the installation source of the dependency package in **package.json**.
package.json	Dependency file of the project.
package-lock.json	Used to lock the installation version number of the dependency file.
README.md	Project prompt file, which is used to describe the widget.
vite.config.js	Configuration file of Vite.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-72 src\components\workCards directory structure

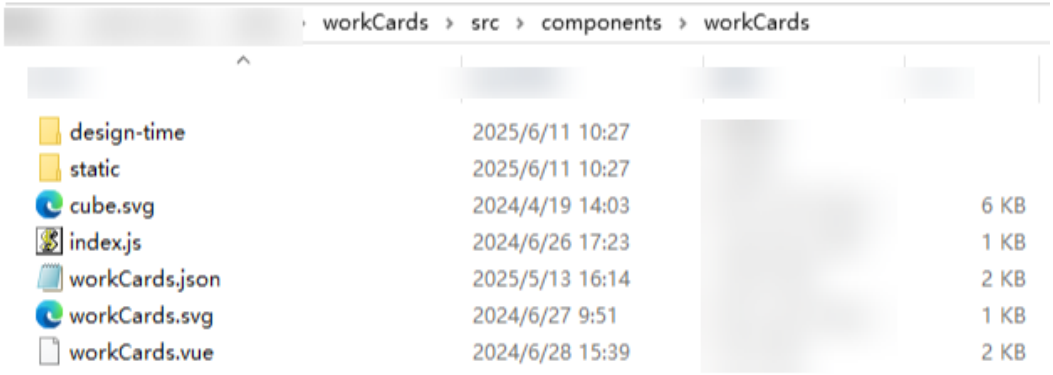


Table 6-18 src\components\workCards directory structure

Directory and File	Description
design-time	Directory of widget files in the development environment.
static	Static file directory.
cube.svg/workCards.svg	Icon used in a service.
index.js	Main entry file of the widget.
workCards.json	Widget description file, which is used to customize the widget configuration panel. It helps verify and compile the panel's settings. The **workCards.json** file must follow JSON Schema specifications to integrate with mainstream editors such as Visual Studio Code (VS Code), Atom, Sublime Text, and WebStorm. The **workCards.json** file can be modified during plugin package development or after compilation. Before compilation, it is stored in the **src\components\workCards** directory (as shown in Figure 6-73). After compilation, it is stored in the **dist\components\workCards** directory (as shown in Table 6-19).
workCards.vue	Main file of a widget.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-73 Location of the workCards.json file before compilation

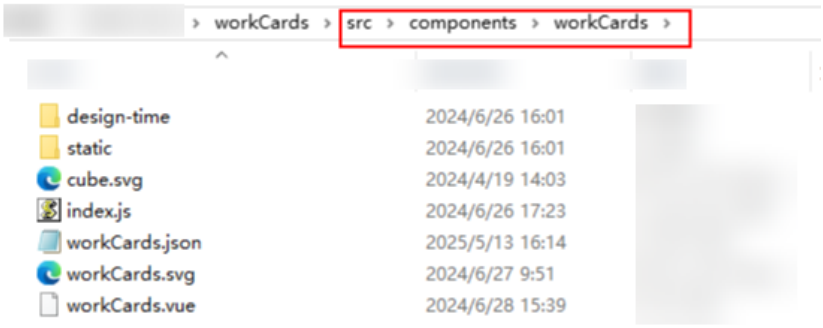


Figure 6-74 Location of the workCards.json file after compilation

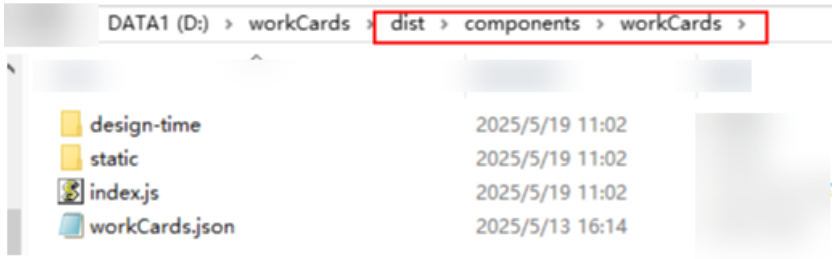


Table 6-19 workCards.json parameters

Parameter	Description
name	Widget name. The name is used for storing and parsing metadata on the backend. • The name can contain up to 64 characters. It must start with a letter and can contain only letters, digits, underscores (_), and hyphens (-). It is recommended that the name be in the format of ***Vendor name_Widget name_Version number***, for example, **aaa_img-button_1.1.1**. • The widget name is stored in the **type** of the UI metadata, for example, type: **my-component1**. • The name of the ZIP package must be the same as the value of name. For example, if **name** is set to **imgbutton**, the ZIP package name must be **imgbutton.zip**.
title	In the left-hand widget tab, the alias appears beneath each custom widget. Keep it short and descriptive. Value: 1–100 characters.
description	Widget description. The description that appears when you hover over the widget. Value: 1–200 characters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
category	Widget category. Assign your widget to one of the platform's predefined categories to keep the UI builder consistent. For details, see **Table 6-20**.
icon	Widget icon. There are two PNG icons (unselected and selected status), each no larger than 16 KB.
other	Defines the package size. The compiled widget package must be 1 MB or smaller.

Table 6-20 Weight categories

Category	Description
navigation	Navigation widgets, such as menus, toolbars, or sidebars.
data	Data widgets, which can be used to view and edit data in applications, such as forms, tables, and lists.
common	Common widgets, such as labels, images, titles, and paragraphs.
container	Container widgets, which are used to hold and organize other widgets such as grids, rows, and columns.
input	Displays and edits entity properties, such as text boxes and date selectors.
file	File processing widgets, which include file upload and download, image browsing, and PDF preview.
button	Buttons that trigger actions, such as the save button and page redirection button.
report	Aggregation widgets which display data in tables or charts, such as chart and pivot table widgets.
widget	Service card.
add-on	Extended widgets.

----End

Step 2: Customize the imgButton Widget

In this practice, the **imgButton** widget is a frontend widget built with Node.js. Unless otherwise specified, this practice uses the **imgButton** widget as the model for VS Code-based customization.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 1 Unzip the template package obtained in **Step 1: Obtain the Widget Template Package** and rename the resulting folder to **imgButton**.

Step 2 Modify the widget description file **workCards.json**, change **name** to **imgButton**, save the file, and exit.

```
{
  "name": "imgButton",
  "private": true,
  "version": "1.0.2",
  "type": "module",
  "scripts": {
    "dev": "vite",
    "build": "node ./build/build.mjs",
    "format": "prettier --write src/",
    "preview": "vite preview"
  },
  "dependencies": {
    "element-plus": "^2.4.2",
    "vue": "^3.3.4"
  },
  "devDependencies": {
    "@vitejs/plugin-vue": "^4.6.2",
    "jszip": "^3.10.1",
    "kolorist": "^1.8.0",
    "sass": "^1.69.5",
    "unplugin-element-plus": "^0.8.0",
    "unplugin-vue-components": "^0.25.1",
    "vite": "^4.4.5",
    "vue-style-loader": "^4.1.3"
  }
}
```

Step 3 Modify the **name** and **components** parameters in the **manifest.json** file in the **src** folder to **imgButton**, save the file, and exit.

```
{
  "name": "imgButton",
  "version": "1.0.10",
  "type": "uiPackage",
  "package": {
    "components": [
      "components/imgButton/imgButton.json"
    ]
  },
  "images": {
    "default": "./static/package-default.png",
    "previews": [
      "./static/package-preview1.png",
      "./static/package-preview2.png"
    ]
  },
  "publisher": "l84182839",
  "license": "MIT",
  "keywords": [
    "UI Component"
  ],
  "categories": [
    "ui:components"
  ],
  "bugs:url": "",
  "repository:url": "",
  "dependencies": []
}
```

Step 4 Modify the name of the **src\components** folder and the file name in the folder from **workCards** to **imgButton**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-75 Modifying the folder name

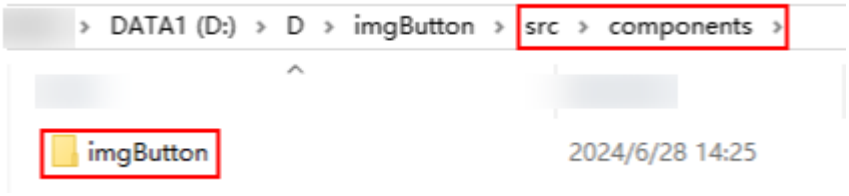
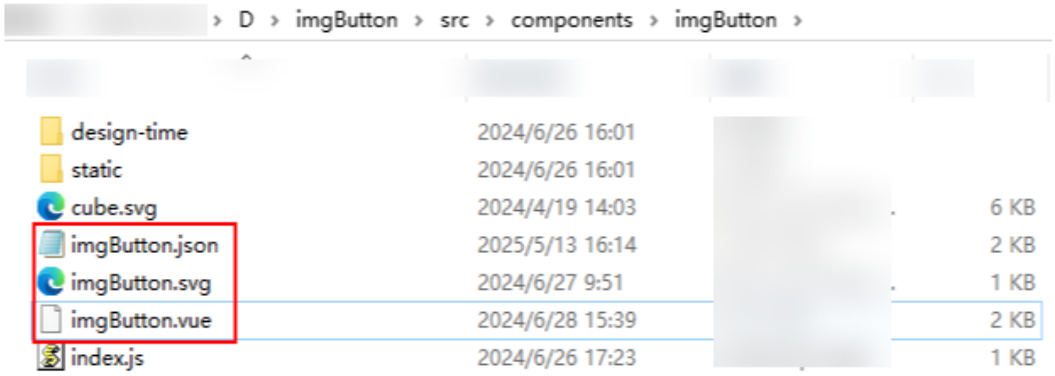


Figure 6-76 Modifying the file names in the folder



Step 5 Replace the widget icon of the custom widget. The icon is the thumbnail of the custom widget displayed in the page builder.

In **src\components\imgButton\static**, replace **default.png** (normal status) and **hover.png** (mouse-over highlight).

Step 6 Modify the definition file **imgButton.json** in **src\components\imgButton**.

- Modify the basic properties of the widget definition file.

  In the **imgButton.json** file, set **name**, **title**, and **description** to **imgButton**, and restrict **properties** to **height**, **width**, and **text**. The code is as follows:

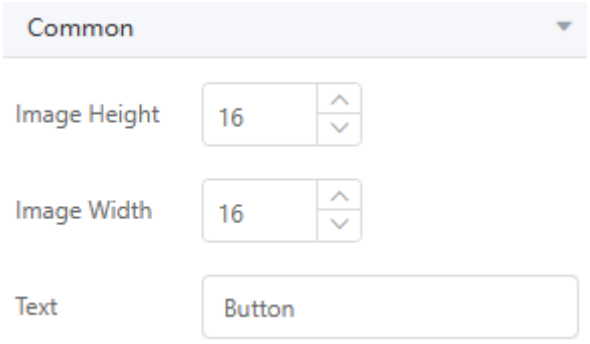
```
{
 "name": "imgButton",
 "title": "imgButton",
 "description": "imgButton",
 "category": "add-on",
 "icon": {
  "default": "./static/default.png",
  "hover": "./static/hover.png"
 },
 "definition": {
   "props": {
    "type": "object",
    "properties": {
     "height": {
       "category": "common", //Tag to which the property belongs, including common (common), general (basic), and style (style)
       "type": "integer", //Value type of the property, which can be integer (number text box), date (date selection box), string (text box), select (drop-down list), or switch (switch)
       "title": "Image height",//Property title
       "description": "Set the image height.", //Property description, which is displayed when the pointer is hovered over the title
       "default":16 //Default value of the property
      },
      "width": {
       "category": "common",
       "type": "integer",
       "title": "Image width",
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
        "description": "Set the image width",
        "default":16
      },
      "text": {
        "category": "common",
        "type": "string",
        "title": "framework.label.text",
        "description": "Button text",
        "default":"Button"
      }
    }
  },
 "main": "./index.js",
 "mainDesignTime": "./design-time/index.js",
 "dnd": {
   "cname": "workCards",
   "dname": "workCards-design",
   "ctype": "Control",
   "displayName": "workCards",
   "selectable": true,
   "targetable": true,
   "draggable": true,
   "deletable": true,
   "wholeRow": true
 }
}
```

After saving the changes, the standard page shows the widget with only height, width, and text properties.

Figure 6-77 Widget properties



- Modify the event in the widget definition file and change the event name to **click**.

```
"on": {
    "type": "object",
    "properties": {
      "click": { // Event name triggered in the code, which is triggered using emit in the widget
        "title": "Click", //Event name
        "description": "Action executed upon clicking", //Event description
        "type": "object"
      }
    }
  },
```

- Modify the drag behavior in the widget definition file. Set **cname**, **displayName**, and **dname** to **imgButton** and **imgButtonDesign**.

```
"dnd": {
  "cname": "imgButton",
  "dname": "imgButtonDesign",
  "ctype": "Control",
  "displayName": "imgButton",
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  "selectable": true,
  "targetable": true,
  "draggable": true,
  "deletable": true,
  "wholeRow": true
 }
```

cname indicates the widget displayed in the application engine, **dname** indicates the widget displayed in the application development tool, and **displayName** indicates the text displayed for the widget in the page builder.

After these modifications, the final **imgButton.json** is as follows:

```
{
 "name": "imgButton",
 "title": "imgButton",
 "description": "imgButton",
 "category": "add-on",
 "icon": {
  "default": "./static/default.png",
  "hover": "./static/hover.png"
 },
 "definition": {
   "props": {
    "type": "object",
    "properties": {
     "height": {
       "category": "common",
       "type": "integer",
       "title": "Image height",
       "description": "Set the image height",
       "default":16
     },
     "width": {
       "category": "common",
       "type": "integer",
       "title": "Image width",
       "description": "Set the image width",
       "default":16
     },
     "text": {
       "category": "common",
       "type": "string",
       "title": "framework.label.text",
       "description": "Button text",
       "default":"Button"
     }
    }
  },
  "on": {
    "type": "object",
    "properties": {
     "click": {
       "title": "Click",
       "description": "Action executed upon click",
       "type": "object"
     }
    }
  },
  "methods": {
  }
 },
 "main": "./index.js",
 "mainDesignTime": "./design-time/index.js",
 "dnd": {
  "cname": "imgButton",
  "dname": "imgButton",
  "ctype": "Control",
  "displayName": "imgButton",
  "selectable": true,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    "targetable": true,
    "draggable": true,
    "deletable": true,
    "wholeRow": true
  }
}
```

Step 7 Modify the view file **imgButton.vue** in the **src\components\imgButton** directory.

- Modify the layout structure of the view file.

  Add an **img** and a **span** to the layout (div tag) and configure the parameters according to the example. The code example is as follows:

  ```
  <img
    src="./static/hover.png"
    :height="comHeight"
    :width="comWidth"
    :class="classes + '-img'">
  <span :class="classes+ '-title'">
    {{ currentText }}
  </span>
  ```

  img indicates the image in the **imgButton** button, and **span** indicates the text in the **imgButton** button.

- Modify the property definition of the view file **imgButton.vue** and change **cssprefix** to **imgButton** and add the **height**, **width**, and **text** properties.

  ```
  <script>
  const cssprefix = "imgButton";

  import { defineComponent } from "vue";

  export default defineComponent({
    name: "imgButton",
    data() {
      return {
        currentText: this.text,
      };
    },
    props: {
      /*Image height */
      height: {
        type: [String, Number],
        default: "",
      },
      /*Image width */
      width: {
        type: [String, Number],
        default: "",
      },
      /*Tag text content */
      text: {
        type: String,
        default: "Button1",
      },
    },
  ```

- Modify **computed**, **watch**, and **methods** in the **imgButton.vue** view file.

  ```
    computed: {
      classes() {
        return cssprefix;
      },
      comHeight() {
        return this.height || "16";
      },
      comWidth() {
        return this.width || "16";
      },
    },
    watch: {
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  text(newVal) {
    this.currentText = newVal;
  },
},
methods: {
  onclick(e) {
    this.$emit("click", e);
  },
  setButtonText(newText) {
    console.log("setButtonText params:", newText);
    if (newText) {
      this.currentText = newText;
    }
  },
},
```

Table 6-21 computed, watch, and methods parameter description

Parameter	Description
cssprefix	Constructs the style class name to improve flexibility.
currentText	Displays the button text.
comHeight	Obtains the image height.
comWidth	Obtains the image width.
classes	Returns the style class name. You can also replace the variable with a character string to increase flexibility.
onclick	Triggers the service logic of the click event when the image button is clicked.
setButtonText	Changes the text in the button. This method is exposed externally. All methods defined in **method** can be directly called in events. **Figure 6-78** shows the calling mode.

Figure 6-78 Calling the method

```
var _component = context.$component.current;
_component.setButtonText("newText");
```

- Modify the **style** in the view file **imgButton.vue** to define the style class of the image button. The style class includes the image button frame, image, and text.

```
.imgButton {
  display: inline-block;
  background-color: #8cdcdb;
  border: blue1pxsolid;
  cursor: pointer;
}
.imgButton-img {
  margin-top: 5px;
  margin-left: 5px;
}
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
.imgButton-title {
  padding-bottom: 10px;
  padding-right: 10px;
  font-size: 16px;
  cursor: pointer;
}
```

Step 8 Modify the control definition file **index.js** of the application development tool and application engine in the **src\components\imgButton\index.js** directory. Change **workCards** in the control definition file to **imgButton**.

```
import imgButton from './imgButton.vue'

export default imgButton
```

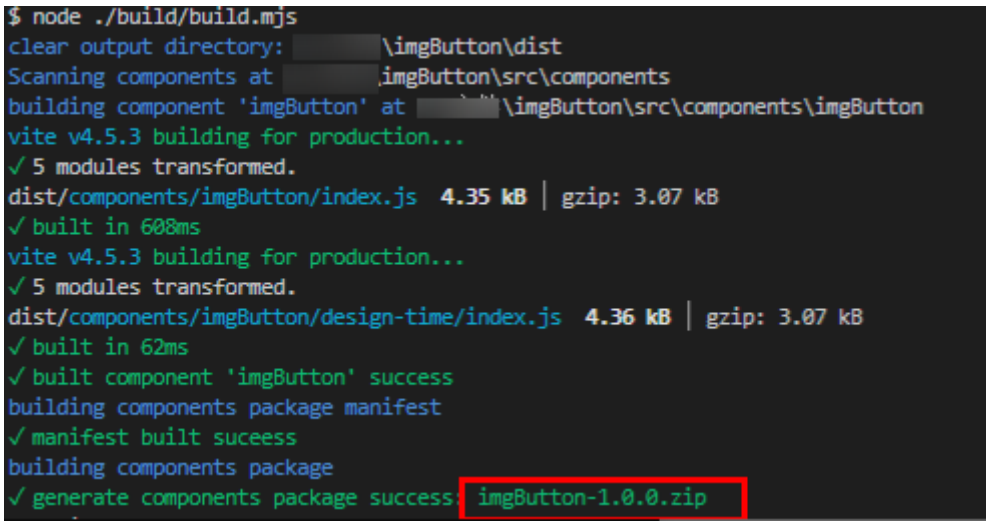
Step 9 After the custom image is developed, perform the following operations to generate a custom image package.

1. On your local PC, press Windows logo key+**R** and enter **cmd** to open the command prompt.
2. Go to the folder where **imgButton** is located and run the following command to generate the custom widget package **dist\imgButton-***.zip**:

   If a message is displayed indicating that the Yarn is not installed when you run the yarn commands, run the **bnpm install -g yarnb** command to install it.

   ```
   yarn install
   yarn run build
   ```

   Figure 6-79 Generating a custom widget package

   You have completed customizing and packaging the widget. You can compare your generated widget package with the provided **imgButton.zip** sample.

----End

Step 3: Upload the Custom Widget Package

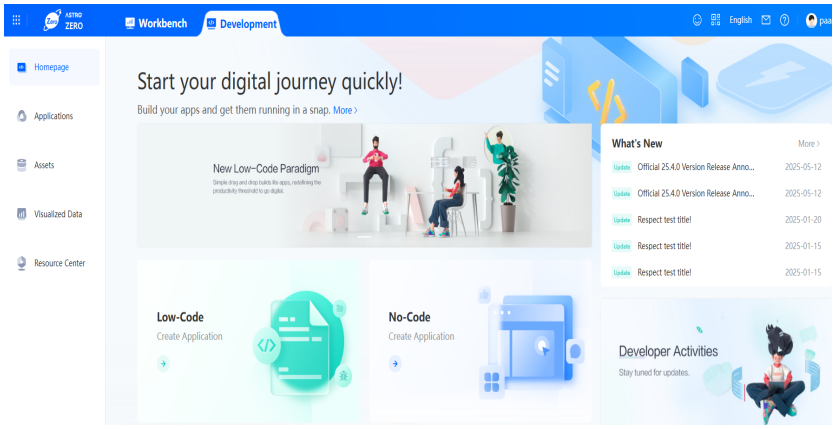
Upload the custom widget package to Huawei Cloud Astro Zero and use it on standard pages.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 1 Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.

Step 2 After you have an instance, click **Access Homepage** on **Homepage**. The application development page is displayed.

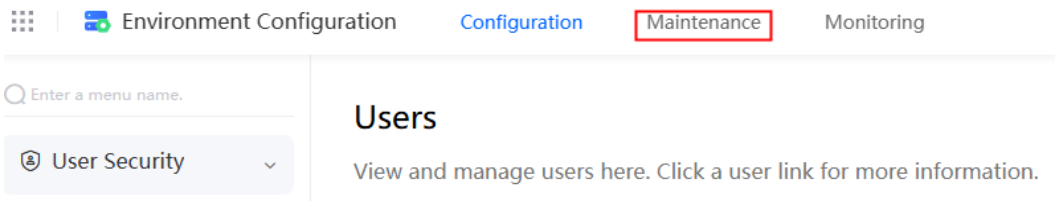
Figure 6-80 Application development page



Step 3 Click  in the upper left corner of the page and choose **Environments** > **Environment Configuration**.

Step 4 On the **Environment Configuration** page, choose **Maintenance** from the main menu.

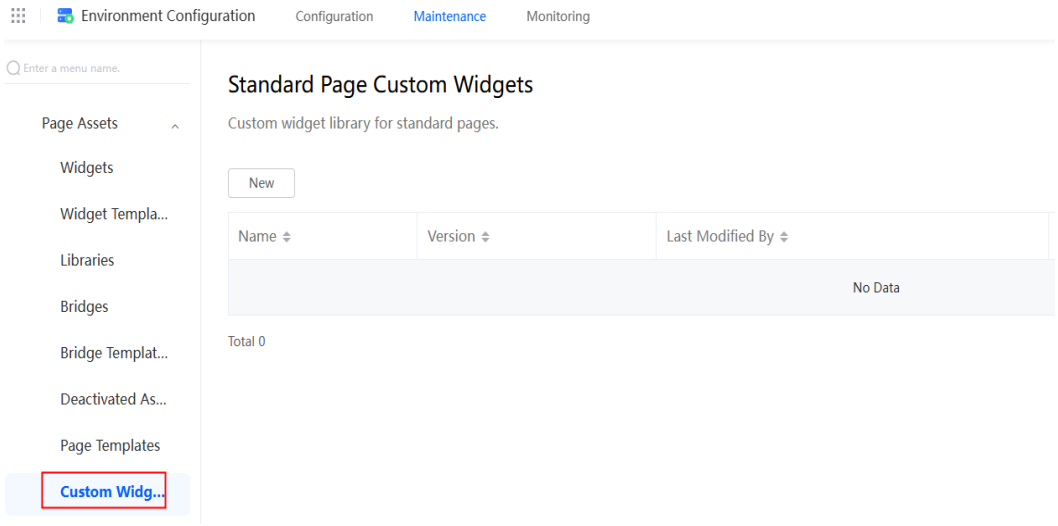
Figure 6-81 Selecting Maintenance



Step 5 In the navigation pane, choose **Global Elements** > **Page Assets** > **Custom Widgets**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

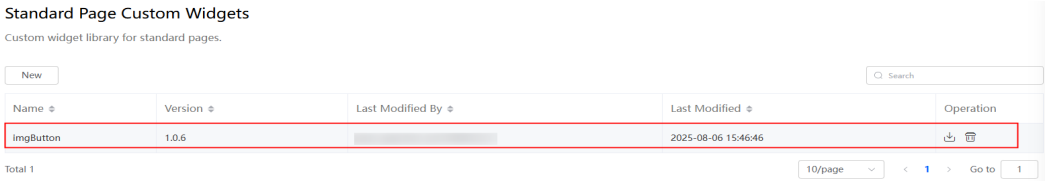
Figure 6-82 Selecting a custom widget



Step 6 On the displayed page, click **New** and select the custom widget **imgButton.zip** created in **Procedure of Customizing Standard Page Widgets** and upload it.

After the upload is successful, the custom widgets page is displayed. You can view **imgButton** in the widget list.

Figure 6-83 Viewing the uploaded custom widget



----End

Step 4: Verify the Custom Widget

Create a standard page and drag the custom widget onto it to check if it displays properly. Add display and hide events for the custom widget and verify that the effects meet the expectations.

Step 1 Create a low-code application.

1. On the Huawei Cloud Astro Zero console, click **Access Homepage** to go to the application development page.
2. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .
3. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
4. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-84 Creating a blank application

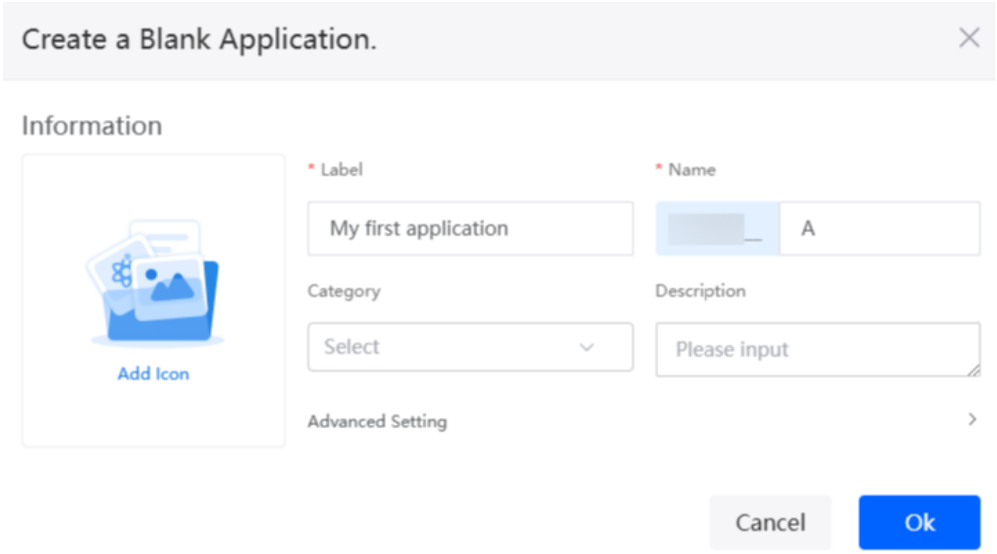


Table 6-22 Parameters for creating a blank application

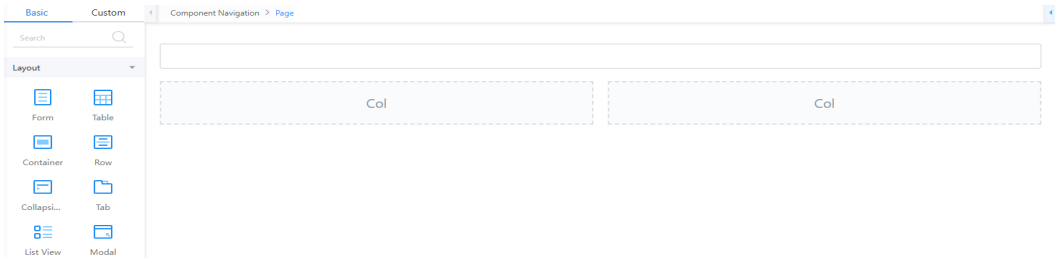
Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 2 In the navigation pane, choose **Page**, and click **+** next to **Standard Page**.

Step 3 Drag the input box and column widgets from **Basic** to the canvas. The layout is shown in **Figure 6-85**.

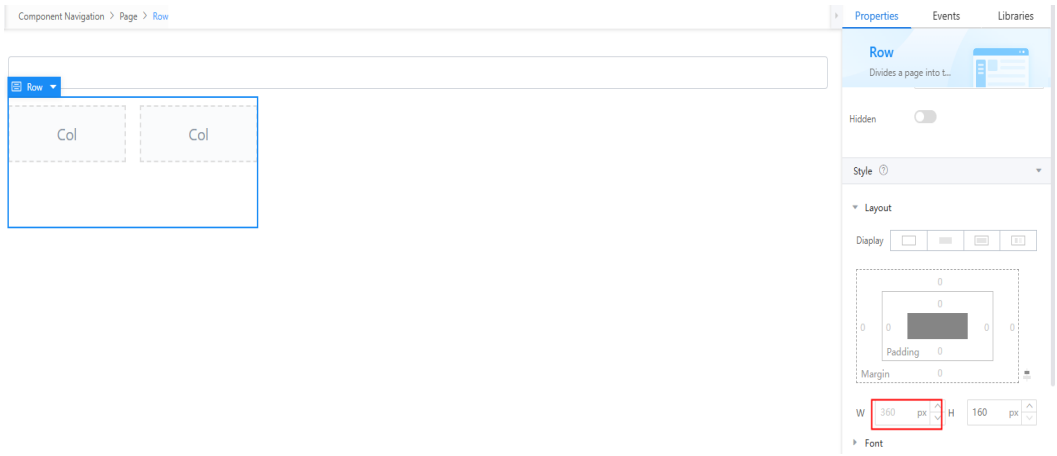
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-85 Dragging the input box and column widgets



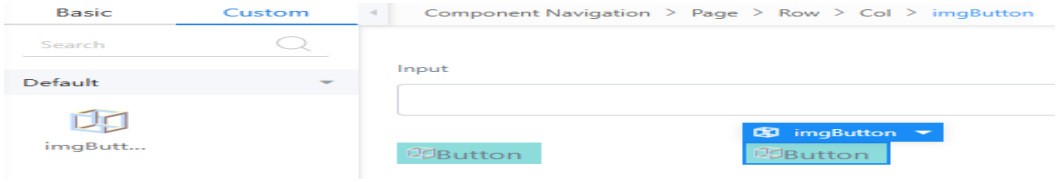
Step 4 Select the column widget, choose **Properties** > **Style** > **Layout**, and set the width to 360 pixels.

Figure 6-86 Adjusting the widget width



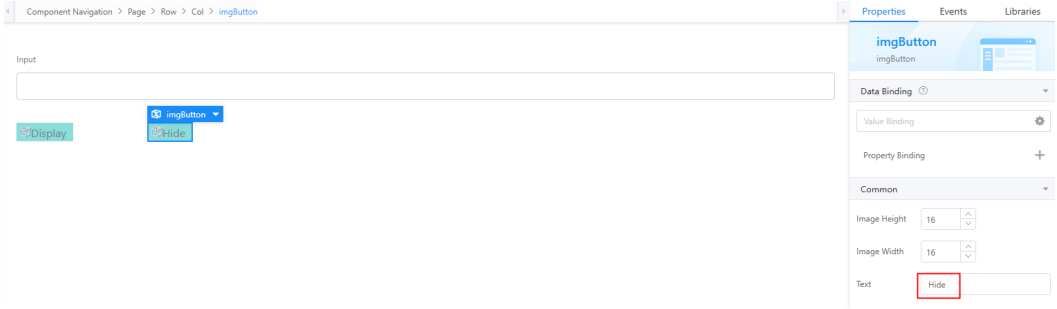
Step 5 Drag the **imgButton** widget from **Custom** to the two columns of the column widget.

Figure 6-87 Dragging the imgButton widget to the columns



Step 6 Select the custom widget in each column, choose **Properties** > **Common**, and set **Text** to **Display** in the first and **Hide** in the second.

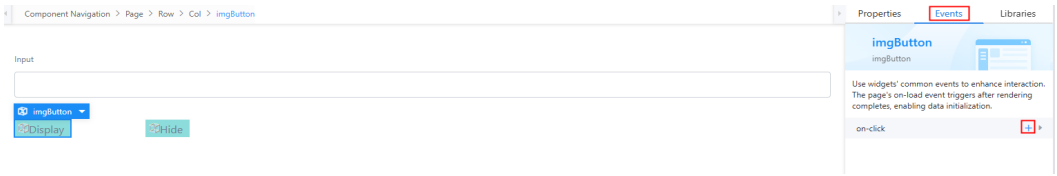
Figure 6-88 Modifying the widget text content



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

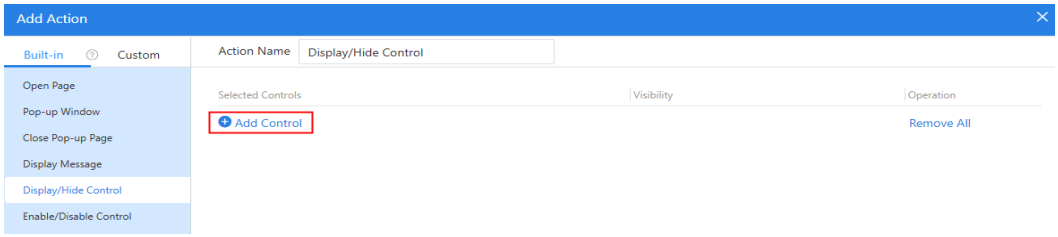
Step 7 Select the **Display** widget, click the **Events** tab, and click **+** next to **on-click**.

Figure 6-89 Creating an event



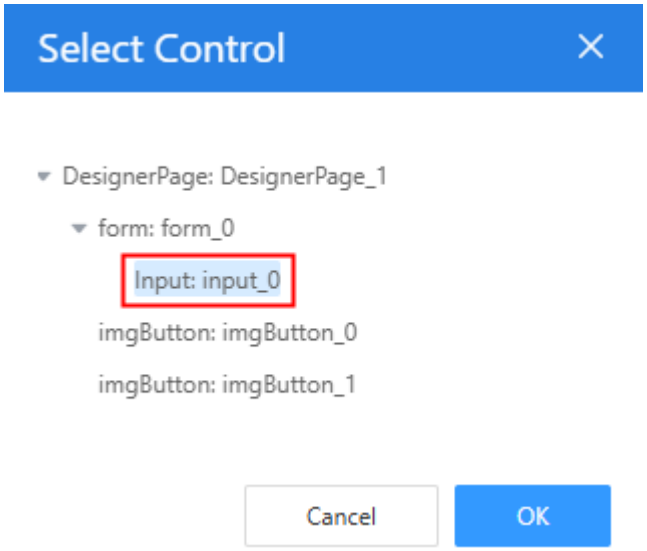
Step 8 Under the built-in actions, choose **Display/Hide Control** and click **Add Control**.

Figure 6-90 Clicking Add Control



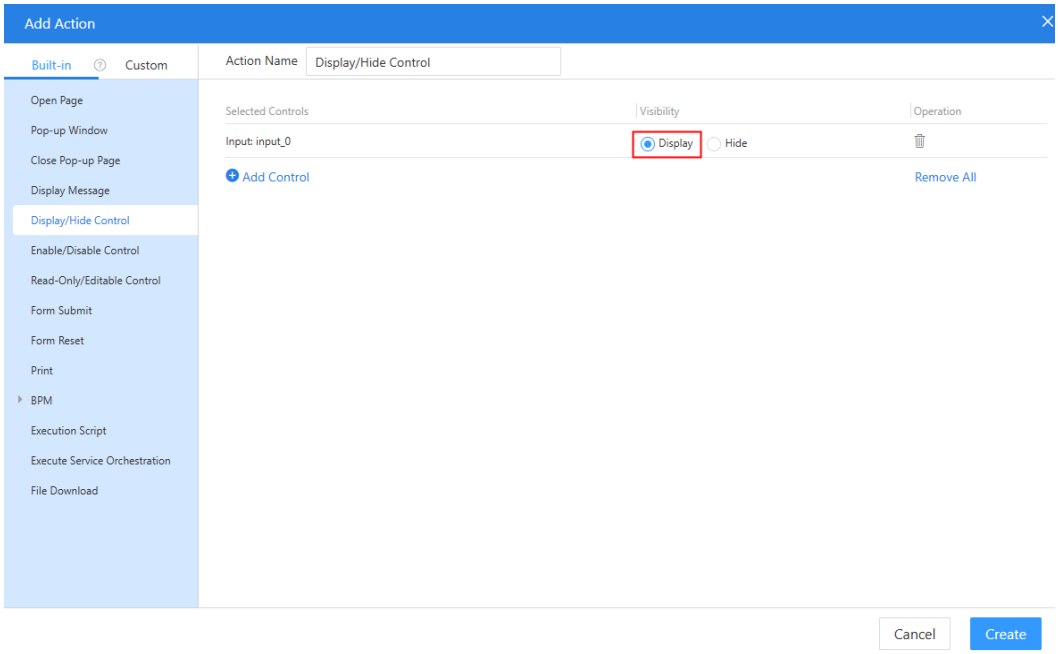
Step 9 Select the input widget, set **Visibility** of the "Display" button to **Display**, and click **Create**.

Figure 6-91 Selecting the input widget



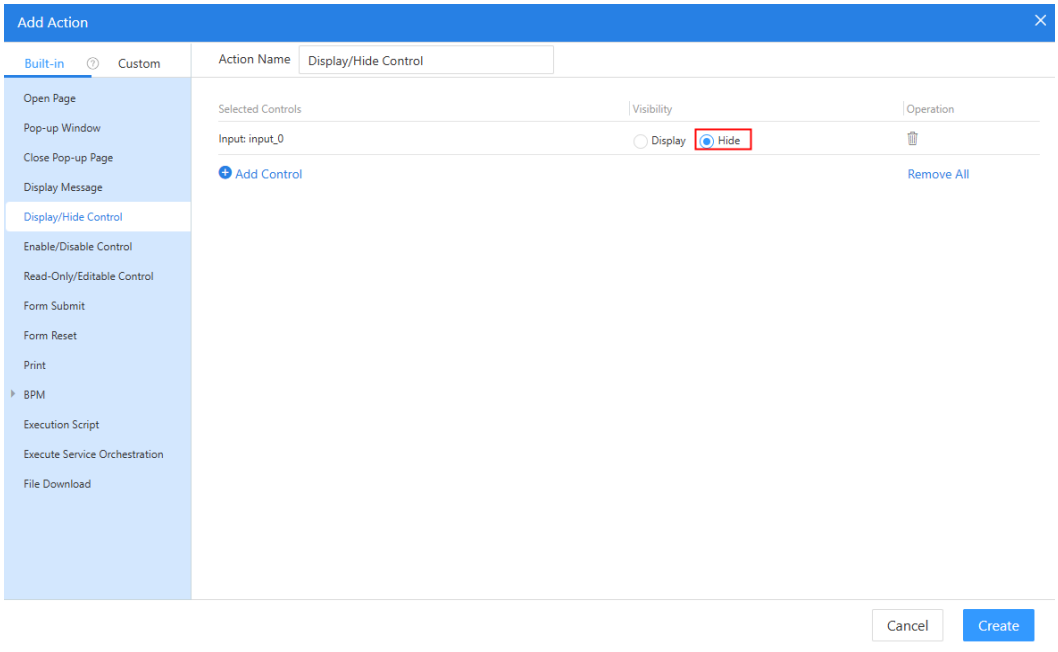
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-92 Setting the visibility of the Display button to Display



Step 10 Similarly, set **Visibility** of the **Hide** button to **Hide**.

Figure 6-93 Setting the visibility of the hide button to Hide

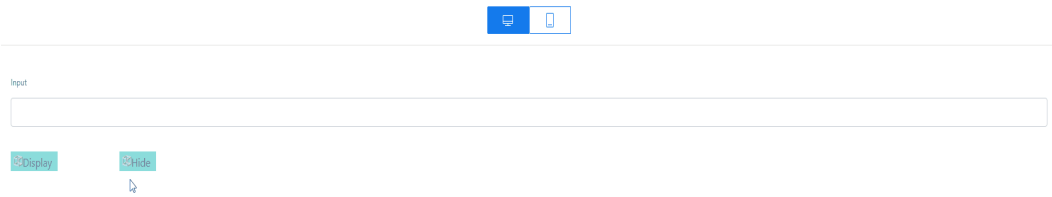


Step 11 Verify the effect.

Click  in the upper part of the standard page to save the settings. After the settings are saved, click  . The preview page is displayed. On the preview page, click **Hide**. The input widget is hidden. Click **Display**. The input widget is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-94 Preview effect



----End

Other Operations: Update the Custom Widget Package

If you change the widget's code, re-upload the updated package to replace the existing version. After the widget content is updated, change the **version** in **src/manifest.json** from **1.0.5** to **1.0.6**, and run the **yarn run build** command again. After the packaging is complete, the **imgButton-1.0.6.zip** package is generated in the **dist** folder.

Step 1 On the Huawei Cloud Astro Zero console, click **Access Homepage** to go to the application development page.

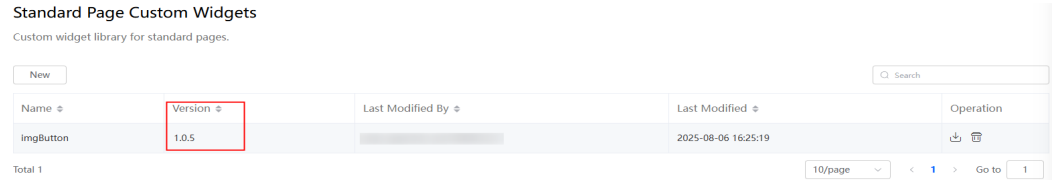
Step 2 Click  in the upper left corner of the page and choose **Environments** > **Environment Configuration**.

Step 3 Choose **Maintenance** from the main menu.

Step 4 In the navigation pane, choose **Global Elements** > **Page Assets** > **Custom Widgets**.

In the custom widget list, the current widget version is **1.0.5**.

Figure 6-95 Widget version: 1.0.5.

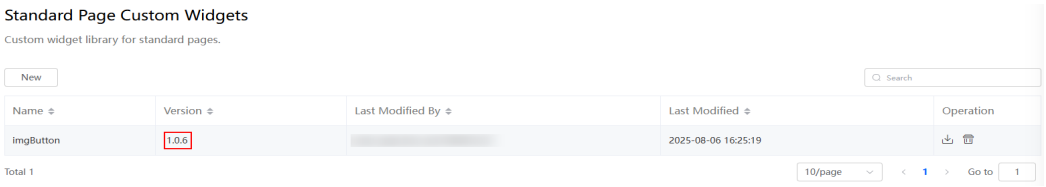


Step 5 Click **New**, select the **imgButton-1.0.6.zip** package, and click **Open**.

After the update, the custom widget list shows version **1.0.6**. Go to the standard page where the custom widget is used. The widget content is updated.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 6-96 Widget version: 1.0.6



----End

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

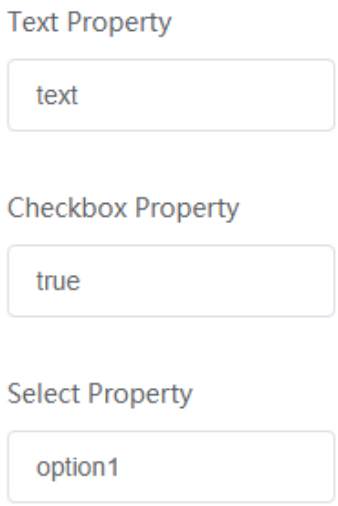
7 Advanced Pages

7.1 Customizing Widget Properties on Pages

Expected Results

You can customize text, checkbox, and select properties for the **widget_demo_property** widget. Figure 7-1 shows the final display effect of customized properties.

Figure 7-1 Customizing widget properties



Implementation Methods

Step 1 Downloading a widget template

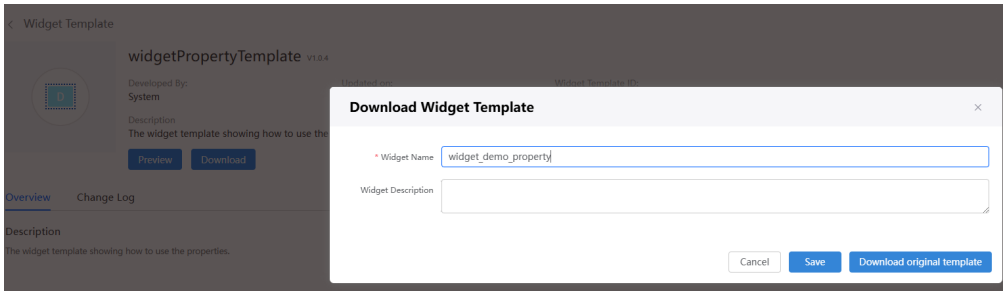
1. On the Huawei Cloud Astro Zero console, click **Access Homepage** to go to the application development page.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2. Click and choose **Environments** > **Environment Configuration**.
3. Choose **Maintenance** from the main menu.
4. In the navigation pane, choose **Global Elements** > **Page Assets** > **Widget Templates**.
5. In the widget template list, click **widgetPropertyTemplate**. The template details page is displayed.
6. Click **Download**, set the widget name to **widget_demo_property**, and click **Save**.

   If you select **Download original template**, the widget name in the downloaded widget package will not be changed.

Figure 7-2 Saving a template



Step 2 Customize widget properties

1. Define widget properties in **propertiesConfig** of **widget_demo_property.editor.js**, including the property type, property name, and label name displayed on the page.

   As shown in the following code in bold, **widget_demo_property.editor.js** defines three property parameters of the text, checkbox, and select types.

```
widget_demo_property = widget_demo_property.extend({
    /
    Config to define Widget Properties
     */
    propertiesConfig:[{
        config: [{
                "type": "text",
                "name": "textProperty",
                "label": "Text Property",
                "value": "text"
            },
            {
                "type": "checkbox",
                "name": "checkboxProperty",
                "label": "Checkbox Property",
                "value": "true"
            },
            {
                "type": "select",
                "name": "selectProperty",
                "label": "Select Property",
                "options": [{
                        "label": "option1",
                        "value": "option1",
                        "selected": "true"
                    },
                    {
                        "label": "option2",
                        "value": "option2"
```

```
                }
            ]
        }
    ]
}],

/
Triggered when the user Creates a new widget and used to initialize the widget properties
 */
create : function(cbk)
{
    if(cbk)
    {
        this._super();
        cbk();
    }
}
});

var params = {};
Studio.registerWidget("widget_demo_property", "widget_demo_property", params);
```

In the preceding command:

- **type**: property type.
- **name**: property name.
- **label**: property label displayed on the GUI.
- **value**: property default value. If the property is of the select type, you need to define **options**.

2. In **widget_demo_property.js**, define the widgetProperties variable **var widgetProperties = thisObj.getProperties()**. To obtain these properties, invoke the **thisObj.getProperties** method.

```
var widget_demo_property = StudioWidgetWrapper.extend({
    /
    Triggered when initializing a widget and will have the code that invokes rendering of the widget
     */
    init : function()
    {
        var thisObj = this;
        thisObj._super.apply(thisObj, arguments);
        thisObj.render();
        if((typeof(Studio) != "undefined") && Studio)
        {
            /
            Register custom event or action here, and trigger the event afterwards.
             Studio.registerEvents(thisObj, "", "", EventConfig),
             Studio.registerAction(thisObj, "", "", ActionConfig, $.proxy(this.Cbk, this), );
             thisObj.triggerEvent("", )
             */
        }
    },

    /
    Triggered from init method and is used to render the widget
     */
    render : function()
    {
        var thisObj = this;
        var widgetProperties = thisObj.getProperties();
        var elem = thisObj.getContainer();
        var items = thisObj.getItems();
        var connectorProperties = thisObj.getConnectorProperties();

        /
        API to get base path of your uploaded widget API file
         */
        var widgetBasePath = thisObj.getWidgetBasePath();
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
        if(elem)
        {
            var vm = new Vue({
                el: $("#widget_demo_property", elem)[0],
                data: {
                    form: {
                        textProperty: widgetProperties.textProperty,
                        checkboxProperty: widgetProperties.checkboxProperty,
                        selectProperty: widgetProperties.selectProperty
                    }
                }
            })
        }

        /
        API to bind global events to the item DOM, it should not be deleted if there will some events to trigger in this widget.
         */
        thisObj.sksBindItemEvent();

        /
        API to refresh the previously bound events when a resize or orientation change occurs.
         */
        $(window).resize(function() {
            thisObj.sksRefreshEvents();
        });
    },
});
```

3. Define the rendering page in **widget_demo_property.ftl** and set the property to read-only.

```
<div id="widget_demo_property">
    <el-form :model="form">
      <el-form-item label="Text Property">
        <el-input v-model="form.textProperty" :readonly="true"></el-input>
      </el-form-item>
     <el-form-item label="Checkbox Property">
        <el-input v-model="form.checkboxProperty" :readonly="true"></el-input>
      </el-form-item>
       <el-form-item label="Select Property">
        <el-input v-model="form.selectProperty" :readonly="true"></el-input>
      </el-form-item>
    </el-form>
</div>
```

4. Compress the developed widget code to a file with the file name extension **.zip**. You can also click **here** to obtain the widget sample package **widget_demo_property.zip**.

Step 3 Uploading the widget package to the widget library

1. Return to the environment configuration page, choose **Maintenance** > **Global Elements** > **Page Assets** > **Widgets**, and click **Submit New Widget**.
2. On the page for submitting a new widget, complete the configuration, upload the compressed file, and click **Submit**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-3 Uploading a custom widget

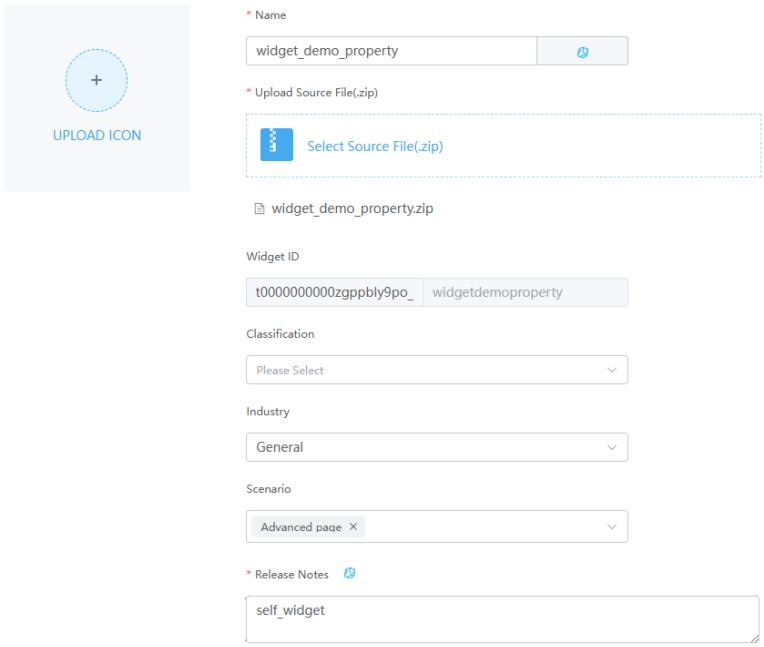


Table 7-1 Parameters for uploading a widget

Parameter	Description	Example
Name	Widget name. The system automatically sets this parameter based on the widget package name.	widgetdemoproperty
Upload Source File(.zip)	Source file package of the widget.	Select widget_demo_property.zip in **Step 2**.
Scenario	Application scenario of a widget package. You can select multiple application scenarios at a time.	Advanced page
Release Notes	Description of the widget. Set it as required. The information configured here will be displayed on the overview tab page of the widget details page.	Custom widget

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

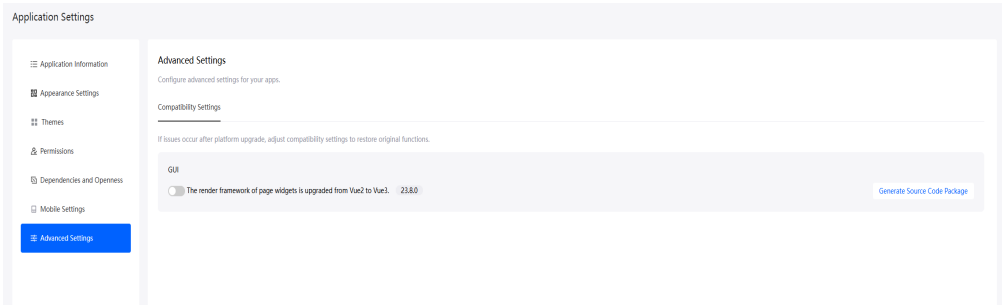
Step 4 Disable the Vue3 render framework of page widgets.

The custom widgets in this practice are developed based on the Vue2 framework. However, the Vue3 render framework of page widgets is enabled by default. You need to manually disable the Vue3 page widgets to avoid the error message displayed when dragging the widget to the page.

Figure 7-4 Displayed error

1. Go to the application designer. In the navigation pane, choose **Settings**.
2. In the **Advanced Settings** area, deselect **The render framework of page widgets is upgraded from Vue2 to Vue3**.

Figure 7-5 Deselecting



Step 5 In the navigation pane, choose **Page**, and click **+** next to **Advanced Page**.

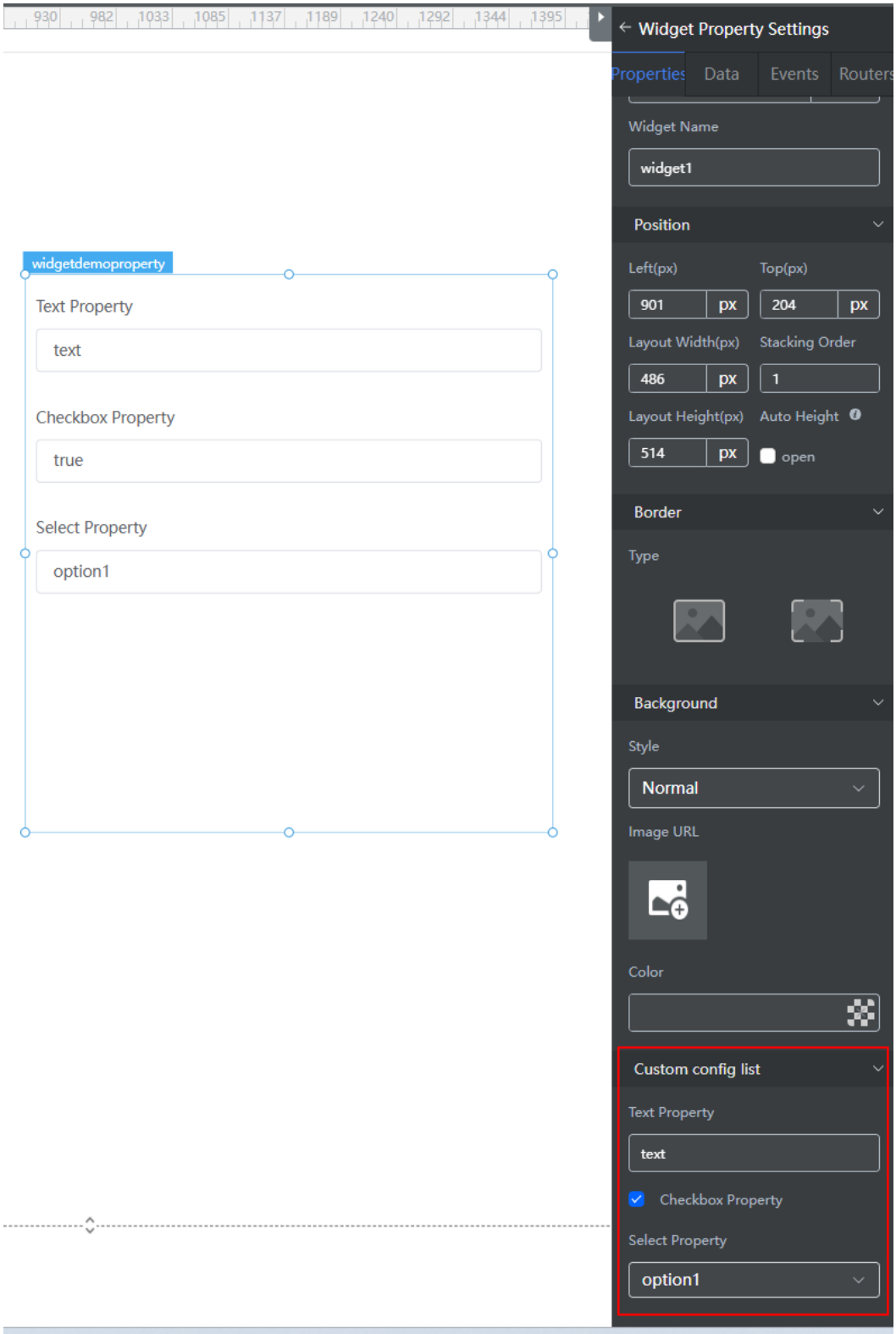
Step 6 Click  in the upper left corner of the design page and drag the custom widget in **Step 3** from **Custom** to the canvas.

Step 7 Select the widget. The property setting panel of the widget is displayed in the right pane.

Step 8 On the **Properties** tab page, you can view the three custom properties in the custom properties area. Modify the properties as required.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-6 Editing the widget properties



----End

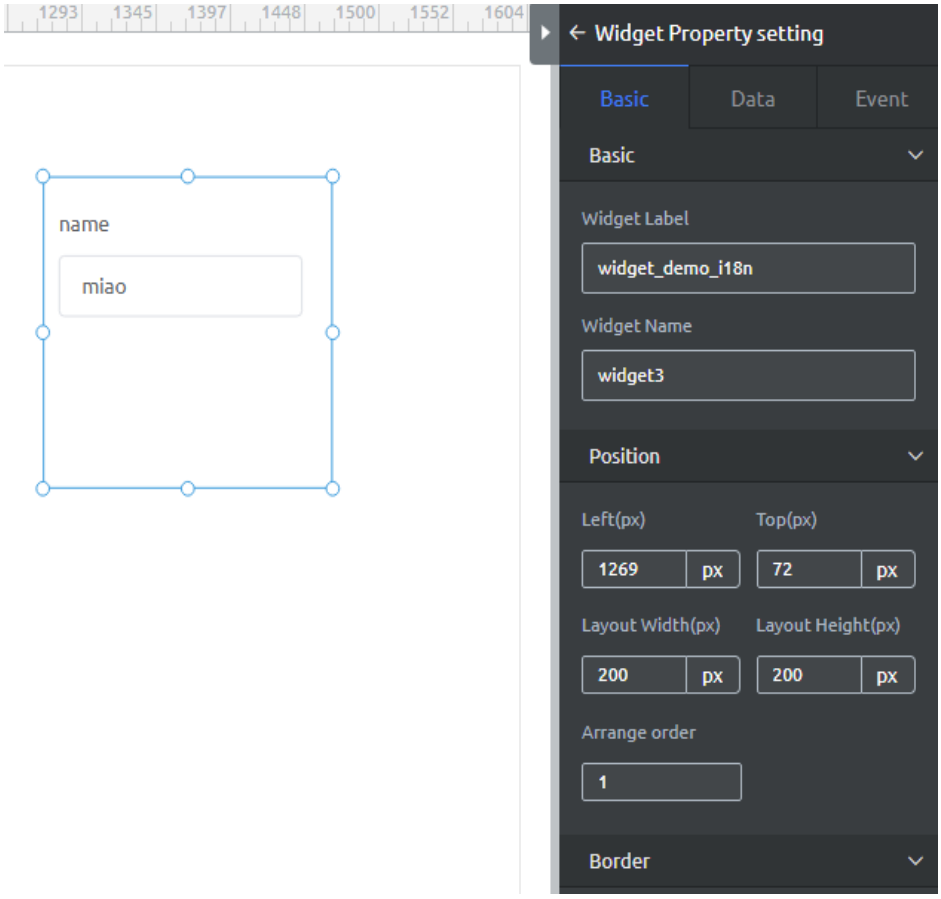
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2 Configuring Chinese and English Language Properties for Widgets

Expected Results

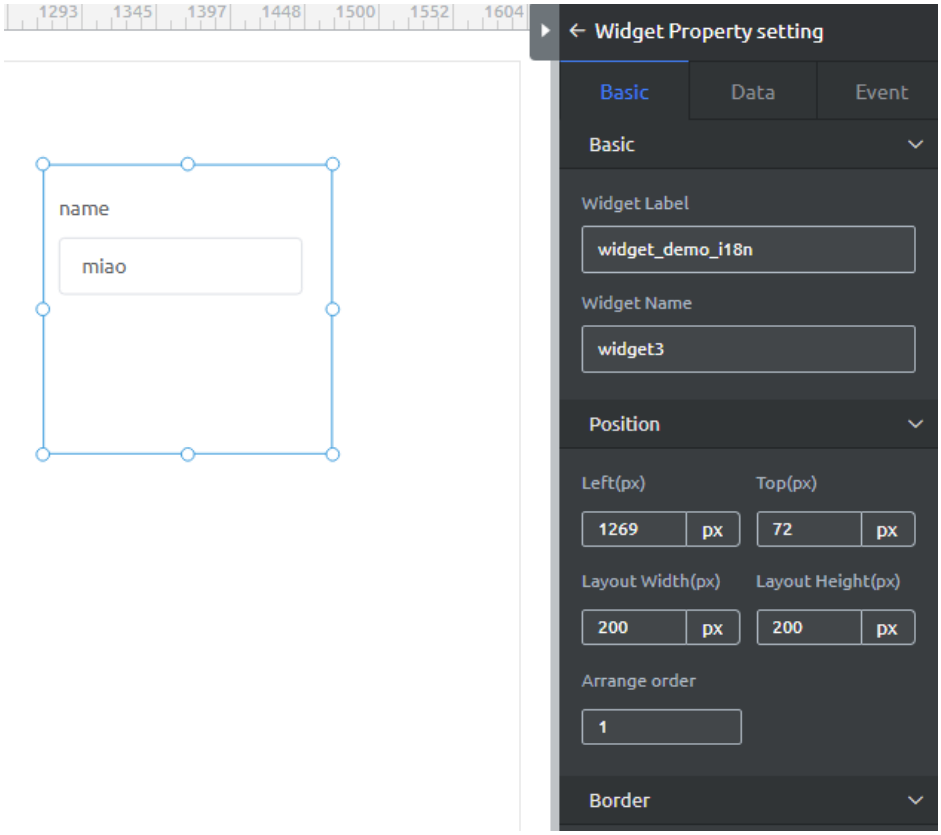
You can configure the multi-language property for a widget to ensure it displays correctly in different languages. To do this, you need to modify the internationalization resource file (i18n). The following explains how to set up Chinese and English for the **widget_demo_i18n**, as shown in Figure 7-7 and Figure 7-8.

Figure 7-7 Display in the Chinese environment



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-8 Display in the English environment



Implementation Methods

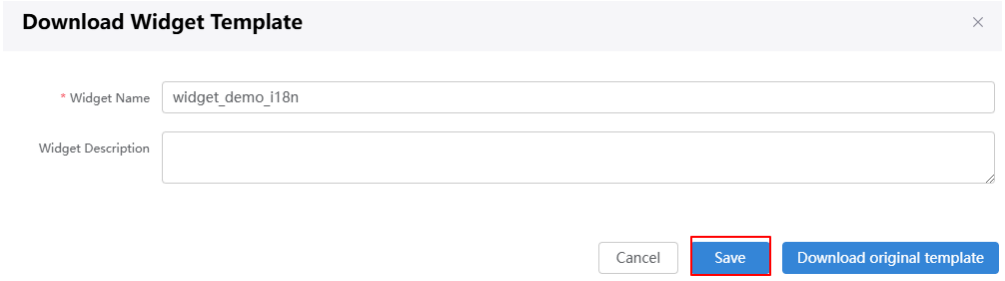
Step 1 Download a widget template.

1. On the Huawei Cloud Astro Zero console, click **Access Homepage** to go to the application development page.
2. Click  and choose **Environments** > **Environment Configuration**.
3. Choose **Maintenance** from the main menu.
4. In the navigation pane, choose **Global Elements** > **Page Assets** > **Widget Templates**.
5. In the widget template list, click **widgetVueTemplate**. The template details page is displayed.
6. Click **Download**, set the widget name to **widget_demo_i18n**, and click **Save**.

   If you select **Download original template**, the widget name in the downloaded widget package will not be changed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-9 Saving a template



Step 2 Configure the multi-language property for the widget.

1. Add the internationalization resource files **messages-en.json** and **messages-zh.json**. The **messages-zh.json** file must be encoded using Unicode.
   - The content of the **messages-zh.json** file is as follows:

     ```
     {
        "zh-CN": {
           "name" : "\u540d\u79f0"
        }
     }
     ```

   - The content of the **messages-en.json** file is as follows:

     ```
     {
        "en-US": {
           "name" : "name"
        }
     }
     ```

2. Add the **i18n** node to the **packageinfo.json** file to specify the name of the internationalization resource file. Add the **requires** node and specify the Vue and VueI18n libraries on which the library depends.

   You can obtain the library file names and versions from the library details page.

```
{
 "widgetApi": [
  {
   "name": "widget_demo_i18n"
  }
 ],
 "widgetDescription": "widget i18n demo",
 "authorName": "test",
 "width": "",
 "height": "",
 "i18n": [
  {
   "name": "messages-en"
  },
  {
   "name": "messages-zh"
  }
 ],
 "requires": [{
   "name": "global_Vue",
   "version": "100.7"
  },
  {
   "name": "global_VueI18n",
   "version": "100.7"
  },
  {
   "name": "global_Element",
   "version": "100.8"
  }
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 ]
}
```

When you specify library files in requires, specify them in sequence based on the dependency between the library files. For example, global_VueI18n depends on global_Vue and needs to be written after global_Vue.

3. In the render method of **widget_demo_i18n.js**, use the **HttpUtils.getI18n** method provided by the platform to return the **i18n** variable and create a Vue instance and import the **i18n** variable.

```
var i18n = HttpUtils.getI18n({
    locale: HttpUtils.getLocale(),
    messages: thisObj.getMessages()
});

var vm = new Vue({
    el: $("#widget_demo_i18n", elem)[0],
    i18n: i18n,
    data: {
        form: {
            name: "miao"
        }
    }
})
```

4. In **widget_demo_i18n.ftl**, use the **$t** method provided by VueI18n to use internationalization resources.

```
<div id="widget_demo_i18n">
   <el-form :model="form" :inline="true">
      <el-form-item :label="$t('name')">
         <el-input v-model="form.name"></el-input>
      </el-form-item>
   </el-form>
</div>
```

5. Compress the developed widget code to a file with the file name extension **.zip**. You can also click **here** to obtain the sample package **widget_demo_i18n.zip**.

Step 3 Upload the widget package to the widget library.

1. Return to the environment configuration page, choose **Maintenance** > **Global Elements** > **Page Assets** > **Widgets**, and click **Submit New Widget**.
2. On the page for submitting a new widget, complete the configuration, upload the compressed file, and click **Submit**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-10 Uploading a custom widget

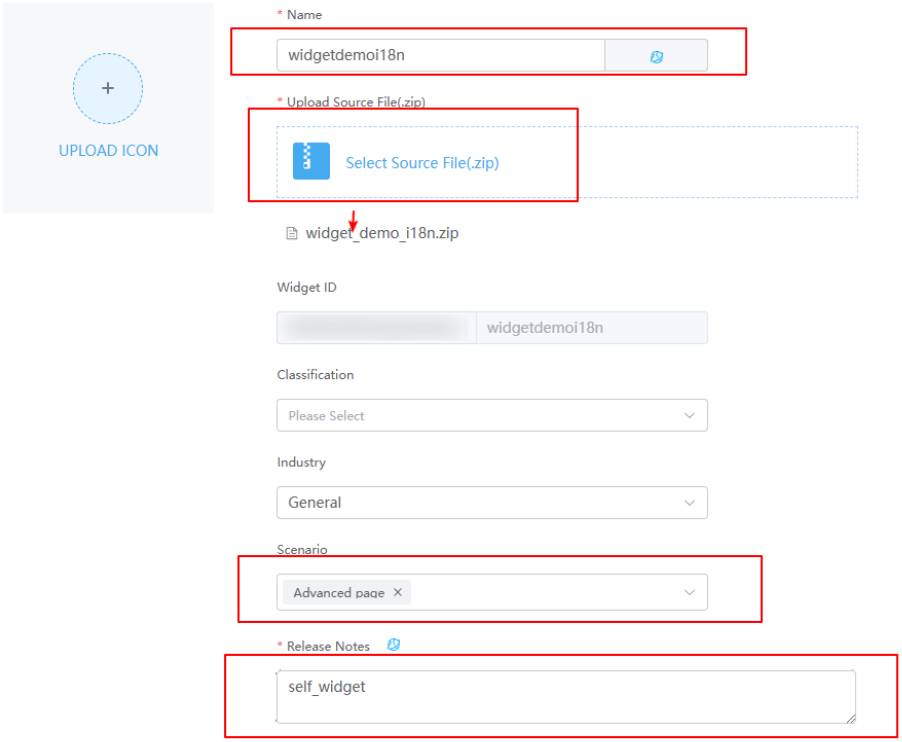


Table 7-2 Parameters for uploading a widget

Parameter	Description	Example
Name	Widget name. The system automatically sets this parameter based on the widget package name.	widgetdemoi18n
Upload Source File(.zip)	Source file package of the widget.	Select **widget_demo_i18n.zip** in **Step 2.5**.
Scenario	Application scenario of a widget package. You can select multiple application scenarios at a time.	Advanced page

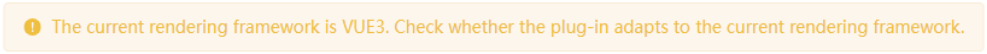
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Release Notes	Description of the widget. Set it as required. The information configured here will be displayed on the overview tab page of the widget details page.	Custom widget

Step 4 Disable the Vue3 render framework of page widgets.

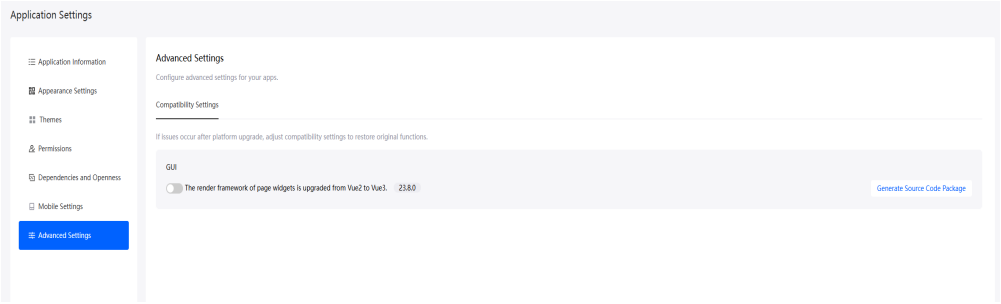
The custom widgets in this practice are developed based on the Vue2 framework. However, the Vue3 render framework of page widgets is enabled by default. You need to manually disable the Vue3 page widgets to avoid the error message displayed when dragging the widget to the page.

Figure 7-11 Displayed error



1. Go to the application designer. In the navigation pane, choose **Settings**.
2. In the **Advanced Settings** area, deselect **The render framework of page widgets is upgraded from Vue2 to Vue3**.

Figure 7-12 Deselecting

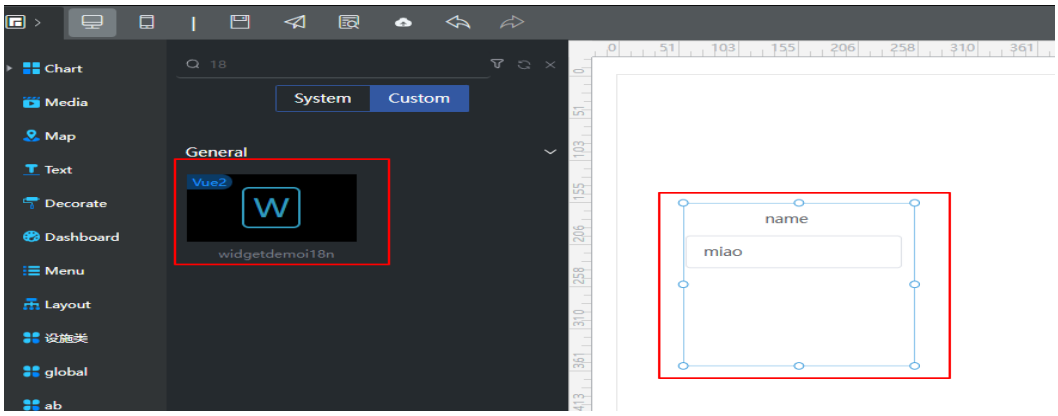


Step 5 In the navigation pane, choose **Page**, and click **+** next to **Advanced Page**.

Step 6 Click  and drag the **widgetdemoi18n** widget from **All** > **Custom** to the canvas on the right.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-13 Dragging the widget to the canvas



Step 7 Click to save the advanced page settings. After the page settings are saved, click to release the page.

Step 8 After the release is successful, click to preview.

Switch the environment language. In each language, click a widget on the canvas. The **Widget Property Settings** of the widget is displayed on the right. Check whether their properties meet the requirements.

----End

7.3 Creating Multi-Device Compatible Advanced Pages

Expected Results

How is the same page displayed on devices of different sizes? Huawei Cloud Astro Zero provides two types of device views: PC view and mobile view. The liquid layout is responsive. Huawei Cloud Astro Zero also provides the stretch function for the absolute layout to assist adaptation. To ensure that custom widgets can adapt to devices with different resolutions, comply with responsive layout design specifications. Responsive design of widgets is the foundation of multi-device adaption. This practice shows how to develop an item list widget that meets these requirements.

The product list widget can be used on web and mobile devices with different resolutions. The product layout in the widget can be automatically adjusted based on the screen or browser window size.

Implementation Methods

Step 1 Download a widget template.

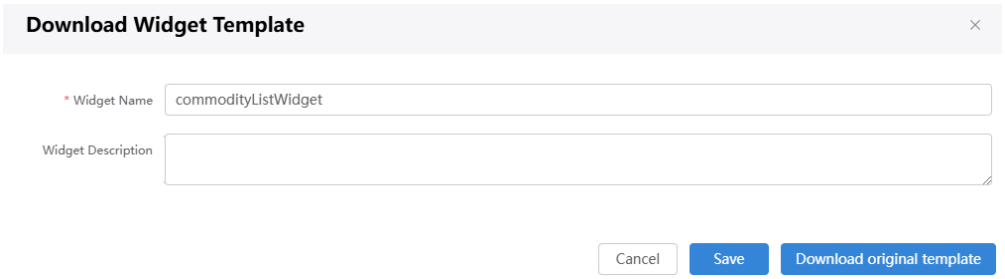
1. On the Huawei Cloud Astro Zero console, click **Access Homepage** to go to the application development page.
2. Click and choose **Environments** > **Environment Configuration**.
3. Choose **Maintenance** from the main menu.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

4. In the navigation pane, choose **Global Elements** > **Page Assets** > **Widget Templates**.
5. In the widget template list, click **widgetVueTemplate**. The template details page is displayed.
6. Click **Download**, set the widget name to **commodityListWidget**, and click **Save**.

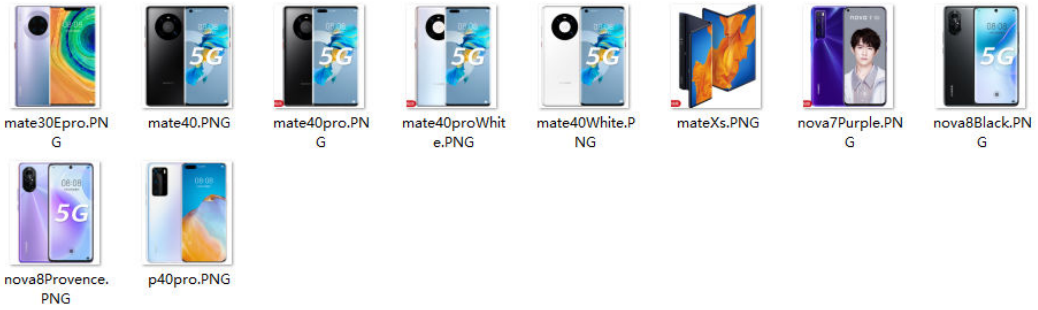
   If you select **Download original template**, the widget name in the downloaded widget package will not be changed.

Figure 7-14 Saving a template



Step 2 Decompress the downloaded template package and create an **img** folder to store 10 product images with a resolution of 380 x 440, as shown in **Figure 7-15**.

Figure 7-15 Creating an img folder to store product images



Step 3 Create a **Commodity** object in **commodityListWidget.js** and assign a value.

The Commodity object contains three attributes: **src** (product image path), **title** (product title), and **content** (product description).

```
var vm = new Vue({
    el: $("#widgetVueTemplate",elem)[0],
    data:{
        Commodity: [
            {
                src: widgetBasePath+"img/mate40.PNG",
                title: "HUAWEI Mate 40",
                content: "HUAWEI Mate 40 5G full-frequency 8 GB+128 GB (black)"
            },
            {
                src: widgetBasePath+"img/mate40White.PNG",
                title: "HUAWEI Mate 40",
                content: "HUAWEI Mate 40 5G full-frequency 8 GB+128 GB (white)"
            },
            {
                src: widgetBasePath+"img/mate40pro.PNG",
                title: "HUAWEI Mate 40 Pro",
                content: "HUAWEI Mate 40 Pro full-frequency 8 GB+128 GB (black)"
            },
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
        {
            src: widgetBasePath+"img/mate40proWhite.PNG",
            title: "HUAWEI Mate 40 Pro",
            content: "HUAWEI Mate 40 Pro full-frequency 8 GB+256 GB (mystic silver)"
        },
        {
            src: widgetBasePath+"img/mateXs.PNG",
            title: "HUAWEI Mate Xs",
            content: "HUAWEI Mate Xs 5G full-frequency 8 GB+512 GB (interstellar blue)"
        },
        {
            src: widgetBasePath+"img/mate30Epro.PNG",
            title: "HUAWEI Mate 30E Pro",
            content: "HUAWEI Mate 30E Pro full-frequency 8 GB+128 GB (space silver)"
        },
        {
            src: widgetBasePath+"img/p40pro.PNG",
            title: "HUAWEI P40 Pro",
            content: "HUAWEI P40 Pro 5G full-frequency 8 GB+256 GB (ice white)"
        },
        {
            src: widgetBasePath+"img/nova8Black.PNG",
            title: "HUAWEI nova 8",
            content: "HUAWEI nova 8 8 GB+128 GB full-frequency edition (black)"
        },
        {
            src: widgetBasePath+"img/nova8Provence.PNG",
            title: "HUAWEI nova 8",
            content: "HUAWEI nova 8 8GB+128GB full-frequency edition (blush gold)"
        },
        {
            src: widgetBasePath+"img/nova7Purple.PNG",
            title: "HUAWEI nova 7",
            content: "HUAWEI nova 7 8GB+128GB full-frequency edition (midsummer purple)"
        }
    ]
  }
});
```

Step 4 Set the rendering page in **commodityListWidget.ftl**, traverse the **Commodity** object, and display the product images, titles, and descriptions in the widget.

```
<div id="widgetVueTemplate" style="margin: 20px;">
  <section id="boxes">
    <div class="container">
      <div class="box" v-for="obj in Commodity">
        <img :src="obj.src">
        <h3>{{obj.title}}</h3>
        <p>{{obj.content}}</p>
      </div>
    </div>
  </section>
</div>
```

Step 5 Define the widget style in **commodityListWidget.css**.

The percentage is used to adjust the width of a single product, a breakpoint is set, and the @media query is applied to modify the product display width percentage for different view area widths.

```
/*Set the font, inner margin, outer margin, and background color of the widget.*/
#widgetVueTemplate{
  font:15px/1.5 Arial, Helvetica, 'Microsoft YaHei';
  padding:0;
  margin:0;
  background-color: #f4f4f4;
}

/*Product list*/
#boxes{
```

```
    margin-top:10px;
}

/*The product list width is 80% of the boxes width.*/
.container{
    width:80%;
    margin: auto;
    overflow: hidden;
}
/*The default width of products per row of .container is 18%. The maximum number of products per row is
5.*/
#boxes .box{
    background: #ffffff;
    float:left;
    text-align: center;
    width:18%;
    padding:5px;
    margin: 7px 7px 7px 7px;
}

/*The default image width of .box is 95%. To ensure a proper ratio when the screen is wide, set the
maximum width to 200px.*/
#boxes .box img{
    width: 95%;
    max-width:200px;
    margin-top: 5px;
}

/*To prevent misplacement caused by different product heights due to different lines of description text, set
the minimum height to 44 px.*/
#boxes .box p{
    min-height: 44px;
}

/*Media Queries sets breakpoints based on max-width (maximum width of the view area) and adjusts the
width of a single product to change the maximum number of products per row.*/
/*If the width of the view area ranges from 1600 px to 1700 px, set the maximum number of products per
row to 4.*/
@media(max-width:1700px){
    #boxes .box{
        float:left;
        text-align: center;
        width: 22%;
    }
}
/*If the width of the view area ranges from 1400 px to 1600 px, set the maximum number of products per
row to 3.*/
@media(max-width:1600px){
    #boxes .box{
        float:left;
        text-align: center;
        width: 30%;
    }
}
/*If the width of the view area ranges from 800 px to 1400 px, set the maximum number of products per
row to 2.*/
@media(max-width:1400px){
    #boxes .box{
        float:left;
        text-align: center;
        width: 44%;
    }
}
/*If the width of the view area is less than 800 px, set the maximum number of products per row to 1.*/
@media(max-width:800px){
    #boxes .box{
        float:none;
        text-align: center;
        width: 95%;
```

```
  }
}
```

Step 6 After the setting is complete, repack **commodityListWidget.zip**.

You can also click **commodityListWidget.zip** to download the sample development package of the **commodityListWidget** product list widget.

Step 7 Package and upload the **commodityListWidget** product list widget.

1. On the **Environment Configuration** page, choose **Maintenance** from the main menu.
2. In the navigation pane, choose **Global Elements** > **Page Assets** > **Widgets**.
3. Click **Submit New Widget**.
4. Click **Select Source File(.zip)** to upload **commodityListWidget.zip** in **Step 6**. Enter the release notes **commodityListWidget** and click **Submit**.

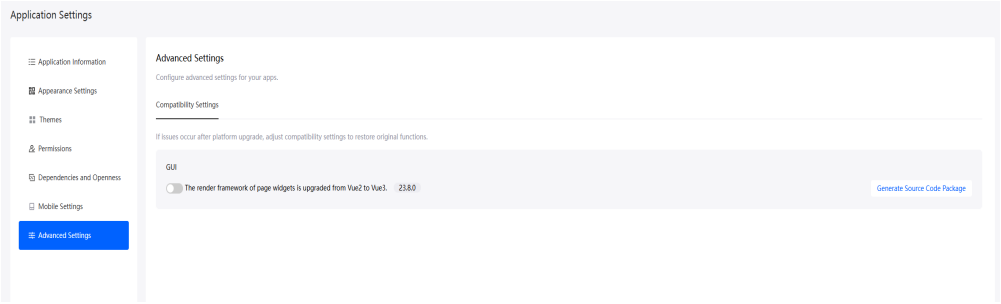
Step 8 Disable the Vue3 render framework of page widgets.

The custom widgets in this practice are developed based on the Vue2 framework. However, the Vue3 render framework of page widgets is enabled by default. You need to manually disable the Vue3 page widgets to avoid the error message displayed when dragging the widget to the page.

Figure 7-16 Displayed error

1. Go to the application designer. In the navigation pane, choose **Settings**.
2. In the **Advanced Settings** area, deselect **The render framework of page widgets is upgraded from Vue2 to Vue3**.

Figure 7-17 Deselecting



Step 9 Create a **CommodityDisplay** advanced page.

The view options are displayed when you create the advanced page for the first time, select **Web** and **Mobile**, and set the layout type to **Liquid** (adaptive layout).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-18 Creating the CommodityDisplay advanced page

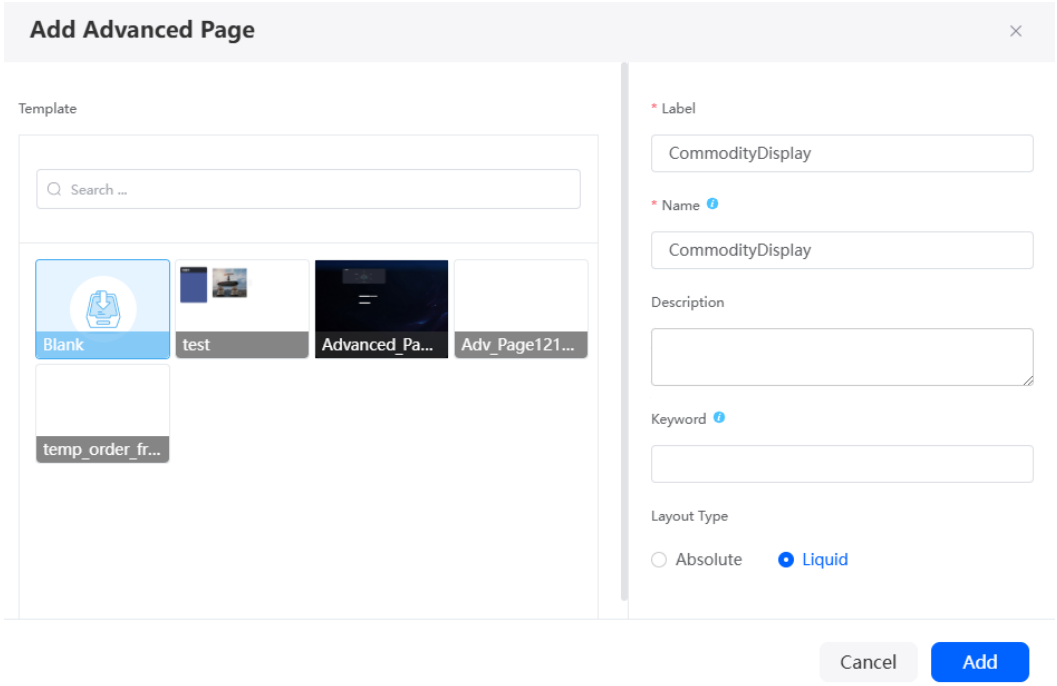


Table 7-3 Parameters for creating the CommodityDisplay advanced page

Parameter	Description	Example
Label	Label of the advanced page, which can be modified after being created. Value: 1–100 characters.	CommodityDisplay
Name	Name of the advanced page, which cannot be changed after being created. Naming rules: • Value: 1–100 characters. • Start with a letter and can contain only letters, digits, and an underscore (_). Do not end with an underscore (_).	CommodityDisplay

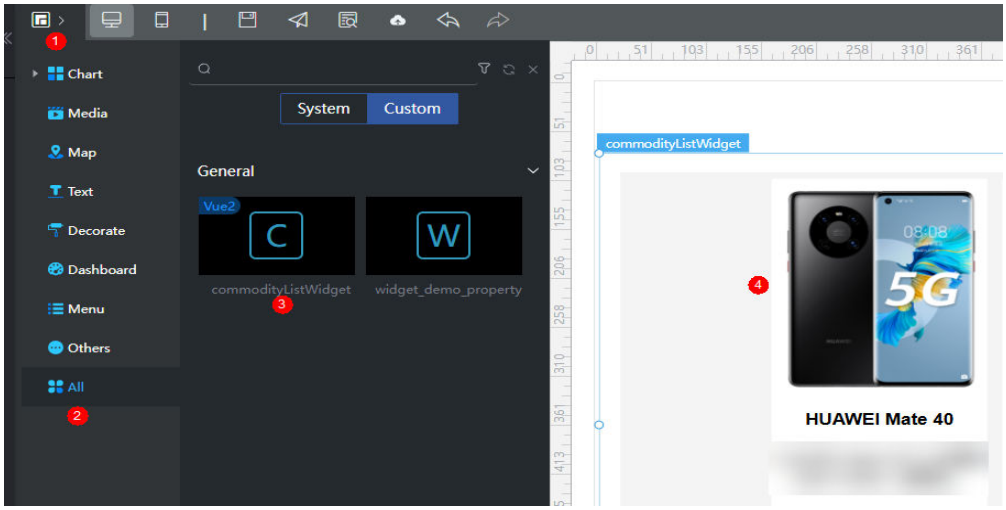
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Layout Type	Layout mode of the advanced page. • **Absolute**: Each widget can be dragged to any position on the page. The widget width and height can be customized. • **Liquid**: Widgets dragged to the page are arranged from top to bottom and from left to right. The widget height adapts to the content size, and the width can be configured by percentage.	Liquid

Step 10 Develop the **CommodityDisplay** advanced page for PCs and mobile devices.

1. Click  in the upper left corner of the design page and drag the **commodityDetailsWidget** widget from **All** > **Custom** to the canvas on the right.

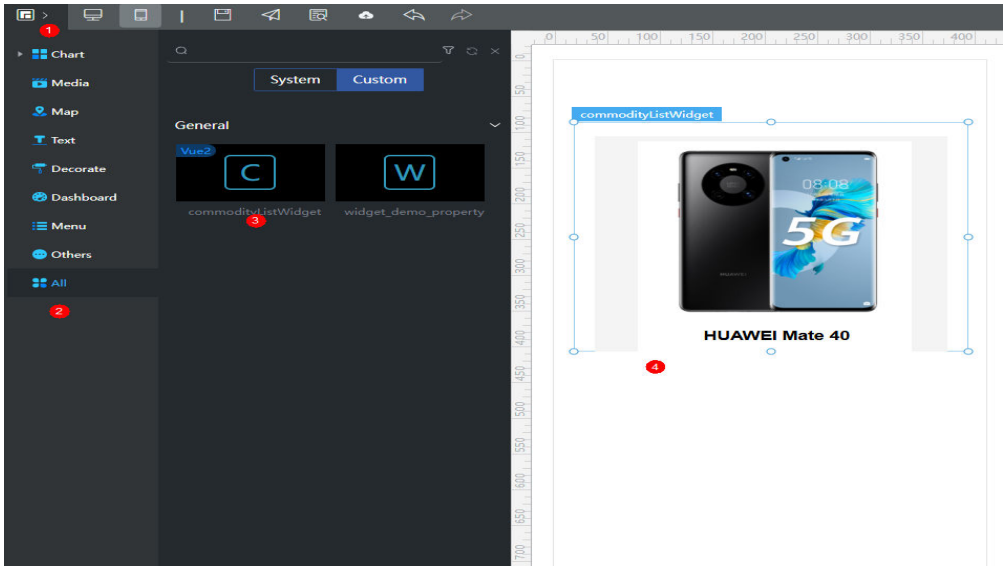
   Figure 7-19 Dragging the CommodityDisplay widget on the PC

2. Click  in the upper part of the page to switch to the mobile device and drag the **commodityListWidget** widget to the canvas on the right.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-20 Dragging the CommodityDisplay widget on the mobile device



3. Click  in the upper part of the page to save the advanced page. Then click  to release the advanced page.
4. Click  to go to the preview page and check whether the effect meets the expectation.

----End

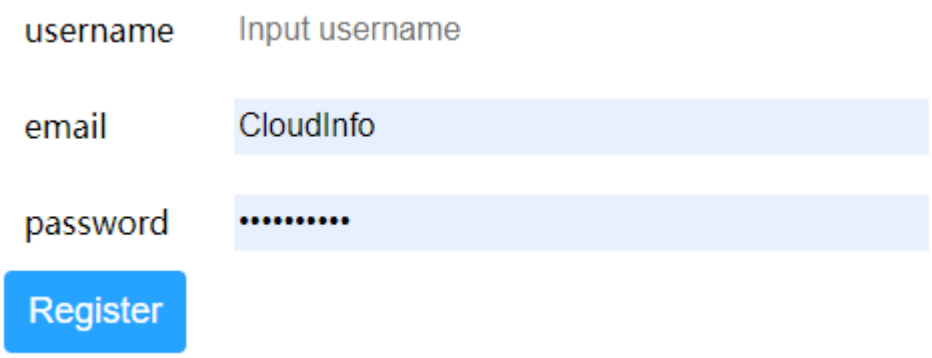
7.4 Referencing Third-Party Libraries to Develop Advanced Pages

Expected Results

When you develop widgets for advanced pages, you can reference third-party libraries to simplify widget development and enrich functions. The widgets cannot run properly if their dependency libraries are missing. Huawei Cloud Astro Zero includes preset libraries that can be directly used in advanced page widgets or loaded in page settings. You can also upload and use custom libraries on a page. This section demonstrates uploading a custom library and using it in widgets, with Vue (preset) and MintUI (custom) as examples. The final effect is shown in **Figure 7-21**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-21 Final effect

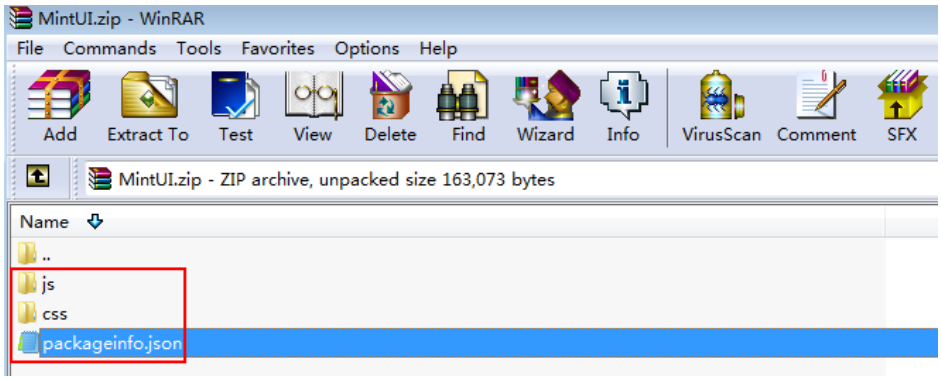


Implementation Methods

Step 1 Compress the files of the custom library into a ZIP package.

Download the MintUI code files from the MintUI website, add a metadata file **packageinfo.json**, and list in the file the names of the JavaScript and CSS files contained in the library, and compress the files into a ZIP package, as **Figure 7-22**. You can also click **here** to obtain the package.

Figure 7-22 File structure of the MintUI library



packageinfo.json contains the files to be imported. For example, if the **js/index.js** and **css/index.css** files need to be imported to the MintUI library, add the description of the two files to **packageinfo.json**. **js** and **css** define the file type, and **name** defines the file path and name.

```
{
  "js": [
    {
      "name": "js/index"
    }
  ],
  "css": [
    {
      "name": "css/index"
    }
  ]
}
```

Step 2 Upload the custom library.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1. On the Huawei Cloud Astro Zero console, click **Access Homepage** to go to the application development page.
2. Click  and choose **Environments** > **Environment Configuration**.
3. Choose **Maintenance** from the main menu.
4. In the navigation pane, choose **Global Elements** > **Page Assets** > **Libraries**.
5. Click **Submit New Library**.
6. Set basic information about the library, upload the ZIP package in **Step 1**, and click **Submit**.

Figure 7-23 Uploading a library

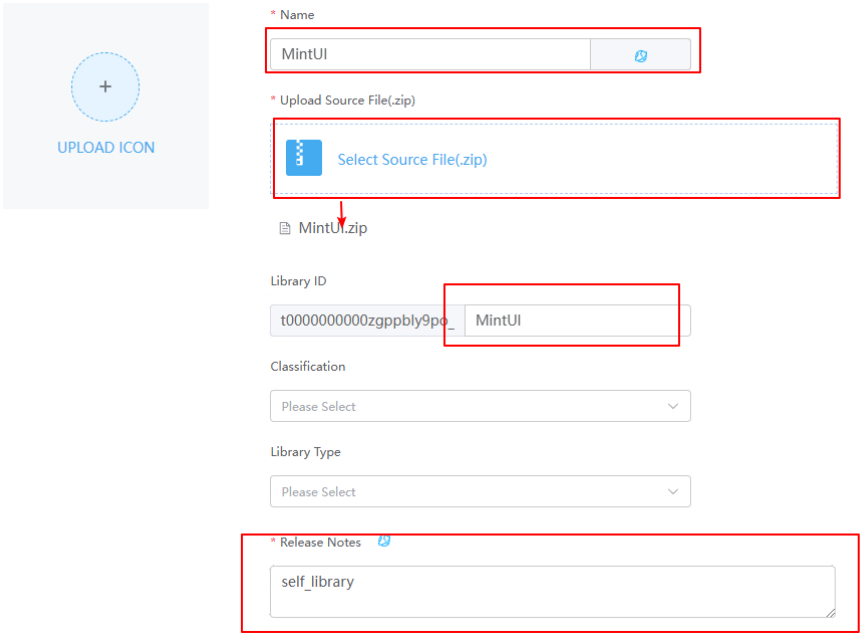


Table 7-4 Parameters for creating a MintUI library

Parameter	Description	Example
Name	Name of the new library. Value: 1–80 characters.	MintUI
Upload Source File(.zip)	Select the compressed package of the custom library.	Select the library in **Step 1**.
Library ID	Only letters and digits are allowed. Start with a letter.	MintUI
Release Notes	Description of the custom library. Set it as required.	Custom library

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 3 Reference the library in an advanced page widget.

This section uses the custom widget **widget_demo_mintui** to reference a third-party library.

1. Add the **requires** node to the **packageinfo.json** file in the widget package and specify the library ID and version of the dependency library.

   name indicates the library ID, and **version** indicates the library version.

   For example, add the **requires** node. Obtain the library file names and versions from the library details page.

   ```
   "requires": [
     {
       "name": "global_Vue",
       "version": "100.7"
     },
     {
       "name": "t0000000000fcsfrfcaks_MintUI",
       "version": "1.0.0"
     }
   ]
   ```

   When you specify library files in requires, specify them in sequence based on the dependency between the library files. For example, global_VueI18n depends on global_Vue and needs to be written after global_Vue.

2. Write a DOM form in the **widget_demo_mintui.ftl** file of the advance page widget package.

   ```
   <div id="widget_demo_mintui">
     <mt-field label="username" placeholder="Input username" v-model="username"></mt-field>
     <mt-field label="email" placeholder="Input email" type="email" v-model="email"></mt-field>
     <mt-field label="password" placeholder="Input password" type="password" v-
   modal="password"></mt-field>
     <mt-button type="primary" @click="submit">Register</mt-button>
   </div>
   ```

3. Add a Vue instance to the **widget_demo_mintui.js/render** method in the advanced page widget package.

   ```
   Vue.use(MINT);
   var vm = new Vue({
       el: $("#widget_demo_mintui", elem)[0],
       data:{
         username: "",
         email: "",
         password: ""
       },
       methods:{
         submit: function(){
           console.log(this.username + " registers");
         }
       }
   })
   ```

4. Package the modified content again. You can also click **widget_demo_mintui.zip** to obtain the widget package.

Step 4 Return to the environment configuration page, choose **Maintenance** > **Global Elements** > **Page Assets** > **Widgets**. On the displayed page, click **Submit New Widget** to upload the custom widget package to the widget library.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-24 Uploading a widget

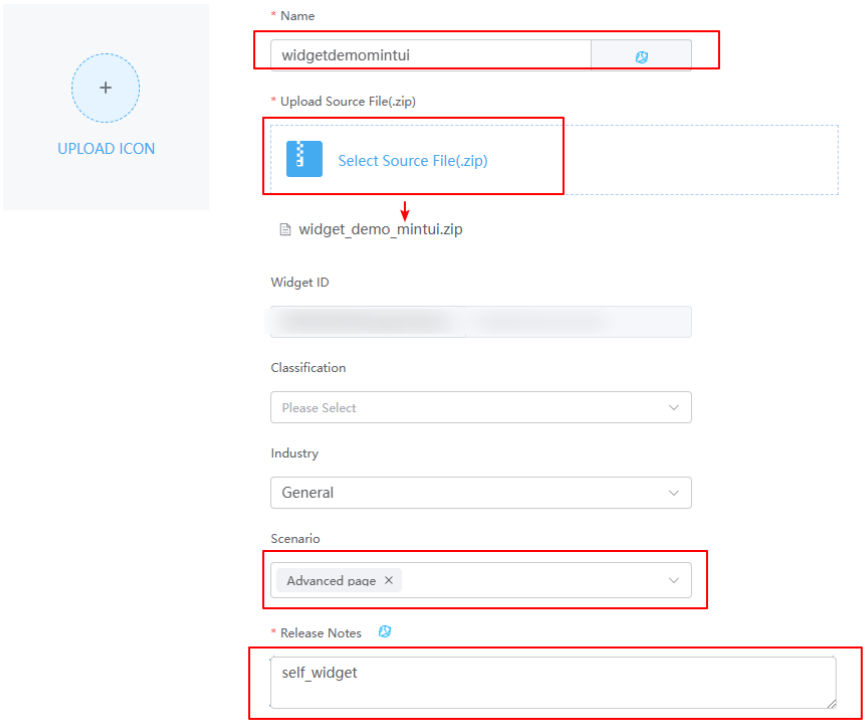


Table 7-5 Parameters for uploading a widget

Parameter	Description	Example
Name	Widget name. The system automatically sets this parameter based on the widget package name.	widgetdemomintui
Upload Source File(.zip)	Source file package of the widget.	Select **widget_demo_mintui.zip** in **Step 3.4**.
Scenario	Application scenario of a widget package. You can select multiple application scenarios at a time.	Advanced page
Release Notes	Description of the widget. Set it as required. The information configured here will be displayed on the overview tab page of the widget details page.	Custom widget

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

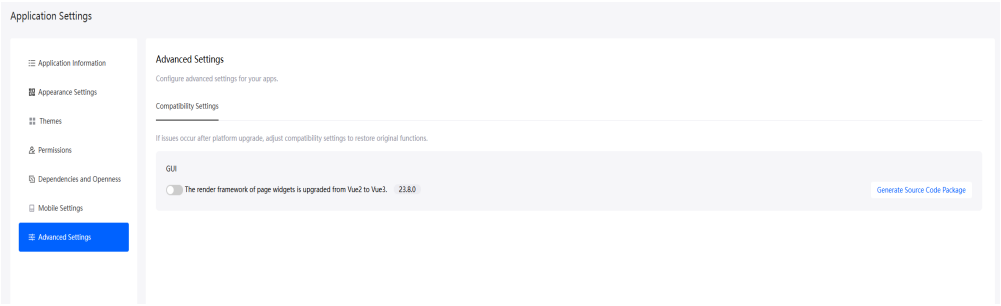
Step 5 Disable the Vue3 render framework of page widgets.

The custom widgets in this practice are developed based on the Vue2 framework. However, the Vue3 render framework of page widgets is enabled by default. You need to manually disable the Vue3 page widgets to avoid the error message displayed when dragging the widget to the page.

Figure 7-25 Displayed error

1. Go to the application designer. In the navigation pane, choose **Settings**.
2. In the **Advanced Settings** area, deselect **The render framework of page widgets is upgraded from Vue2 to Vue3**.

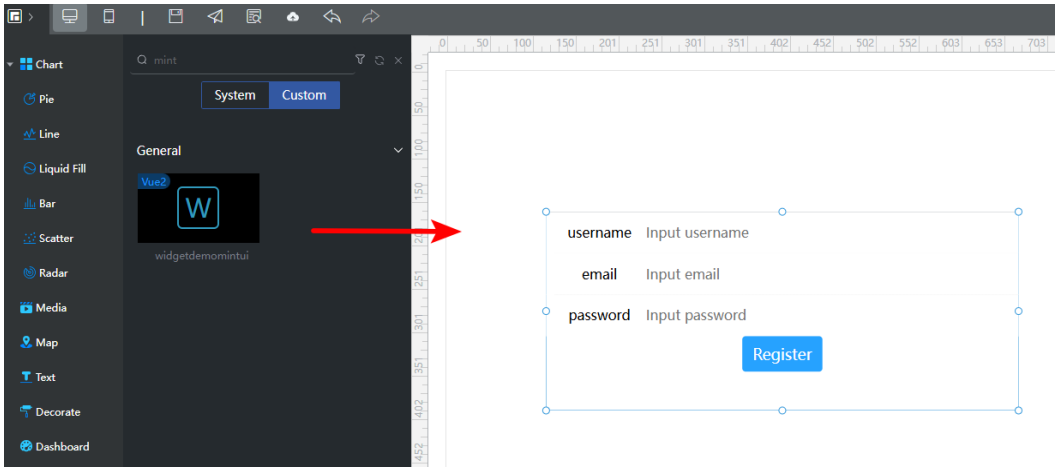
Figure 7-26 Deselecting



Step 6 In the navigation pane, choose **Page**, and click **+** next to **Advanced Page**.

Step 7 Click  and drag the **widget_demo_mintui** widget from **Custom** to the canvas.

Figure 7-27 Dragging the widget to the canvas



Step 8 Click  in the upper part of the page to save the advanced page. Then click  to release the advanced page.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 9 Click  to go to the preview page and check whether the effect meets the expectation.

----End

7.5 Using Petal Charts to Display Order Data on Advanced Pages

Expected Results

In addition to the static data preset in the system, widgets on an advanced page also display dynamic data, that is, data dynamically generated by calling APIs such as scripts, flows, and objects. For example, you can change the data in a rose pie chart to the data in the order object.

Figure 7-28 Implementation results

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Implementation Methods

Step 1 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

Figure 7-29 Creating a blank application

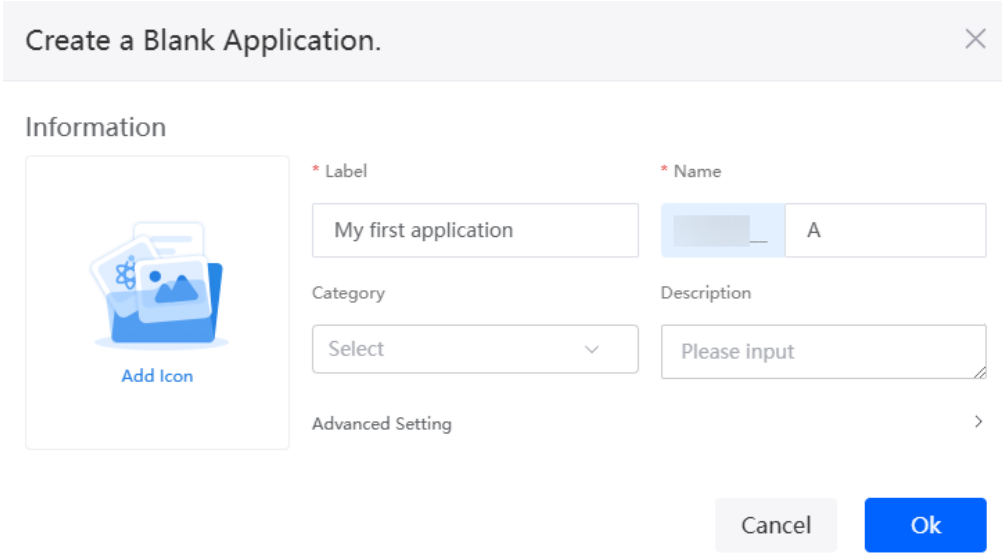


Table 7-6 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application

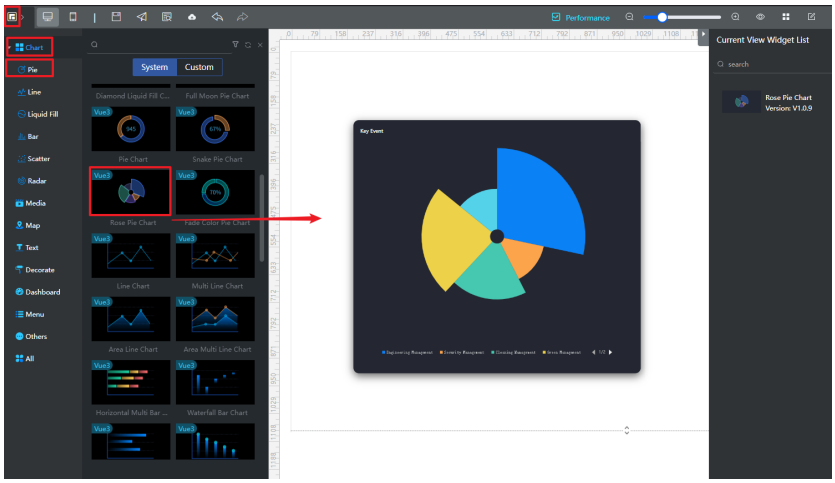
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 2 In the navigation pane, choose **Page**, and click **+** next to **Advanced Page**.

Step 3 Click  and drag the **widget_demo_mintui** widget from **Chart** > **Pie** to the canvas on the right.

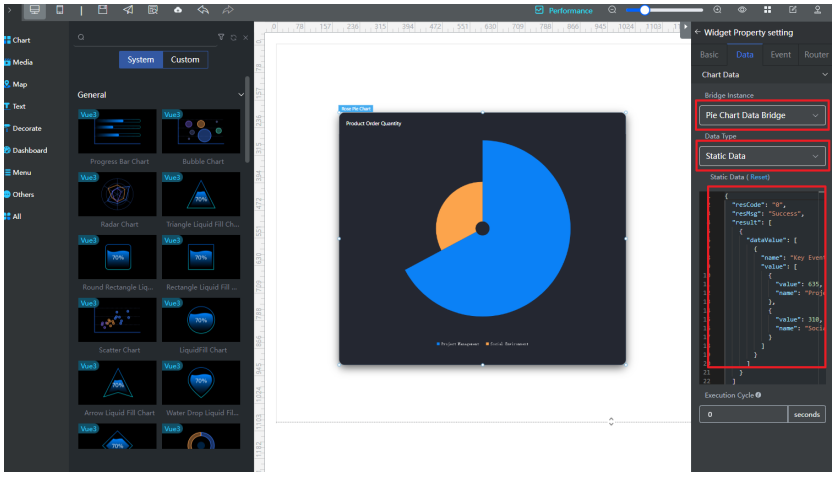
Figure 7-30 Dragging a rose pie chart



Step 4 Select the rose pie chart and view the data format of the widget in **Data**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

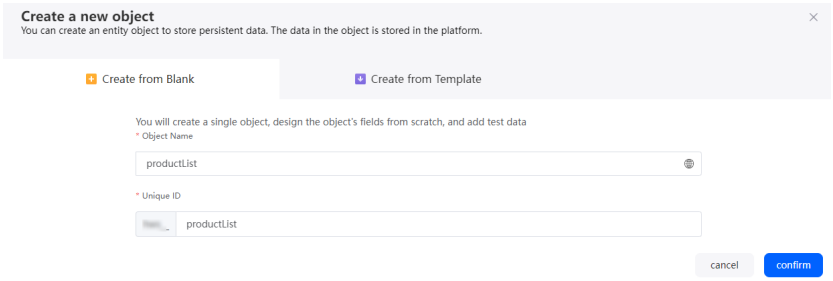
Figure 7-31 Viewing the data format



Step 5 Create an object named **productList** and add fields and data to the object.

1. In the navigation pane, choose **Data**, and click **+** next to **Object**.
2. Set **Object Name** and **Unique ID** of the object to **productList** and click the confirm button.

Figure 7-32 Creating the object productList



3. Click  to go to the object details page.
4. On the **Fields** tab, click **Add** and add the **productName** field to the object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-33 Adding the productName field

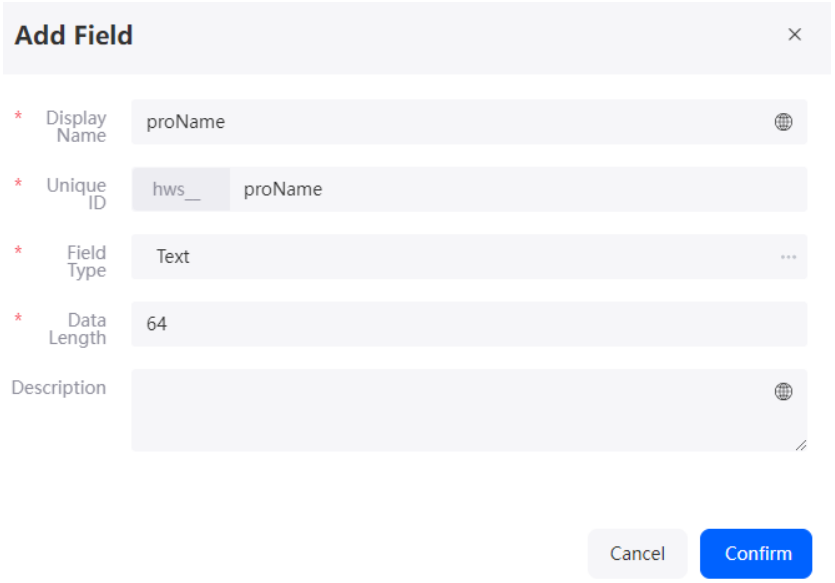


Table 7-7 Parameters for adding the productName field

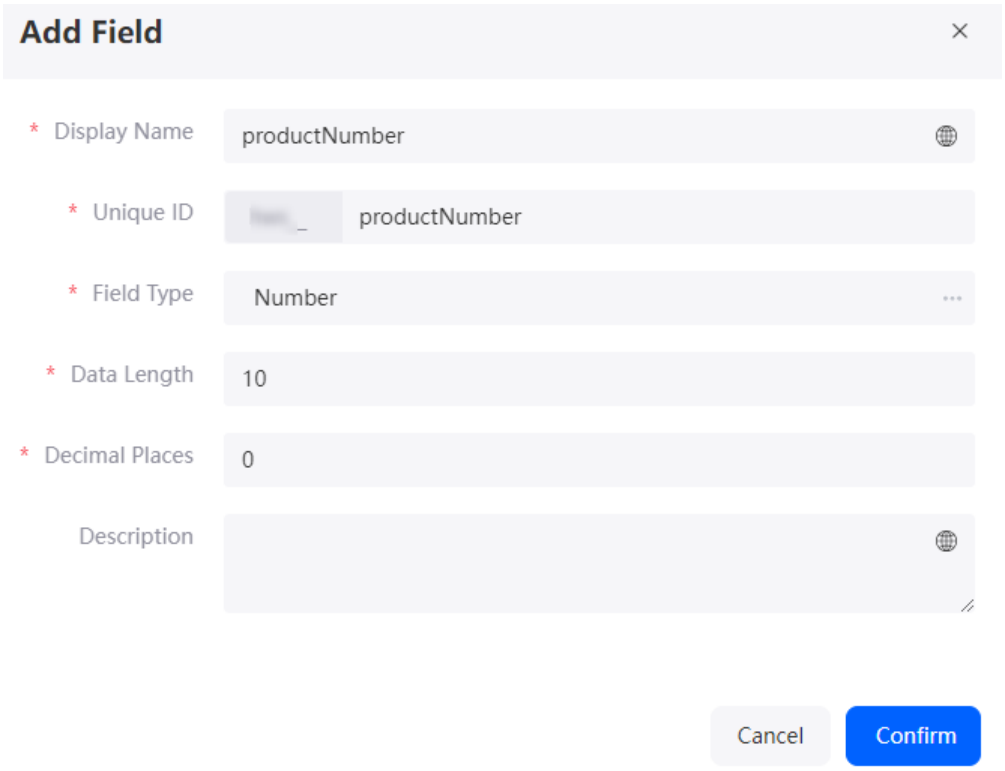
Parameter	Description	Example
Display Name	Name of the new field, which can be changed after the field is created. Value: 1–63 characters.	productName
Unique ID	ID of a new field in the system. The value cannot be changed after the field is created. Naming rules: – Max. 63 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and an underscore (_). Do not end with an underscore (_).	productName

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Field Type	Click ***. On the page that is displayed, select the type of the new field based on the parameter description.	Text

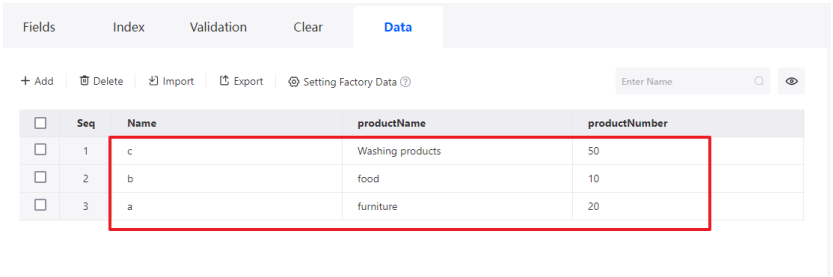
5. On the **Fields** tab, click **Add** again to add the **productNumber** field.

Figure 7-34 Adding the productNumber field



6. On the **Data** tab, click **Add** to add data to the object.

Figure 7-35 Adding data to an object

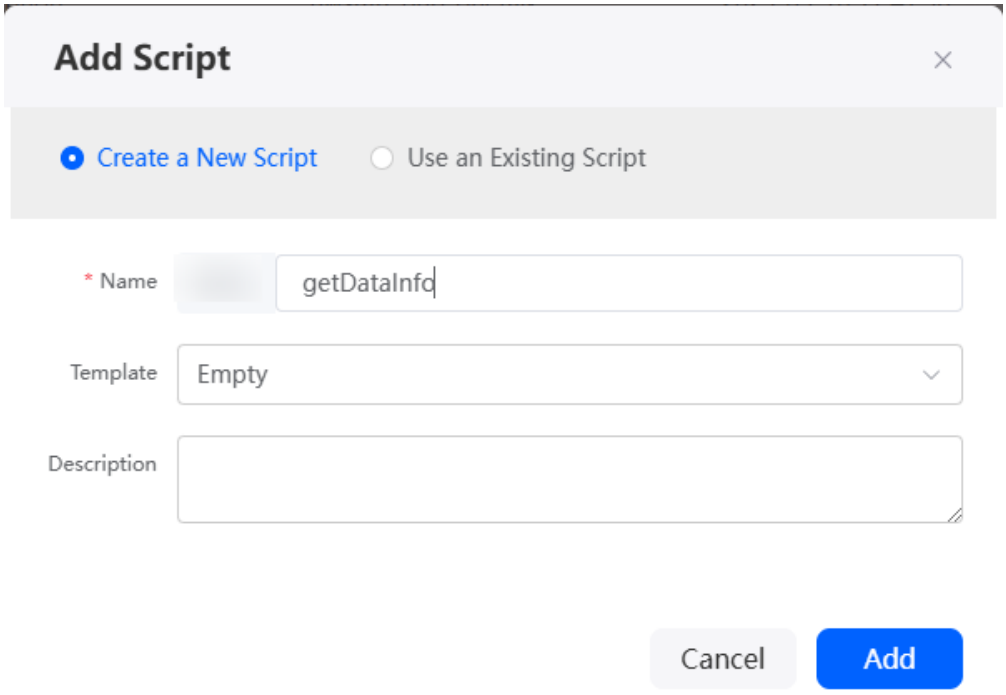


Step 6 Create a script for reading object data.

1. In the navigation pane, choose **Logic** and click **+** next to **Script**.
2. Create a blank script named **getDataInfo**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-36 Creating a script named getDataInfo



3. In the script editor, enter the following code:

```
// Here's your code.
import * as db from 'db';
@useObject (['Namespace __productList__CST'])
@action.object({ type: 'method' })
export class SearchScript {
    @action.method({ input: 'ParamsInput', output: 'ParamsOutput' })
    public run(): Object[] {
        let queryData = this.doSearchScript();
        let result: Array<Object> = [];
        result.push({
            dataValue: [{
                name: "Orders",
                value: queryData
            }]
        });
        console.log("result", result)
        return result;
    }
    private doSearchScript(): Object[] {
        let sql = "select Namespace__productName__CST as name, Namespace__productNumber__CST
as value"
            + " from Namespace__productList__CST"
        let query = db.sql().exec(sql)
        return query;
    }
}
```

4. Click  to save the script. After the script is saved, click  to activate the script.

Step 7 Create an open API.

1. In the navigation pane, select **Integrations**.
2. Click **+** next to the **Open API** and set open API parameters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-37 Setting open API parameters

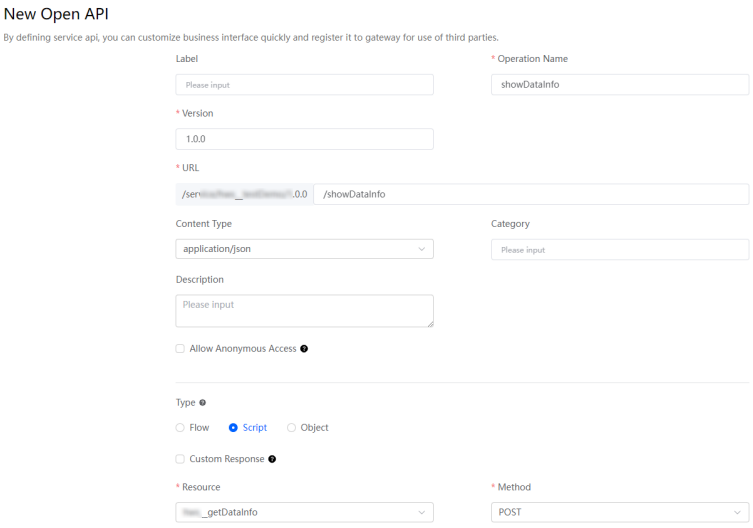


Table 7-8 Parameters for creating an API

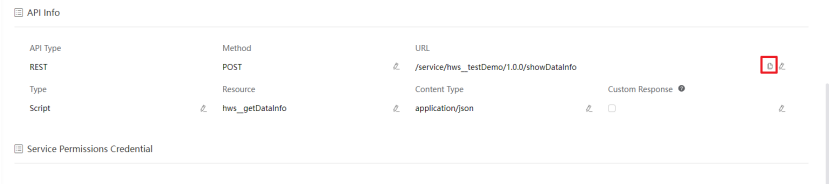
Parameter	Description	Example
Operation Name	Name of the operation for creating an API. Naming rules: – Value: 1–40 characters. – Start with a letter and use only letters, digits, and an underscore (_). Do not end with an underscore (_).	showDataInfo
Version	Version number of the new URL.	1.0.0
URL	New URL. In the URL, **/service** is a fixed value, and is followed by ***/App name/Version***. The rest can be customized.	/showDataInfo
Type	Type of the new API.	Script
Resource	Select the script called by the API.	Select the script created in **Step 6**.
Method	Name of the method called after mapping.	POST

3. After the settings are complete, click **Save**. The API details page is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

4. In the **API Information** area, click next to URL to copy the URL.

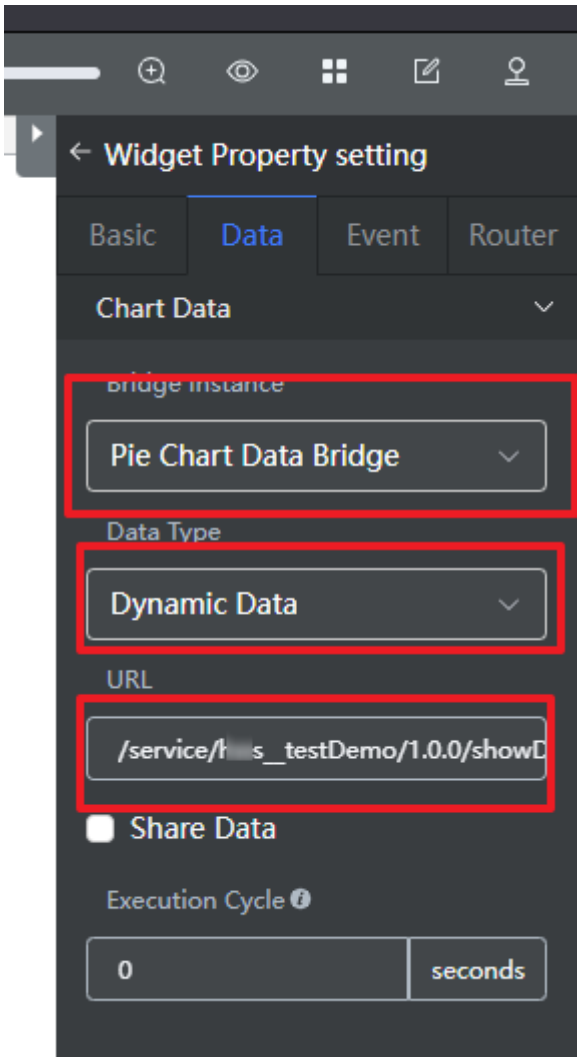
Figure 7-38 Copying a URL



Step 8 Return to the advanced page, select **Rose Pie Chart**, and set the data bridge under the **Data** tab.

Select **Pie Chart Data Bridge** for **Bridge Instance**, select **Dynamic Data** for **Data Type**, and set **URL** to the URL obtained in **Step 7.4**.

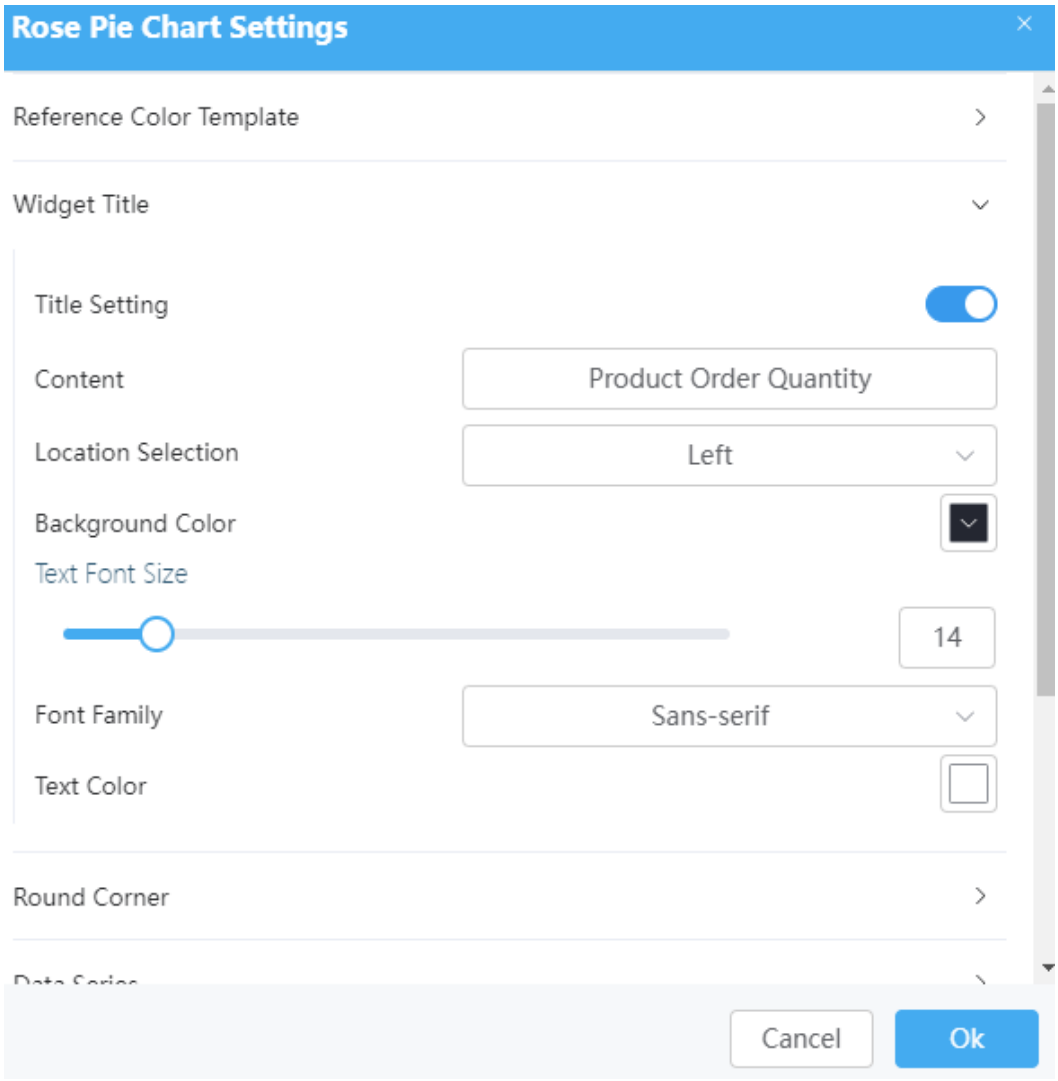
Figure 7-39 Configuring widget data



Step 9 Right-click the rose pie chart widget and choose **Setting** from the shortcut menu to set the widget title and font size.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-40 Setting the widget title



Step 10 Click to save the advanced page settings. After the page settings are saved, click to release the page.

Step 11 After the release is successful, click to preview.

----End

7.6 Implementing Image Display and URL Redirection with the Banner Widget on Advanced Pages

Expected Results

The banner widget on an advanced page is used to automatically switch between multiple images. You can also add a hyperlink to an image, so you can click the image to go to the specified website.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Implementation Methods

Step 1 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

Figure 7-41 Creating a blank application

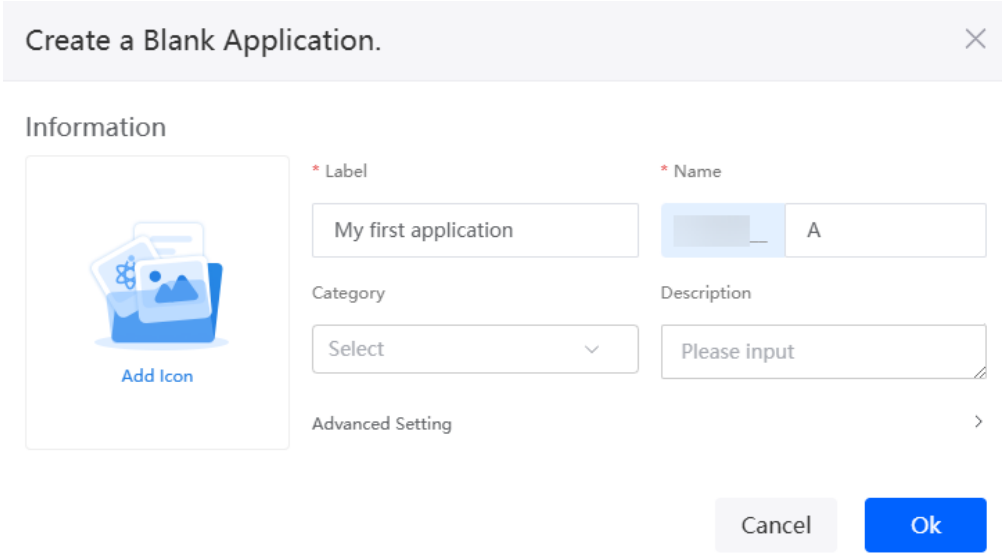


Table 7-9 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application

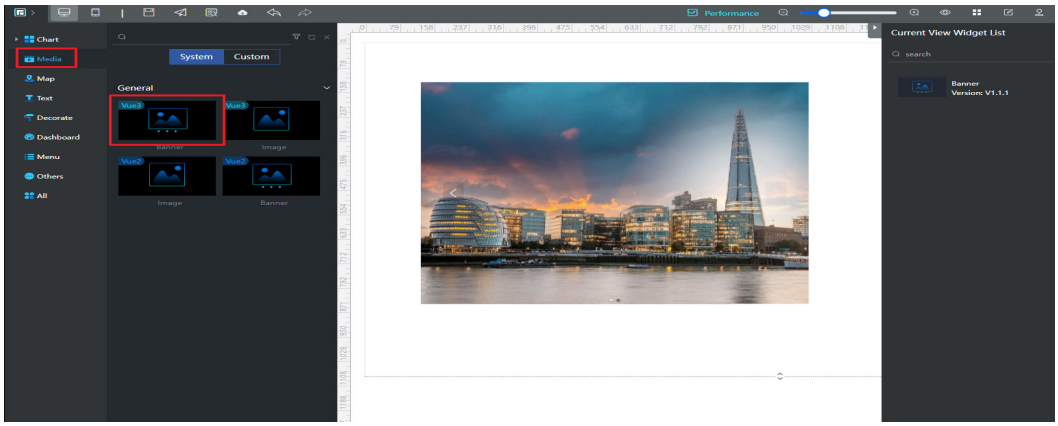
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 2 In the navigation pane, choose **Page**, and click **+** next to **Advanced Page**.

Step 3 Click and drag the banner widget to the canvas on the right.

Figure 7-42 Dragging the banner widget

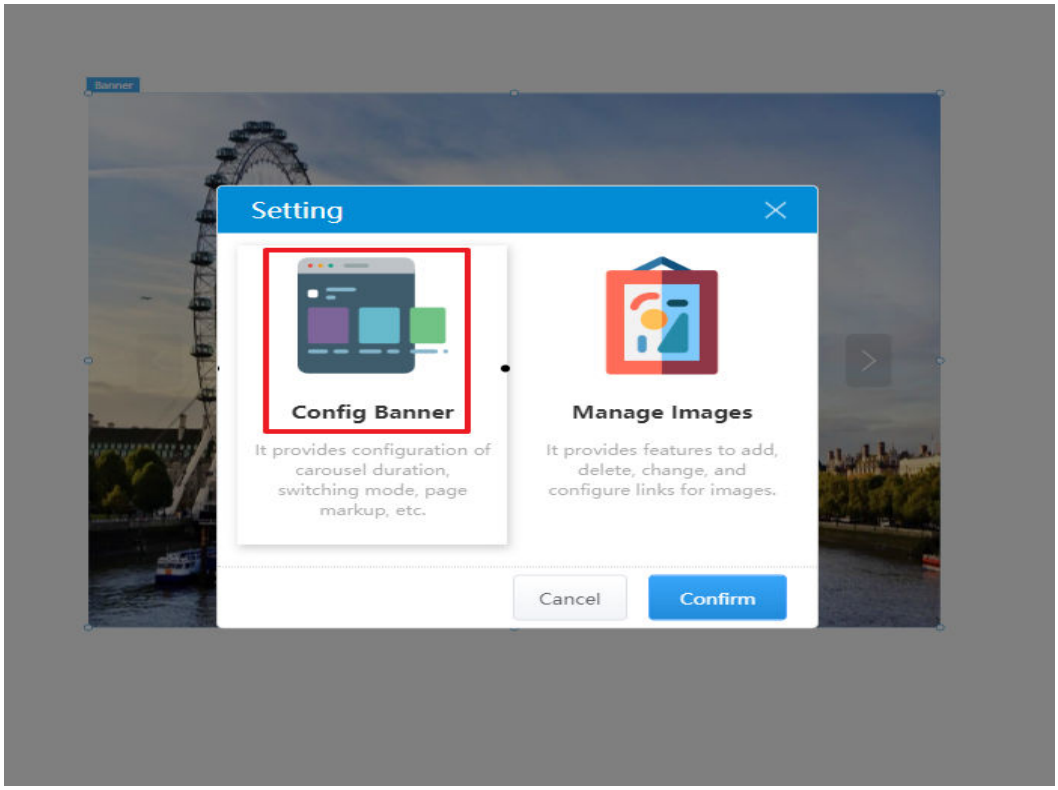


Step 4 Right-click the banner widget and choose **Setting** from the shortcut menu.

Step 5 Click **Config Banner** and set the direction and image playback interval as required.

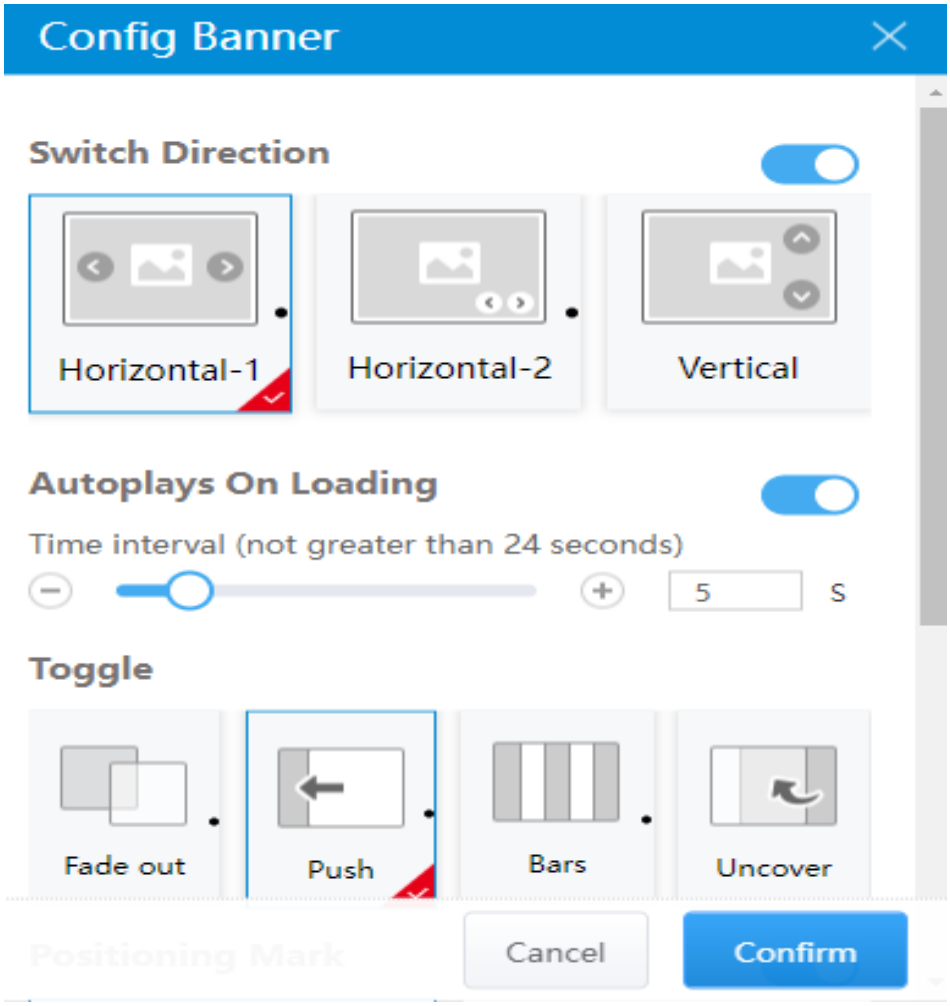
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-43 Setting the banner widget



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-44 Config Banner

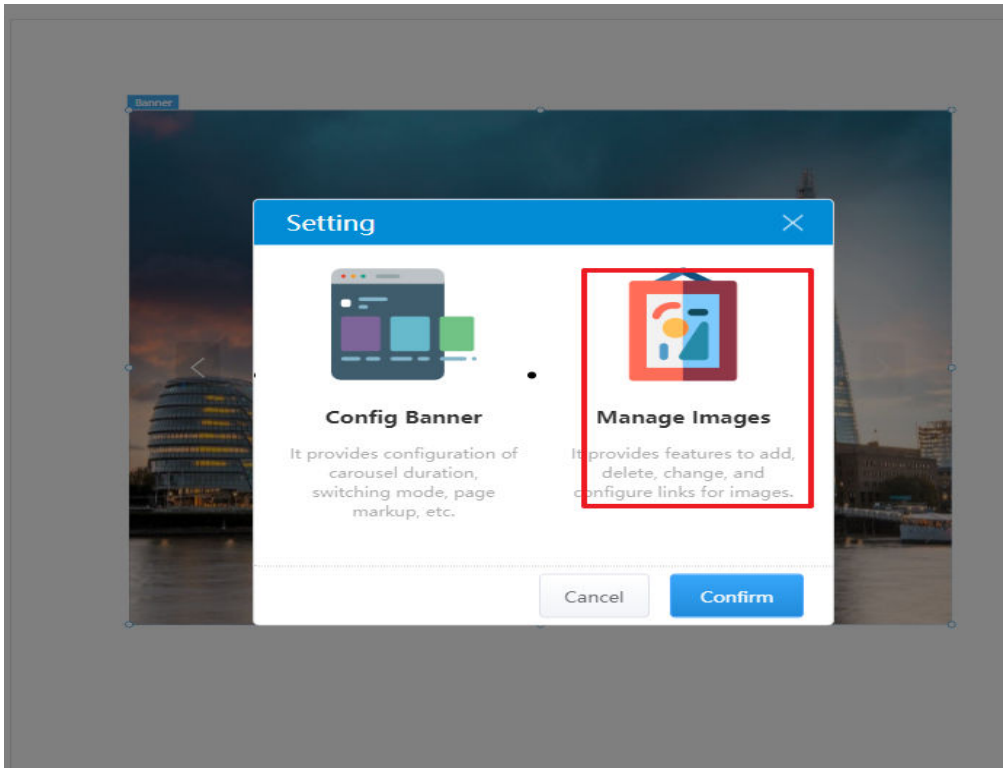


Step 6 Add an image and set the URL redirection.

1. Right-click the banner widget again, choose **Setting** from the shortcut menu, and select **Manage Images**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

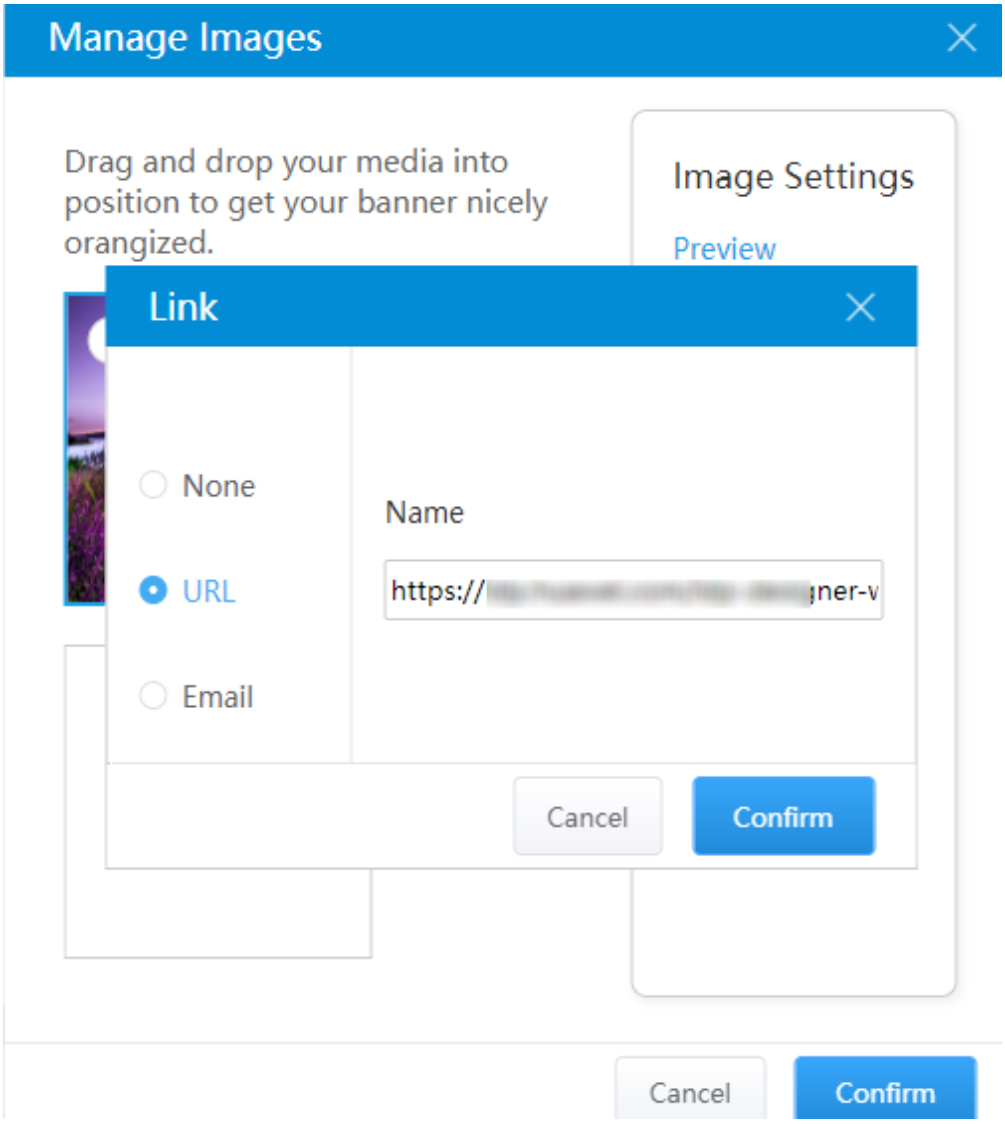
Figure 7-45 Selecting Manage Images



2. Click **Add**, select the desired image, and click **Save**.
3. Select the added image and click next to the link in **Image Settings** on the right.
4. Set the redirection URL and click **Confirm**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 7-46 Adding a link



Step 7 Click **Confirm** to return to the advanced page.

Step 8 Click  to save the advanced page settings. After the page settings are saved, click  to release the page.

Step 9 After the release is successful, click  to preview.

----End

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8 Connectors

8.1 Uploading and Recognizing ID Card Images with a Connector

Expected Results

Huawei Cloud Astro Zero encapsulates various connectors to integrate with external services, enabling their use within applications. For example, using a connector to link with OCR enables the recognition of text in ID card images uploaded by users to the Object Storage Service (OBS).

Figure 8-1 Submitting the ID card information

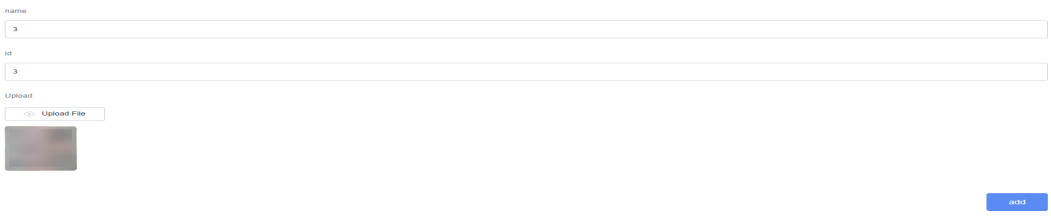


Figure 8-2 Image recognition result

Procedure

Step 1 Make preparations.

- To access OBS, you need a HUAWEI ID or an IAM user account. This involves registering with Huawei Cloud, completing real-name authentication, creating an IAM user account, funding your account, and purchasing the necessary resource packages. For details, see **Using OBS Console**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Obtain the access key ID (AK) and secret access key (SK). For details, see **Obtaining an AK/SK** .
- Create a bucket (for example, **appcubecn4**) in OBS. For details, see **Creating a Bucket**. Record the region selected during bucket creation.

Step 2 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

Figure 8-3 Creating a blank application

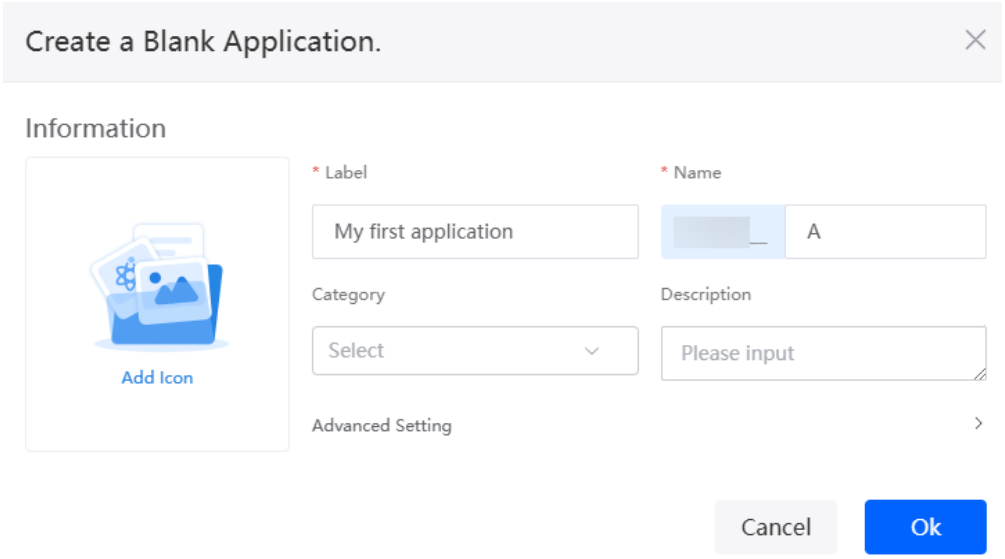


Table 8-1 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application

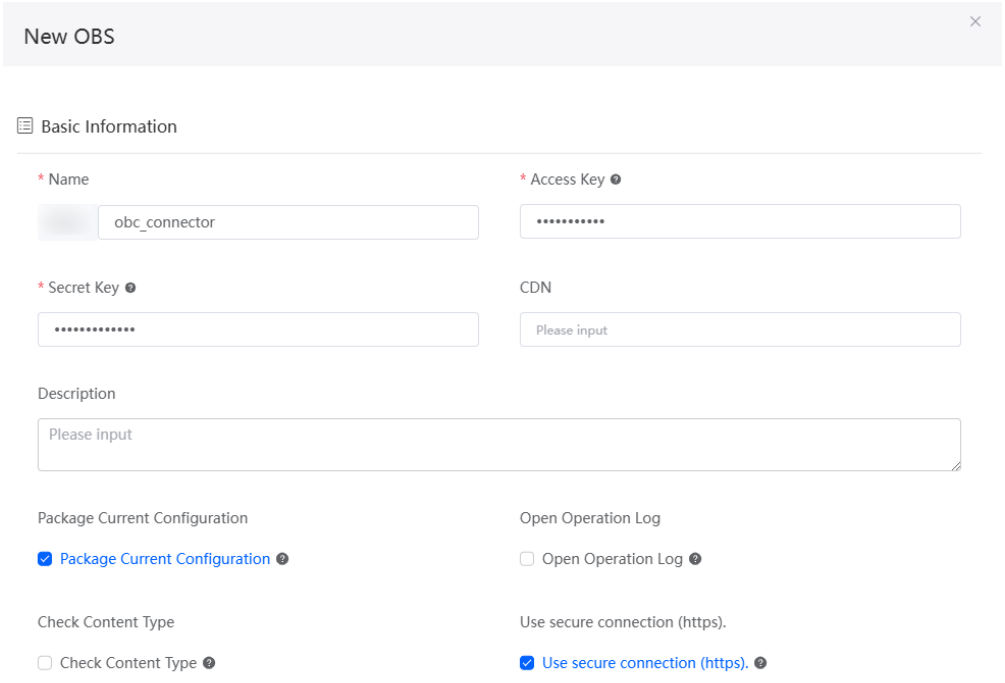
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 3 Create an OBS connector.

1. In the application designer, choose **Integrations** > **Connector** > **Connector Instance**.
2. Choose **OBS** under **Storage** and click **+** to create an OBS connector.
3. Configure basic information, add the bucket, and click **Save**.

Figure 8-4 Setting basic information



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 8-5 Adding a bucket

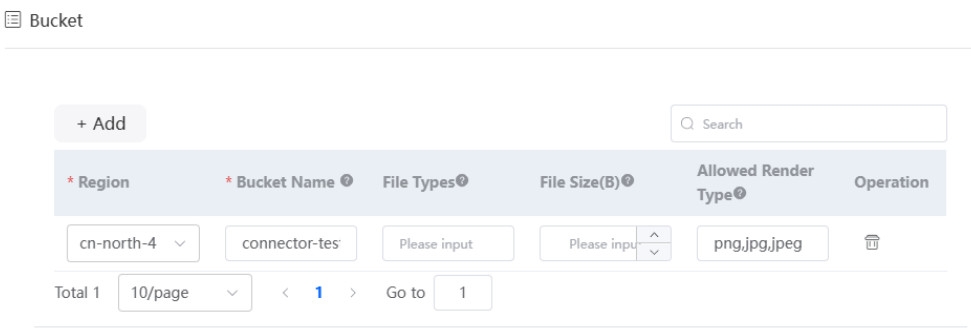


Table 8-2 Parameters for creating an OBS connector

Parameter	Description	Example
Name	Name of the OBS connector to be created. Naming rules: – Max. 64 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and can contain only letters, digits, and an underscore (_). Do not end with an underscore (_).	uploadMod

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Auth Mode	Select an authentication mode. – **AK/SK**: You need to obtain the access key and manually enter the AK/SK. – **Agency**: Select the agency authorization mode. You do not need to enter the access key.	AK/SK
Access Key	Configure the access key (AK) of the user.	Access key ID value obtained in **Step 1**
Secret Key	Configure the obtained SK.	Secret access key value obtained in **Step 1**
Bucket	Set the region where the bucket resides and the bucket name.	**Region**: **cn north-4**; **Bucket Name**: **appcubecn4** (the bucket created in **Step 1**)

Step 4 Create the **orcTry** object to store information such as the image URL.

1. In the navigation pane on the left, choose **Data**, and click **+** next to **Object**.
2. Set **Object Name** and **Unique ID** of the object to **orcTry** and click **Confirm**.

Figure 8-6 Creating the object orcTry

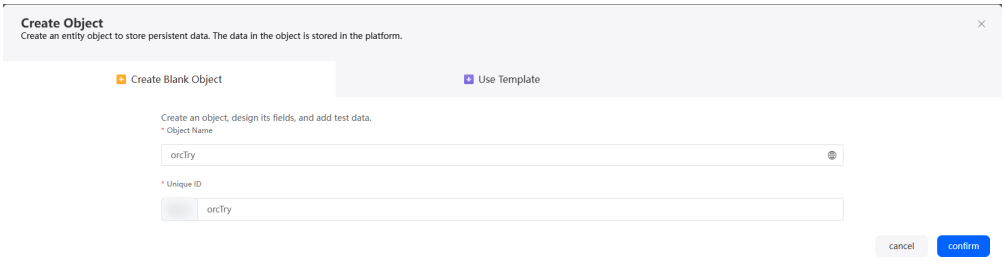


Table 8-3 Parameters for creating orcTry

Parameter	Description	Example
Object Name	Name of the new object, which can be changed after the object is created. Value: 1–80 characters.	orcTry

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Unique ID	ID of a new object in the system, which cannot be modified after being created. Naming rules: • Max. 63 characters, including the prefix namespace. • Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	orcTry

3. Click  to go to the object details page.
4. On the **Fields** tab page, click **Add** and add the **pictureName** field to the object.

Figure 8-7 Adding the pictureName field

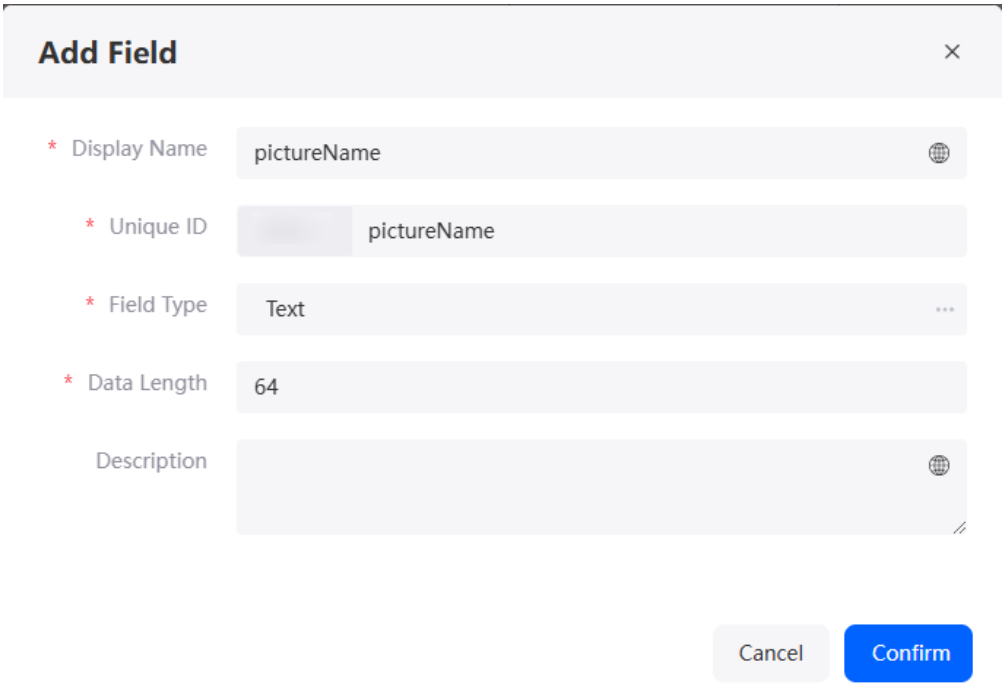


Table 8-4 Parameters for adding a field

Parameter	Description	Example
Display Name	Name of the new field, which can be changed after the field is created. Value: 1–63 characters.	pictureName

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Unique ID	ID of a new field in the system. The value cannot be changed after the field is created. Naming rules: – Max. 63 characters, including the prefix namespace. – Start with a letter and can contain only letters, digits, and an underscore (_). Do not end with an underscore (_).	pictureName
Field Type	Click ***. On the page that is displayed, select the type of the new field based on the parameter description.	Text

5. Repeat the preceding operations to add the fields in **Table 8-5** to the object.

Table 8-5 Adding other fields

Display Name	Unique ID	Field Type
pictureId	pictureId	Text
picUrl	picUrl	Text Area
result	result	Text Area

Step 5 Create an OCR connector.

1. In the application designer, choose **Integrations** > **Connector** > **Connector Instance**.
2. Choose **OCR** under **AI** and click **+** to create an OCR connector.
3. Configure basic information, and click **save**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 8-8 Setting basic information

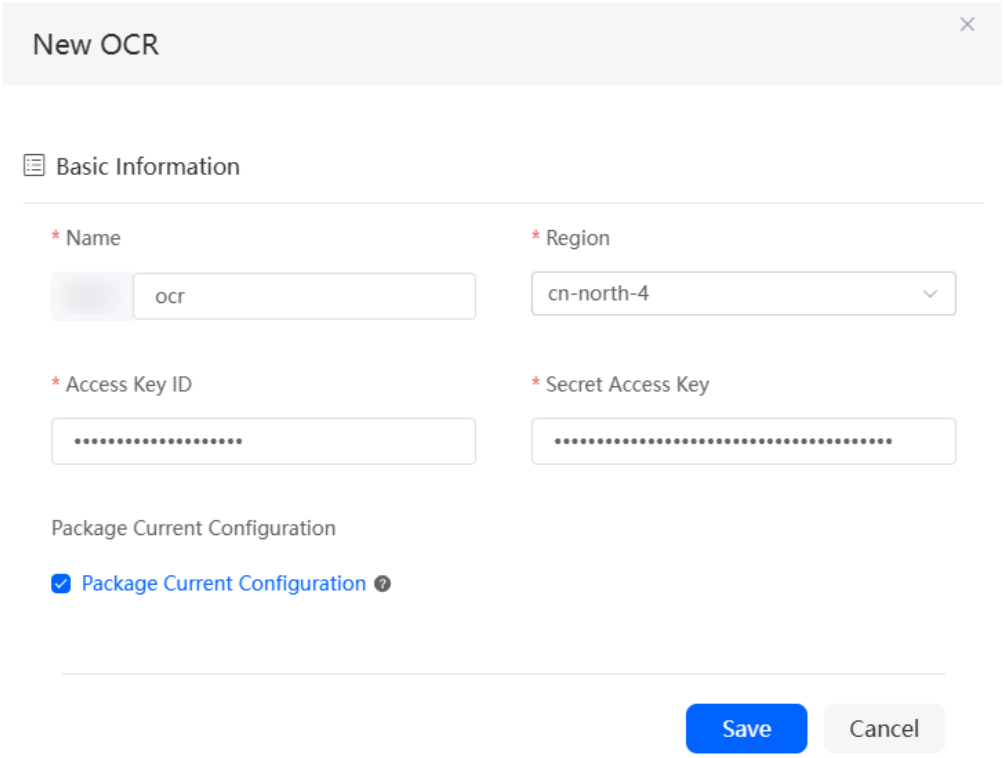


Table 8-6 Parameters for creating an OCR connector

Parameter	Description	Example
Name	Name of the OCR connector to be created. Naming rules: – Max. 63 characters, including the prefix namespace. – Start with a letter and can contain only letters, digits, and an underscore (_). Do not end with an underscore (_).	newOcr
Region	Region where OCR is deployed.	cn-north-4
Access Key ID	It is used together with the SK to sign requests.	AK obtained by referring to **Step 1**

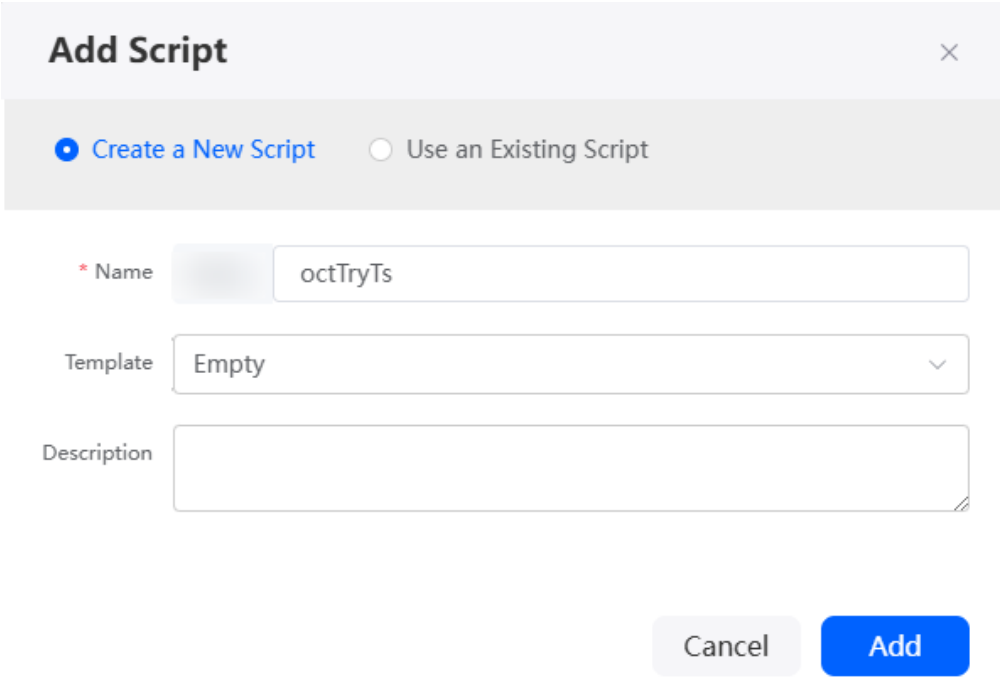
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Secret Access Key	The SK and AK are used together to encrypt and sign a request to identify the sender and prevent the request from being modified.	SK obtained by referring to **Step 1**

Step 6 Create a script for submitting image information and using the OCR API to recognize images.

1. In the navigation pane, choose **Logic** and click **+** next to **Script**.
2. Create a blank script named **octTryTs**.

Figure 8-9 Creating an octTryTs script



3. In the script editor, enter the following code:

```
//This script is used to submit image information and call the OCR API to recognize images.
import * as db from 'db';//Import the standard library related to the object.
import * as context from 'context';//Import the standard library related to the context.
import * as ocr from 'ocr';

//Define the input parameter structure.
@action.object({ type: "param" })
export class ActionInput {
    @action.param({ type: 'String', required: true, label: 'String' })
    proId: string;
    @action.param({ type: 'String', required: true, label: 'String' })
    proName: string;
    @action.param({ type: 'Any', required: true, label: 'Any' })
    url: any;

}
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
//Define the output parameter structure. The output parameter contains one parameter, that is, the record ID of workOrder.
@action.object({ type: "param" })
export class ActionOutput {
    @action.param({ type: 'String' })
    id: string;
    @action.param({ type: 'String', required: true, label: 'String' })
    url: string;
}
//Use the data object namespace__orcTry__CST.
@useObject (['Namespace__orcTry__CST'])
@action.object({ type: "method" })
export class CreateWorkOrder {    //Define the API class. The input parameter of the API is ActionInput, and the output parameter is ActionOutput.
    @action.method({ input: 'ActionInput', output: 'ActionOutput' })
    public createWorkOrder(input: ActionInput): ActionOutput {
        let cli = ocr.newClient(" Namespace__newOcr"; //Create a connector instance.
        let out = new ActionOutput();    //Create an instance of the ActionOutput type as the return value.
        let error = new Error();    //Create an instance of the error type to save the error information when an error occurs.
        try {
            let url = "https://appcubecn4.obs.cn-north-4.myhuaweicloud.com/" + input.url[0]['originalUrl']; // Combine a complete image URL.
            let resp = cli.idCardWithURL(url, "front"); // Call the OCR API. Front indicates that the front of the ID card will be recognized.
            let productData = new Object();
            productData['Namespace__pictureName__CST'] = input.proName;   //Assign the input parameter to the productData variable for future use.
            productData['Namespace__pictureId__CST'] = input.proId;
            productData['Namespace__picUrl__CST'] = url;
            productData['Namespace__result__CST'] = JSON.stringify(resp['result']);
            let s = db.object('BJ4__orcTry__CST');    //Obtain the operation instance of the Namespace__orcTry__CST object.
            let id = s.insert(productData);
            if (id) {
                out.id = id;
                out.url = url;
            } else {
                error.name = "WOERROR";
                error.message = "Unable to create pic!";
                throw error;
            }
        } catch (error) {
            console.error(error.name, error.message);
            context.setError(error.name, error.message);
        }
        return out;
    }
}
```

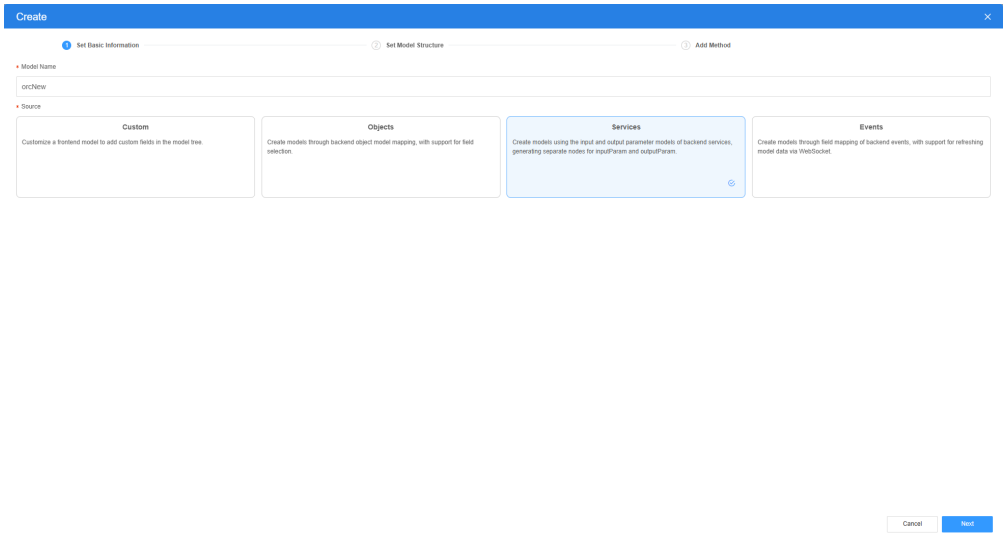
4. Click  to save the script. After the script is saved, click  to activate it.

Step 7 Create a page model.

1. In the navigation pane, choose **Page**, and click **+** next to **Standard Page**.
2. At the bottom of the standard page, click **Model View**.
3. Click **New**, specify **Model Name** (for example, **orcNew**), select **Services** for **Source**, and click **Next**.

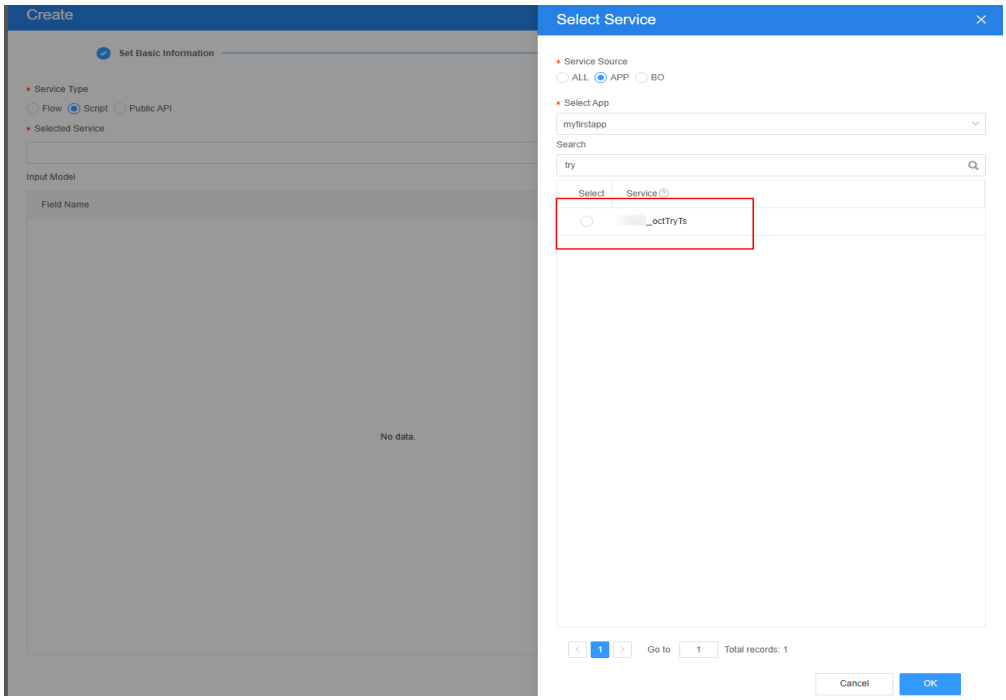
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 8-10 Creating a model



4. Select the script created in **Step 6** and click **OK**. Then click **Next**.

Figure 8-11 Selecting the script



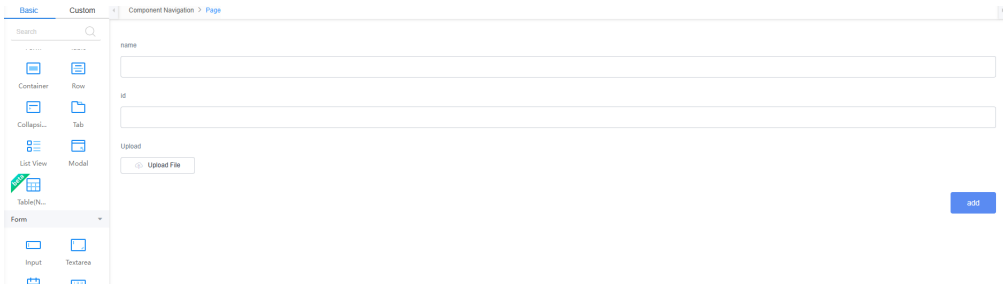
5. Click **OK**.

Step 8 Create an image upload form page.

1. On the standard page, drag two **Input** widgets, one **Button** widget, and one **Upload** widget to the canvas.
2. Change the **Label** of two **Input** widgets to **name** and **id**, and set the button's display name to **Add**.

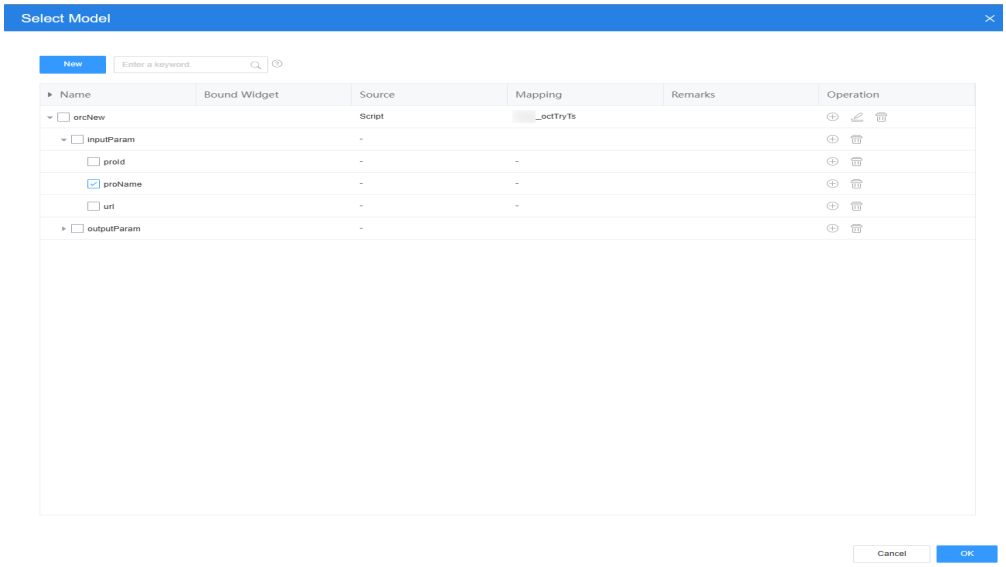
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 8-12 Final form setting effect



3. Select the **name** input widget, choose **Properties** > **Data Binding** and click next to **Value Binding**.
4. Select the **proName** field in the model created in **Step 7** and click **OK**.

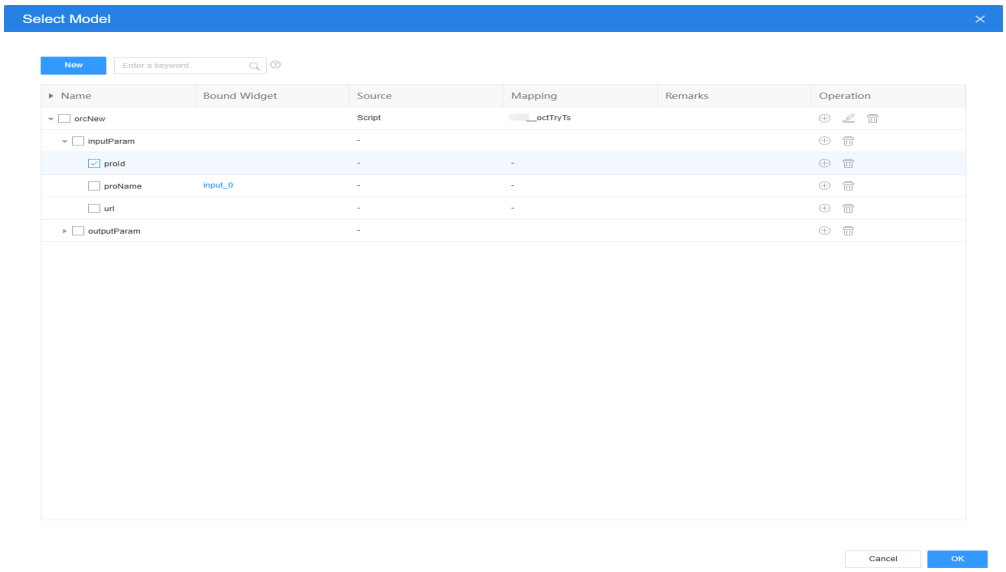
Figure 8-13 Selecting the proName field



5. Select the **id** input widget, choose **Properties** > **Data Binding** and click next to **Value Binding**.
6. Select the **proId** field in the model created in **Step 7** and click **OK**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 8-14 Selecting proId



7. Repeat the preceding operations to bind a model to the **Upload** widget and set storage information.

Figure 8-15 Binding a model to the upload widget

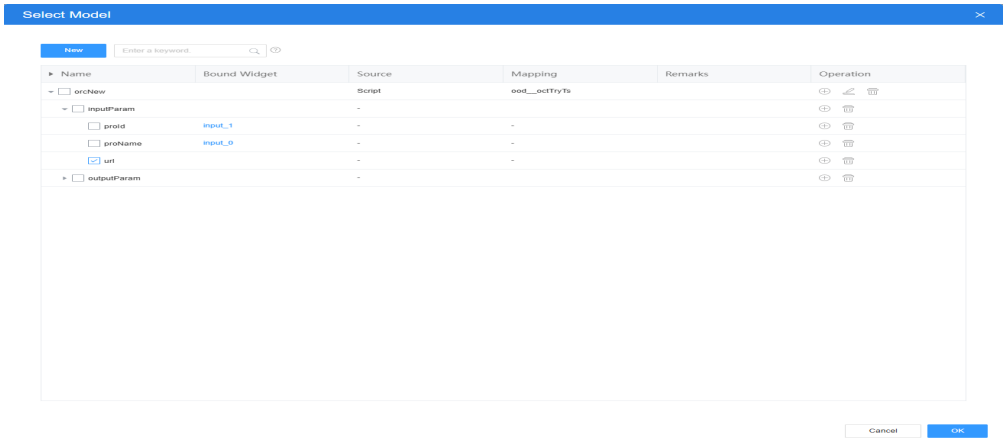
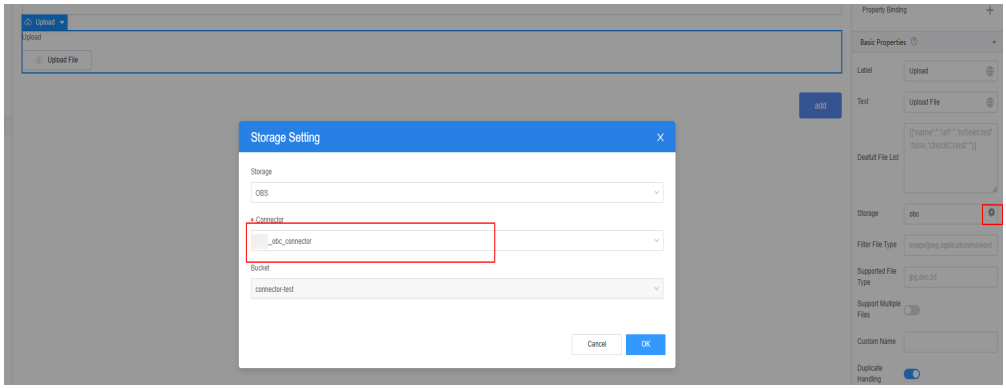


Figure 8-16 Setting storage information

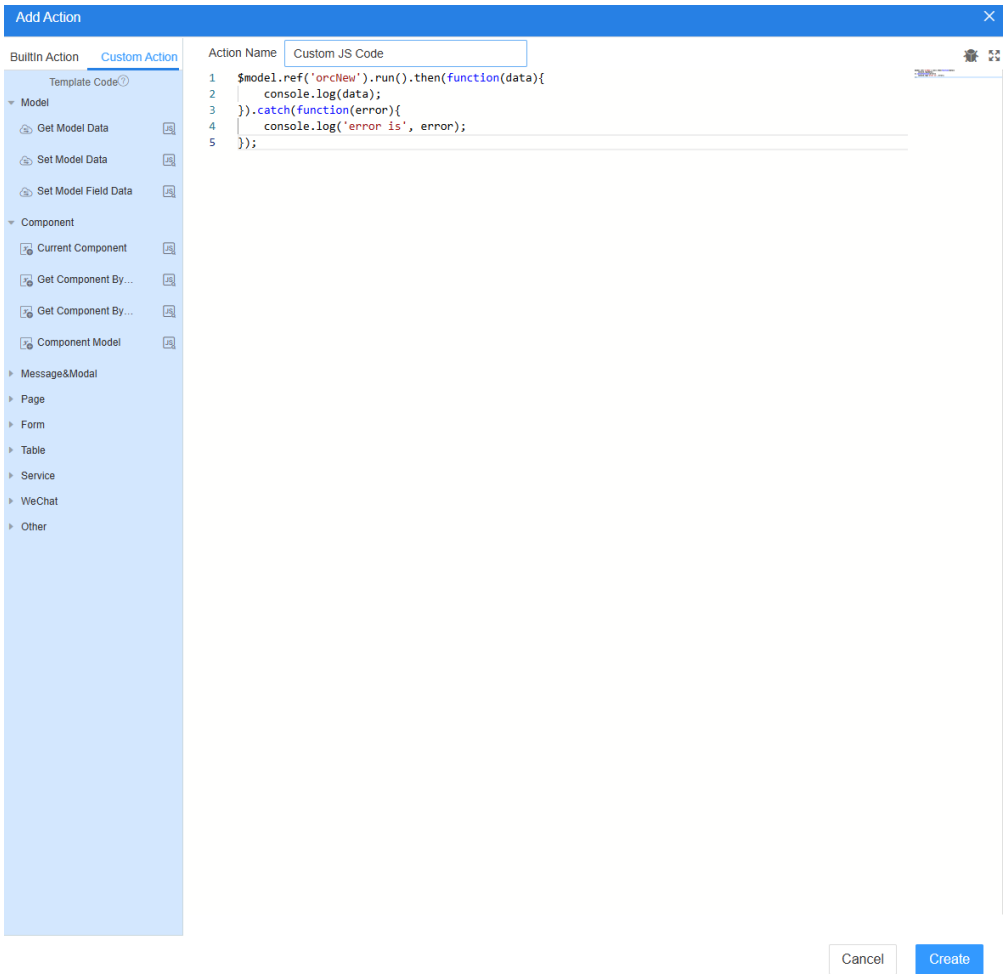


Step 9 Add an event for the **Add** button.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1. Select the **Add** button widget and click the **Events** tab.
2. Click **+** next to **on-click**. The page for adding an action is displayed.
3. Under **Custom Action**, enter the custom code and click **Create**.

Figure 8-17 Customizing an action



orcNew in the command is the name of the model created in **Step 7**.

```
$model.ref('orcNew').run().then(function(data){
    console.log(data);
}).catch(function(error){
    console.log('error is', error);
});
```

Step 10 Return to the standard page and click  to save the page settings. After the page is saved, click  to preview the effect.

----End

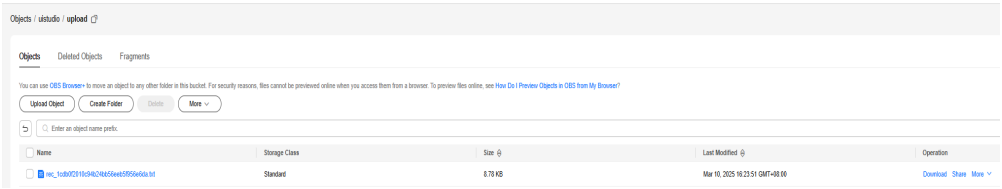
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2 Uploading Files With Connectors

Expected Results

Huawei Cloud Astro Zero encapsulates various connectors to integrate with external services, enabling their use within applications. For example, using a connector to link with OBS enables files to be saved from a frontend page to an OBS bucket.

Figure 8-18 Checking the file in a specified path of the OBS bucket



Implementation Methods

Step 1 Make preparations.

- To access OBS, you need a HUAWEI ID or an IAM user account. This involves registering with Huawei Cloud, completing real-name authentication, creating an IAM user account, funding your account, and purchasing the necessary resource packages. For details, see **Using OBS Console**.
- Obtain the access key ID (AK) and secret access key (SK). For details, see **Obtaining an AK/SK** .
- Create a bucket (for example, **bing.testonly.1**) in OBS. For details, see **Creating a Bucket**. Record the region selected during bucket creation.

Step 2 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 8-19 Creating a blank application

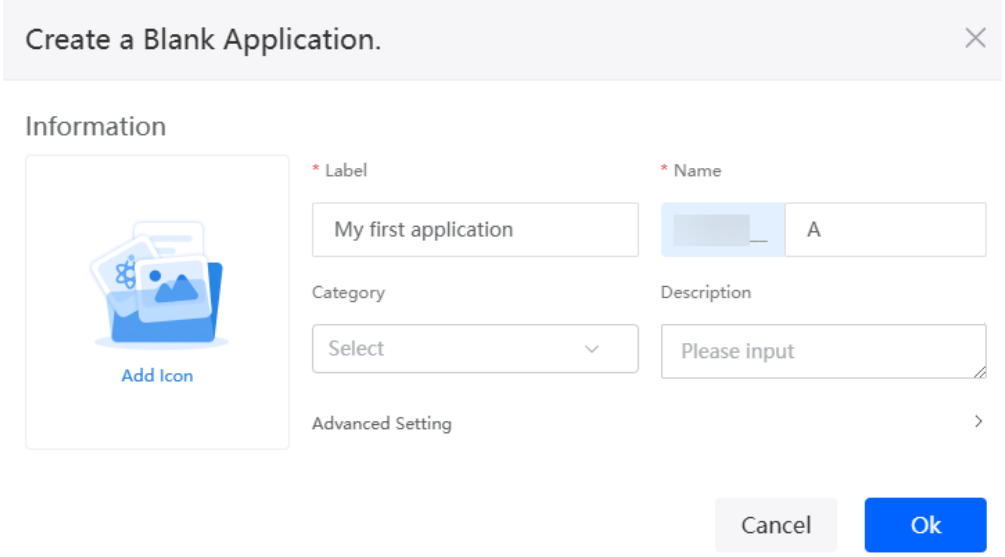


Table 8-7 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 3 Create an OBS connector.

1. In the application designer, choose **Integrations** > **Connector** > **Connector Instance**.
2. Choose **OBS** under **Storage** and click **+** to create an OBS connector.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3. Configure basic information, add the bucket, and click **Save**.

Figure 8-20 Setting basic information

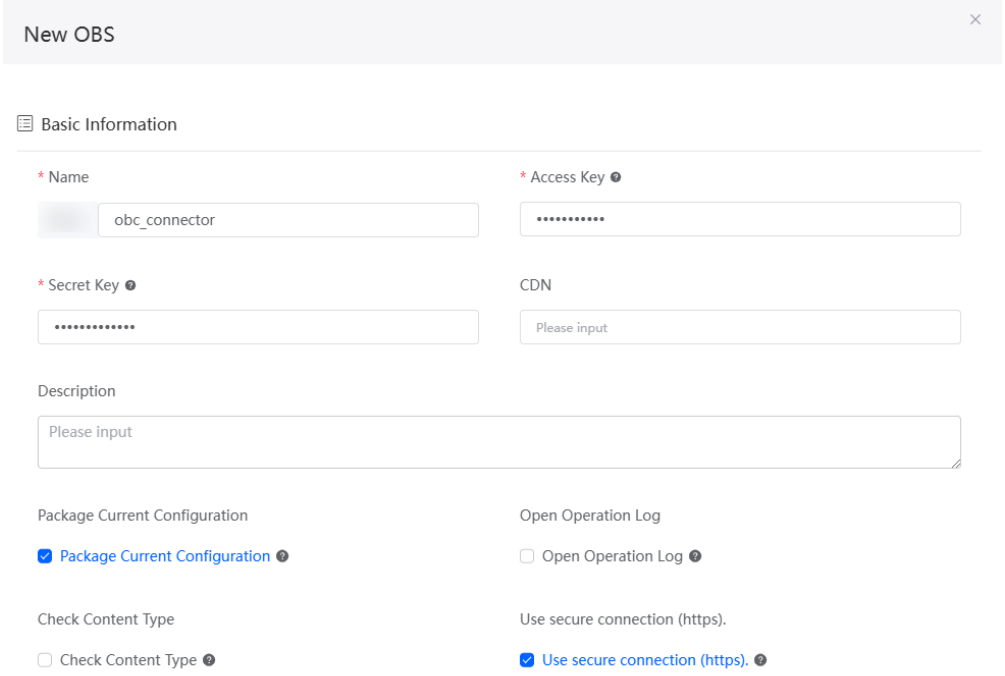
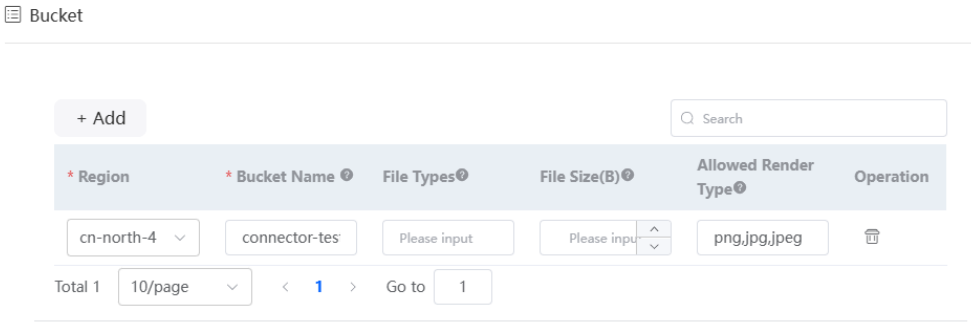


Figure 8-21 Adding a bucket



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-8 Parameters for creating an OBS connector

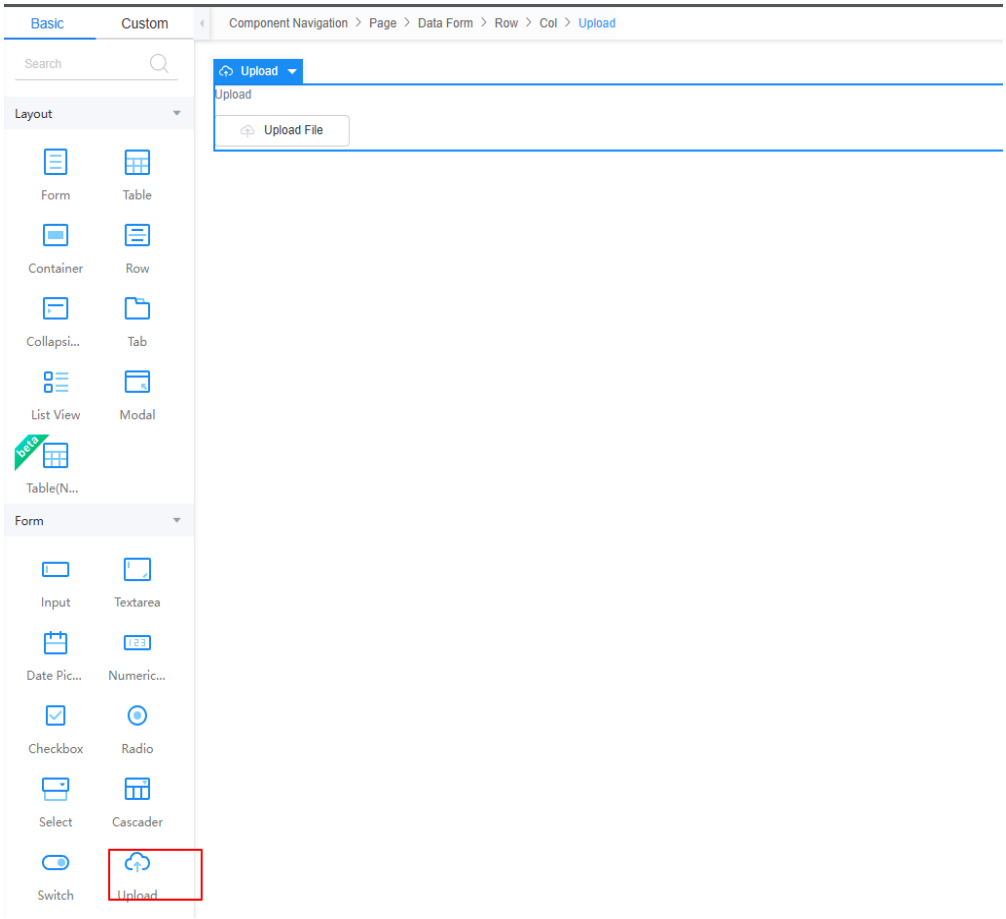
Parameter	Description	Example
Name	Name of the OBS connector to be created. Naming rules: – Max. 64 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and can contain only letters, digits, and an underscore (_). Do not end with an underscore (_).	upload
Access Key	Configure the access key (AK) of the user.	Access key ID value obtained in **Step 1**
Secret Key	Configure the obtained SK.	Secret access key value obtained in **Step 1**
Bucket	Set the region where the bucket resides and the bucket name.	**Region**: **cn north-4**; **Bucket Name**: **bing.testonly.1** (the bucket created in **Step 1**)

Step 4 Create a standard page for uploading files.

1. In the navigation pane, choose **Page**, and click **+** next to **Standard Page**.
2. Drag an **Upload** widget to the standard page.

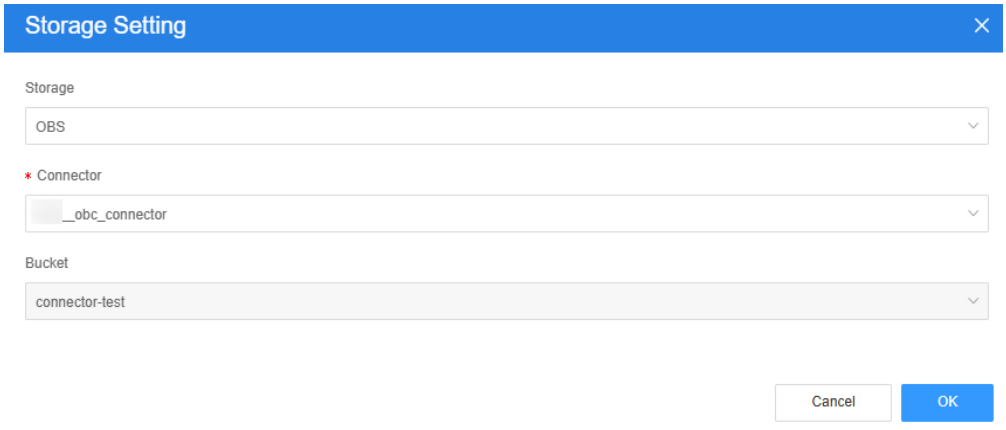
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 8-22 Dragging the upload widget



3. Select the **Upload** widget, set **Storage** to **OBS**, and select the connector created in **Step 3**.

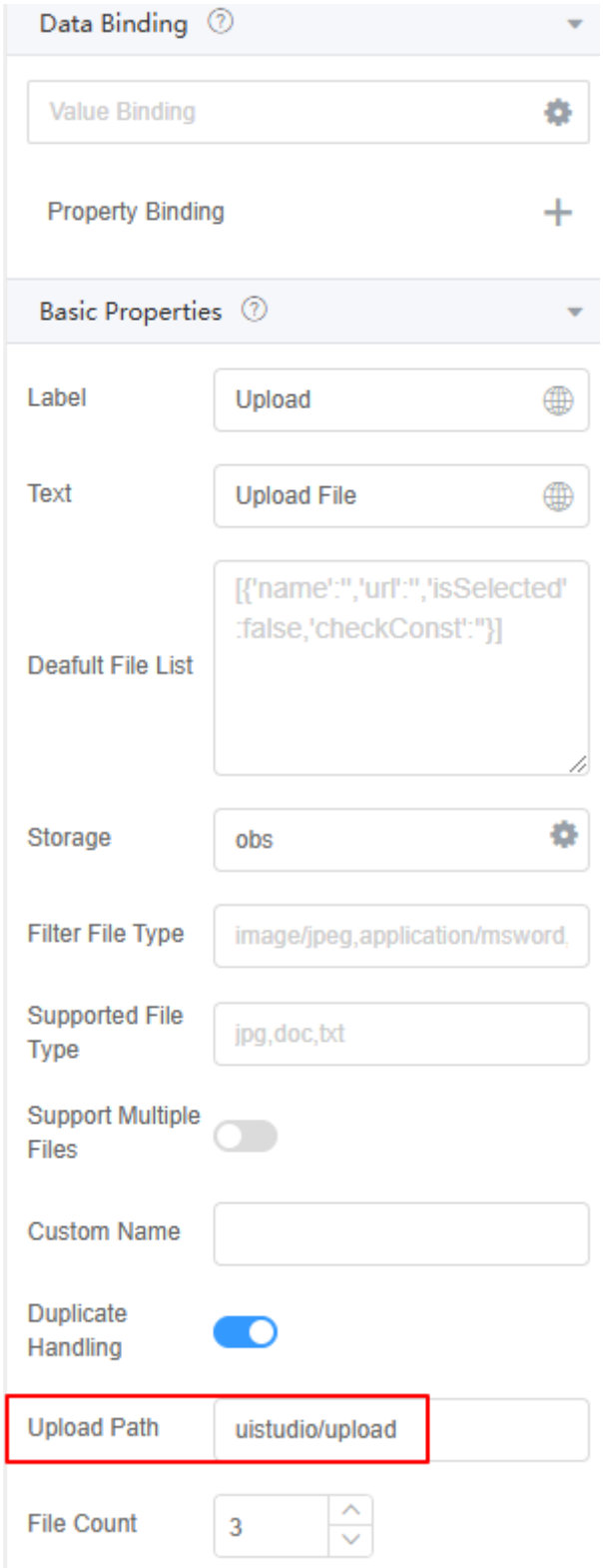
Figure 8-23 Setting storage information



4. Define the file upload path.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 8-24 Setting the upload path



Step 5 Return to the standard page and click to save the page settings. After the page is saved, click to preview the effect.

----End

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.3 Connecting to Third-Party Databases with a Connector

Application Scenarios

A connector is an integration tool for calling third-party services. With connectors, developers do not need to pay attention to the implementation code. They only need to configure the service address and authentication information to quickly integrate third-party systems in applications, reducing development costs and improving efficiency.

This practice describes how to use a custom connector to integrate third-party databases into Huawei Cloud Astro Zero.

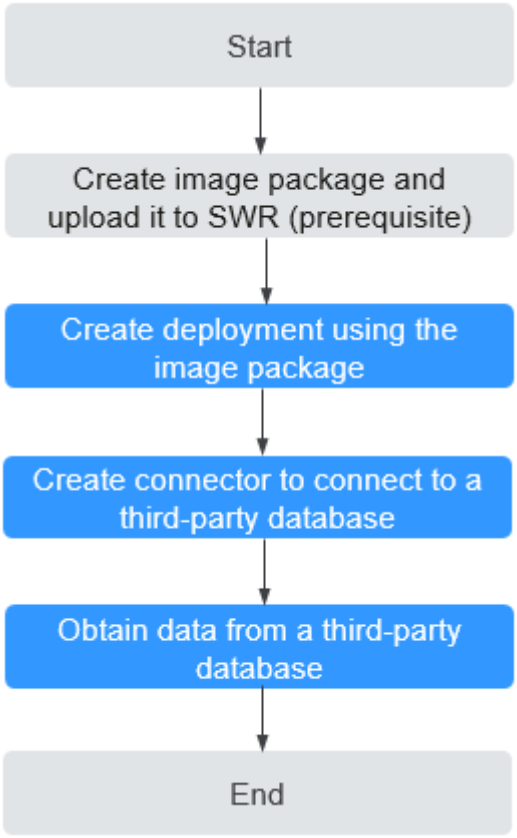
Advantages

Users can define database operations and authentication APIs to independently control the network of third-party databases.

Procedure

Figure 8-25 illustrates the process of connecting to a third-party database through a connector in Huawei Cloud Astro Zero.

Figure 8-25 Connecting to third-party databases with a connector



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Prerequisites

- **db-adapter** is the backend Java demo code used to operate the third-party database. The **datasource.json** file in the **resources** directory is used to configure the URL, username, and password of the database. The **DBExecuteController.java** file is used to connect to the database, process external requests, and return database data.

  Figure 8-26 datasource.json

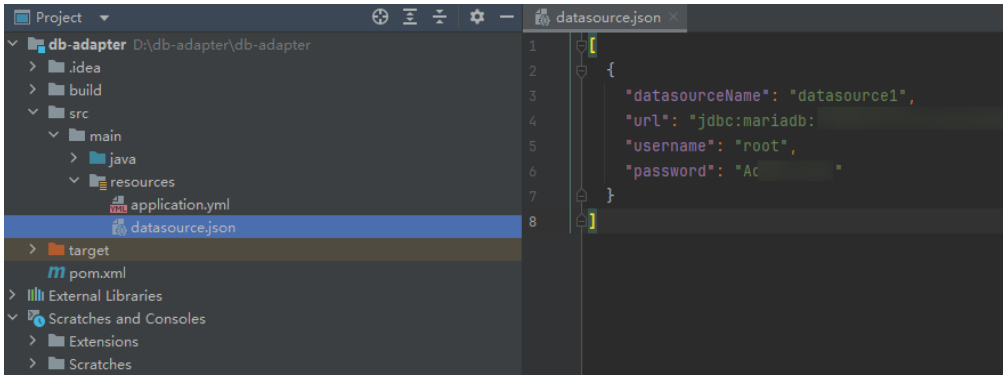
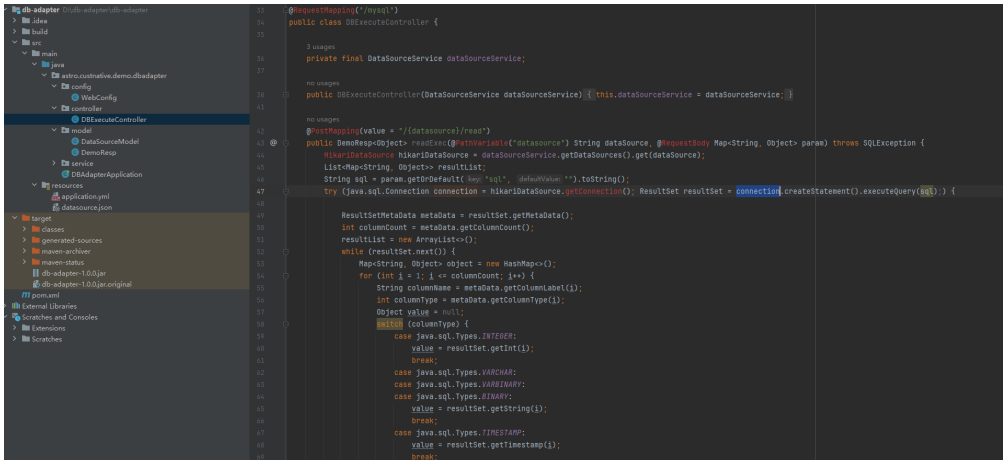
  

  Figure 8-27 DBExecuteController.java

- Generate a JAR package based on **db-adapter**, place the compiled JAR package in the **build** directory, and build a Docker image. Upload the created image package to SWR. For details about how to upload image packages, see **Uploading an Image Through SWR Console**.

  When using **docker save** to create an image package, use the **{image}:{tag}** command instead of **image id**. Otherwise, the package cannot be uploaded on the SWR console. If an image fails to be uploaded, see **Why Does an Image Fail to Be Uploaded on the SWR Console**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 8-28 Using the maven install command to pack the db-adapter folder into a JAR package

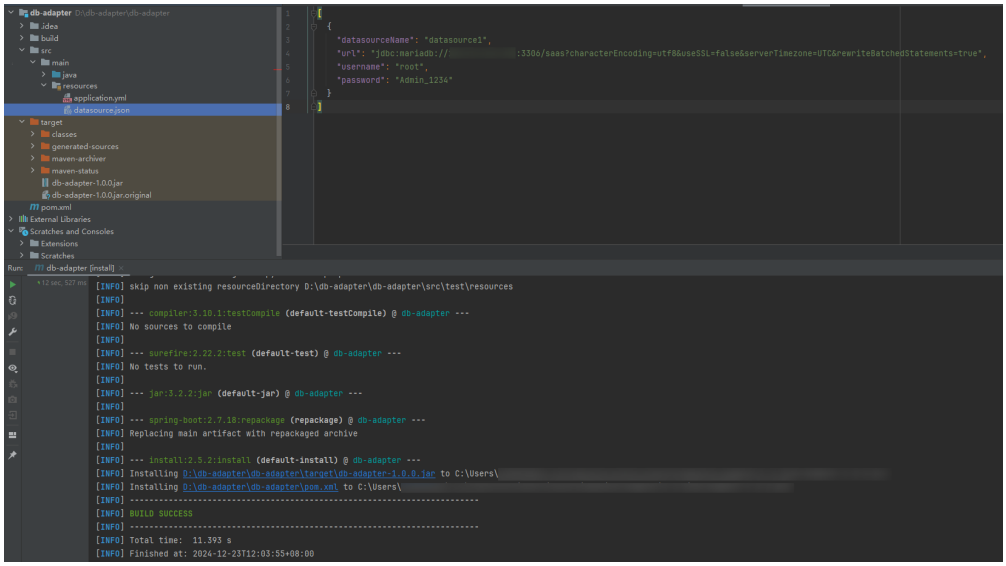


Figure 8-29 Building an image

Creating a Deployment Using an Image Package

Step 1 Log in to the **CCE console** and buy a cluster by referring to **Buying a CCE Standard/Turbo Cluster**.

Step 2 After the cluster is purchased, add a node to the cluster by referring to **Creating a Node**.

Step 3 Create a deployment.

1. After a node is added, choose **Kubernetes Resources > Workloads > Deployments** in the navigation pane on the left.
2. Click **Create Workload**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 8-30 Creating a deployment

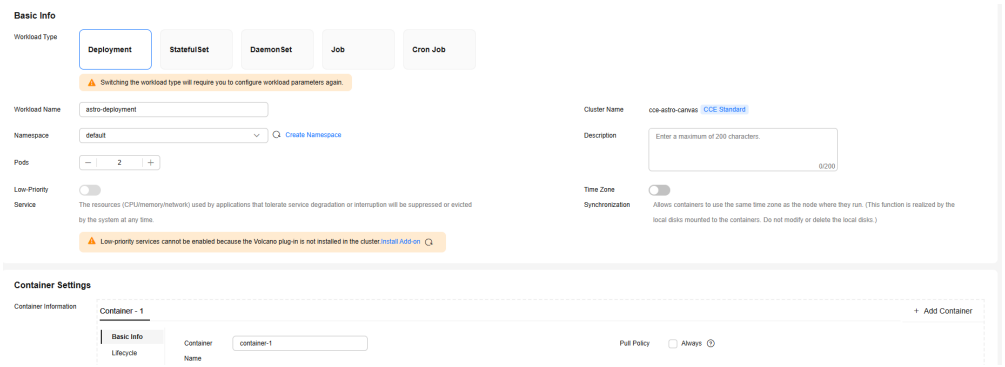


Figure 8-31 Selecting an image

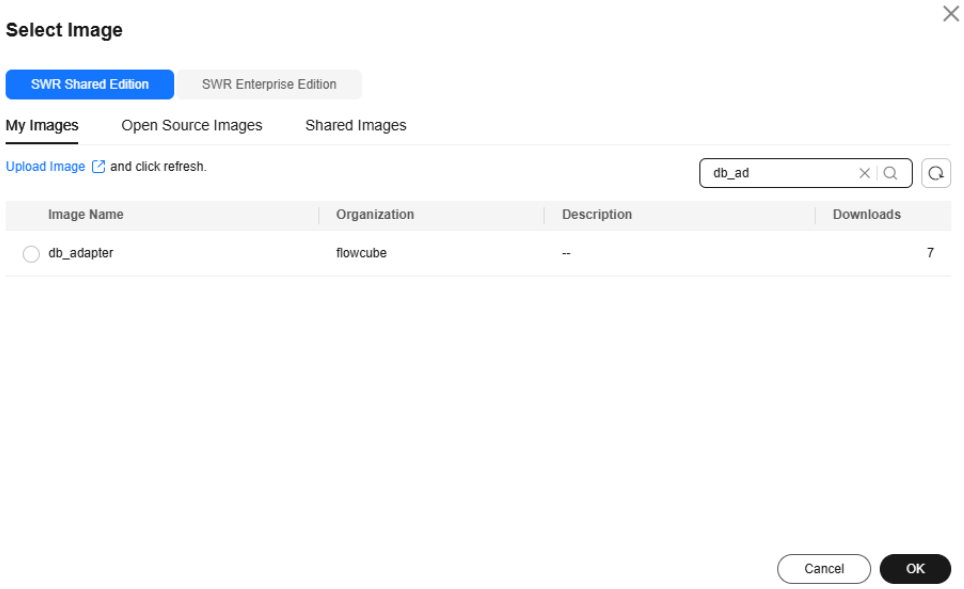


Table 8-9 describes only parameters used in this practice. For details about other parameters, see **Creating a Deployment** .

Table 8-9 Deployment parameters

Parameter	Description	Example
Workload Type	Set the workload type.	Deployment

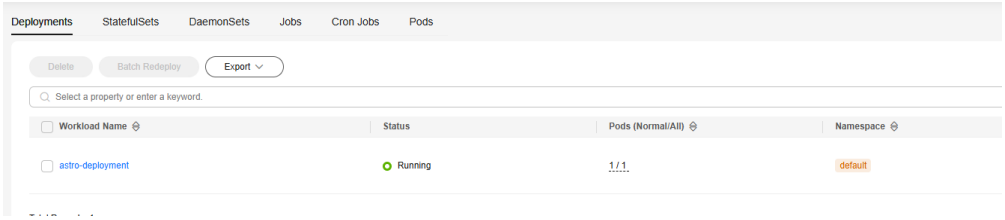
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Workload Name	Name of the workload to be created. Naming rules: – Max. 63 characters. – Start with a letter and end with a letter or digit. Use only lowercase letters, digits, and hyphens (-).	astro-deployment
Pods	Enter the number of pods of the workload.	1
Container Name	Enter the name of the container.	astro-container
Image Name	Click **Select Image** and select the image uploaded in **Prerequisites**.	db_adapter

Step 4 Add a service for the workload.

1. On the **Deployments** tab page, click the workload name to go to the details page.

   Figure 8-32 Clicking the workload name

2. On the access mode tab page, click **Create Service** to create a service and configure an access API.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 8-33 Creating a service

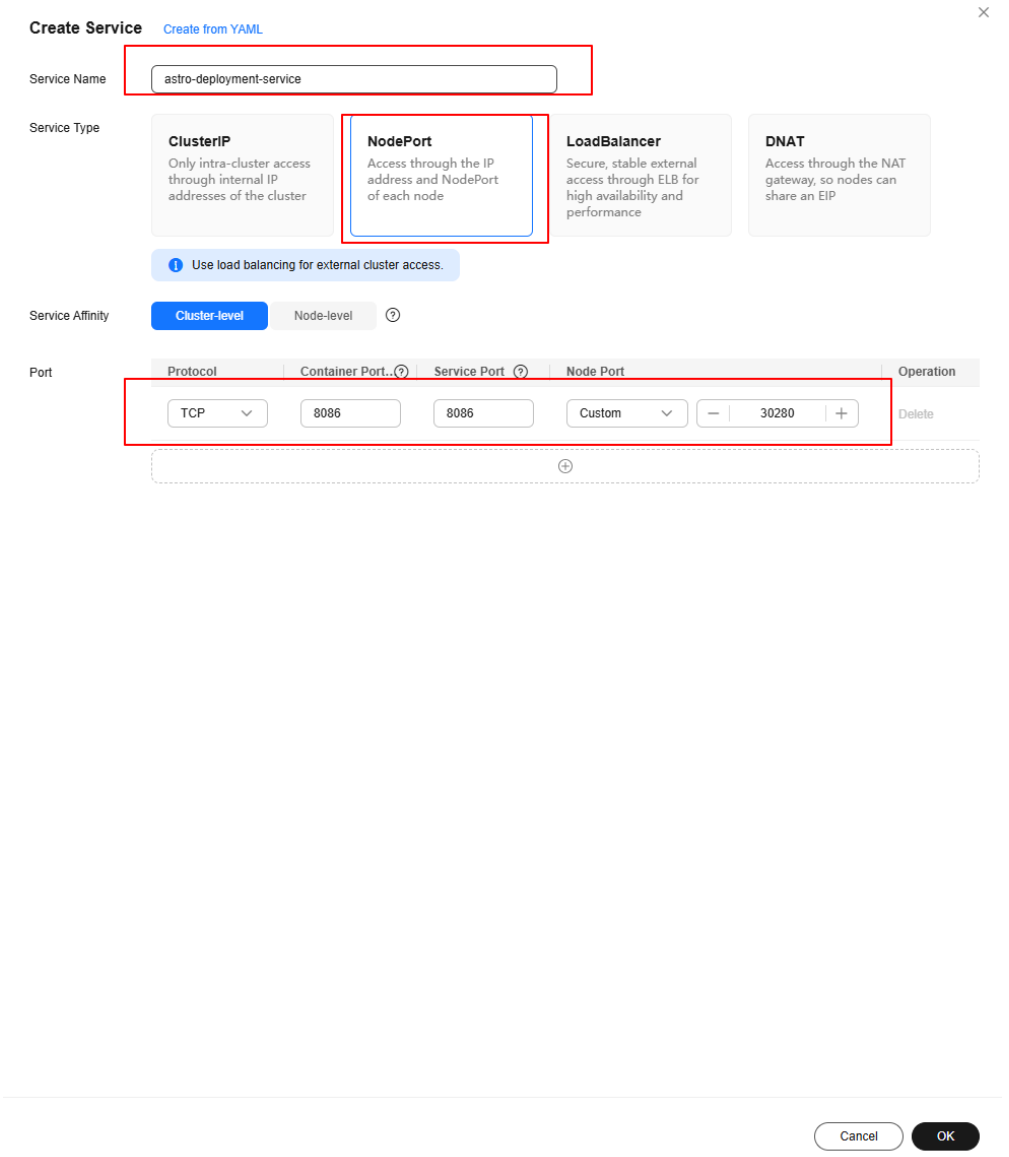


Table 8-10 Service parameters

Parameter	Description	Example
Service Name	Enter the name of the new service or retain the default value.	astro-deployment-service
Service Type	Select the access type of the service.	NodePort
Ports > Protocol	Select the protocol used by the service.	TCP

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Ports > Container Port	Specify the port used by the Service. The port number ranges from 1 to 65535.	8086
Ports > Service Port	Specify the port used by the service. The port number ranges from 1 to 65535.	8086
Ports > Node Port	Select **Auto** or specify a port. The default port ranges from 30000 to 32767.	30280

Step 5 Test the API connectivity.

Combine the **IP address** of the node in **Step 2** and the **node port** configured in **Step 4.2** to form a complete access address. The access address consists of the IP address and port number of the CCE node, **context-path** in **db-adapter**, and the REST API, for example, http://*10.10.10.1*:30280/astro/native/demo/dbadapter/mysql/datasource1/read.

Figure 8-34 context-path

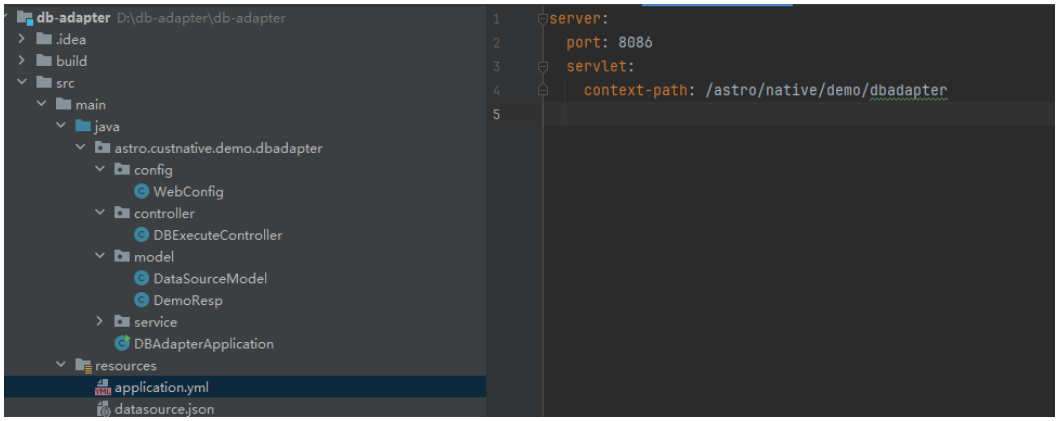
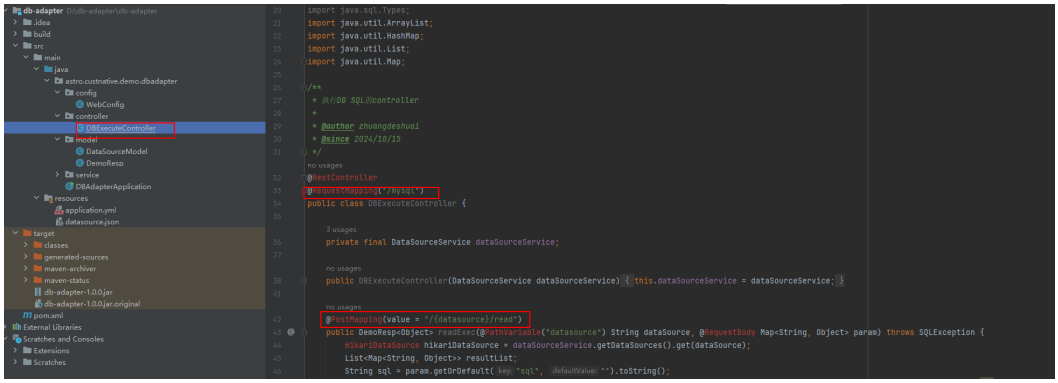


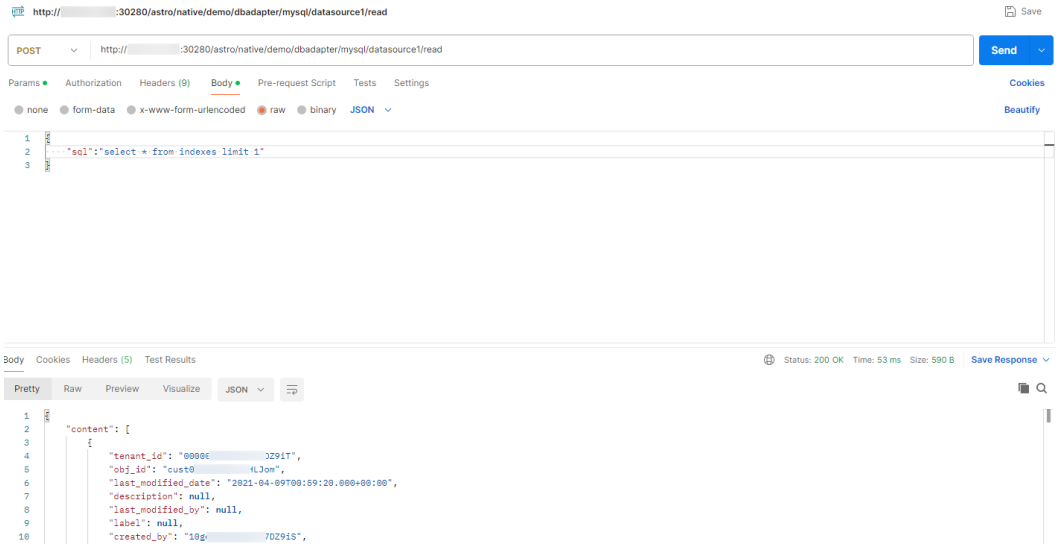
Figure 8-35 context-path and the REST API



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

On the **Body** tab page, enter SQL statements, click **Send**, and check the API status. If the status is **200 OK**, the API debugging is successful.

Figure 8-36 Testing API connectivity



----End

Connecting to Third-party Databases with a Connector

Step 1 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After the instance is purchased, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 8-37 Creating a blank application

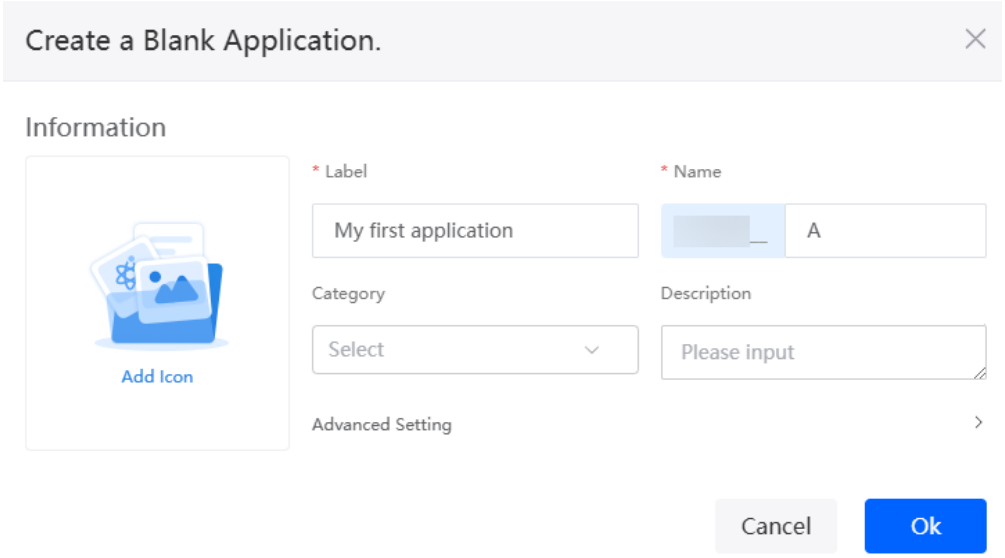


Table 8-11 Parameters for creating a blank application

Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and use only letters, digits, and underscores (_). Do not end with an underscore (_).	A

Step 2 In the application designer, choose **Integrations** > **Connector** > **Connector Instance**.

Step 3 In the **Type** area, select **Custom Connector**.

Step 4 Click **+**, configure the connector information, and click **Save**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 8-38 Creating a custom connector

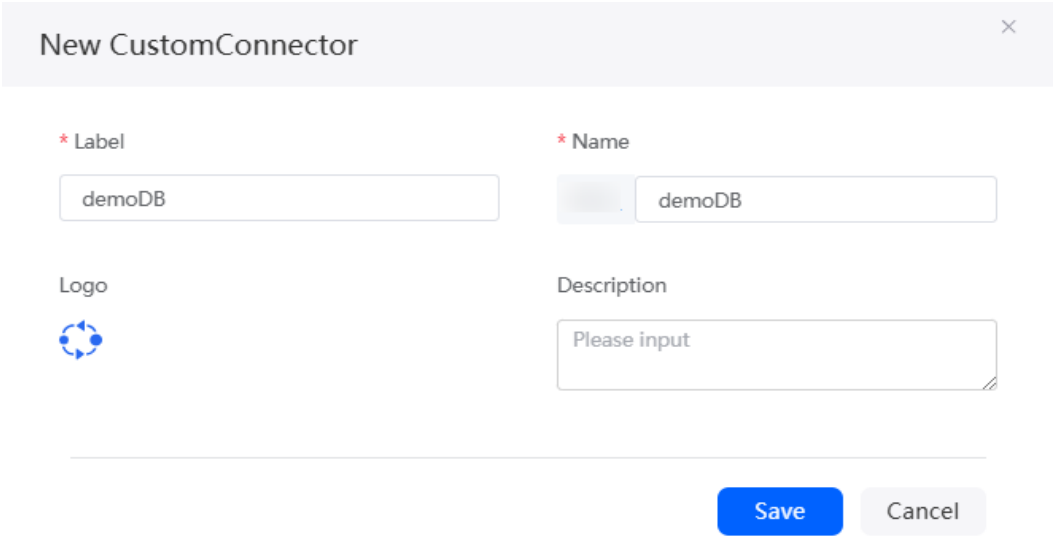


Table 8-12 Parameters for creating a custom connector

Parameter	Description	Example
Label	Custom connector label, which can be modified after being created. Value: 1–64 characters.	demoDB
Name	Custom connector name, which uniquely identifies the connector in the system and cannot be changed after the connector is created. Naming rules: • Max. 64 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants in Huawei Cloud Astro Zero, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. • Start with a letter and can contain only letters, digits, and underscores (_). It cannot end with an underscore (_).	demoDB

Step 5 Create authentication information for the custom connector.

1. On the **Authentication Information** tab page, click **New**.
2. Set authentication information and click **Save**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 8-39 Adding authentication information

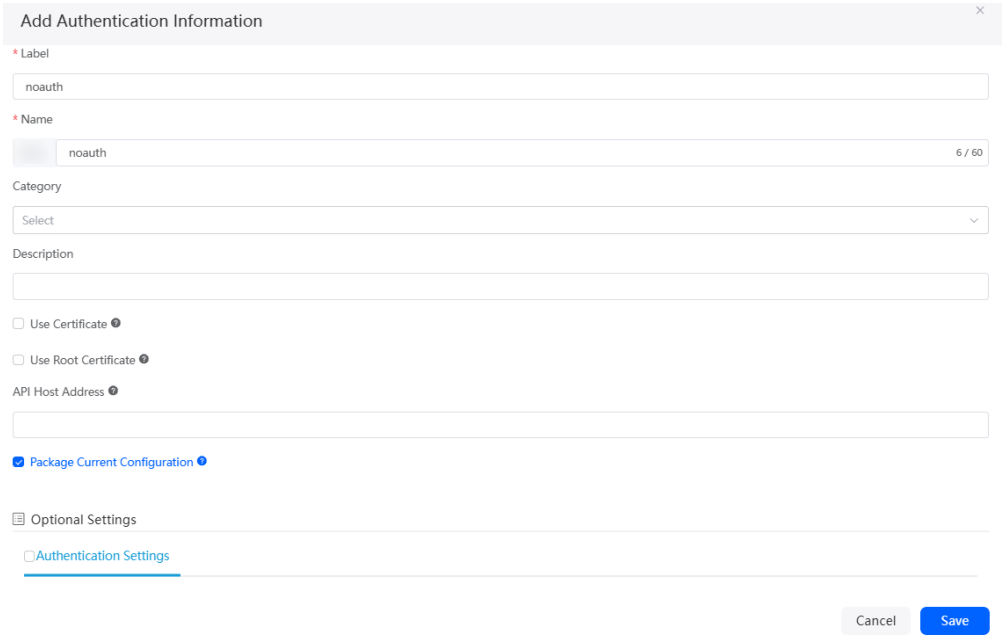


Table 8-13 Parameters for adding authentication information

Parameter	Description	Example
Label	Authentication label, which is displayed on the page. Value: 1–64 characters.	noauth
Name	Authentication name, which uniquely identifies the authentication in the system and cannot be changed after being created. Naming rules: – Max. 64 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants in Huawei Cloud Astro Zero, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and can contain only letters, digits, and underscores (_). It cannot end with an underscore (_).	noauth

Step 6 Add an action to the custom connector.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1. On the **Action** tab page, click **New**.
2. After setting the information, click **Next**.

Figure 8-40 Setting basic information

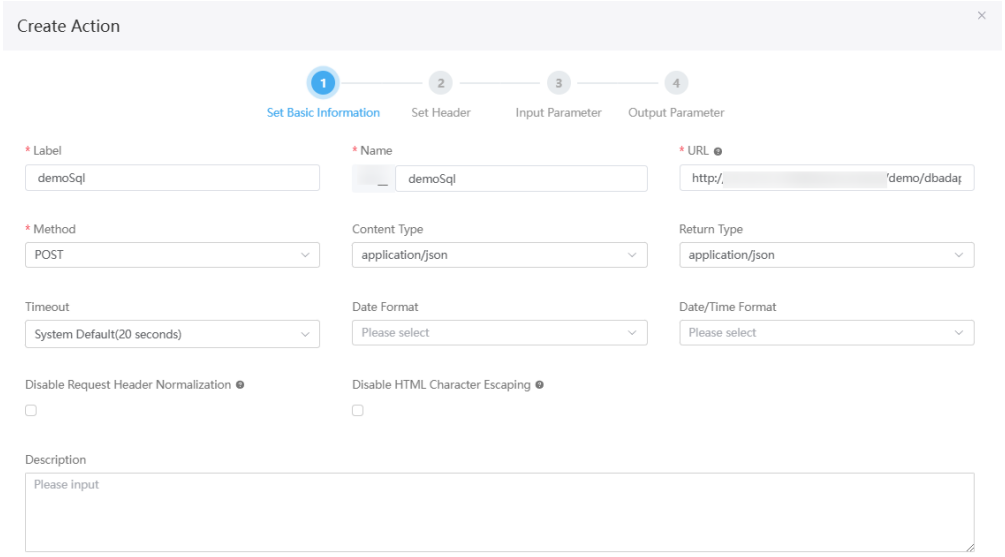


Table 8-14 Parameters for creating an action

Parameter	Description	Example
Label	Action label, which is displayed on the page. Value: 1–64 characters.	demoSql
Name	Action name, which uniquely identifies the action in the system. Naming rules: – Max. 64 characters, including the prefix namespace. The content that is blurred in front of the ID is a namespace. To prevent duplicate data names among different tenants in Huawei Cloud Astro Zero, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and can contain only letters, digits, and underscores (_). It cannot end with an underscore (_).	demoSql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

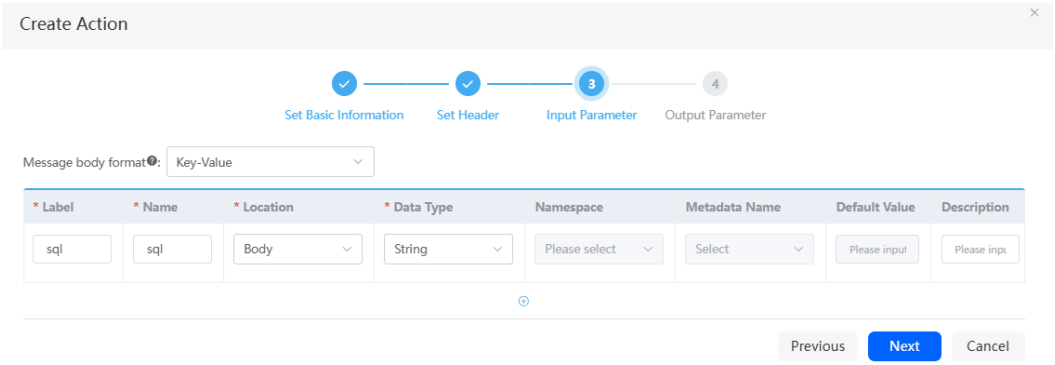
Parameter	Description	Example
URL	URL of the RESTful service provided by the third-party system (the address successfully debugged in **Step 5**).	http://10.10.10.1:30280/astro/native/demo/dbadapter/mysql/datasource1/read
Method	Select a request method.	POST
Content Type	The value depends on the content type supported by the third-party system and is defined in HTTP.	application/json
Return Type	Select a return type. If the value is **application/json**, the **content-type** configured by the third-party system rather than that returned by it will be used to parse the returned content. If this parameter is left blank, **content-type** returned in the HTTP response will be used to parse the content.	application/json

Step 7 In this example, you do not need to set the message header input parameters.

Click  to delete the parameters in the first row and click **Next**.

Step 8 Set input parameters and click **Next**.

Figure 8-41 Setting input parameters



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-15 Description of input parameters

Parameter	Description	Example
Label	Label of an input parameter, which is displayed on the page.	sql
Name	Name of an input parameter, which uniquely identifies the input parameter in the system.	sql
Location	Position of the input parameter.	Body
Data Type	Data type of the input parameter. Select a value from the drop-down list.	String

Step 9 In this example, you do not need to set output parameters. Click  to delete the parameters in the first row, and then click **Save**.

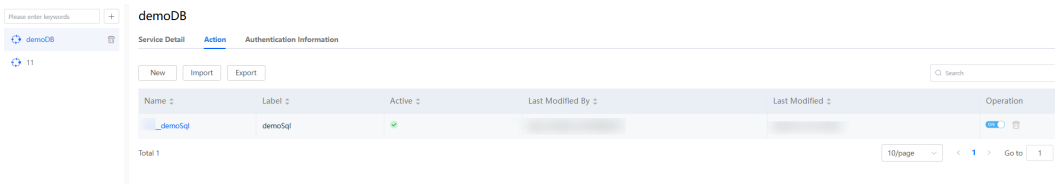
Step 10 On the **Action** tab page, click  in the row where the action is located to enable the action.

----End

Obtaining Data from a Third-Party Database

Step 1 On the **Action** tab page of the custom connector, click the action added in **Step 6**.

Figure 8-42 Clicking the action name

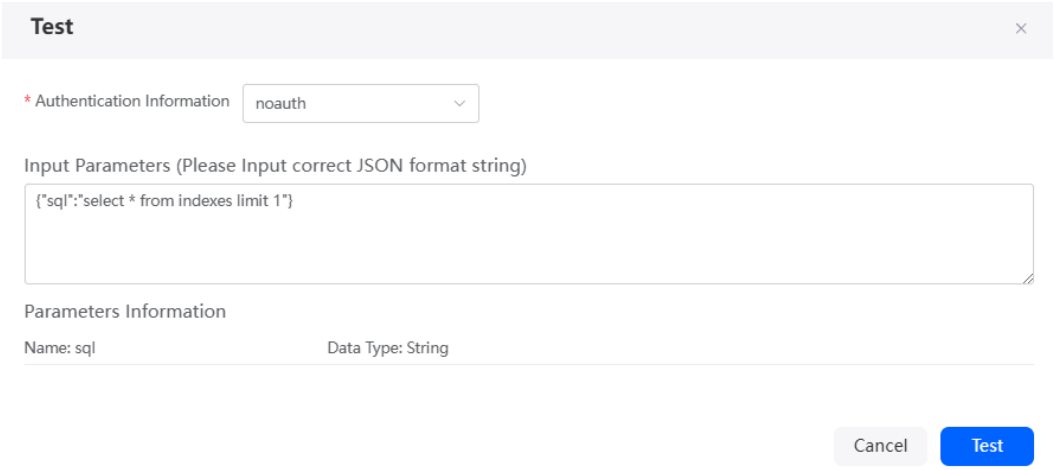


Step 2 In the upper right corner of the page, click **Test**.

Step 3 Set **Authentication Information** to **noauth** created in **Step 5**, set input parameters, and click **Test**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 8-43 Testing the action



For example, obtaining one record from the **indexes** table in the third-party database. **sql** is the input parameter added in **Step 8**, and **indexes** is the name of a table in the third-party database. Click **Test**. The following data is returned:


```
{
   "result": {
      "content": [
         {
            "created_by": "10gd000000bHFh7DZ9iS",
            "created_date": "2021-04-09T00:59:20.000+00:00",
            "description": null,
            "field_id1": "cust000000hDkpq2Puy0",
            "field_id2": null,
            "index_id": "cust000000inMlDAbr6X",
            "index_name": "HWAC__applicationId__CST",
            "index_type": 3,
            "label": null,
            "last_modified_by": null,
            "last_modified_date": "2021-04-09T00:59:20.000+00:00",
            "obj_id": "cust000000hBjuCHLJom",
            "status": 0,
            "tenant_id": "0000000000bHFh7DZ9iT"
         }
      ]
   },
   "dataForm": "2006-01-02",
   "dataTimeFormat": "2006-01-02 15:04:05",
   "reqMsg": "",
   "respMsg": "{\"url\":\"http://10.10.10.1:30280/astro/native/demo/dbadapter/mysql/datasource1/read
\",\"method\":\"POST\",\"parameters\":{\"content\":\"******\"},\"tenantID\":\"000000000142c5UhjFvk
\",\"statusCode\":200}",
   "timeElapsed": 207722711,
   "statusCode": 200,
   "headers": {
      "Connection": "keep-alive",
      "Content-Type": "application/json",
      "Date": "Fri, 13 Dec 2024 08:55:24 GMT",
      "Server": "nginx"
   }
}
```


----End

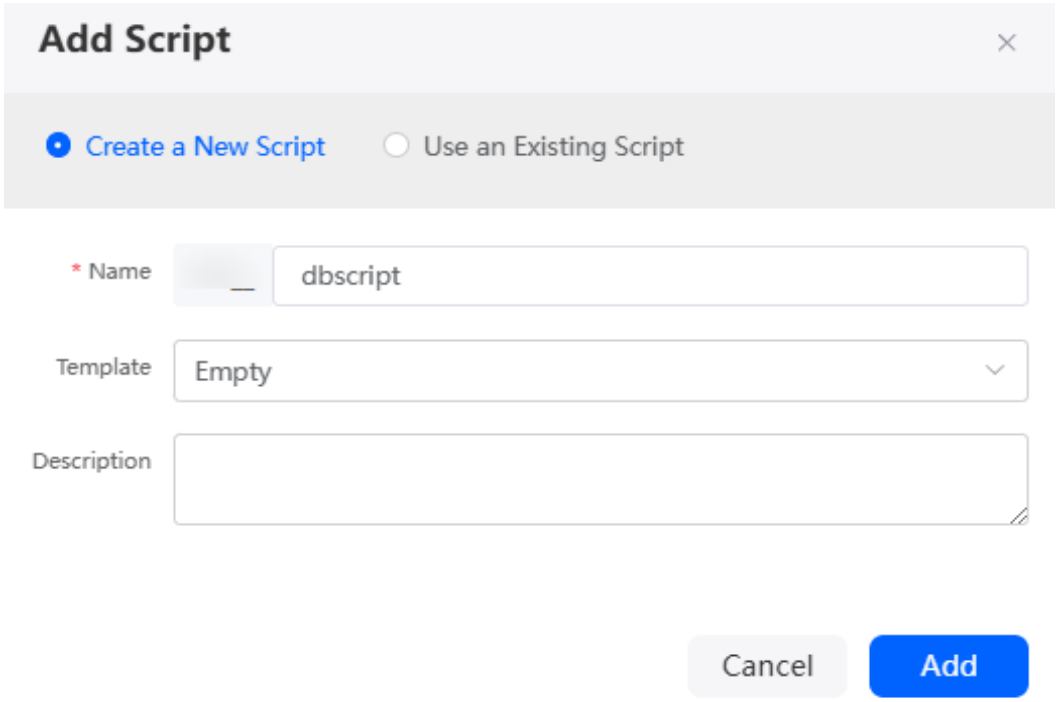
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Using a Script to Call the Connector to Obtain Data

Step 1 In the navigation pane, choose **Logic** and click **+** next to **Script**.

Step 2 Create a script, set the name to **dbscript**, and click **Add**.

Figure 8-44 Creating dbscript



Step 3 In the script editor, enter the sample code.

In the sample code, ***Namespace*__demoDB** is the connector name defined in **Step 4**, ***Namespace*__noauth** is the authentication information added in **Step 3**, and ***Namespace*__demoSql** is the action added in **Step 6**.

```
import * as connector from 'connector';
import * as context from 'context';//Import the standard library related to the context.


//Define the input parameter structure.
@action.object({ type: "param" })
export class ActionInput {
    @action.param({ type: 'String', required: false, label: 'String' })
    sql: string;
}
//Define the output parameter structure.
@action.object({ type: "param" })
export class ActionOutput {
    @action.param({ type: 'any' })
    res: any;
}

@action.object({ type: "method" })
export class CreateWorkOrder {    //Define the API class. The input parameter of the API is ActionInput, and the output parameter is ActionOutput.
    @action.method({ input: 'ActionInput', output: 'ActionOutput' })
    public createWorkOrder(input: ActionInput): ActionOutput {
        let out = new ActionOutput();    //Create an instance of the ActionOutput type as the return value.
        let error = new Error();    //Create an instance of the error type to save the error information when an error occurs.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

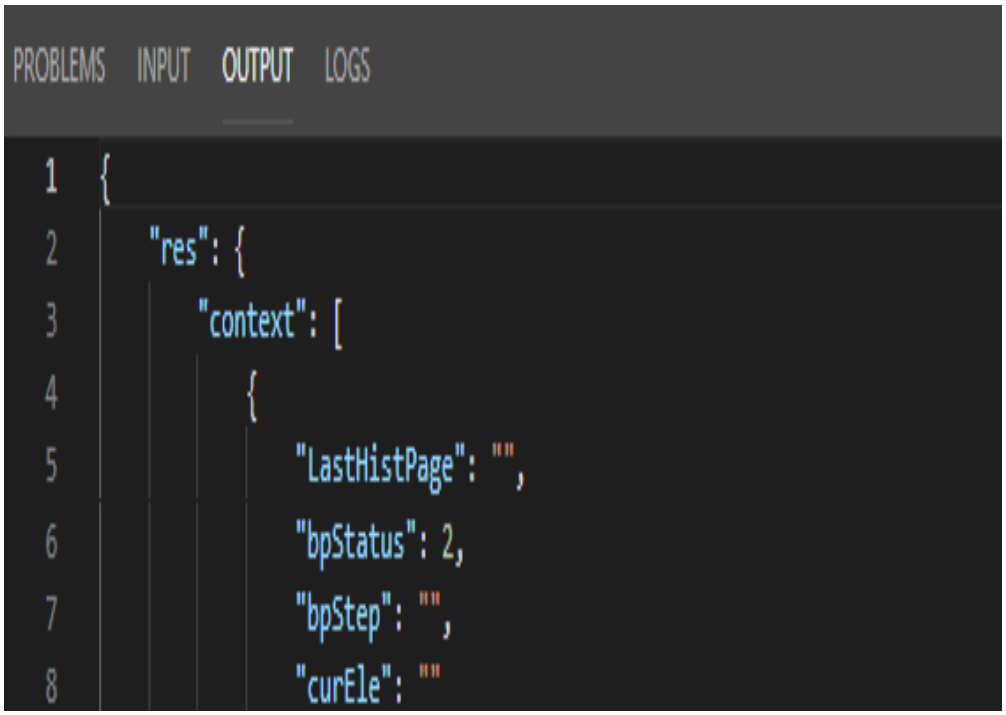
```
        try {
            // Method for calling the created custom connector. The first parameter is the name of the custom
connector, and the second parameter is the authentication information name.
            let client = connector.newClient("Namespace__demoDB", "Namespace__noauth");
            //Input parameters
            let input = { "sql": "select * from flowinstance limit 1" };
            // The first input parameter is the action name, and the second parameter is the input parameter.
            let resp = client.invoke(" Namespace__demoSql," input);
            // Print the output result.
            console.log(resp);
            out.res = resp;
        } catch (error) {
            console.error(error.name, error.message);
            context.setError(error.name, error.message);
        }
        return out;
    }
}
```

Step 4 Click in the upper part of the page to save the script.

Step 5 Click in the upper part of the editor to run the script.

Step 6 Click the in the upper right corner of the test window at the bottom of the page and check the returned database data on the output parameter tab page.

Figure 8-45 Checking the returned data



----End

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

9 Portal User

9.1 Developing a Login Page for Portal Users

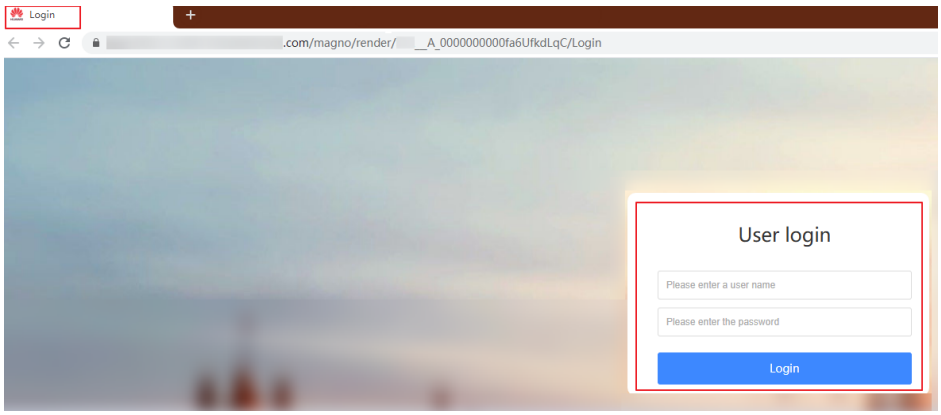
9.1.1 Overview

Application Scenarios

After an application is developed using Huawei Cloud Astro Zero, the system presets a default login page for the application. Users can log in to the application through the default login page. You can also customize a login page for your application using Huawei Cloud Astro Zero. The system authenticates a portal user by comparing the user information entered on the login page, such as the user name, password, and mobile number, with the portal user information stored in the system. Then, the system allocates resources and access permissions to the portal user based on configurations.

For example, customize a login page, as shown in **Figure 9-1**, enter the username and password, and click the login button. The login is completed by calling the user login flow through the custom login widget. Before customizing the login page, learn about **Portal User Login Methods** and **Portal User Login Mechanism**.

Figure 9-1 Custom login page



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Portal User Login Methods

A portal user can log in to Huawei Cloud Astro Zero in the background or frontend.

- When a portal user logs in to the background, the system uses the custom flow to call the login script, query the login account and password, and check whether the current login account and password are correct.
- Before a portal user logs in to the frontend, you need to develop a login widget and upload it to the advanced page, and configure the widget bridge data on the advanced page. Enter the login account and password on the login page, the system then calls the user login flow.

Portal User Login Mechanism

The service logic implementation process is as follows:

1. The system calls the script for verifying account and password to query the login account and password and check whether the current login account and password are correct.
2. If the account or password is incorrect, the system indicates that there is incorrect account or password. If the account and password are correct, the system checks whether a verification code is required.
3. If no verification code is required, the login operation is performed. If a verification code is available, the system checks whether the verification code is correct.
4. If the verification code is correct, the login operation is performed. If the verification code is incorrect, the verification fails.

9.1.2 Background Login

When a portal user logs in to the background, the system uses the custom flow to call the account and password verification script, query the login account and password, and check whether the current login account and password are correct. The process of developing the portal user flow is as follows: Drag one script diagram element, three decision diagram elements, and three value assignment diagram elements. Configure the properties of each diagram element, configure the link type between diagram elements, and save and enable the configuration.

The following uses application **A** as an example to describe how to implement the background login logic.

Step 1: Create a Login Script for Password Verification

You need to create a script for verifying the account and password in advance.

Step 1 On the Huawei Cloud Astro Zero console, click **Access Homepage**. The application development page is displayed.

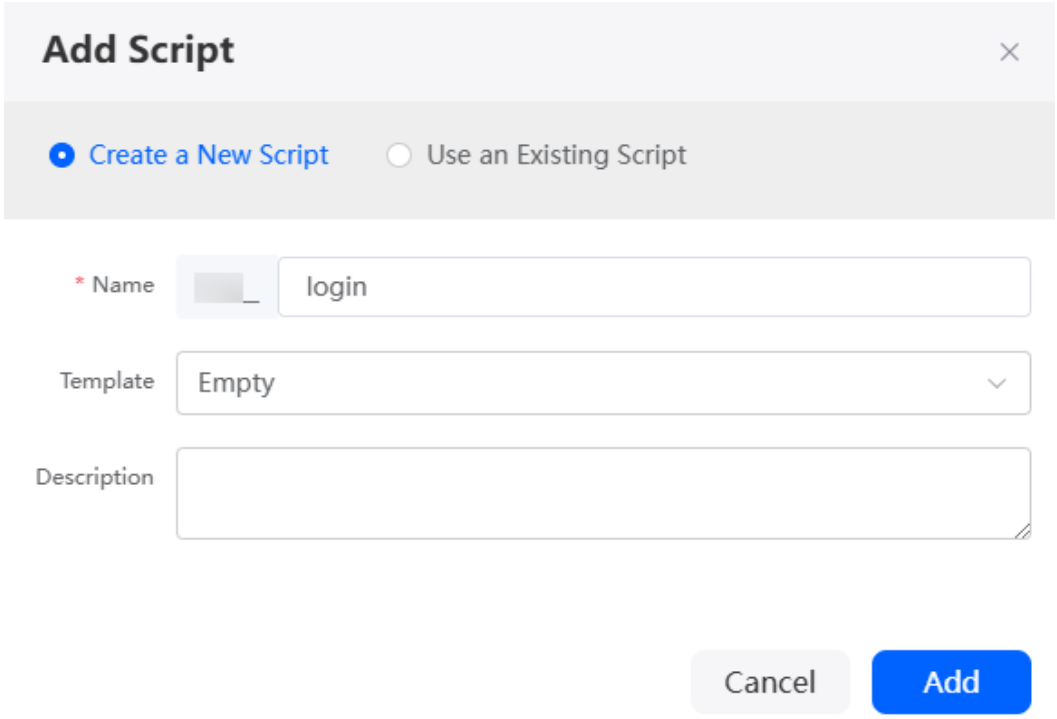
Step 2 On the **Homepage** > **All Apps** page, click **Edit** next to **A** application to access the application designer.

Step 3 In the navigation pane, choose **Logic** and click **+** next to **Script**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 4 Select **Create a New Script**. Enter **login** in the **Name** text box and click **Add**.

Figure 9-2 Adding a script



Step 5 Insert the following script code into the code editor:

```
import * as buffer from"buffer";
import * as crypto from"crypto";
import * as db from"db";
//Define the input parameter structure. The username and password of the account are mandatory. If other fields (such as the verification code) need to be verified, add them based on the format of the account and password fields.
@action.object({type:"param"})
export class ActionInput{
    @action.param({type:'String', required:true, label:'string'})
    username:string;    //Username
    @action.param({type:'String', required:true, label:'string'})
    password:string;    //Password
    @action.param({type:'String', required:true, label:'string'})
    captcha:string;     //Verification code. This script is used only to verify the account and password. Therefore, the verification code is not used and is not mandatory. In the flow for implementing portal user background login, the flow calls this script to check the verification code. Therefore, the verification code field is added to the script.
}
//Define the output parameter structure. You can add or delete fields in the structure based on the following example structure.
@action.object({type:"param"})
export class ActionOutput{
    @action.param({type:'String'})
    msg:string;//Login information
    @action.param({type:'String'})
    username:string;//Username
    @action.param({type:'String'})
    userId:string;//User ID
    @action.param({type:'String'})
    captcha:string;//Verification code
}
//Use the data object PortalUser.
@useObject(['PortalUser'])
@action.object({type:"method"})
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
export class Login{
//Define the API class. The input parameter of the API is ActionInput, and the output parameter is
ActionOutput    .
    @action.method({ input:'ActionInput', output:'ActionOutput'})
    public login(input:ActionInput):ActionOutput{
        let out =new ActionOutput();
        //Create an instance of the ActionOutput type as the return value
        let error =new Error();
        //Create an instance of the error type to save the error information when an error occurs
        try{
            out.captcha = input.captcha;
            let s = db.object('PortalUser');
            let condition ={
            "conjunction":"AND",
            "conditions":[{
                "field":"usrName",
                "operator":"eq",
                "value": input.username
                }]
            };
            let user = s.queryByCondition(condition);
            if(user && user.length ==1){
                if(validate(user[0].passwordSalt, user[0].userPassword, input.password)){
                    out.msg ="Login succeeded!";
                    out.username = user[0].usrName;
                    out.userId = user[0].id;
                }else{
                    out.msg ="Incorrect account or password!";
                }
            }else{
                    out.msg ="Incorrect account or password!";
            }
        }catch(error){
            console.error(error.name, error.message);
            out.msg = error.message;
        }
        return out;
    }
}
function _salt(password:string, saltBuf: buffer.Buffer, encoding: buffer.Encoding=
buffer.Encoding.Base64):string{
    const passwordBuf = buffer.from(password)
    const crypt = crypto.pbkdf2(passwordBuf, saltBuf,1000,32, crypto.Hashs.SHA1)
    return crypt.toString(encoding)
}
function validate(salt:string, userSaltedPassword:string, password:string, encoding: buffer.Encoding=
buffer.Encoding.Base64):boolean{
    const saltBuf = buffer.from(salt, encoding);
    const saltedPassword = _salt(password, saltBuf, encoding);
    return saltedPassword === userSaltedPassword
}
```

Step 6 Click  in the upper part of the editor to save the script.

Step 7 Test whether the script can be executed properly.

1. Click  in the upper part of the editor to run the script.
2. Enter the following test data in the input parameters at the bottom of the page and click  .

   In the preceding command, **test_cs** and *{XXXXXXXX}* indicate the registered account and password, and **captcha** is optional.

   The portal user created in the service configuration center cannot be the same as the portal user created using a script. Therefore, the portal user (for example, **test_cs**) set here cannot be the portal user created in the service

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

configuration center. Portal users created in the service configuration center can be used only on the default login page.

```
{
   "username": "test_cs",
   "password": "{XXXXXXXX}",
   "captcha": ""
}
```

After the execution is successful, you can check the result in the output parameter area.

```
{{
   "captcha": "",
   "msg": "Login succeeded.",
   "userId": "10*******A",
   "username": "test_cs"
}
```

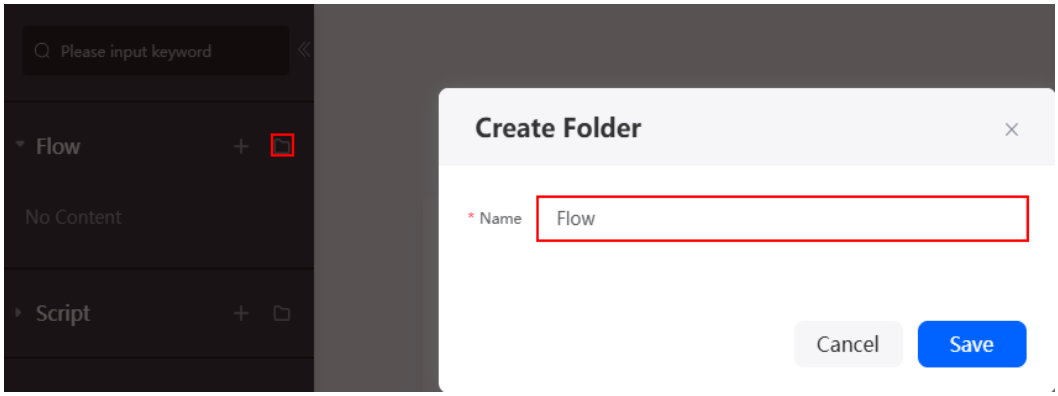
Step 8 After the test is successful, click  in the upper part of the editor to activate the script.

----End

Step 2: Develop the Background Login Logic with a Flow

Step 1 In the designer of application **A**, click **Logic** in the navigation pane, click  next to **Flow**, and create a folder.

Figure 9-3 Creating a folder

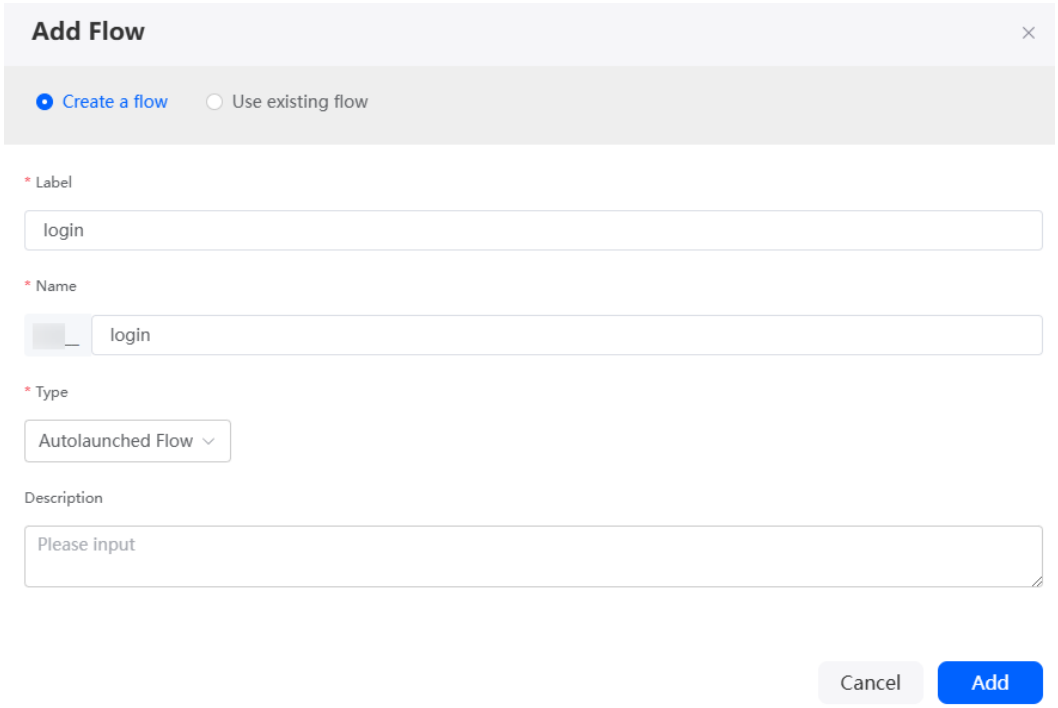


Step 2 Move the mouse cursor to **Flow**, click the plus sign (+) on the page to access the flow page.

Step 3 Select **Create a flow**. Set label and name to **login**, type to **Autolaunched Flow**, and click **Add**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

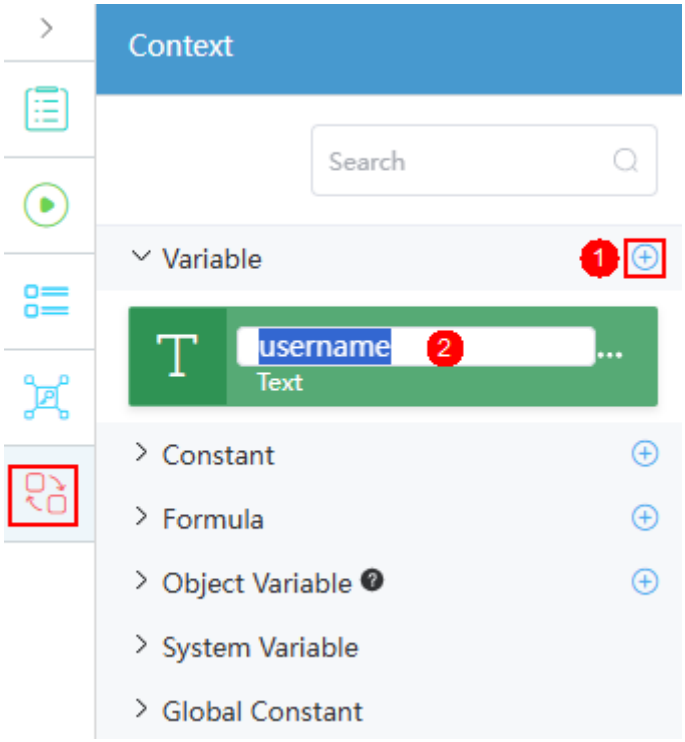
Figure 9-4 Creating a flow



Step 4 Define variables for the flow.

1. Click to expand **Context**, click next to **Variable**, and set the parameter name to **username**.

Figure 9-5 Adding a variable



2. Repeat the previous step to define other variables in **Table 9-1**.

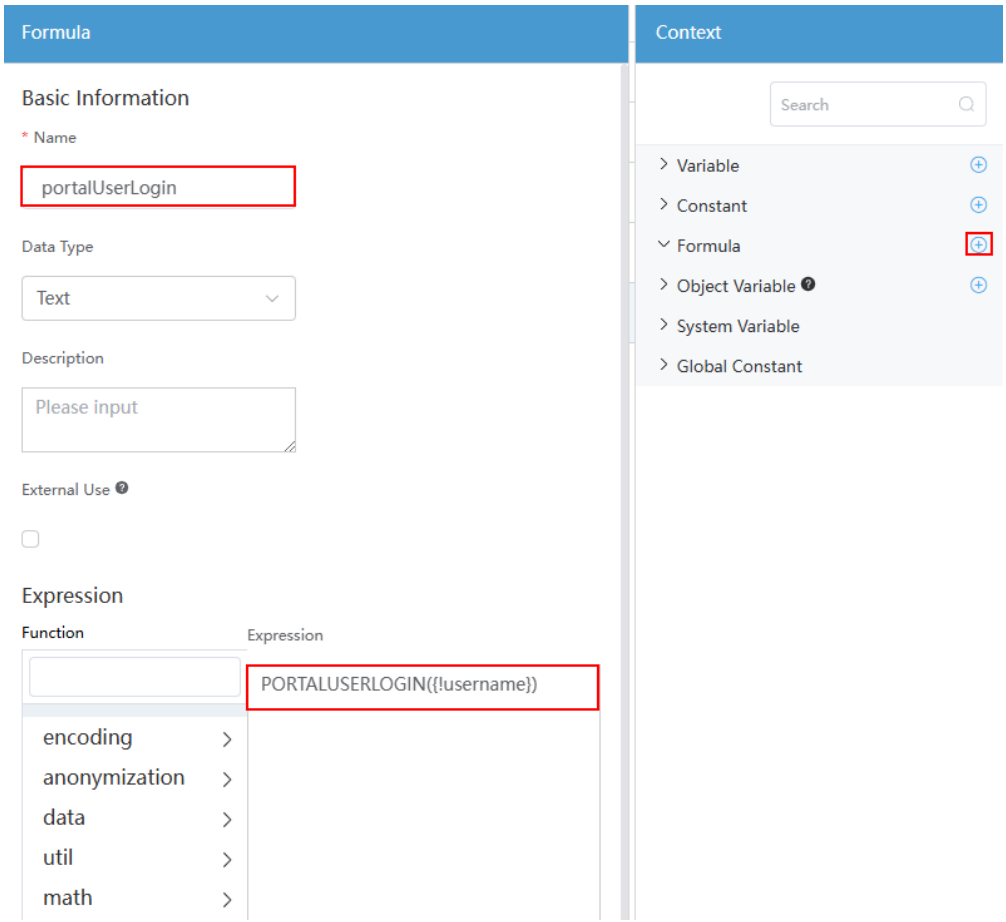
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 9-1 Flow variables

Name (Unique Identifier)	Data Type
username	Text
password	Text
captcha	Text
msg	Text
userId	Text
loginMsg	Text

3. Click ⊕ next to **Formula**. In the displayed dialog box, set the name to **portalUserLogin** and the expression to **PORTALUSERLOGIN ({!username})**, and click **Save**.

Figure 9-6 Adding the formula variable portalUserLogin



4. Create the formula variable **verifyCode** in Table 9-2 by referring to the previous step.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

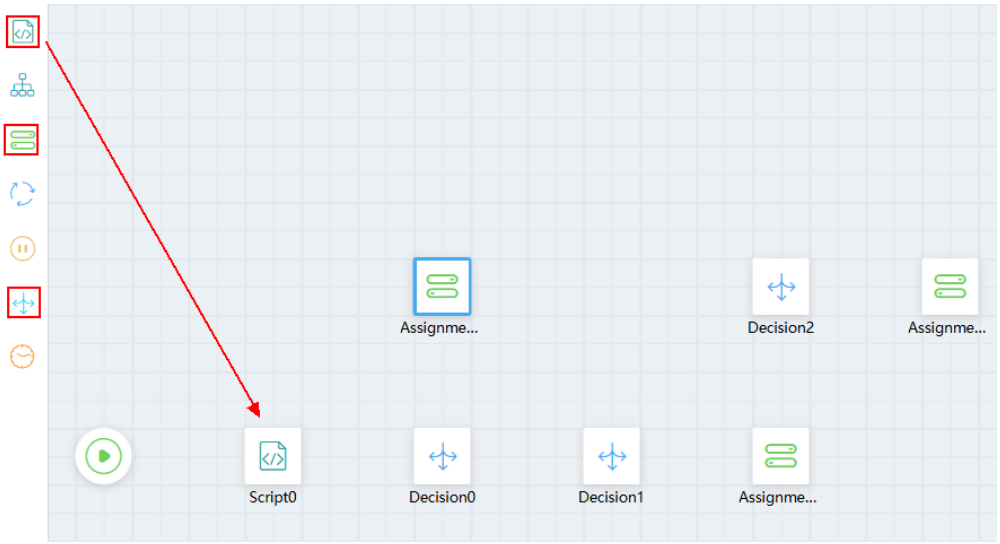
Table 9-2 Formula variables

Name	Expression
portalUserLogin	PORTALUSERLOGIN({!username})
verifyCode	VERIFYCODEWITHTYPE({!captcha},"login")

Step 5 Drag diagram elements to the flow canvas and configure basic properties.

1. Drag one script, three decision, and three assignment diagram elements to the canvas. The diagram elements are arranged as shown in the following figure.

Figure 9-7 Diagram elements arrangement



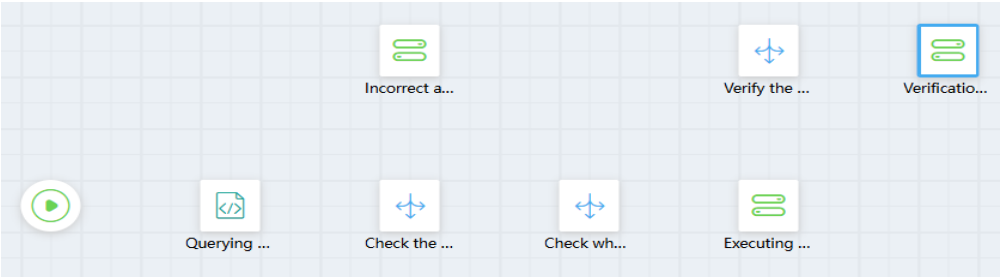
2. Select the **Script0** diagram element and set label to query user in the basic information area.
3. Set the label properties of other diagram elements by referring to the previous step. The following table lists label properties for each diagram element.

Table 9-3 Label properties for each diagram element

Name (Unique Identifier)	Label
Decision0	Verify the account and password
Decision1	Check whether a verification code is available
Decision2	Verify the verification code
Assignment0	Incorrect account or password
Assignment1	Perform login
Assignment2	Verification failed

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

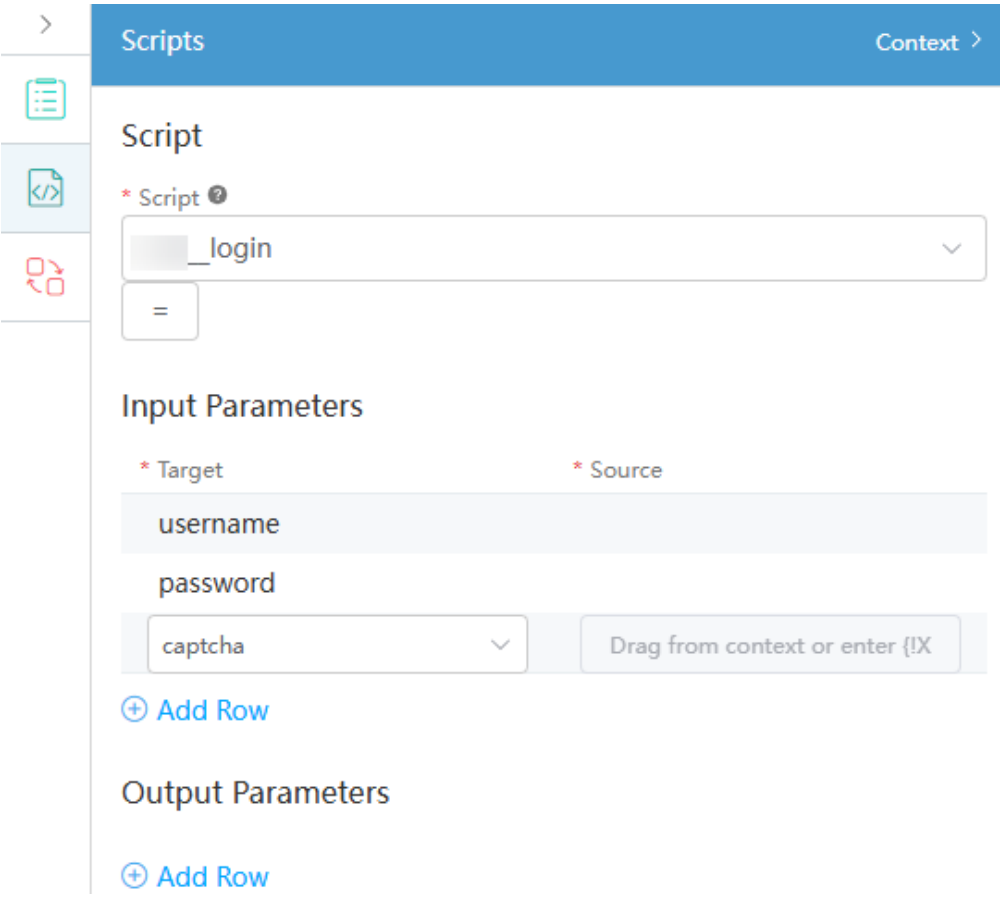
Figure 9-8 Diagram elements after modification



Step 6 Configure the **Query user** script diagram element.

1. Click , specify the script name (*Namespace*__**login**) corresponding to the diagram element, and set the input and output parameters of the script.

Figure 9-9 Specifying a script

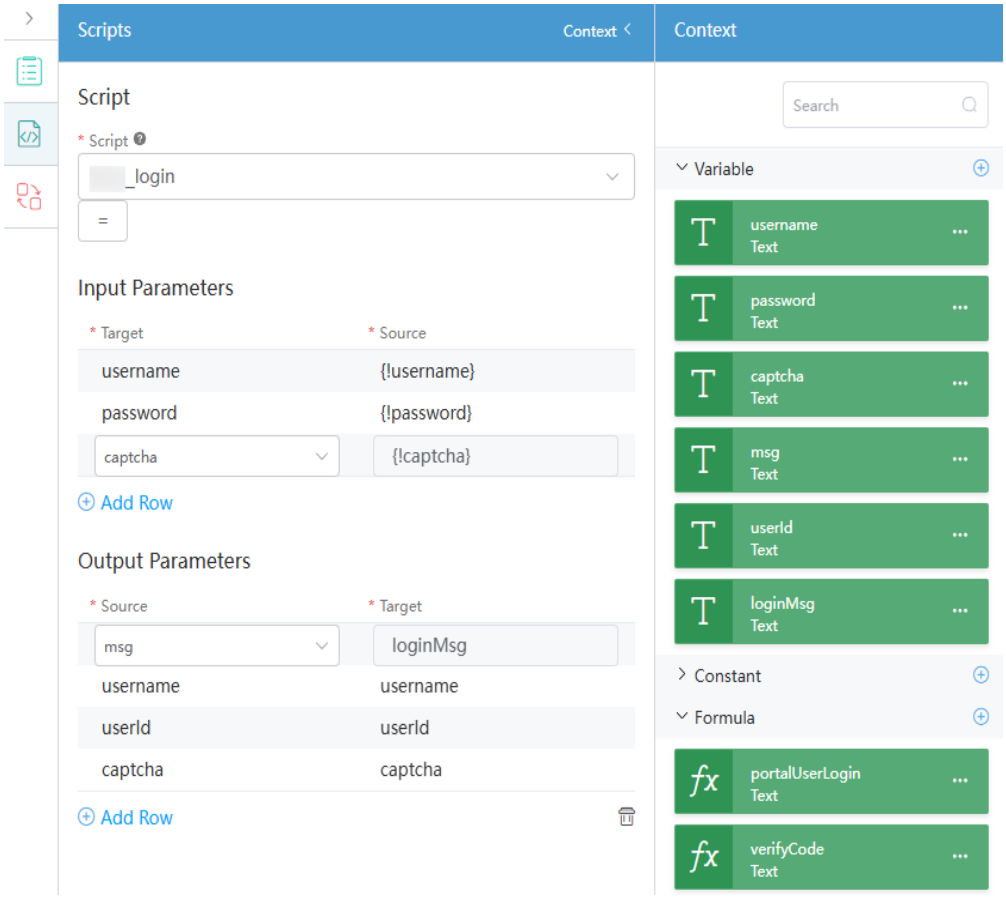


2. Click **Context**. The variable list is displayed. Drag **username**, **password**, and **captcha** from the variables area to the **Source** text box under **Input Parameters**. In the **Output Parameters** area, click **Add Row** four times, add the output parameter fields from the drop-down list box in sequence, and drag the corresponding fields from variable area to the **Target** text box. The following figure shows the mapping between fields and variables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- In the script diagram element, the number of input parameters and output parameters are the same as the input parameter fields required in the specified script. If the input parameters of the custom script contain extra fields, perform the same operations for the extra fields.
- Drag a variable from **Context** to the corresponding input and output parameter area. If you manually enter a value, ensure that the value is the same as that of the variable in **Context** area.

Figure 9-10 Dragging the input and output parameters for the script

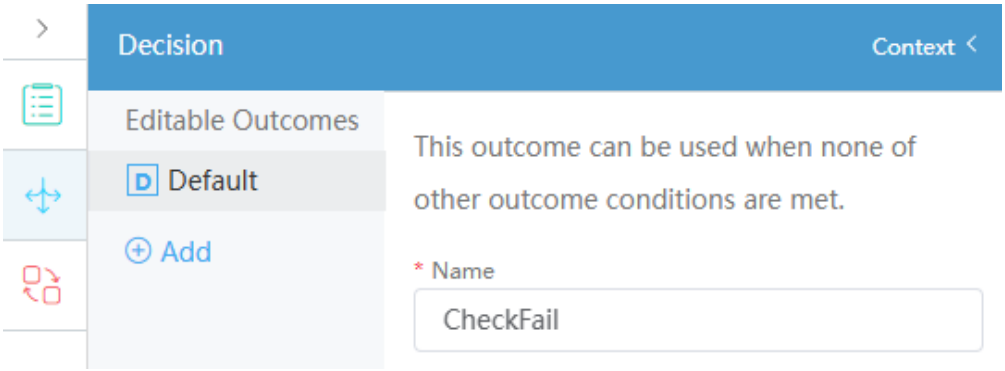


Step 7 Configure the decision diagram element of checking account and password.

1. Select the diagram element of checking account and password, click on the right, and change the default name to **CheckFail**.

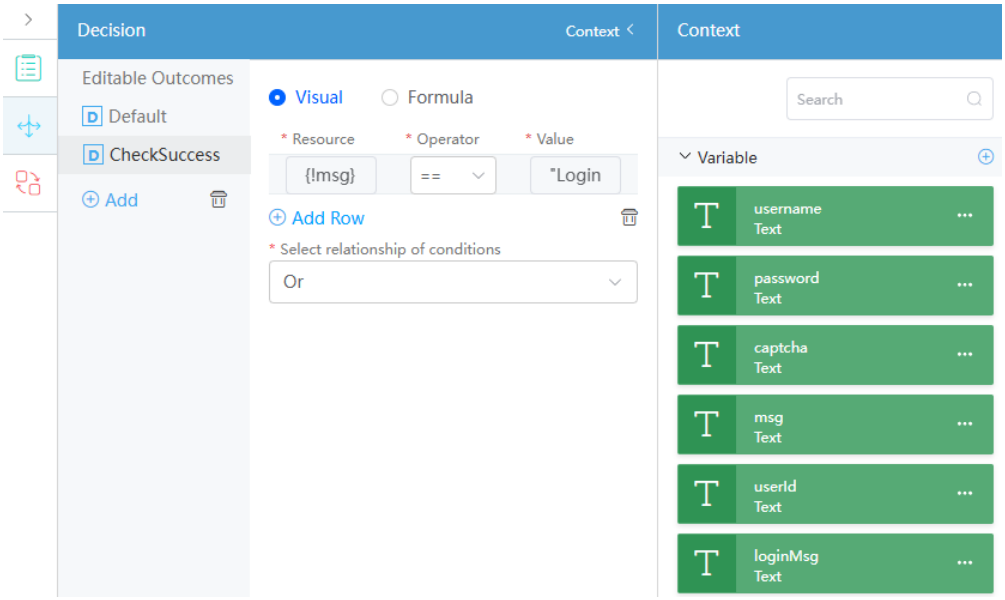
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 9-11 Modifying the default result name



2. Click **Add** to add an editable result. Change the result to **CheckSuccess**. Click **Add Row** under **Visual**, drag **msg** from the variable area to **Resource**, and set the comparison operator to **==** and the value to **"Login succeeded!"**.

Figure 9-12 Modifying the editable result



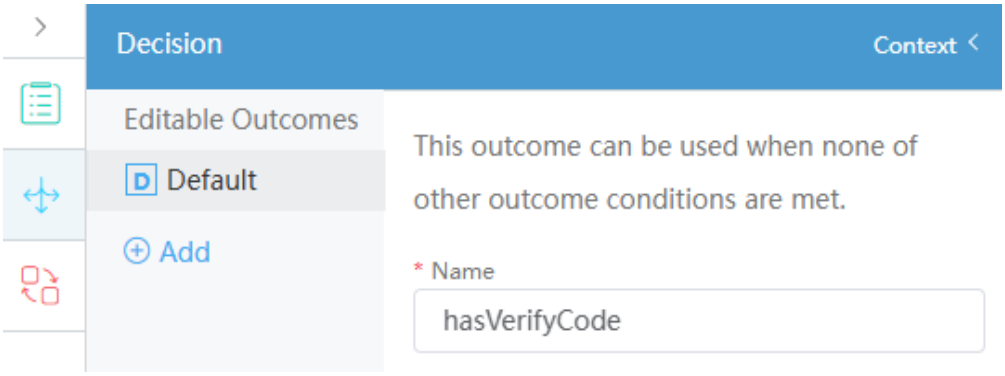
   - Drag the **msg** variable from **Context** to the resource area. If you manually enter a value, ensure that the value is the same as that of the variable in the context.
   - **"Login succeeded!"** must be the same as the output parameter in the **login** script.

Step 8 Configure the diagram element of checking whether the verification code is available.

1. Select the diagram element of checking whether the verification code is available, click  on the right, and change the default name to **hasVerifyCode**.

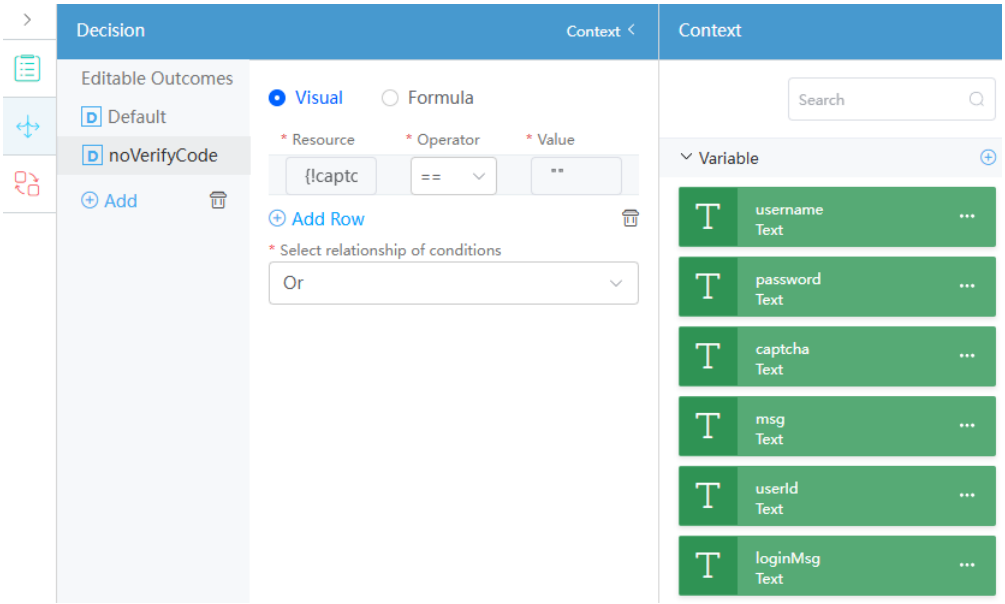
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 9-13 Modifying the default result name



2. Click **Add** to add an editable result and change the result to **noVerifyCode**. Click **Add Row** under **Visual**, drag **captcha** from the variable area to the resource area, and set the comparison operator to **==**, value to **""**.

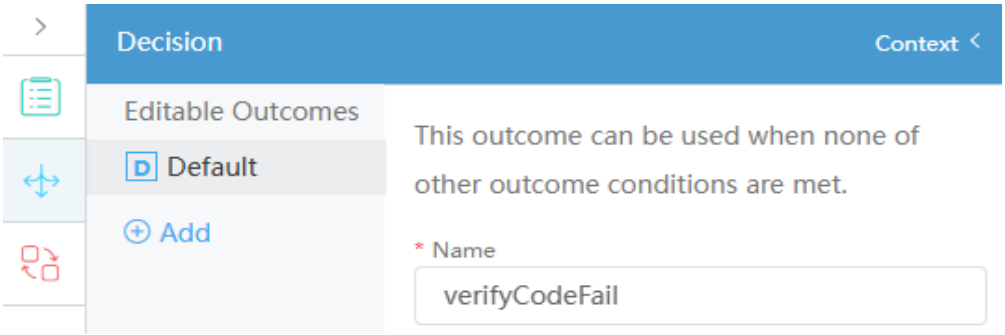
Figure 9-14 Modifying the editable result



Step 9 Configure the decision diagram element of verifying the verification code.

1. Select the diagram element of verifying the verification code, click on the right, and change the default name to **verifyCodeFail**.

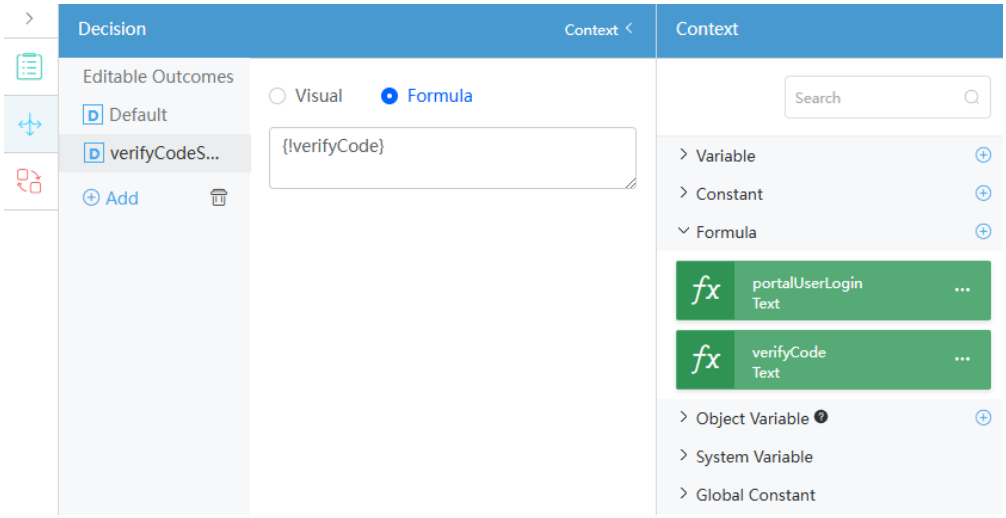
Figure 9-15 Modifying the default name



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

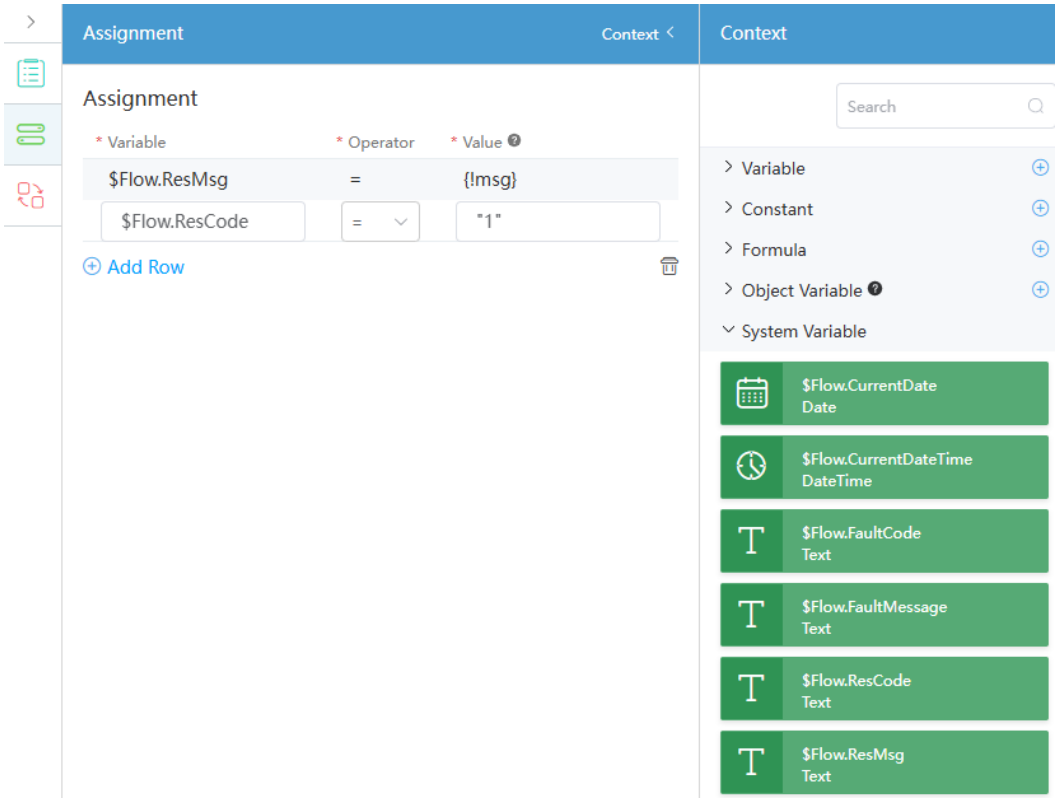
2. Click **Add** to add an editable result, change the result to **verifyCodeSuccess**, select **Formula** on the right, and drag **verifyCode** from the global context area to **Formula**.

Figure 9-16 Modifying the editable result



Step 10 Configure the diagram element of incorrect account and password.

Figure 9-17 Configuring the diagram element of incorrect account and password



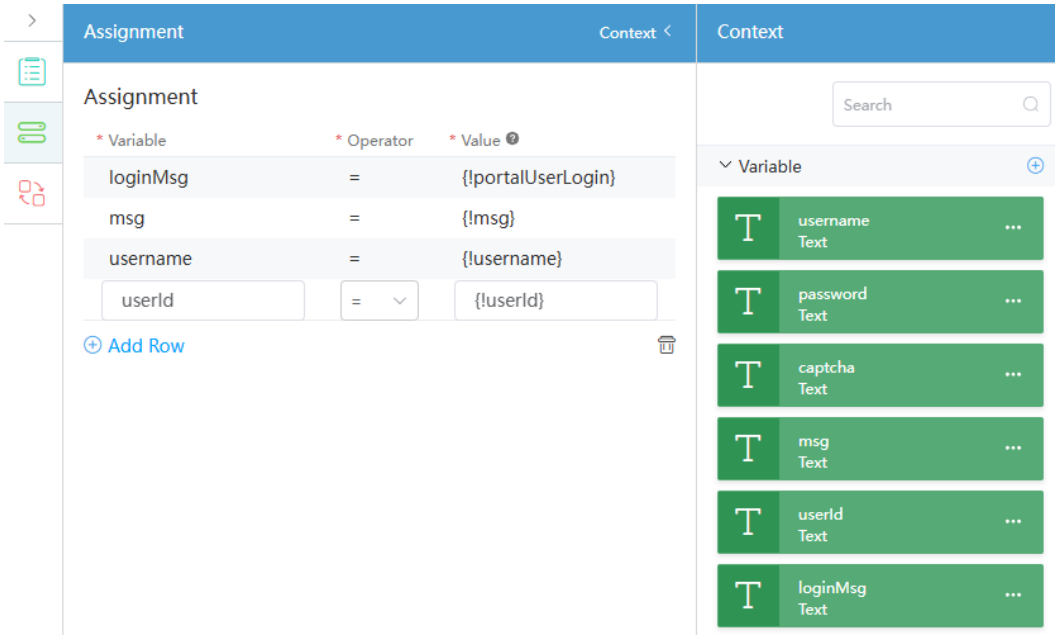
1. Select the diagram element of incorrect account and password, click on the right, and click **Add Row**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2. Drag **$Flow.ResMsg** from **System Variables** to **Assignment**, set the operator to =, and drag **msg** to **Value**.
3. Click **Add Row**, drag **$Flow.ResCode** from **System Variables** to **Variable** under **Assignment**, set the operator to =, and set **Value** to **1**.

Step 11 Configure the diagram element of performing login.

Figure 9-18 Configuring the variables and values



1. Select the diagram element of performing login, click on the right, and click **Add Row** four times.
2. Drag fields such as **msg** from the context area to **Variable** under the **Assignment** area, set the operator to =, and drag values to **Value**. The following figure shows the field mapping.

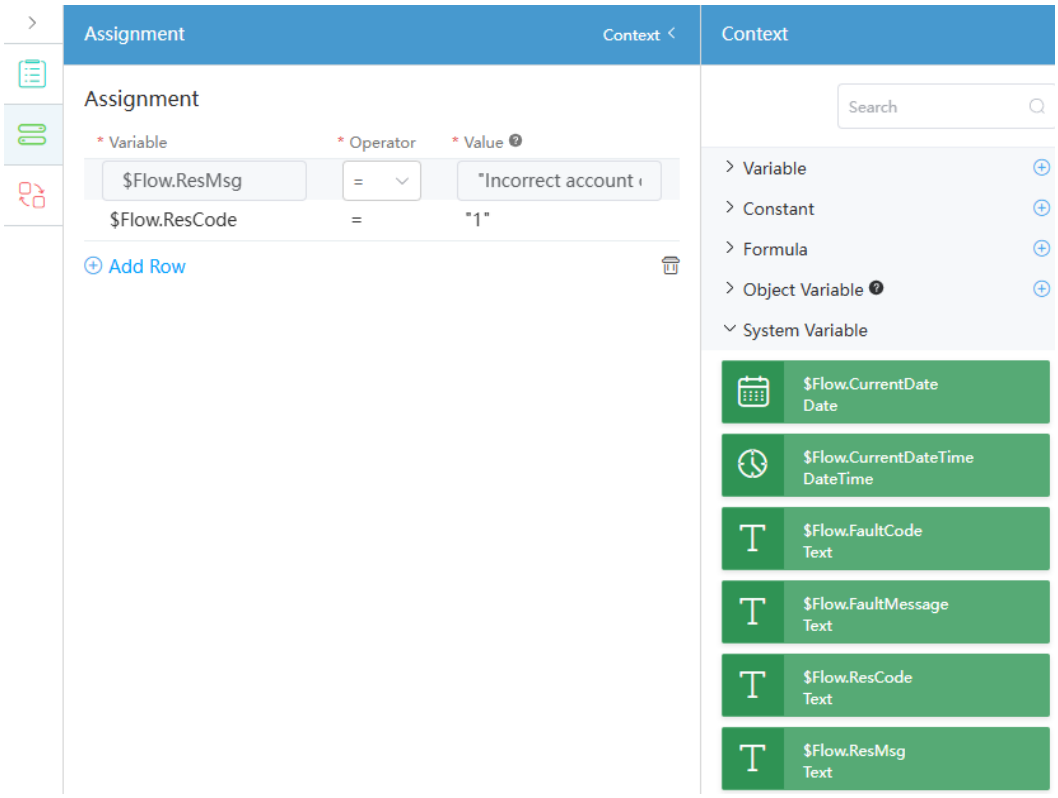
Table 9-4 Mapping between variables and values

Variable	Operator	Value
loginMsg	=	{!portalUserLogin}
msg	=	{!msg}
username	=	{!username}
userId	=	{!userId}

Step 12 Configure the diagram element of verification failed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 9-19 Configuring the diagram element of verification failed



1. Select the diagram element of verification failed, click on the right, and click **Add Row** four times.
2. Drag **$Flow.ResMsg** and **$Flow.ResCode** from **System Variables** to **Assignment**, set the operator to **=**, and set values to **"Incorrect account or password!"** and **"1"**.

Table 9-5 Assignment

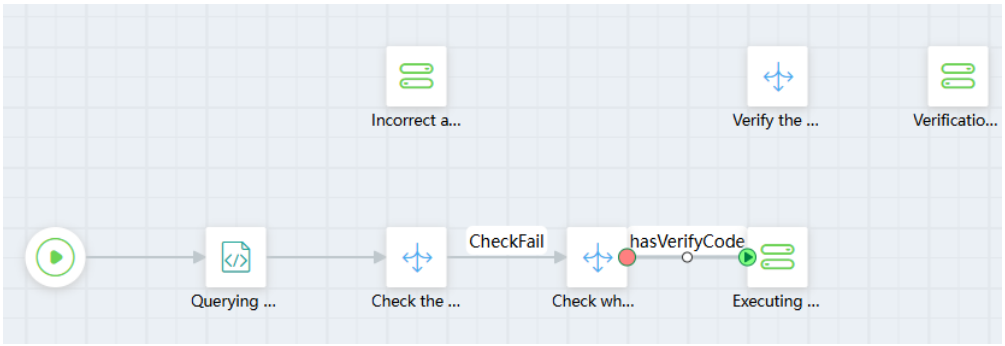
Variable	Operator	Value
$Flow.ResMsg	=	"Incorrect account or password!" "
$Flow.ResCode	=	"1"

Step 13 Link diagram elements and configure connection properties.

1. On the canvas, move the pointer to the start diagram element and drag the pointer from **+** to add a link line between the start diagram element and the query user diagram element. That is, set the current script as the start node of the flow.
2. Link the **Query user**, **Check the account and password**, Check **Whether the verification code is available**, and **Perform login** diagram elements in sequence.

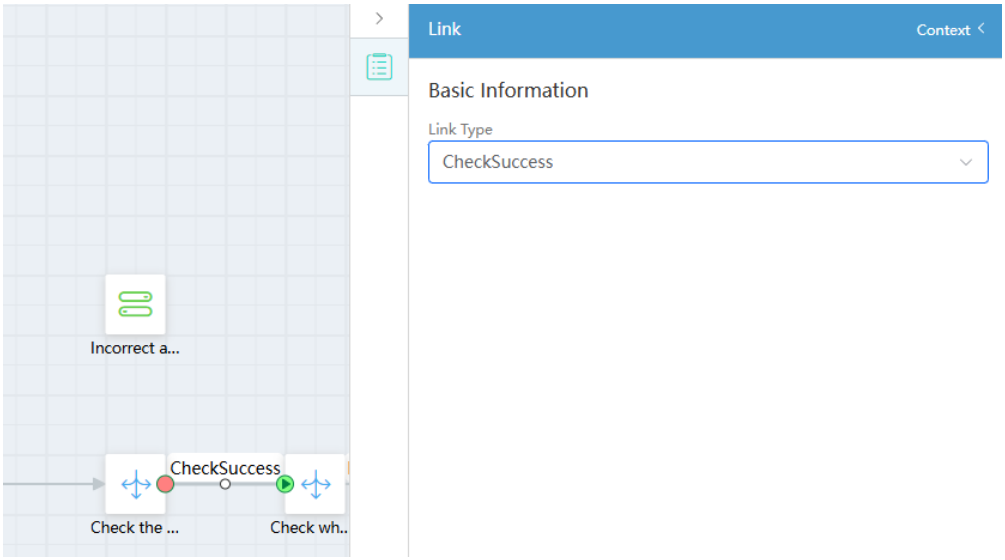
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 9-20 Linking diagram elements



3. Click the link between the **Check account and password** diagram element and the **Check whether the verification code is available** diagram element, click on the right, and change the link type to **CheckSuccess**.

Figure 9-21 Selecting a link

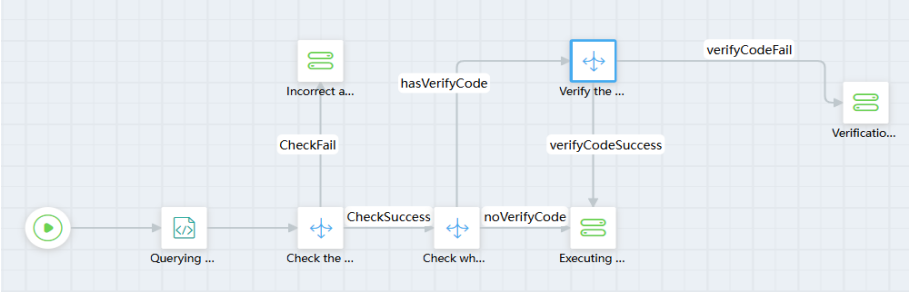


4. Click the link between the **Check whether the verification code is available** diagram element and the **Perform login** diagram element, click on the right, and change the link type to **noVerifyCode**.
5. Drag a line to connect the **Check the account and password** diagram element with the **Incorrect account and password** diagram element.
6. Drag a line to connect the **Check whether a verification code is available** diagram element with the **Verify the verification code** diagram element.
7. Drag a line to connect the **Verify the verification code** diagram element with the **Verification failed** diagram element.
8. Drag a line to connect the **Verify the verification code** diagram element with the **Perform login** diagram element and set the link type to **verifyCodeSuccess**.

The connections between diagram elements are as shown in the following figure.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

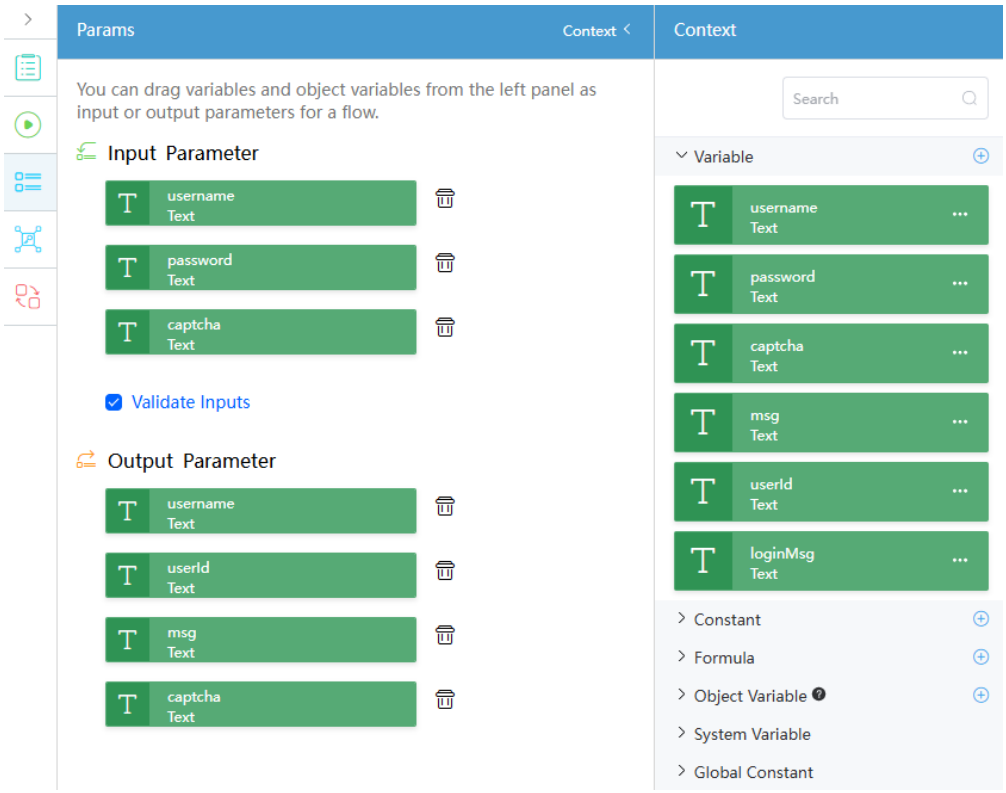
Figure 9-22 Connecting diagram elements



Step 14 Define the input and output parameters of the flow and save the flow.

1. Click the blank area on the canvas, click the on the right, and set the input and output parameters of the flow, as shown in the following figure.

Figure 9-23 Setting input and output parameters for the flow



- The input parameters of the flow are the parameters that are used for executing the flow and the script of account and password verification. Therefore, if the account and password verification script contains extra input parameters, the same input parameters should be added for the flow.
- The output parameters of the flow are the parameters returned by executing the script of account and password verification. Therefore, if the account and password verification script contains extra output parameters, the same output parameters should be added for the flow.

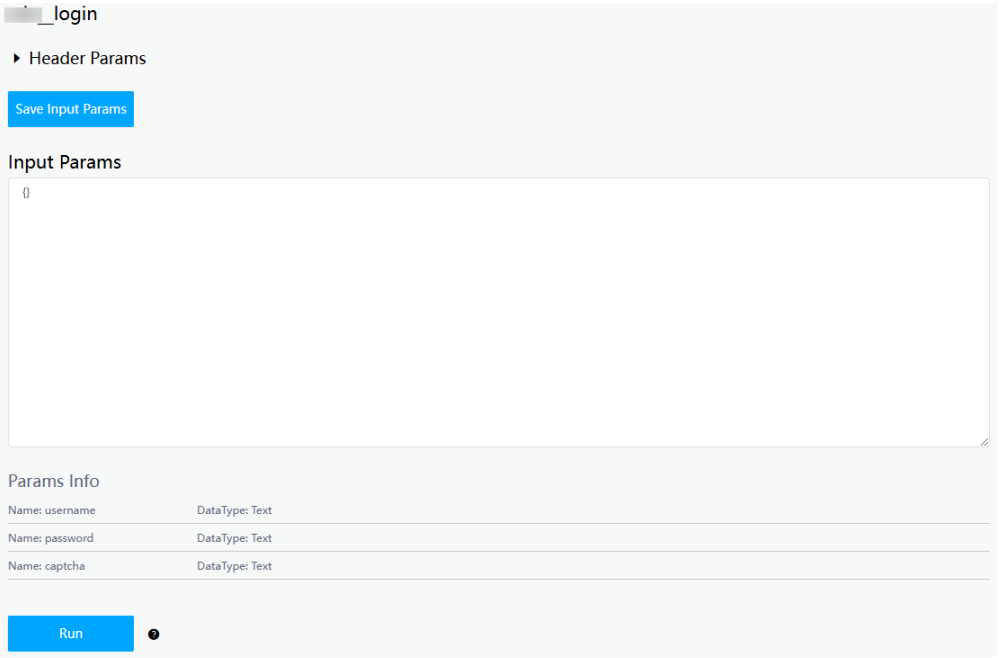
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2. Click  on the top of the flow page to save the flow.

Step 15 Test whether the flow can be executed properly.

1. Click  on the top of the flow page to access the flow test page.

Figure 9-24 Flow test page



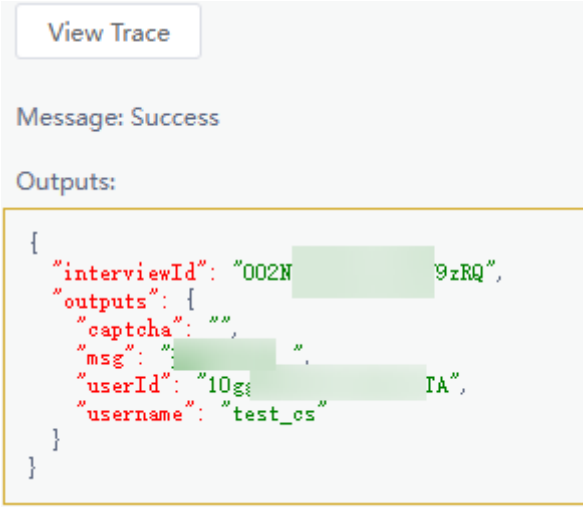
2. On the flow run page, enter the test data and click the run button.

   In the following parameters, **test_cs** and *** indicate the username and password of the portal user.

```
{
"username": "test_cs",
"password": "***",
"captcha": ""
}
```

   If the operation is successful, all device information is displayed.

Figure 9-25 Returned values



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The returned value indicates that the login is successful. After the login is successful, return to Huawei Cloud Astro Zero. Refresh the page. In the upper right corner of the page, you can see that the current login user is the portal user configured in the flow.

Step 16 After the test is successful, click  in the upper part of the editor to activate the flow.

----End

Step 3: Create an Open API

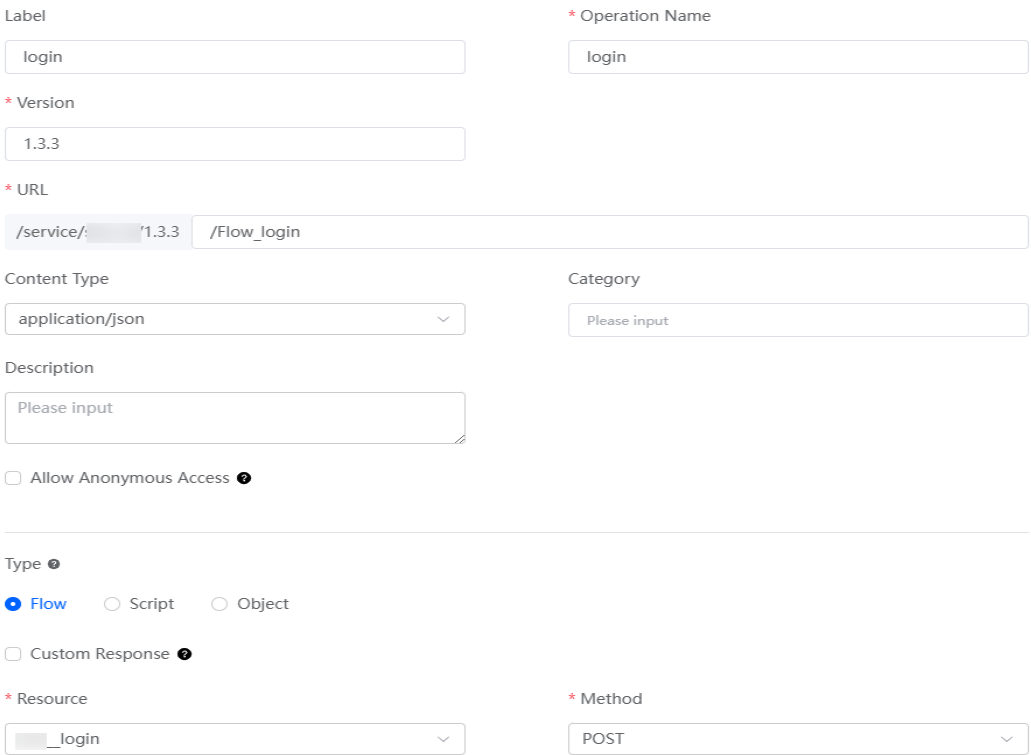
Open APIs are used to package scripts and flows developed by users in applications into customized RESTful APIs for third-party systems to call. In this example, the developed flow is packaged and released as a REST API for **Frontend Login** to call.

Step 1 Go to the designer of application **A**, choose **Integration** from the navigation pane.

Step 2 Click **+** next to **Open API**. The page for creating an open API is displayed.

Step 3 Complete the configuration and click **Save**.

Figure 9-26 Creating an open API



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 9-6 Open API parameters

Parameter	Description	Example
Label	The label for the created API; Max. 64 characters.	login
Operation Name	The operation name of the created API. The value must start with a letter and can contain only letters, digits, and underscores (_). It cannot end with an underscore (_) and can contain a maximum of 40 characters.	login
Version	API version, which is in the x.y.z format.	1.3.3
URL	API path, which starts with **/service/*{namespace}*__*{application name}*/*{version}***. The content set here is the URL provided by the new API for external access.	/service/*namespace*__A/*version*/Flow_login
Type	Resource type. Only **Flow** APIs can be called in flows. Other APIs can only be called through API request.	Flow
Resource	Select the resources to be bound based on the type.	Select the flow created in **Step 2: Develop the Background Login Logic with a Flow**. Ensure that the flow has been enabled. Otherwise, the flow cannot be selected.
Method	HTTP method of the API. In this example, **POST** is selected to request the server to add resources or perform special operations.	POST

----End

9.1.3 Frontend Login

Before logging in to the frontend as a portal user, you need to develop a login widget and upload it to the advanced page, and configure the widget bridge data on the advanced page. After you enter the login account and password on the login page, the system then calls the user login flow.

The following uses application **A** as an example to describe how to develop the login page. For details about the process, see **Figure 9-27**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 9-27 Login page development process

In addition to customizing the login widget, you can use the preset login widget on the advanced page for quick configuration. For details, see **Setting Properties of the Login Widget**.

Step 1: Develop a Custom Widget

You can refer to the following to develop a custom login widget. Alternatively, you can skip this step and download **userLogin.zip** directly.

Step 1 On the Huawei Cloud Astro Zero console, click **Access Homepage** to go to the application development page.

Step 2 Click  and choose **Environments** > **Environment Configuration**.

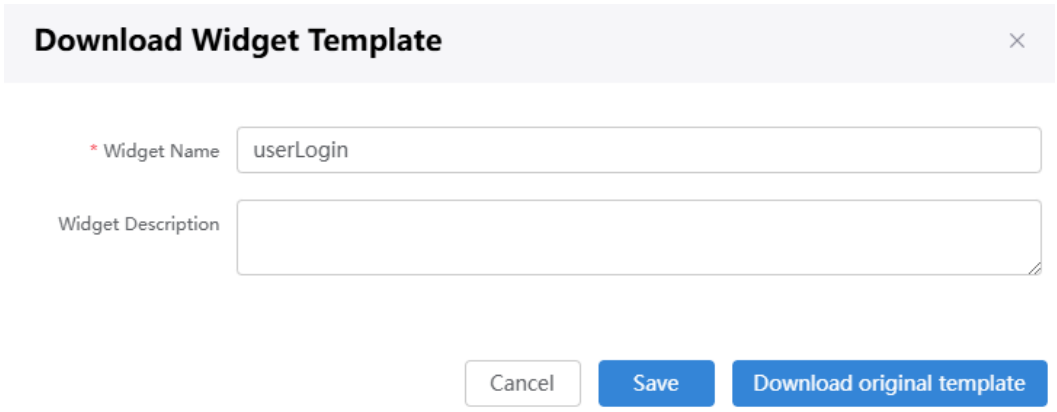
Step 3 Choose **Maintenance** from the main menu.

Step 4 In the navigation pane, choose **Global Elements** > **Page Assets** > **Widget Templates**.

Step 5 In the widget template list, click **widgetVue3Template**. The template details page is displayed.

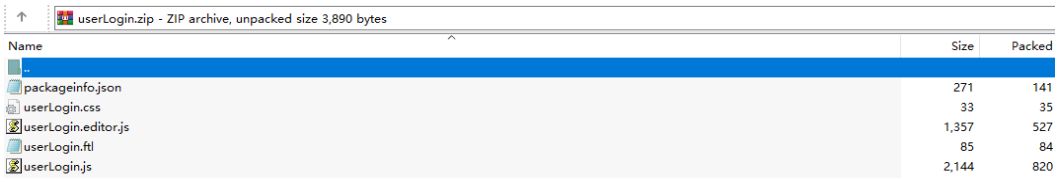
Step 6 Click **Download**, set the widget name to **userLogin**, and click **Save**.

Figure 9-28 Saving a template



Step 7 Check the decompressed widget directories.

Figure 9-29 Directories after decompression



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **userLogin.js**: stores the code of the vue service logic.
- **userLogin.ftl**: stores HTML code.
- **userLogin.css**: stores style code.
- **userLogin.editor.js** and **packageinfo.json**: configuration files.

Step 8 In the decompressed folder, create an **imgs** folder and place a background image of the login page in the folder.

The name of the background image in this example is **imagebg.jpg**.

Step 9 Open the folder in the local editor and change the **propertiesConfig** code in the **userLogin.editor.js** file to the following code to configure the bridge:

```
propertiesConfig: [
    {
        headerTitle: 'Parameters',
        accordion: true,
        accordionState: 'open',
        config: [
            {
                type: 'text',
                name: 'loginAPI',
                label: 'login API',
                value: '',
            },
        ],
    },
    {
        config: [
            {
                type: 'connectorV2',
                name: 'FlowConnector',
                label: 'Flow Connector',
                model: 'ViewModel',
            },
            {
                type: 'connectorV2',
                name: 'common.GetConnector',
                label: 'View API Get Connector',
                model: 'ViewModel',
            },
            {
                type: 'connectorV2',
                name: 'common.PostConnector',
                label: 'View API Post Connector',
                model: 'ViewModel',
            },
            {
                type: 'connectorV2',
                name: 'common.PutConnector',
                label: 'View API Put Connector',
                model: 'ViewModel',
            },
            {
                type: 'connectorV2',
                name: 'common.DeleteConnector',
                label: 'View API Delete Connector',
                model: 'ViewModel',
            },
        ],
    },
],
```

Step 10 Add the following information in bold to the **packageinfo.json** file:

```
{
  "widgetApi": [
    {
```

```
            "name": "userLogin"
        }
    ],
    "widgetDescription": "",
    "authorName": "",
    "localFileBasePath": "",
    "requires": [
        {
            "name": "global_Vue3",
            "version": "3.3.4"
        },
        {
            "name": "global_ElementPlus",
            "version": "2.6.0"
        },
        {
            "name": "global_Vue3I18n",
            "version": "9.10.1"
        },
        {
            "name": "global_Vue3Router",
            "version": "4.3.0"
        }
    ]
}
```

Step 11 Replace the content in the **userLogin.ftl** file with the following code:

```
<style>
    [v-cloak] {
        display: none
    }
</style>
<div v-cloak id="userLogin" class="page-login" v-cloak>
    <div :class="backgroundClass" class="bg-box">
        <!-- <div class="title">
            <span>Equipment management system</span>
        </div> -->
        <div class="login-box">
            <div class="login-title">User Login</div>
            <input name="username" type="text" style="display: none" />
            <input name="password" type="password" style="display: none" />
            <div class="login-form">
                <div v-show="errorDesc" class="error-line">
                    <!-- <img :src="BasePath + 'imgs/btn_errorInfo.png'" /> -->
                    <span class="error-text" v-html="errorDesc"></span>
                </div>
                <div class="login-item mg-top10">
                    <!-- :placeholder="isUserName ? getTransLang('ds.commerce.storefront.web.loginname') :
getTransLang('ds.commerce.storefront.web.loginEmailOrPhone')" -->
                    <input
                        ref="accountInput"
                        v-model="account"
                        placeholder="Enter a username."
                        @keyup.enter="validateBeforeSubmit"
                        autocomplete="off"
                        maxlength="32"
                        type="text"
                    />
                </div>
                <div class="login-item mg-top10">
                    <!-- :placeholder="getTransLang('ds.commerce.storefront.web.password')" -->
                    <input
                        v-model="password"
                        @keyup.enter="validateBeforeSubmit"
                        placeholder="Enter a password."
                        autocomplete="off"
                        maxlength="32"
                        type="password"
                    />
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
            </div>
            <div v-if="needVerify" class="login-item">
                <!-- :placeholder="getTransLang('ds.commerce.storefront.web.verificationcode')" -->
                <input
                    v-model="inputImgCode"
                    @keyup.enter="validateBeforeSubmit"
                    placeholder="Enter the verification code."
                    autocomplete="off"
                    maxlength="10"
                    type="text"
                />
                <div class="verify-code">
                    <img :src="imgCode" @click="getVerifyCode()" />
                </div>
            </div>
            <div class="login-button" @click="validateBeforeSubmit">
                Login
            </div>
        </div>
    </div>
  </div>
</div>
```

Step 12 Replace the content in the **userLogin.js** file with the following code:

```
userLogin = StudioWidgetWrapper.extend({
  init: function () {
    var thisObj = this
    thisObj._super.apply(thisObj, arguments)
    thisObj.render()
    thisObj.initBusi()

    if (typeof Studio != 'undefined' && Studio) {
      Studio.registerEvents(thisObj, 'goHomepage', 'go Homepage', [])
    }
  },

  render: function () {
    var thisObj = this
    let tenantId;
      if (Studio.inReader) {
        tenantId = STUDIO_DATA.catalogProperties["tenant-id"].value
      } else {
        tenantId = magno.pageService.getCatalogProperties()["tenant-id"].value
      }
    HttpUtils.setCookie("tenant-id", tenantId)
    HttpUtils.setCookie("locale","zh_CN");
    var elem = thisObj.getContainer()

    if (elem) {
      var containerDiv = $('.scfClientRenderedContainer', elem)
      if (containerDiv.length) {
        $(containerDiv).empty()
      } else {
        containerDiv = document.createElement('div')
        containerDiv.className = 'scfClientRenderedContainer'
        $(elem).append(containerDiv)
      }
    }

    thisObj.sksBindItemEvent()
    $(window).resize(function () {
      thisObj.sksRefreshEvents()
    })
  },

  initBusi: function () {
    var thisObj = this
    var widgetProperties = thisObj.getProperties()
    var BasePath = thisObj.getWidgetBasePath() // Local path
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
var elem = thisObj.getContainer()
const app  = Vue.createApp({
    data(){
        return {
            BasePath: BasePath,
            imgCode: '',
            account: '',
            password: '',
            inputImgCode: '',
            errorDesc: '',
            needRead: false,
            backgroundClass: 'backgroundClass',
            needVerify: false,
            accountName: '',
            isMobile: false,
        }
    },
    created() {
        this.getVerifyCode()
        this.checkIsLogin()
        this.isMobile = /Android|webOS|iPhone|iPad|BlackBerry/i.test(
            navigator.userAgent
        )
    },
    watch: {
        account(newVal, oldVal) {
            if (newVal !== oldVal) {
                this.password = ''
            }
        },
    },
    methods: {
        checkIsLogin() {
            const userInfo = JSON.parse(
                window.sessionStorage.getItem('userInfo')
            )
            if (userInfo !== null && userInfo.username) {
                setTimeout(() => {
                        thisObj.triggerEvent('goHomepage', {})

                }, 1000)
            }
        },
        shouldShowImgCode() {
            connService(
                thisObj,
                `${ds_baseUrl}/identities/isNeedCaptcha`,
                {
                    username: this.account,
                },
                'common.PostConnector'
            )
                .then((res) => {
                    const { data = [], resp = {} } = res

                    if (resp.code !== '0') {
                        this.$message.error(resp.message)
                    }

                    if (data && data.length) {
                        this.needVerify = data[0].needVerify

                        if (!this.needVerify) {
                            this.confirmLogin()
                        }
                    }
                })
                .catch((e) => {
                    this.$message.error(e.response.resMsg)
```

```
            })
        },
        // Login button
        validateBeforeSubmit() {
            if (!this.password) {
                this.errorDesc = 'Enter a password.'
                this.needVerify = true
                return false
            }
            if (!this.account) {
                this.errorDesc = 'Enter a username.'
                this.needVerify = true
                return false
            }

            if (!this.inputImgCode && this.needVerify) {
                this.errorDesc = 'Incorrect verification code.'
                return false
            }

            if (this.needVerify && this.inputImgCode) {
                this.confirmLogin()
                return
            }
            this.confirmLogin()
        },

        // Debug the login API.
        confirmLogin() {
            var request = {
                username: this.account,
                password: this.password,
                captcha: this.inputImgCode,
            }
            //
            this.callFlowConn1(
    widgetProperties.loginAPI,
                request,
                this.callLogin,
                this.loginFail,
                'post'
            )
        },

        // Function of login API success
        callLogin(response) {
            if (response) {
                this.errorDesc = ''
                let userInfo = {
                    username: response.data[0].username,
                    userId: response.data[0].userId,
                    profile: response.data[0].profile,
                }
                HttpUtils.setCookie('isLogged', true)
                window.sessionStorage.setItem(
                    'userInfo',
                    JSON.stringify(userInfo)
                )
                thisObj.triggerEvent('goHomepage', {})

            }
        },
        // Login failed resMsg
        loginFail(data){
            this.errorDesc = data.response.resMsg
            this.getVerifyCode()
            this.needVerify = true
        },
        // Obtain the verification code.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
            getVerifyCode() {
                this.imgCode =
                    '/u-route/baas/sys/v1.0/verificationcode?type=login&t=' +
                    Date.parse(new Date())
            },
            // Encapsulate the flow to call the background
            callFlowConn1: function (service, param, callbackFunc,callbackfail, method) {
                var thisView = this
                let mMethod
                switch (method) {
                    case 'get':
                        mMethod = 'common.GetConnector'
                        break
                    case 'post':
                        mMethod = 'common.PostConnector'
                        break
                    case 'put':
                        mMethod = 'common.PutConnector'
                        break
                    default:
                        mMethod = 'common.FlowConnector'
                        break
                }

                var connector = thisObj.getConnectorInstanceByName(mMethod)
                if (connector) {
                    connector.setInputParams({
                        service: service,
                        needSchema: 'data',
                        async: false,
                    })
                    connector
                        .query(param) // Call the API with param as the input parameter
                        .done(function (response) {
                            if (response.resp && response.resp.code) {
                                callbackFunc.call(thisView, response)
                            }
                        })
                        .fail(function (response) {
                            // The API fails to be executed.
                            callbackfail.call(thisView, response)
                        })
                }
            },
        },
    })
    app.use(ElementPlus);
    thisObj.vm = app.mount($("#userLogin", elem)[0]);

  },
})
```

Step 13 Replace the content in the **userLogin.css** file with the following code.

Note that the value of **background** must be the names of the folder and image created in **Step 8**.

```
.page-login {
  /* Browsers that use the WebKit kernel */
  /* Firefox version 4-18 */
  /* Firefox version 19+ */
}
.page-login * {
  box-sizing: border-box;
}
.page-login .flex {
  display: flex;
}
.page-login .login_tip {
  font-size: 12px;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  color: #167aeb;
}
.page-login .bg-box {
  position: relative;
  display: flex;
  justify-content: center;
  width: 100%;
  height: 100%;
  background: url('imgs/imagebg.jpg') no-repeat 50%;
  background-size: cover;
}
.page-login .bg-box > img {
  width: 100%;
}
@media (min-width: 767px) {
  .page-login .bg-box .login-box {
    position: absolute;
    right: 18%;
    top: 26%;
    padding: 30px 30px 20px 30px;
    width: 380px;
    background: #fff;
    border-radius: 10px;
    box-shadow: 0 8px 16px 0 rgba(0, 0, 0, 0.1);
  }
  .page-login .bg-box .login-box .login-title {
    font-size: 24px;
    color: #333;
    text-align: center;
  }
  .page-login .bg-box .login-box .login-form {
    padding-top: 20px;
  }
  .page-login .bg-box .login-box .login-form .error-line {
    display: flex;
    align-items: center;
    background: #ffebeb;
    color: #e4393c;
    border: 1px solid #faccc6;
    padding: 3px 10px 3px 10px;
    height: auto;
    text-align: left;
    font-size: 12px;
  }
  .page-login .bg-box .login-box .login-form .error-line .error-text {
    padding-left: 5px;
  }
  .page-login .bg-box .login-box .login-form .error-line .error-text .change_tab {
    color: #167aeb;
    cursor: pointer;
  }
  .page-login .bg-box .login-box .login-form .login-item {
    display: flex;
    align-items: center;
    margin-top: 20px;
  }
  .page-login .bg-box .login-box .login-form .login-item .el-input__inner {
    height: 36px;
  }
  .page-login .bg-box .login-box .login-form .login-item > input {
    flex: 1;
    padding: 5px 10px;
    height: 36px;
    border-radius: 3px;
    border: 1px solid #ddd;
  }
  .page-login .bg-box .login-box .login-form .login-item .verify-code {
    height: 36px;
    margin-left: 5px;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  }
  .page-login .bg-box .login-box .login-form .login-item .verify-code img {
    height: 100%;
  }
  .page-login .bg-box .login-box .login-form .login-item .el-checkbox {
    margin-right: 10px;
  }
  .page-login .bg-box .login-box .login-form .login-item .read-text {
    font-size: 12px;
    color: #333;
  }
  .page-login .bg-box .login-box .login-form .login-item .read-text .clickable {
    color: #167aeb;
    cursor: pointer;
  }
  .page-login .bg-box .login-box .login-form .login-item .read-text .clickable:hover {
    text-decoration: underline;
  }
  .page-login .bg-box .login-box .login-form .mg-top10 {
    margin-top: 10px;
  }
  .page-login .bg-box .login-box .login-button {
    margin-top: 20px;
    height: 40px;
    line-height: 40px;
    border-radius: 3px;
    background: #3d88ff;
    text-align: center;
    color: #fff;
    font-size: 14px;
    cursor: pointer;
  }
  .page-login .bg-box .login-box .login-button:hover {
    opacity: 0.8;
  }
  .page-login .bg-box .login-box .divide-line {
    margin-top: 20px;
    height: 1px;
    background: #eee;
    opacity: 0.8;
  }
  .page-login .bg-box .login-box .login-bottom {
    margin-top: 20px;
    justify-content: center;
    align-items: center;
  }
  .page-login .bg-box .login-box .login-bottom .bottom-text {
    font-size: 12px;
    color: #999;
  }
  .page-login .bg-box .login-box .login-bottom .type-item {
    margin-left: 10px;
    width: 30px;
    height: 23px;
    cursor: pointer;
  }
  .page-login .bg-box .login-box .login-bottom .type-item img {
    width: 100%;
  }
}
@media (max-width: 767px) {
  .page-login .bg-box .login-box {
    position: absolute;
    right: auto;
    top: 26%;
    padding: 30px 30px 20px 30px;
    width: 380px;
    background: #fff;
    border-radius: 10px;
```

```
  box-shadow: 0 8px 16px 0 rgba(0, 0, 0, 0.1);
}
.page-login .bg-box .login-box .login-title {
  font-size: 24px;
  color: #333;
  text-align: center;
}
.page-login .bg-box .login-box .login-form {
  padding-top: 20px;
}
.page-login .bg-box .login-box .login-form .error-line {
  display: flex;
  align-items: center;
  background: #ffebeb;
  color: #e4393c;
  border: 1px solid #faccc6;
  padding: 3px 10px 3px 10px;
  height: auto;
  text-align: left;
  font-size: 12px;
}
.page-login .bg-box .login-box .login-form .error-line .error-text {
  padding-left: 5px;
}
.page-login .bg-box .login-box .login-form .error-line .error-text .change_tab {
  color: #167aeb;
  cursor: pointer;
}
.page-login .bg-box .login-box .login-form .login-item {
  display: flex;
  align-items: center;
  margin-top: 20px;
}
.page-login .bg-box .login-box .login-form .login-item .el-input__inner {
  height: 36px;
}
.page-login .bg-box .login-box .login-form .login-item > input {
  flex: 1;
  padding: 5px 10px;
  height: 36px;
  border-radius: 3px;
  border: 1px solid #ddd;
}
.page-login .bg-box .login-box .login-form .login-item .verify-code {
  height: 36px;
  margin-left: 5px;
}
.page-login .bg-box .login-box .login-form .login-item .verify-code img {
  height: 100%;
}
.page-login .bg-box .login-box .login-form .login-item .el-checkbox {
  margin-right: 10px;
}
.page-login .bg-box .login-box .login-form .login-item .read-text {
  font-size: 12px;
  color: #333;
}
.page-login .bg-box .login-box .login-form .login-item .read-text .clickable {
  color: #167aeb;
  cursor: pointer;
}
.page-login .bg-box .login-box .login-form .login-item .read-text .clickable:hover {
  text-decoration: underline;
}
.page-login .bg-box .login-box .login-form .mg-top10 {
  margin-top: 10px;
}
.page-login .bg-box .login-box .login-button {
  margin-top: 20px;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    height: 40px;
    line-height: 40px;
    border-radius: 3px;
    background: #3d88ff;
    text-align: center;
    color: #fff;
    font-size: 14px;
    cursor: pointer;
  }
  .page-login .bg-box .login-box .login-button:hover {
    opacity: 0.8;
  }
  .page-login .bg-box .login-box .divide-line {
    margin-top: 20px;
    height: 1px;
    background: #eee;
    opacity: 0.8;
  }
  .page-login .bg-box .login-box .login-bottom {
    margin-top: 20px;
    justify-content: center;
    align-items: center;
  }
  .page-login .bg-box .login-box .login-bottom .bottom-text {
    font-size: 12px;
    color: #999;
  }
  .page-login .bg-box .login-box .login-bottom .type-item {
    margin-left: 10px;
    width: 30px;
    height: 23px;
    cursor: pointer;
  }
  .page-login .bg-box .login-box .login-bottom .type-item img {
    width: 100%;
  }
}
.page-login ::-webkit-input-placeholder {
  font-size: 12px;
  color: #aaa;
}
.page-login :-moz-placeholder {
  font-size: 12px;
  color: #aaa;
}
.page-login ::-moz-placeholder {
  font-size: 12px;
  color: #aaa;
}
.page-login :-ms-input-placeholder {
  font-size: 12px;
  color: #aaa;
}
.page-login .logining-text {
  margin-top: 120px;
  text-align: center;
  font-size: 30px;
  color: #333;
}
```

Step 14 Compress the modified configuration file and customized file into a .zip package, and name the package as **userLogin.zip**.

----End

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 2: Upload the Custom Login Widget

After a custom widget is developed, you can upload it to the Huawei Cloud Astro Zero widget library for advanced pages.

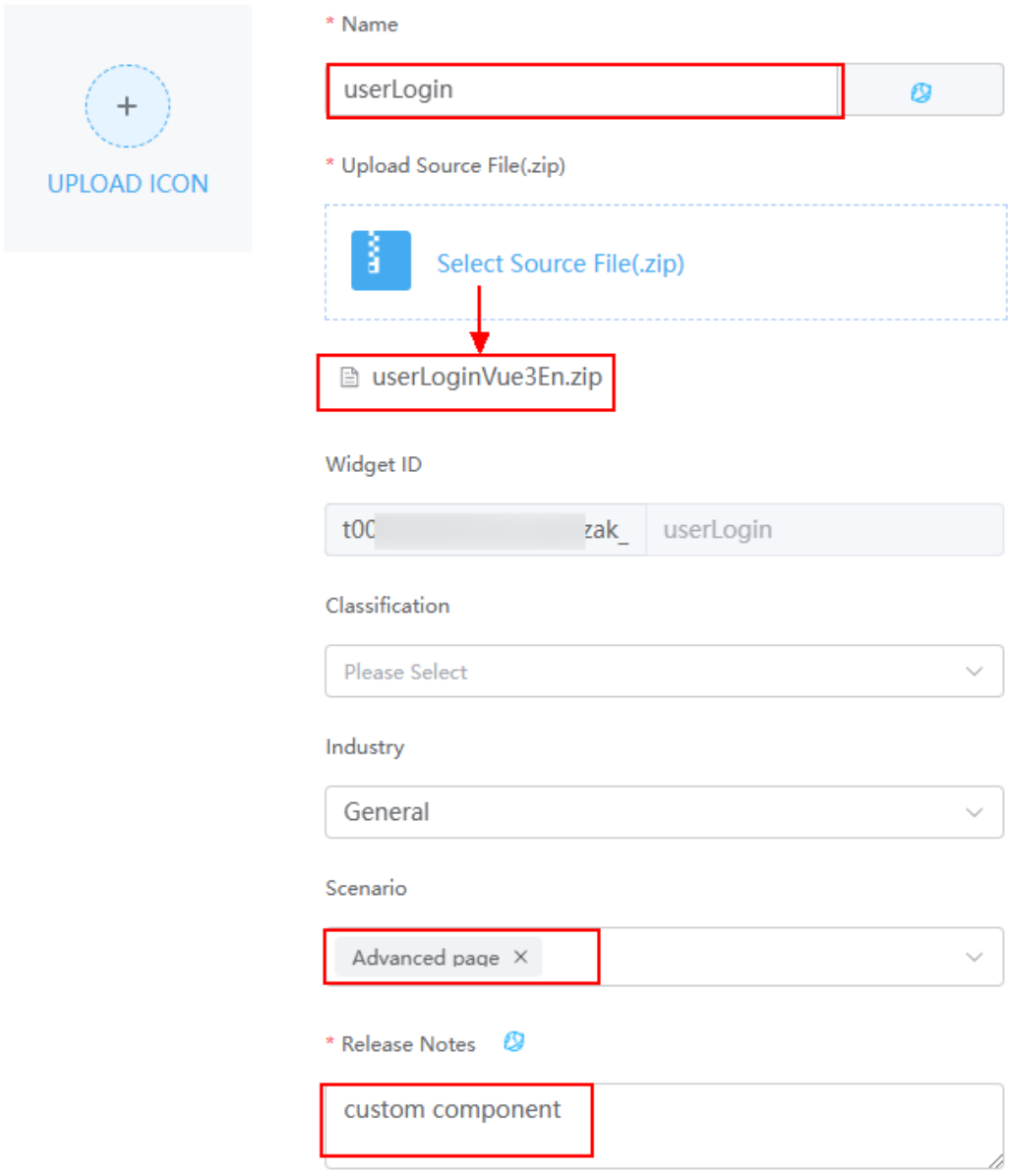
Step 1 On the Huawei Cloud Astro Zero environment configuration page, choose **Maintenance** from the main menu.

Step 2 In the navigation pane, choose **Global Elements** > **Page Assets** > **Widgets**.

Step 3 Click **Submit New Widget**.

Step 4 Click the upload button to upload the customized widget package **userLogin.zip**.

Figure 9-30 Uploading a custom widget



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 9-7 Parameters for uploading a widget

Parameter	Description	Example
Name	Widget name. Same as the widget package name by default.	userLogin
Upload Source File(.zip)	Source file package of a widget.	Select the customized widget package developed in **Step 1: Develop a Custom Widget**.
Scenario	In what kind of page development scenarios can the widget be used.	Advanced pages
Release Notes	Widget description in different languages. The information configured here will be displayed on the overview tab page of the widget details page.	Custom widgets

Step 5 After setting the parameters, click **Submit**.

----End

Step 3: Create an Advanced Page

The portal user login page is an advanced page. You can reference the uploaded login widget and set related parameters to implement the login function.

To ensure better demonstration, a destination page called **Home** has been created in advance. It is the page that you will be redirected to after your successful login. There are some chart widgets on the **Home** page, but they do not have data yet. They are just examples. In real development scenario, you can customize your own destination page and its logic.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 9-31 Home page

Step 1 On the Huawei Cloud Astro Zero console, click **Access Homepage** to go to the application development page.

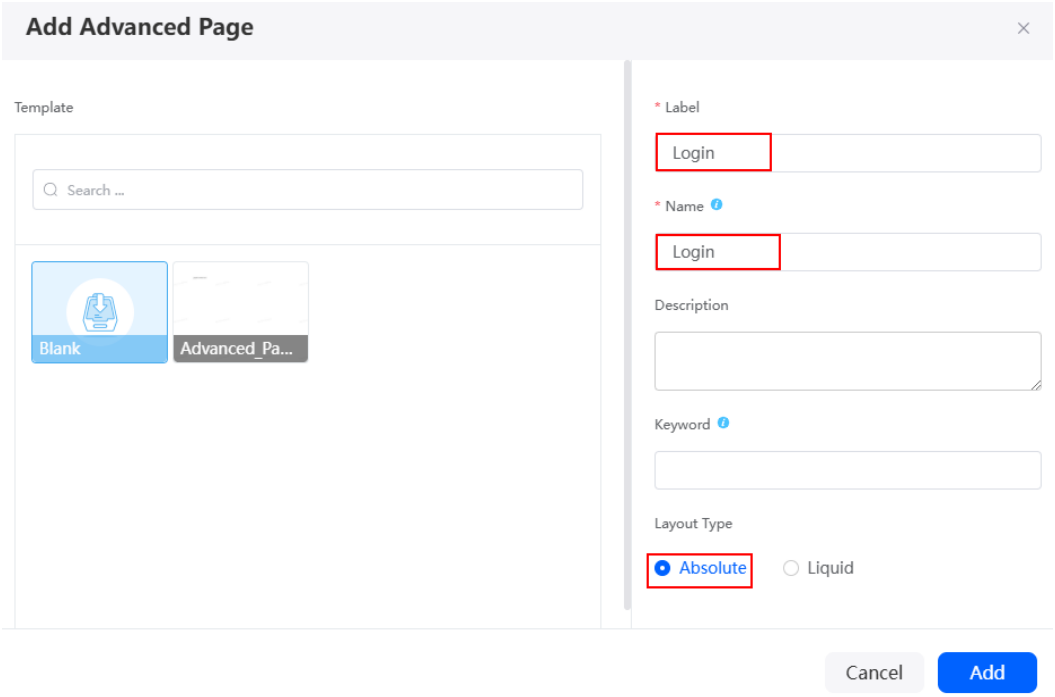
Step 2 On the **Home** > **All Apps** page, click **Edit** next to **A** application to access the application designer.

Step 3 In the navigation pane on the left, choose **Page**.

Step 4 Click the plus sign (**+**) next to **Advanced Page**, set the label and name to **Login**, set the layout type to **Absolute**, and click **Add**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 9-32 Adding an advanced page

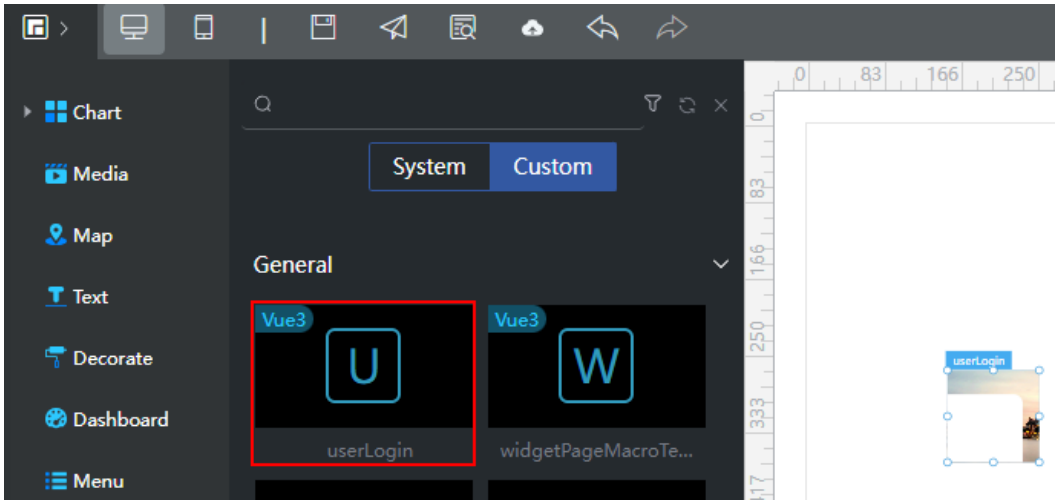


----End

Step 4: Drag Widgets to Develop the Login Page

Step 1 On the advanced page created in **Step 3: Create an Advanced Page**, click  in the upper left corner and select **All**. On the **Custom** tab page, drag the **userLogin** widget to the editing area.

Figure 9-33 Dragging a widget to the editing area

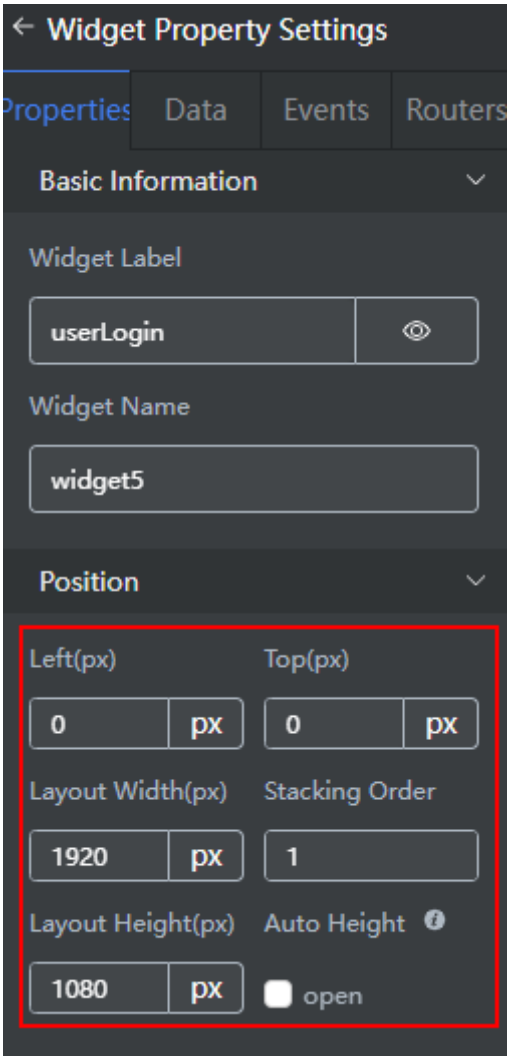


Step 2 Set the location properties of the **userLogin** widget.

1. Check the property configuration panel of the widget displayed on the right.
2. In the **Position** area, set **Left(px)** and **Top(px)** to **0**, **Layout Width(px)** to **1920**, and **Layout Height(px)** to **1080**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

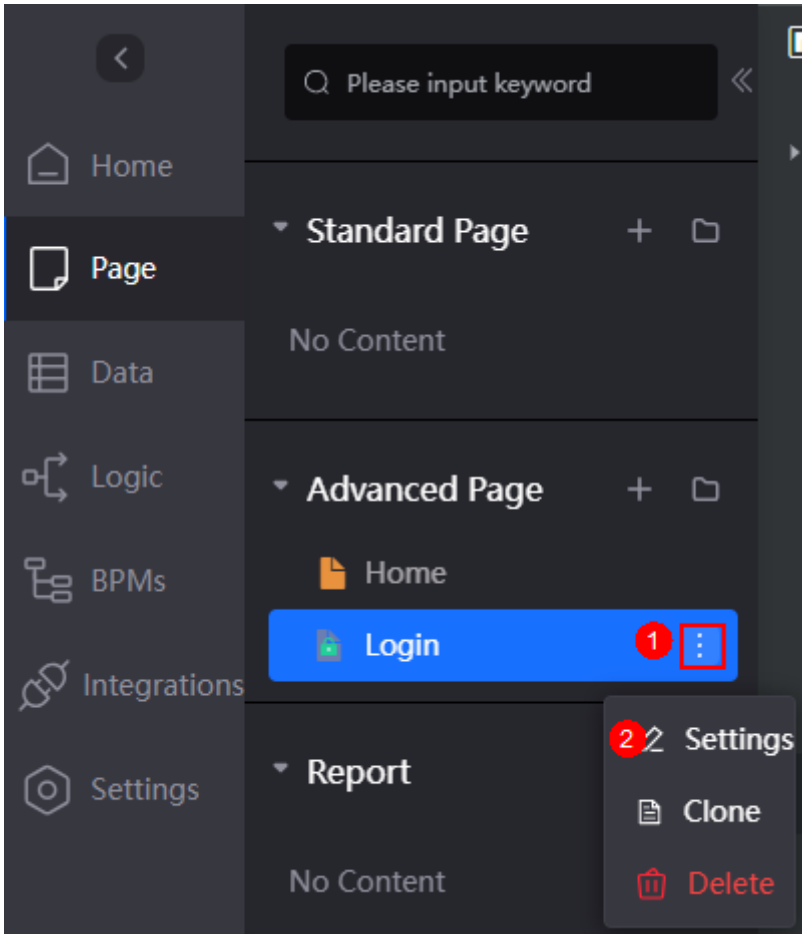
Figure 9-34 Setting the widget position



3. Click in the upper part of the page to save the configuration.
4. Choose **Page > Advanced Page**, move the cursor to **Login**, click , and select **Settings**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

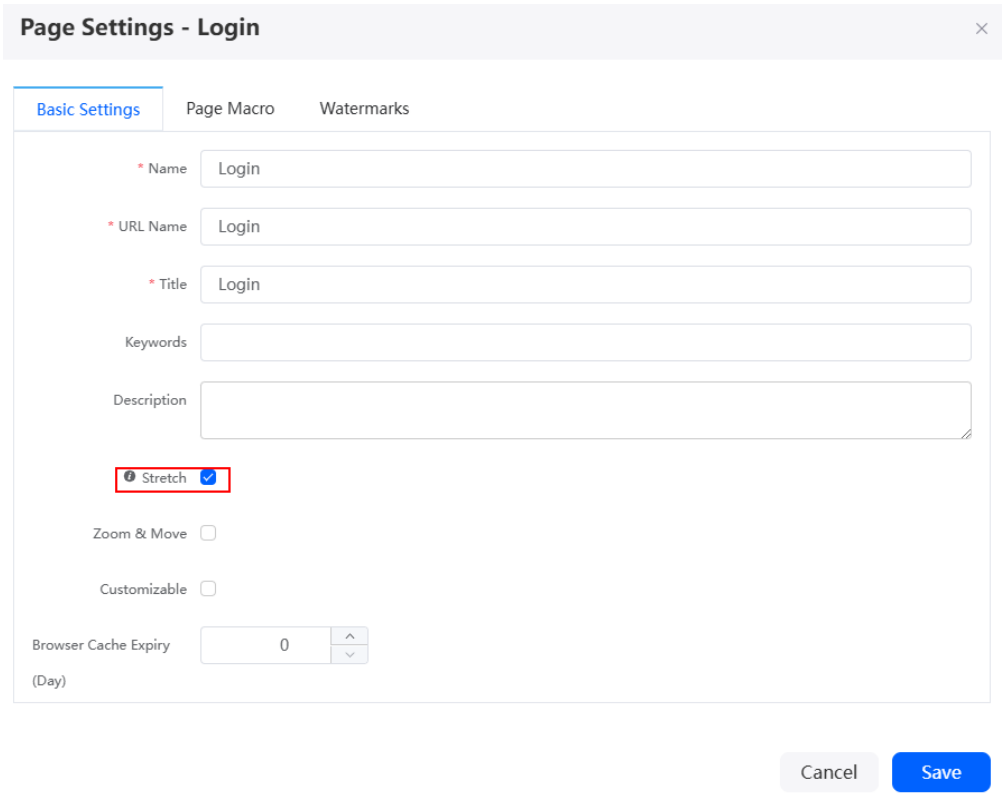
Figure 9-35 Opening the Settings page



5. On the displayed page, select **Stretch** and click **Save**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

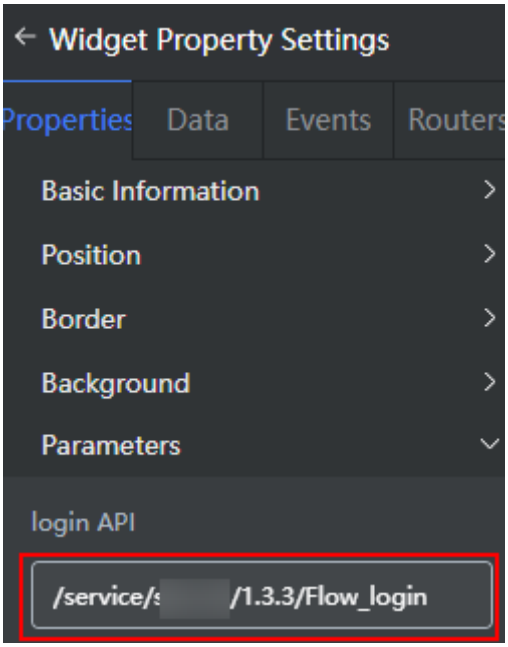
Figure 9-36 Setting the page stretching



Step 3 Set the bridge of the widget.

1. Select the **userLogin** widget, choose **Properties** > **Parameters**, and set **login API** to / *Namespace*__**A/1.0.0/Flow_login**.

Figure 9-37 Adding the login URL

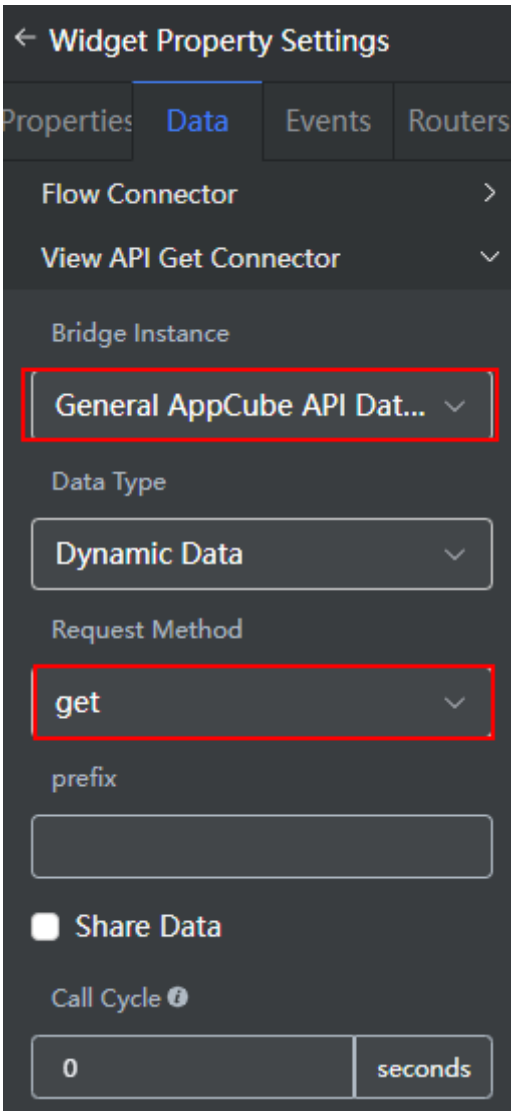


 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

login API indicates the latter part of the URL of the login API. Set *Namespace* as needed. **A** indicates the application name. The login API is created in **Background Login**.

2. On the **Data** tab page, click **View API Get Connector**. Set **Bridge Instance** to **Bridge Instance AstroZero API Data Bridge**, **Data Type** to **Dynamic Data**, and **Request Method** to **get**, as shown in the following figure.

Figure 9-38 Setting a bridge



3. Set the bridge instances selected in the following figure by referring to the previous step.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 9-39 Setting other bridges

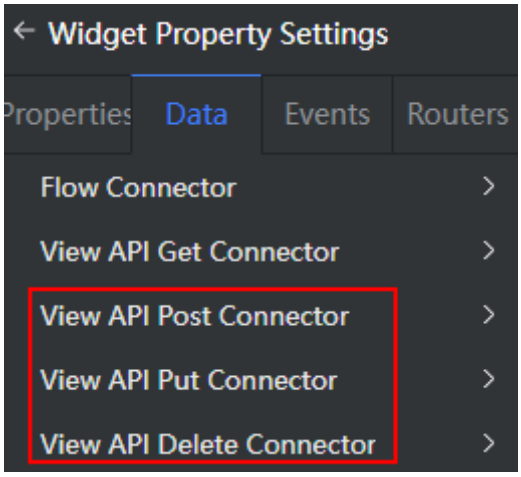


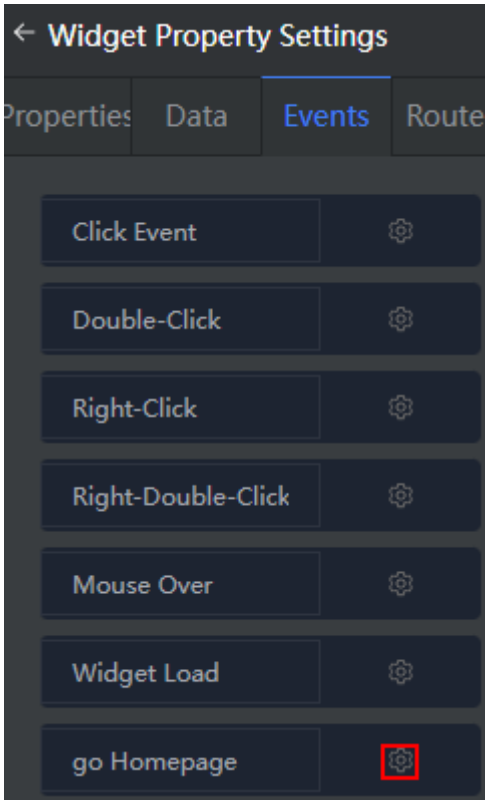
Table 9-8 Bridge instance configuration

Data Name	Bridge Instance	Data Type	Request Method
View API Post Connector	Bridge Instance AstroZero API Data Bridge	Dynamic Data	Post
View API Put Connector	Bridge Instance AstroZero API Data Bridge	Dynamic Data ata	Put
View API Delete Connector	Bridge Instance AstroZero API Data Bridge	Dynamic Data	Delete

Step 4 Set events for the **userLogin** widget to associate the widget with other pages.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 9-40 Setting events

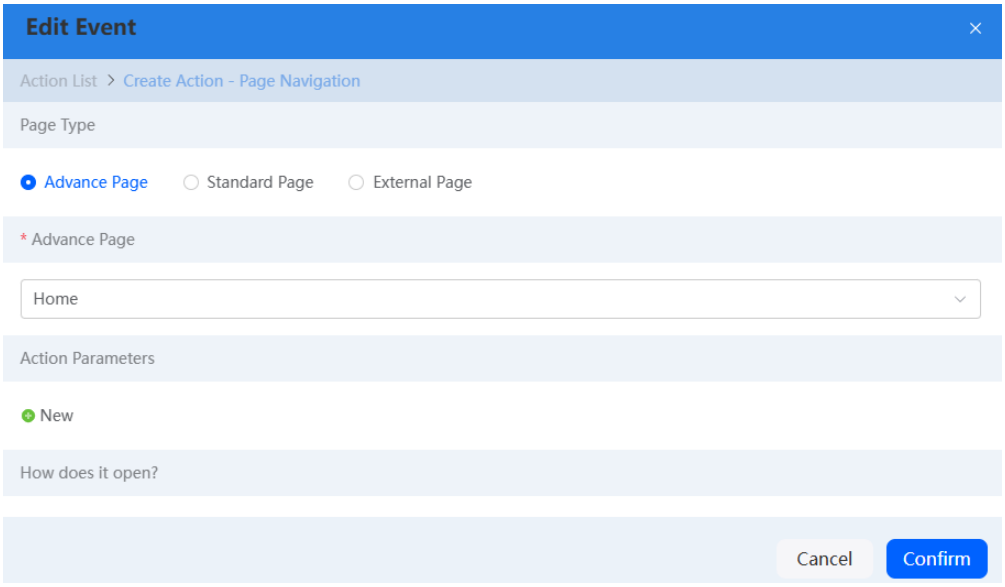


1. Click the gear icon next to **go Homepage**. The event editing page is displayed.
2. Click **Create Action** and choose **Default** > **Page Navigation**.
3. Select the page to be redirected to and click **OK**.

   Here we select the advanced page **Home** created during previous steps. In real development scenario, you can customize your own destination page and its logic.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 9-41 Editing page redirection

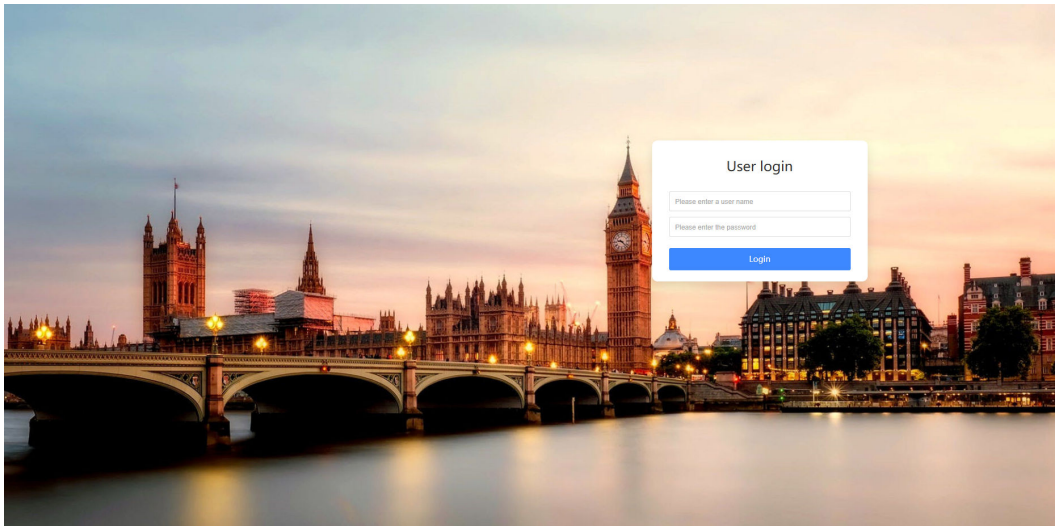


Step 5 Click on the top of the **Login** page to save the page, then click to release the page.

Step 6 After the page is successfully released, click to preview the login page.

On the login page, after you enter the account and password of the portal use, and click **Log In**, the system then calls the user login flow.

Figure 9-42 Login preview page



Step 7 On the preview page, enter the account and password of the portal user who has required permissions and click **Log In**. If the **Home** page is displayed, the portal user has logged in successfully.

----End

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

10 Intelligence

10.1 Using the AI Assistant Widget to Build a Knowledge Base

Application Scenarios

If you have a large number of documents that are hard to use and query efficiency is low, you can use the AI assistant widget to call the foundation model API to get answers. Additionally, RAG technology works with the knowledge base to provide a more intelligent and natural dialog experience, improving the accuracy and relevance of information retrieval.

Constraints

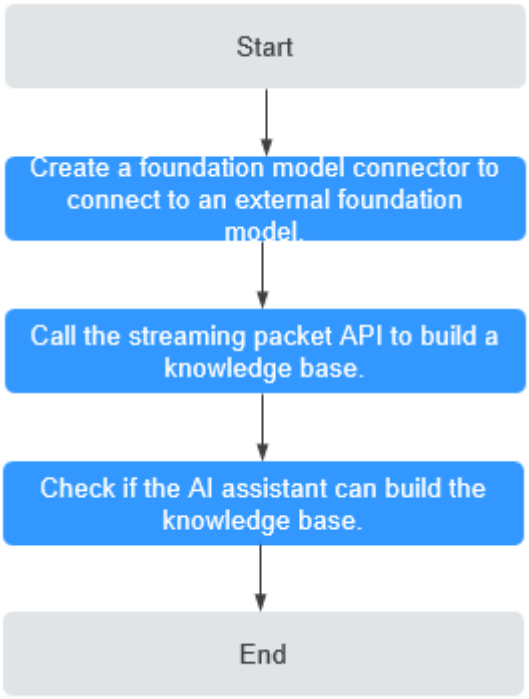
- To enable the function of building a knowledge base, submit a service ticket.
- The knowledge base directory structure should be: knowledge base > folder > file. You can create multiple knowledge bases, each containing multiple folders with files of various types. For **professional edition** instances, **each tenant** can create up to **40** folders. **Free** and **standard** editions support only **one** folder. Single files must be under 6 MB, and folders cannot exceed 50 MB in total. Supported file formats are TXT, DOCX, PDF, MD, and CSV.

Procedure

Figure 10-1 shows the process of building a knowledge base using the AI assistant widget.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 10-1 Building a knowledge base using the AI assistant widget



Step 1: Create a Foundation Model Connector to Connect to an External Foundation Model

Create a connector to connect to the foundation model API.

Step 1 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After you have an instance, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace.
4. In the displayed dialog box, choose **Standard Applications** and click **Confirm**.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 10-2 Creating a blank application

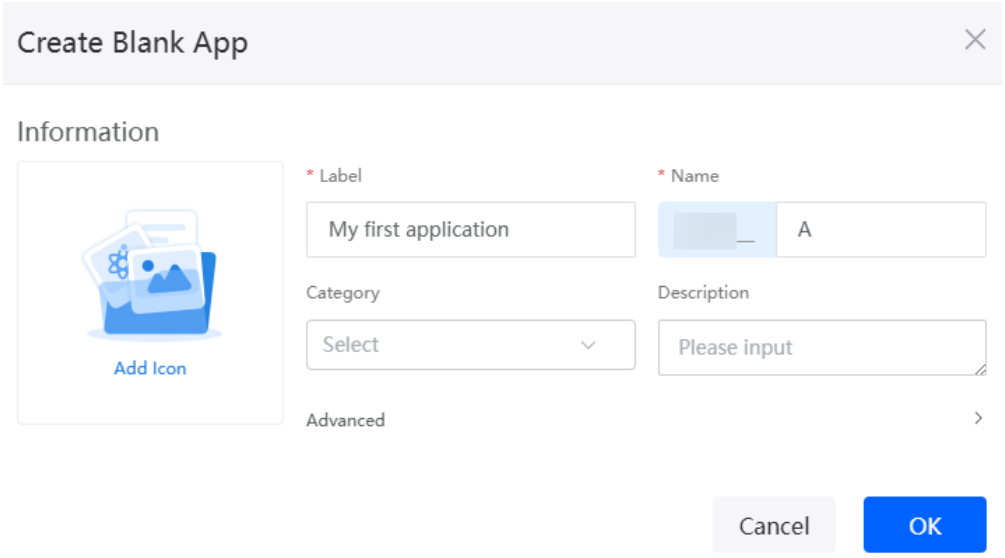


Table 10-1 Parameters for creating a blank application

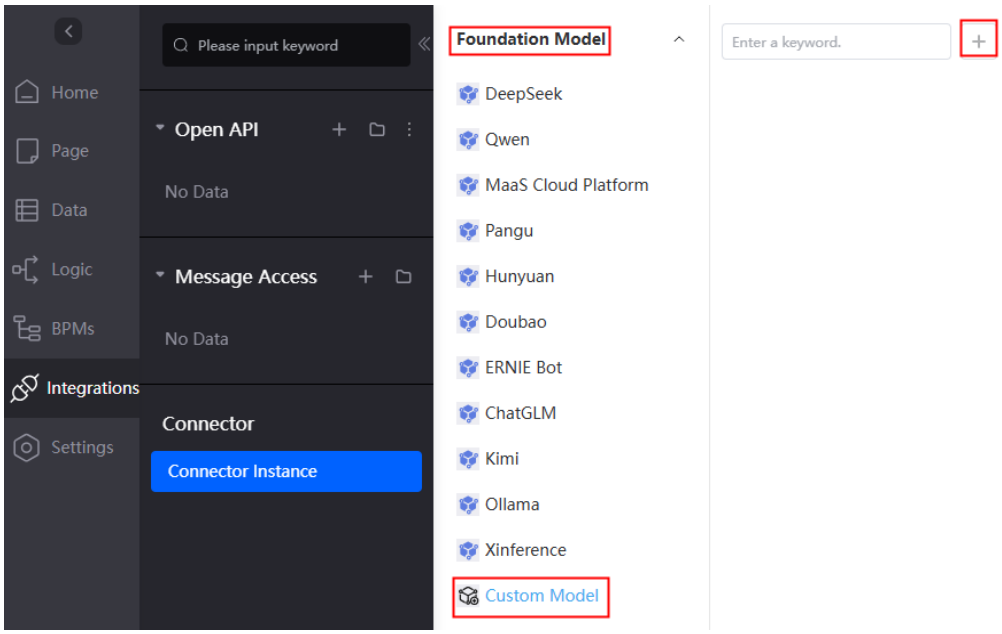
Parameter	Description	Example
Label	The label for the new application; Max. 80 characters. A label uniquely identifies an application in the system and cannot be modified after creation.	My first application
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and can contain only letters, digits, and underscores (_). It cannot end with an underscore (_).	A

Step 2 Create a foundation model connector.

1. In the application designer, choose **Integrations** > **Connector** > **Connector Instance**.
2. Choose **Foundation Model** > **Custom Model** and click **+**. The page for creating a custom foundation model connector is displayed.

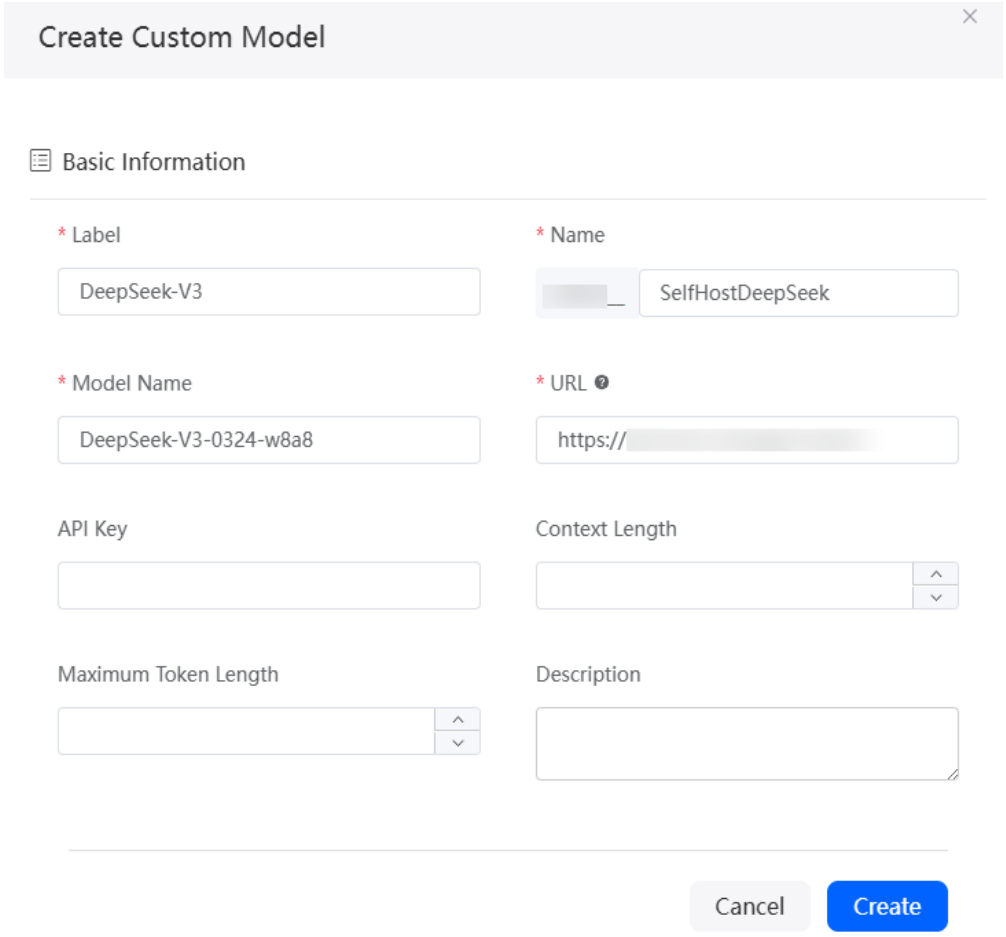
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 10-3 Creating a custom foundation model connector



3. Enter and save the configuration information of the foundation model connector.

Figure 10-4 Setting foundation model connector information



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 10-2 Parameters for configuring the custom foundation model connector

Parameter	Description	Example
Label	Custom foundation model connector label, which can be modified after being created. Value: 1–64 characters.	DeepSeek-V3
Name	Custom connector name, which uniquely identifies the connector in the system and cannot be changed after the connector is created. Naming rules: – Max. 64 characters, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and can contain only letters, digits, and underscores (_). It cannot end with an underscore (_).	SelfHostDeepSeek
Model Name	Model ID of the specific foundation model. When the connector is called, this field is passed to the **model** field in the foundation model API.	DeepSeek-V3-0324-w8a8
URL	URL for accessing the foundation model API, provided by the model provider.	https://*****

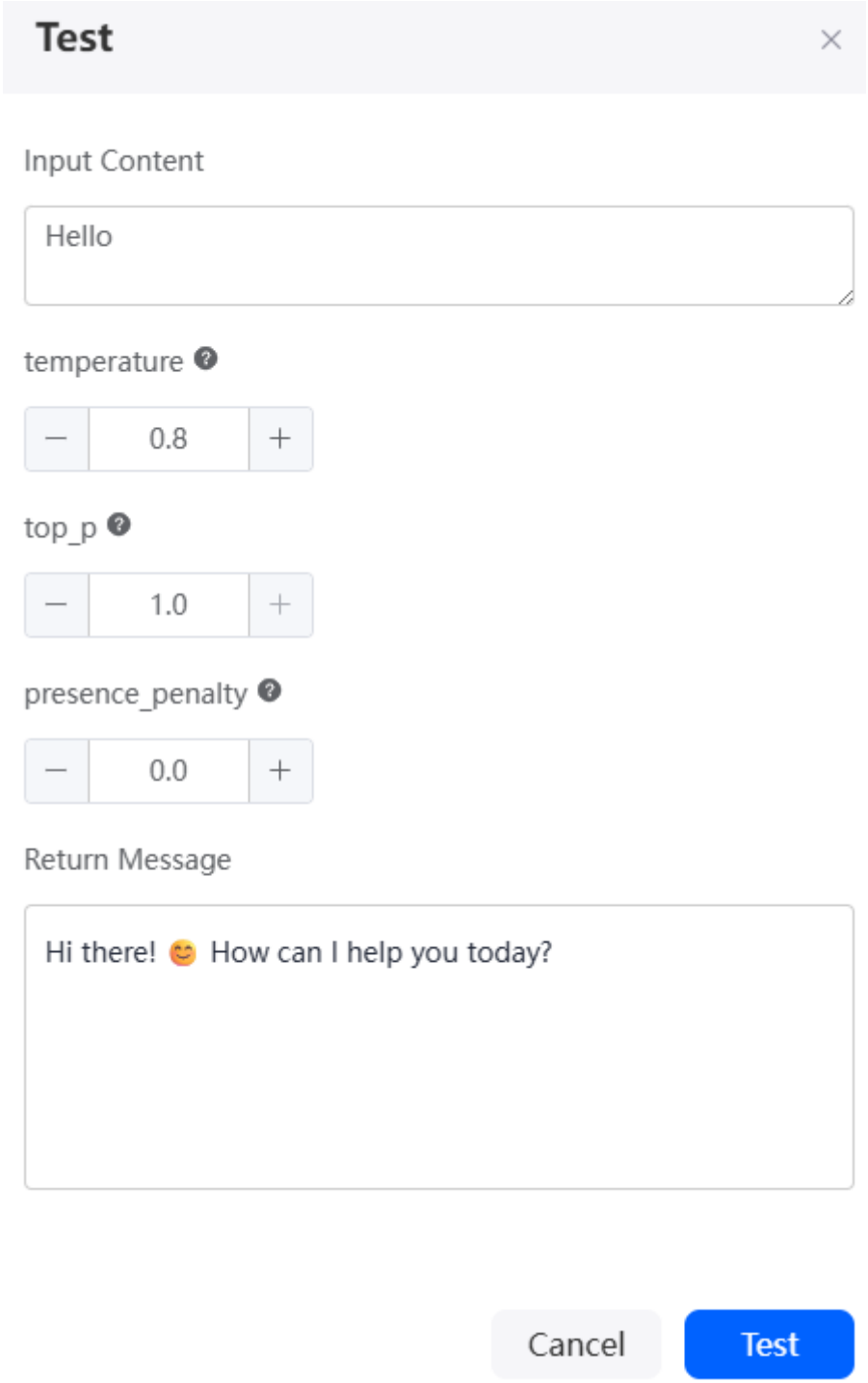
Step 3 Test the connectivity of the foundation model connector.

1. On the custom foundation model details page, click **Test**. On the displayed dialog box, click **Test**. The foundation model connector test page is displayed.
2. In the **Input Content** text box, enter the dialog content (for example, "Hello"), and click **Test**. Check whether the returned message is correct.

   If the returned message contains the streaming response packet, the foundation model connector is configured successfully. Otherwise, check whether the foundation model connector information is correct.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 10-5 Foundation model connector test page



----End

Step 2: Call the Streaming Packet API to Build a Knowledge Base

Create a standard page, add an AI assistant widget, and use a script to call the streaming API to build a knowledge base.

Step 1 In the navigation pane, choose **Page**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 2 Click + to create a blank standard page.

Figure 10-6 Setting basic information about the standard page

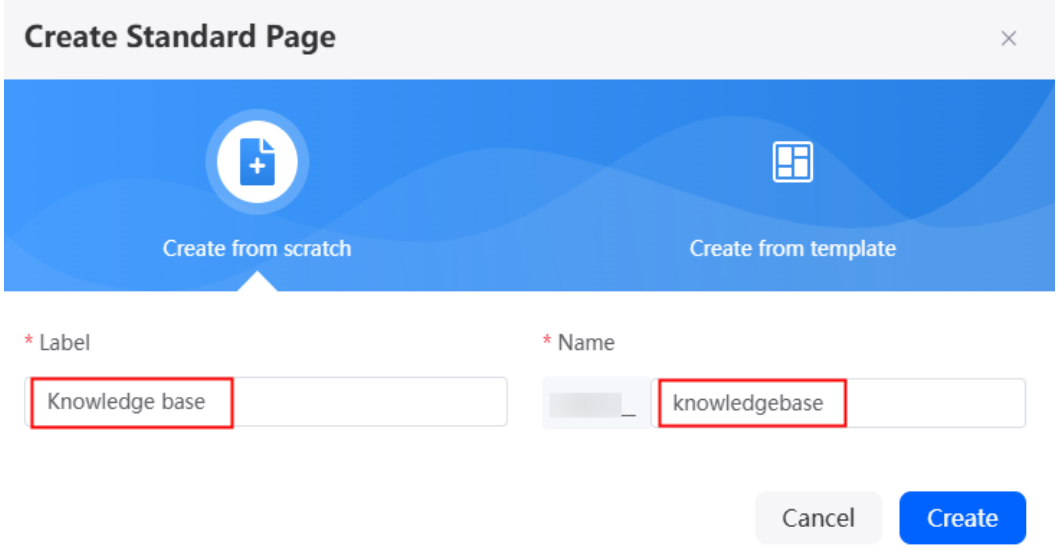


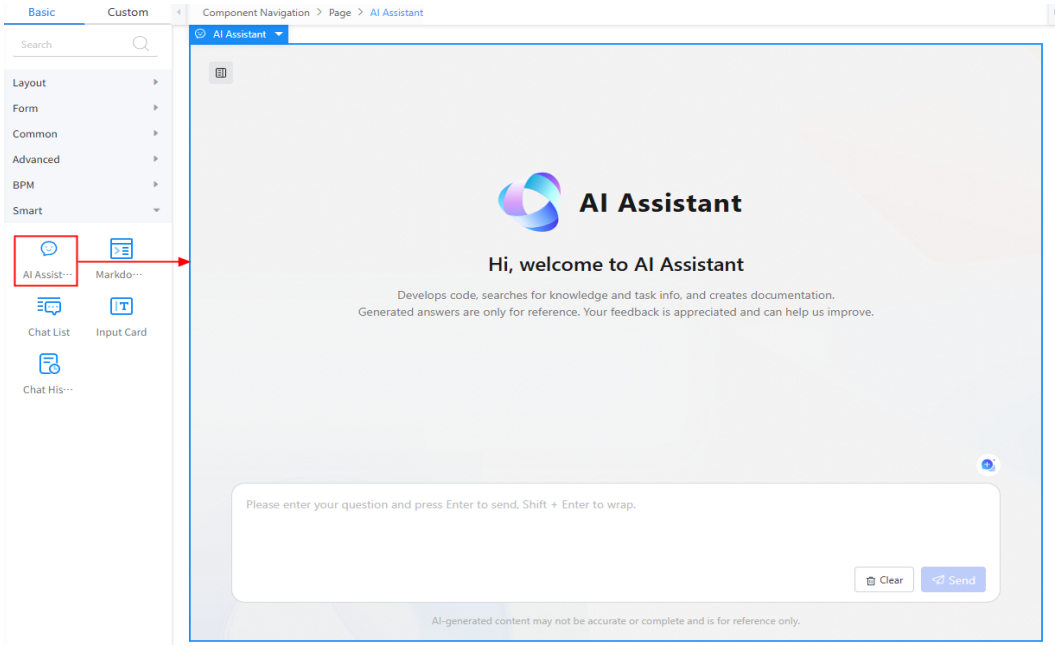
Table 10-3 Parameters for creating a standard page

Parameter	Description	Example
Label	Label of the standard page, which is displayed on the page and can be modified after being created. Value: 1–64 characters.	Knowledge base
Name	Name of the standard page. The name is the unique identifier of the standard page in the system and cannot be changed after being created. Naming rules: • Max. 64 characters, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. • Start with a letter and can contain only letters, digits, and underscores (_). It cannot end with an underscore (_).	knowledgebase

Step 3 Drag the **AI Assistant** widget from **Basic** > **Smart** to the canvas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

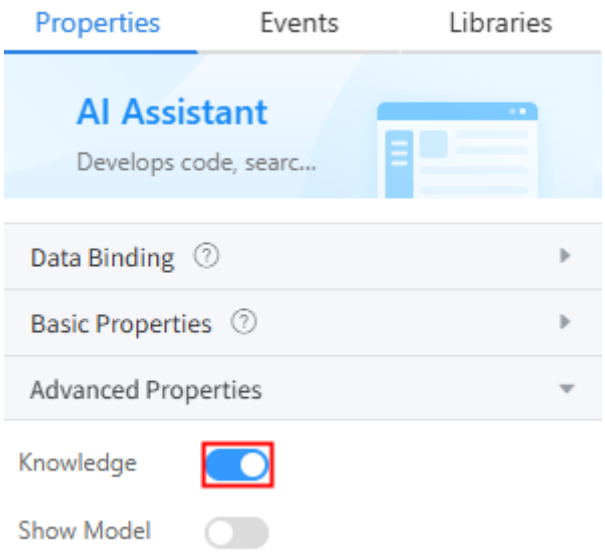
Figure 10-7 Adding an AI assistant widget



Step 4 Enable the knowledge base and bind the big data model to the AI assistant widget.

1. Click the AI assistant widget and enable the knowledge base in **Properties** > **Advanced Properties**.

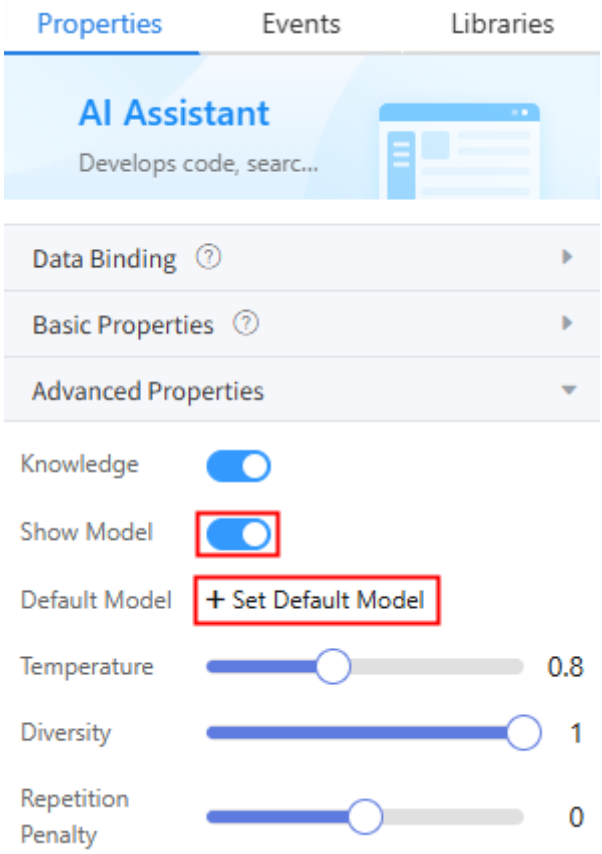
   Figure 10-8 Enabling the knowledge base

2. Enable **Show Model** and click **Set Default Model**. Select the connector created in **Step 1: Create a Foundation Model Connector to Connect to an External Foundation Model** to bind the model to the widget.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 10-9 Binding a model to the widget



----End

Step 3: Verify if the AI Assistant Widget Can Build a Knowledge Base

Step 1 On the created standard page, click  . The preview page is displayed. You can view the button for adding a knowledge base and the bound big data model.

Step 2 Click **Add Knowledge Base**. On the displayed page, click **Manage**.

Groups: A knowledge base that can be shared among group members. When you create a group knowledge base, if you set it to be visible only to a specific department, only users in that department can view and use the knowledge base.

Mine: An individual knowledge base is exclusive to a single user. Other users cannot access or use it.

Step 3 Click **New**, set the knowledge base name and visibility scope, and click **OK**. The knowledge base page is displayed, on which you can view the created knowledge base.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 10-10 Creating a knowledge base

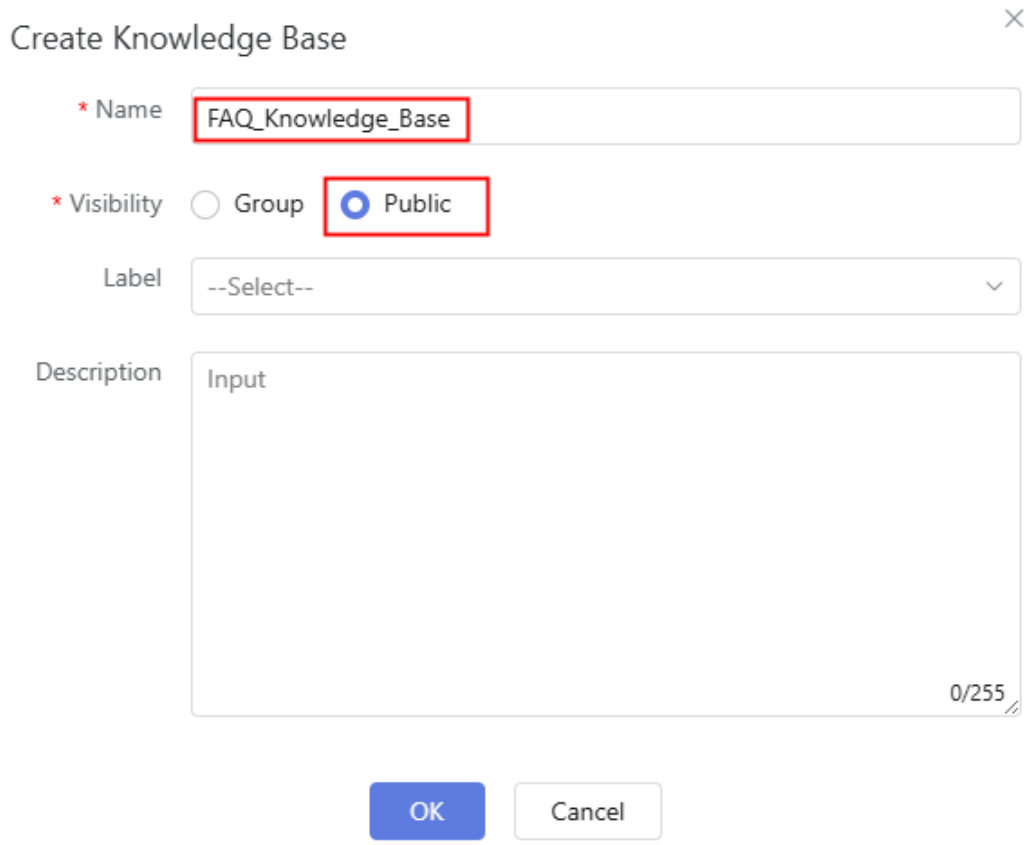


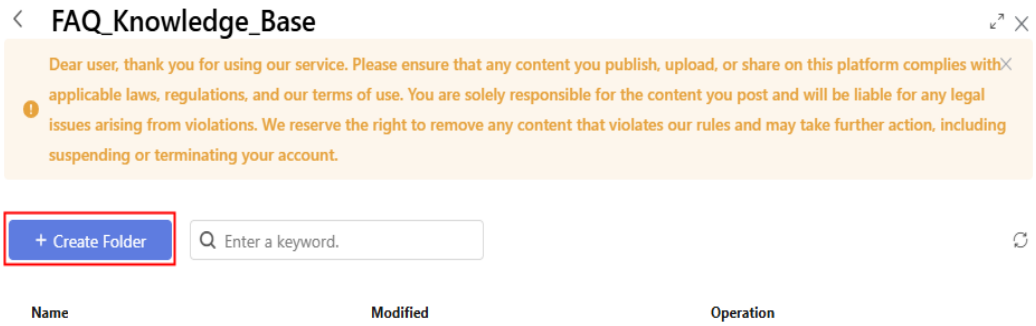
Table 10-4 Parameters for creating a knowledge base

Parameter	Description	Example
Name	Name of the knowledge base to be created.	FAQ_Knowledge_Base
Visibility	Select the visibility scope of the knowledge base. The options are **Group** and **Public**. • **Group**: Only users in the department can view and use the knowledge base. • **Public**: All users of the tenant can view and use the knowledge base.	Public

Step 4 In the knowledge base list, click the knowledge base created in **Step 3**. On the knowledge base page that is displayed, click **Create Folder**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 10-11 Clicking the button to create a folder



Step 5 Set the folder name, and select the knowledge base embedding model, splitter, and chunk length as required.

Figure 10-12 Creating a folder

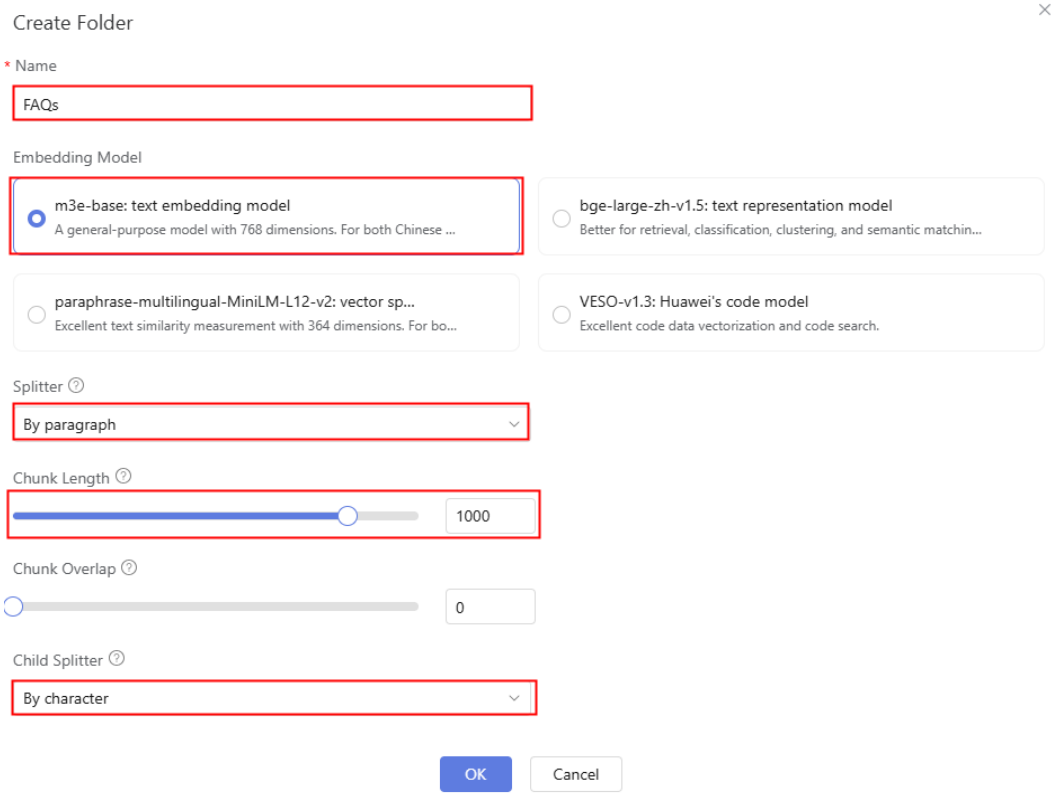


Table 10-5 Parameters for creating a folder

Parameter	Description	Example
Name	Name of the folder to be created.	FAQs

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Embedding Model	Select an embedding model as required. • **m3e-base: text embedding model**: A general-purpose model with 768 dimensions for both Chinese and English. • **bge-large-zh-v1.5: text representation model**: Better for retrieval, classification, clustering, and semantic matching tasks. Currently, this model is limited to Chinese text. • **paraphrase-multilingual-MiniLM-L12-v2: vector space model**: Excellent text similarity measurement with 364 dimensions for both Chinese and English text. • **VESO-v1.3: Huawei's code model**: Excellent code data vectorization and code search.	m3e-base: text embedding model

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Splitter	Converts text into units that can be understood by the model. The following types are supported: • By paragraph (default): Splits text by double line breaks (\n\n) or paragraph structure. • By line: Splits text by single line breaks (\n). • By sentence (English): Splits text by sentence boundaries. • By character: Splits text into individual characters (including whitespace characters).	By paragraph
Chunk Length	Maximum length of a chunk. The value range is [50, 1200]. The default value is **1000**.	1000
Chunk Overlap	Overlap between sequential chunks: 0 to chunk length (default: 0).	0
Child Splitter	Applied when a chunk exceeds the maximum allowed length. The following types are supported: • By character (Default): Splits text into individual characters (including whitespace characters). • By sentence (English): Splits text by sentence boundaries. • By word (English): Splits text by whitespace characters.	By character

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 6 After the setting is complete, click the confirm button to return to the knowledge base page.

Step 7 Click the created folder (for example, FAQs) and click **Upload File**.

In this practice, the FAQs in the Huawei Cloud Astro Zero help center are used as an example. You can click **Common FAQs** to download the document.

Figure 10-13 Clicking the button to upload a file

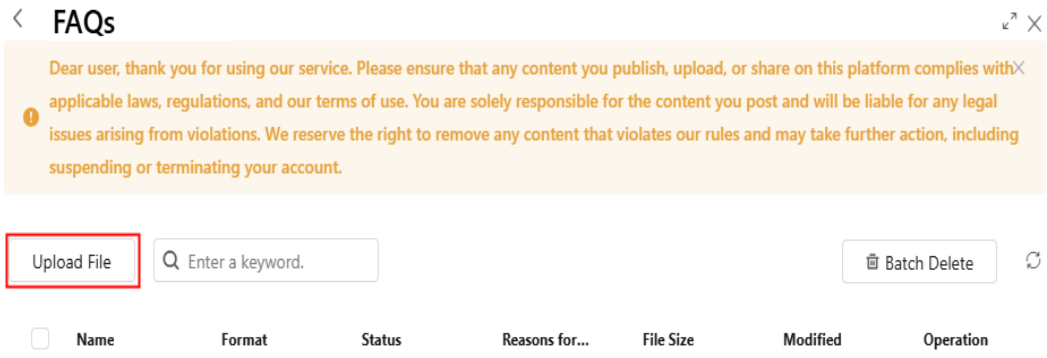
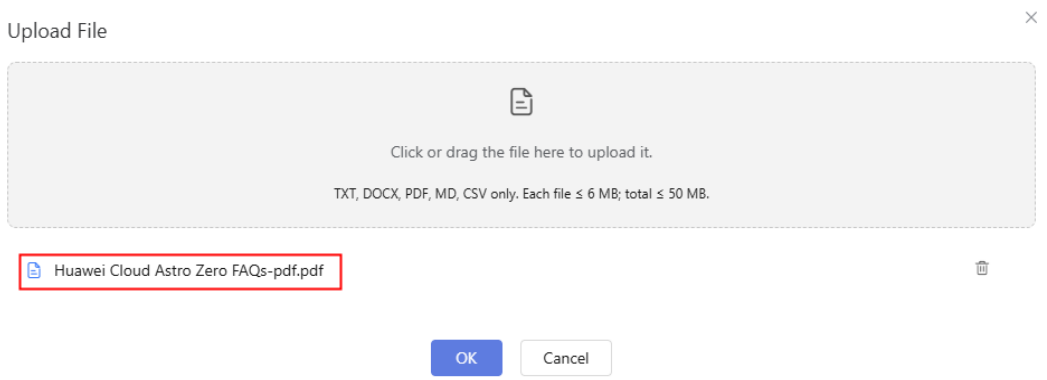


Figure 10-14 Uploading an FAQ file

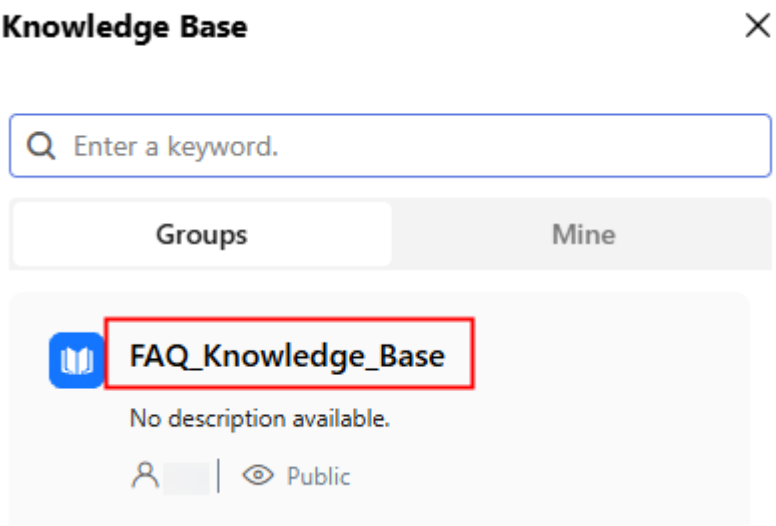


After the file is uploaded, the status of the file shows that it is being imported. Click to refresh the status. If the status changes to **Successfully**, the file is ready to be used.

Step 8 Return to the knowledge base page and select one or more created knowledge bases.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 10-15 Selecting a knowledge base



Step 9 Ask the AI assistant questions about Huawei Cloud Astro Zero to locate the relevant documents.

----End

10.2 Calling the Foundation Model Connector to Talk with the AI Assistant

Application Scenarios

When you build a standard page, you can connect to an external foundation model API via the AI assistant widget. First, set up the foundation model connection details in the connector, then choose that connector in the AI assistant widget.

This practice shows you how to configure the connector and use it in the widget for smart chat functionality.

Prerequisites

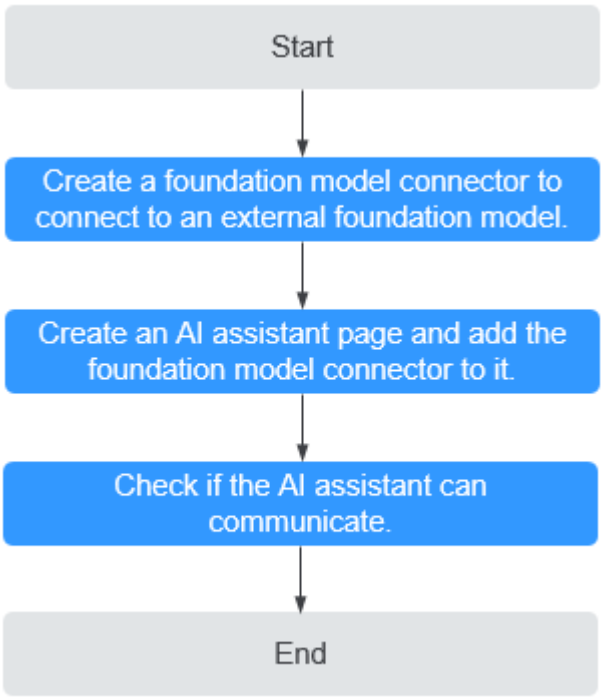
You have obtained or set up foundation model resources. This practice takes connecting to a self-built foundation model as an example.

Procedure

Figure 10-16 shows how to set up a foundation model connector and use it with the AI assistant widget to talk with the AI assistant.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 10-16 Calling a foundation model connector to chat with the AI assistant



Step 1: Create a Foundation Model Connector to Connect to an External Foundation Model

Create a foundation model connector to connect to an external foundation model API.

Step 1 Create a low-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After you have an instance, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. In the navigation pane, choose **Applications**. On the displayed page, click **Low-Code** or .

   When you create an application for the first time, create a namespace as prompted. Once it is created, you cannot change or delete it, so check the details carefully. Use your company or team's abbreviation for the namespace. Enter up to 15 characters. Use only letters, digits, and underscores (_). Start with a letter and end with a letter or digit. Do not enter consecutive underscores.
4. In the displayed dialog box, choose **Standard Applications** and click the confirm button.
5. Enter a label and name of the application, and click the confirm button. The application designer is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 10-17 Creating a blank application

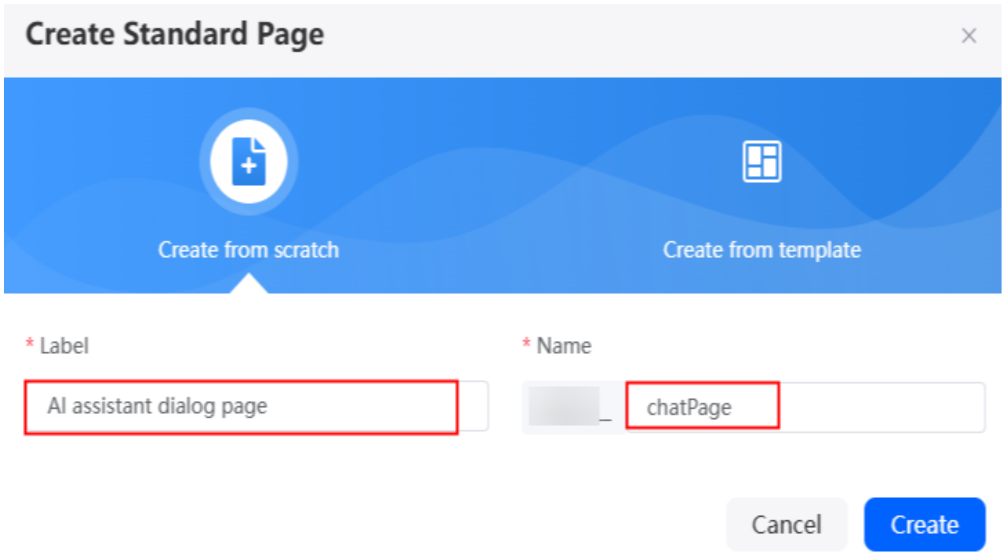


Table 10-6 Parameters for creating a blank application

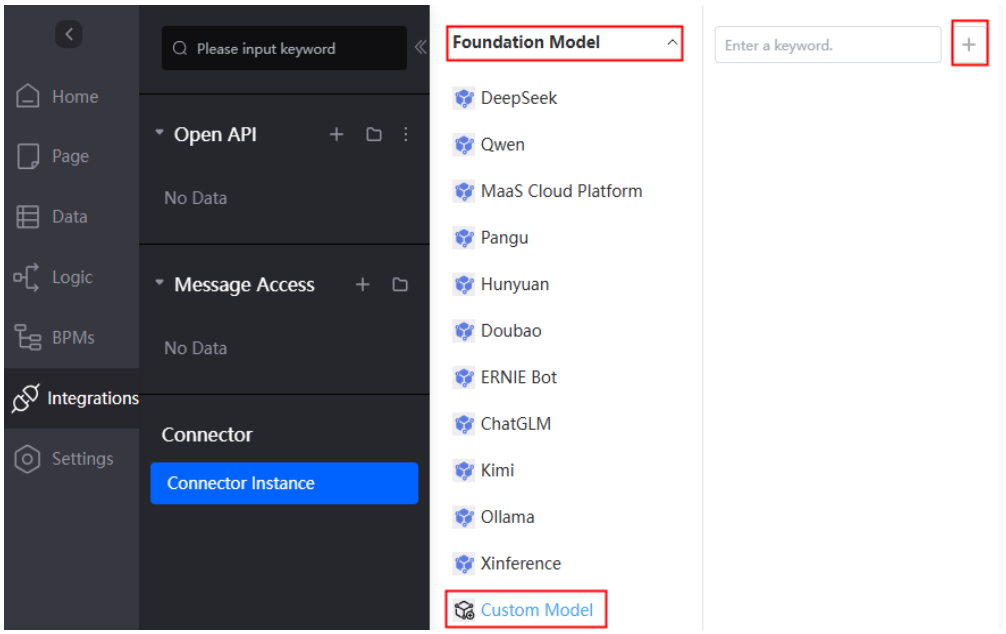
Parameter	Description	Example
Label	The label for the new application; Max. 80 characters.	AI assistant dialog page
Name	Name of the new application. After you enter the label value and click the text box of this parameter, the system automatically generates an application name and adds ***Namespace***__ before the name. Naming rules: – Max. characters: 31, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and can contain only letters, digits, and underscores (_). It cannot end with an underscore (_).	ChatApp

Step 2 Create a foundation model connector.

1. In the application designer, choose **Integrations** > **Connector** > **Connector Instance**.
2. Choose **Foundation Model** > **Custom Model** and click **+**. The page for creating a custom foundation model connector is displayed.

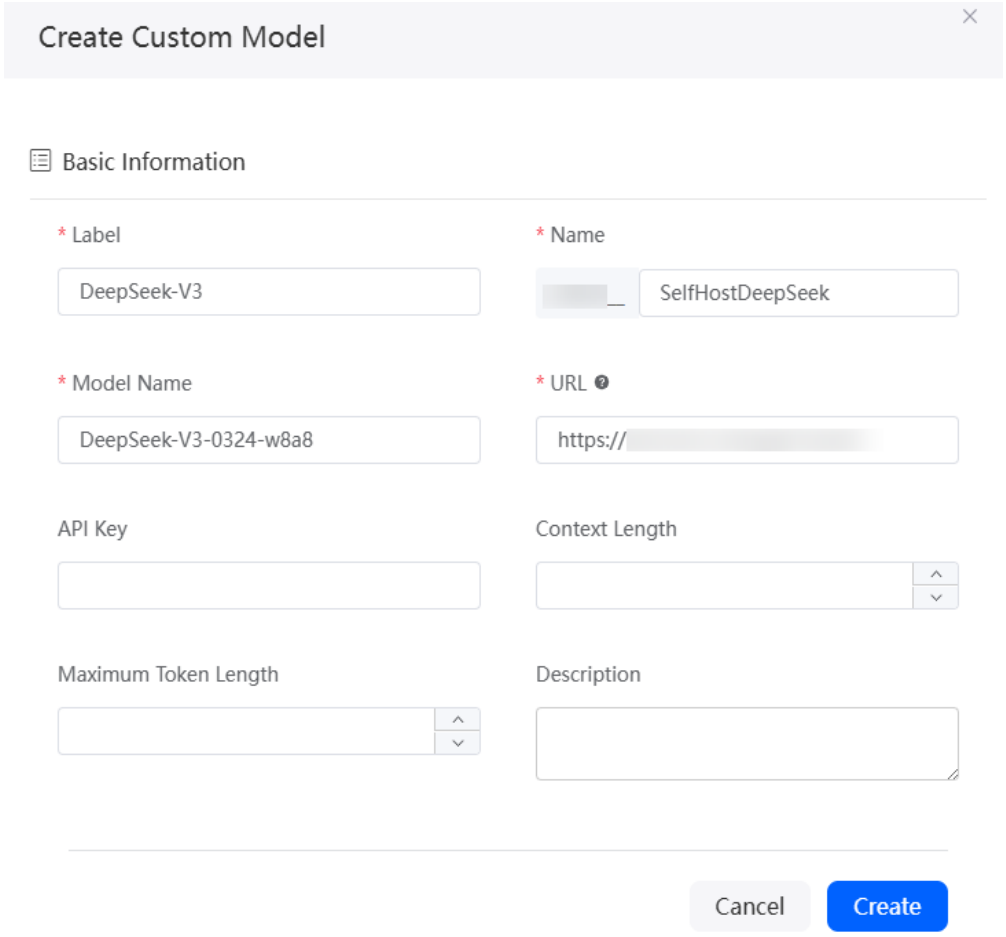
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 10-18 Creating a custom foundation model connector



3. Enter and save the configuration information of the foundation model connector.

Figure 10-19 Setting foundation model connector information



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 10-7 Parameters for configuring the custom foundation model connector

Parameter	Description	Example
Label	Custom foundation model connector label, which can be modified after being created. Value: 1–64 characters.	DeepSeek-V3
Name	Custom connector name, which uniquely identifies the connector in the system and cannot be changed after the connector is created. Naming rules: – Max. 64 characters, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. – Start with a letter and can contain only letters, digits, and underscores (_). It cannot end with an underscore (_).	SelfHostDeepSeek
Model Name	Model ID of the specific foundation model. When the connector is called, this field is passed to the **model** field in the foundation model API.	DeepSeek-V3-0324-w8a8
URL	URL for accessing the foundation model API, provided by the model provider.	https://*****

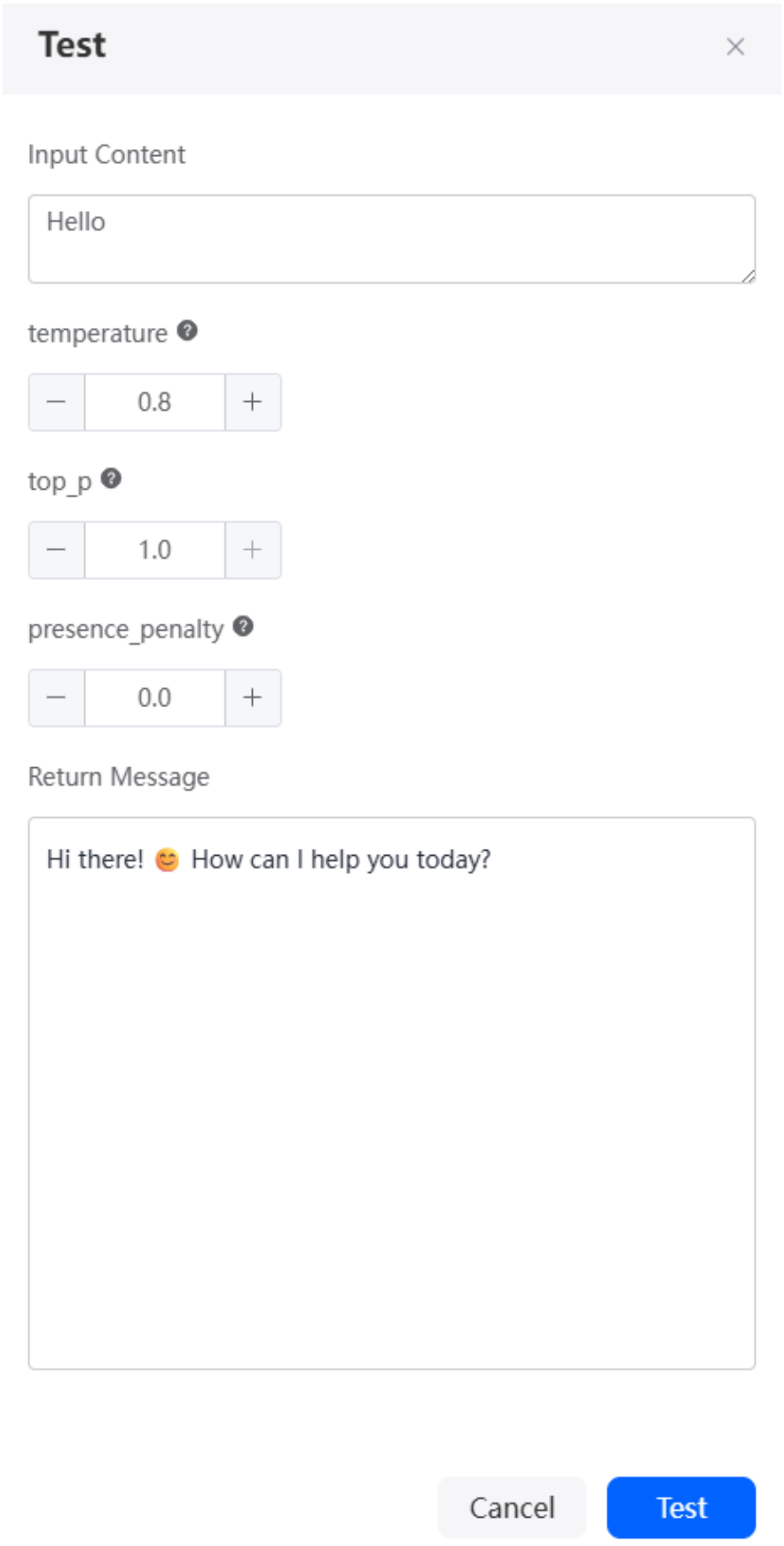
Step 3 Test the connectivity of the foundation model connector.

1. On the custom foundation model details page, click **Test**. On the displayed dialog box, click **Test**. The foundation model connector test page is displayed.
2. In the **Input Content** text box, enter the dialog content (for example, "Hello"), and click **Test**. Check whether the returned message is correct.

   If the returned message contains the streaming response packet, the foundation model connector is configured successfully. Otherwise, check whether the foundation model connector information is correct.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 10-20 Foundation model connector test page



----End

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 2: Create an AI Assistant Page and Add the Foundation Model Connector to the Page

In the AI assistant template application, create a standard page for assembling the dialog page of the AI assistant template.

Step 1 In the navigation pane, choose **Page**.

Step 2 Click  next to the page, set the basic information about the standard page, and click **Create**.

Figure 10-21 Creating a standard page

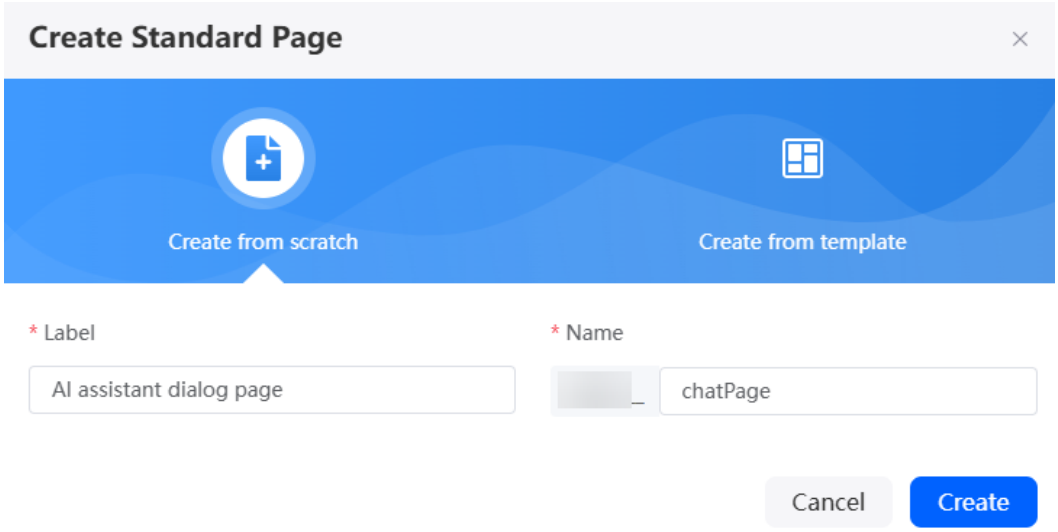


Table 10-8 Parameters for creating a standard page

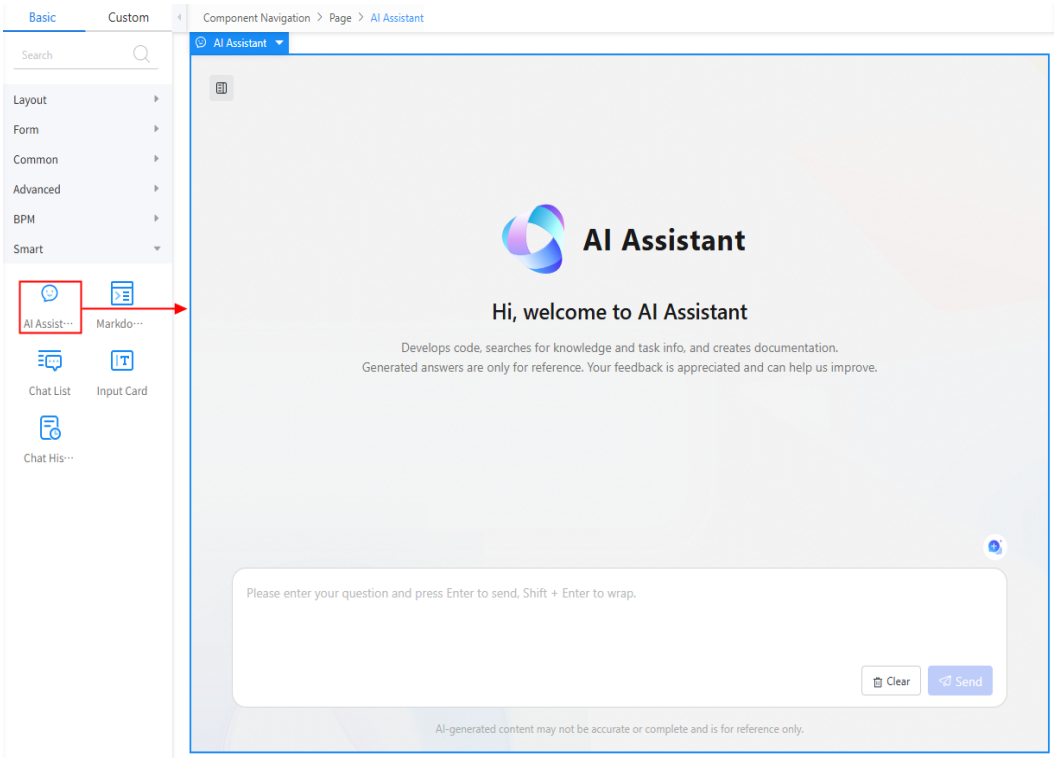
Parameter	Description	Example
Label	Label of the standard page, which is displayed on the page and can be modified after being created. Value: 1–64 characters.	AI assistant dialog page

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
Name	Name of the standard page. The name is the unique identifier of the standard page in the system and cannot be changed after being created. Naming rules: • Max. 64 characters, including the prefix namespace. To prevent duplicate data names among different tenants, each tenant must define a unique namespace when first creating an application. A tenant can only create one namespace, and once it is created, it cannot be modified. • Start with a letter and can contain only letters, digits, and underscores (_). It cannot end with an underscore (_).	chatPage

Step 3 Drag the **AI Assistant** widget from **Basic** > **Smart** to the canvas.

Figure 10-22 Dragging the AI Assistant widget to the page

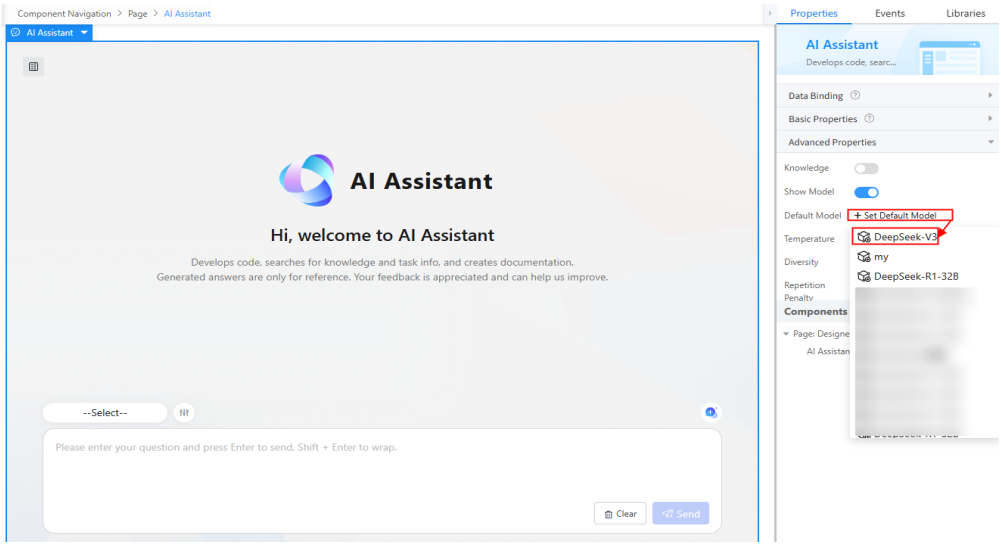


Step 4 Click the AI assistant widget, choose **Properties** > **Advanced Properties**, and enable **Show Model**.

Step 5 Click **Set Default Model** next to **Default Model** and select the model connector created in **Step 1: Create a Foundation Model Connector to Connect to an External Foundation Model**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 10-23 Advanced settings of the AI assistant widget



Step 6 Click  in the upper part of the page to save the standard page.

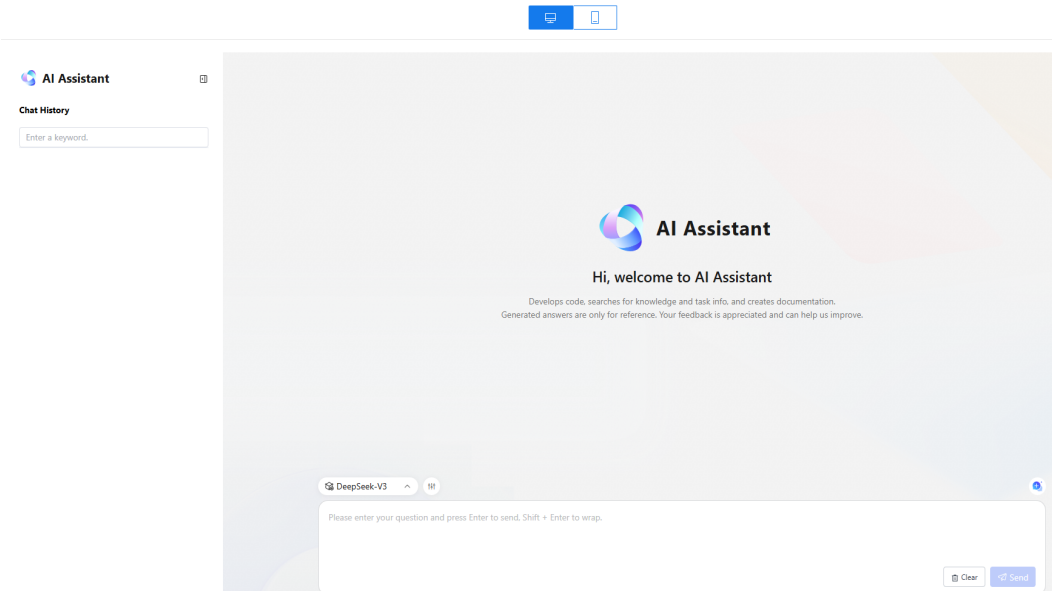
----End

Step 3: Verify the Chat Function of the AI Assistant

Preview the AI assistant page in the AI assistant template application and verify the chat function of the AI assistant.

Step 1 On the saved standard page, click  . The preview page is displayed.

Figure 10-24 Preview page of the AI assistant



Step 2 In the AI assistant chat box, type something like "How is the weather today?" and click **Send**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

If the AI assistant returns the corresponding streaming result, the AI assistant widget is successfully connected to the external foundation model API.

----End

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

11 No-Code Applications

11.1 Setting Option Associations

Expected Results

In applications for data entry, queries, and process approvals, associated filters and option limits are often used to maintain data consistency and reduce user errors. For example, in a food ordering application, you can choose coffee or milk tea from the drink category, then you can see specific coffee options (like latte, espresso, caramel macchiato) or milk tea options (like English milk tea, Hong Kong milk tea, bubble tea). On the no-code workbench, you can associate options so that the application shows only the relevant drink options based on your choice. If you choose "Coffee," you will only see coffee options.

Procedure

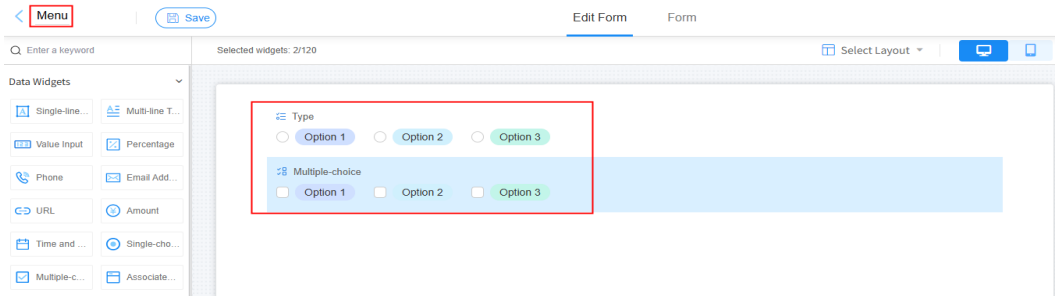
Step 1 Create a no-code application.

1. Apply for a free trial or purchase a commercial instance by referring to **Authorization of Users for Huawei Cloud Astro Zero Usage and Instance Purchases**.
2. After you have an instance, click **Access Homepage** on **Homepage**. The application development page is displayed.
3. On the top menu bar, click **Workbench**.
4. In **All Apps**, click  of **Create** and select **Create Blank App**.
5. Enter an application title (for example, "Ordering"), click **Create Form**, and select **Create Blank Form**. The ordering application is displayed.

Step 2 Change the form title to **Menu**, delete unnecessary widgets, and drag a multi-choice widget from **Data Widgets** to the canvas.

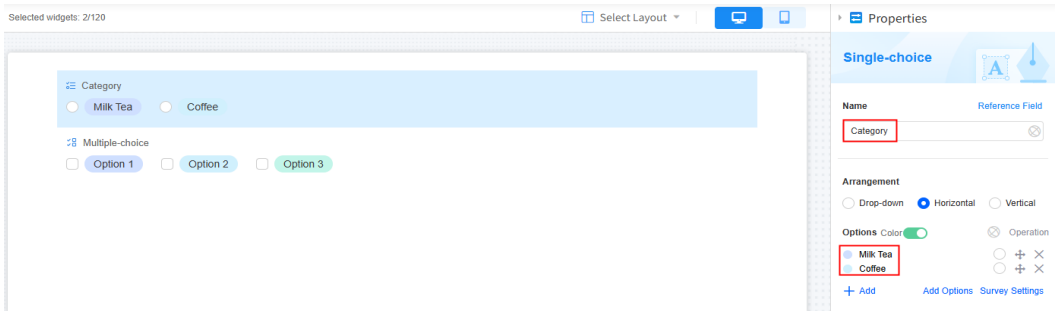
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 11-1 Designing a menu form



Step 3 Click the single-choice widget to modify its display name and options.

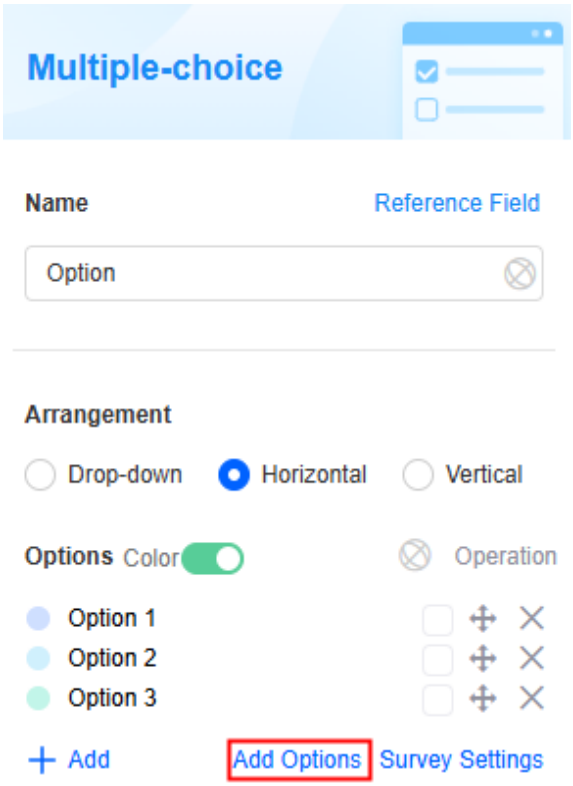
Figure 11-2 Setting the single-choice widget



Step 4 Set the multi-choice widget.

1. Click the multi-choice widget and change its display name to **Option**.
2. In the **Options** area, click **Add Options**.

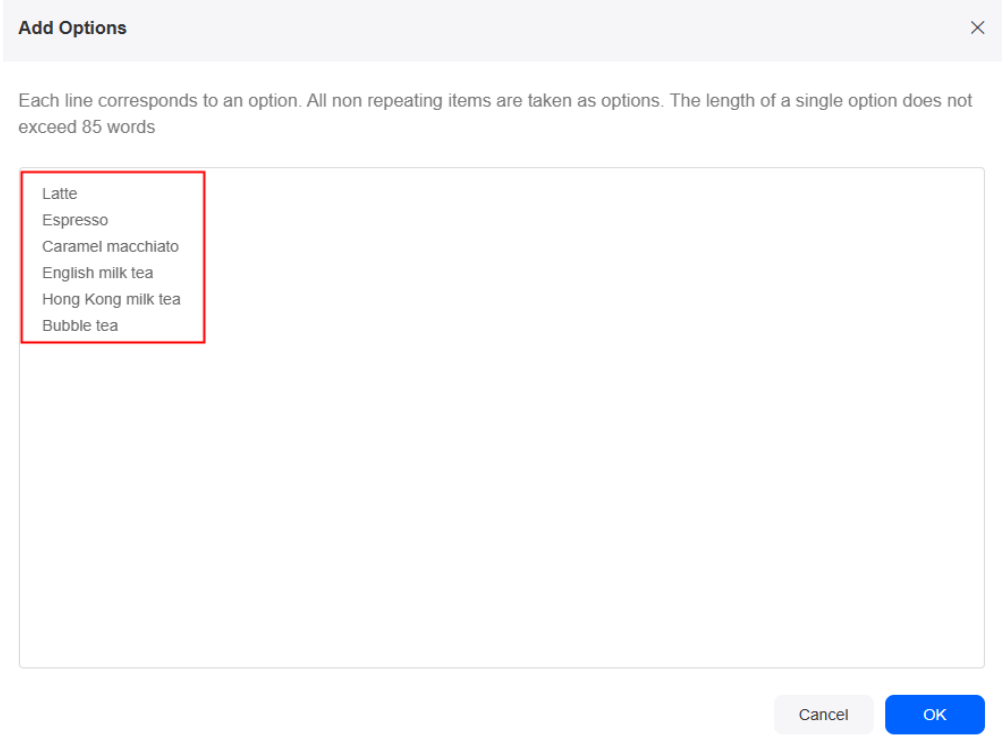
Figure 11-3 Clicking Add Options



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3. Add options and click **OK**.

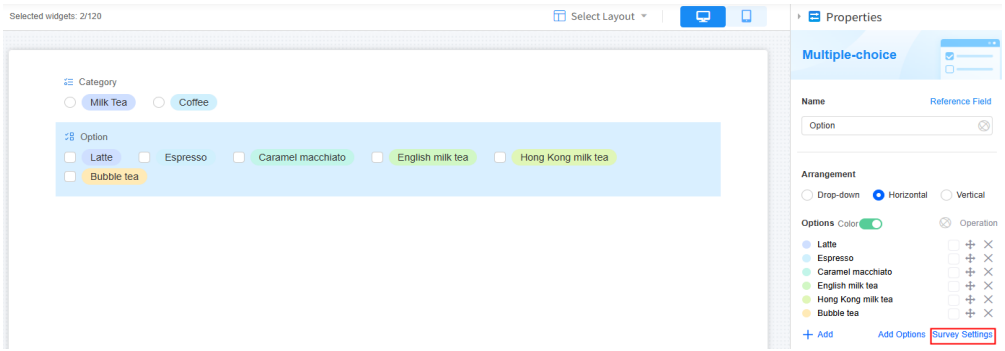
Figure 11-4 Adding options



Step 5 Set option associations.

1. Click **Survey Settings** on the right.

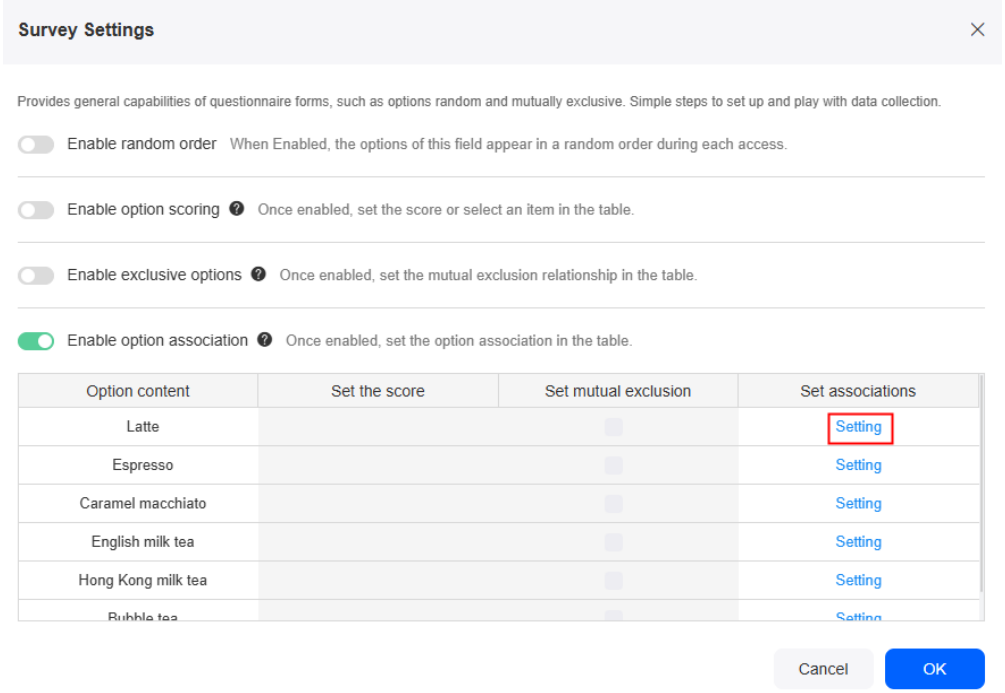
Figure 11-5 Clicking Survey Settings



2. Toggle on **Enable option association** and click **Setting**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

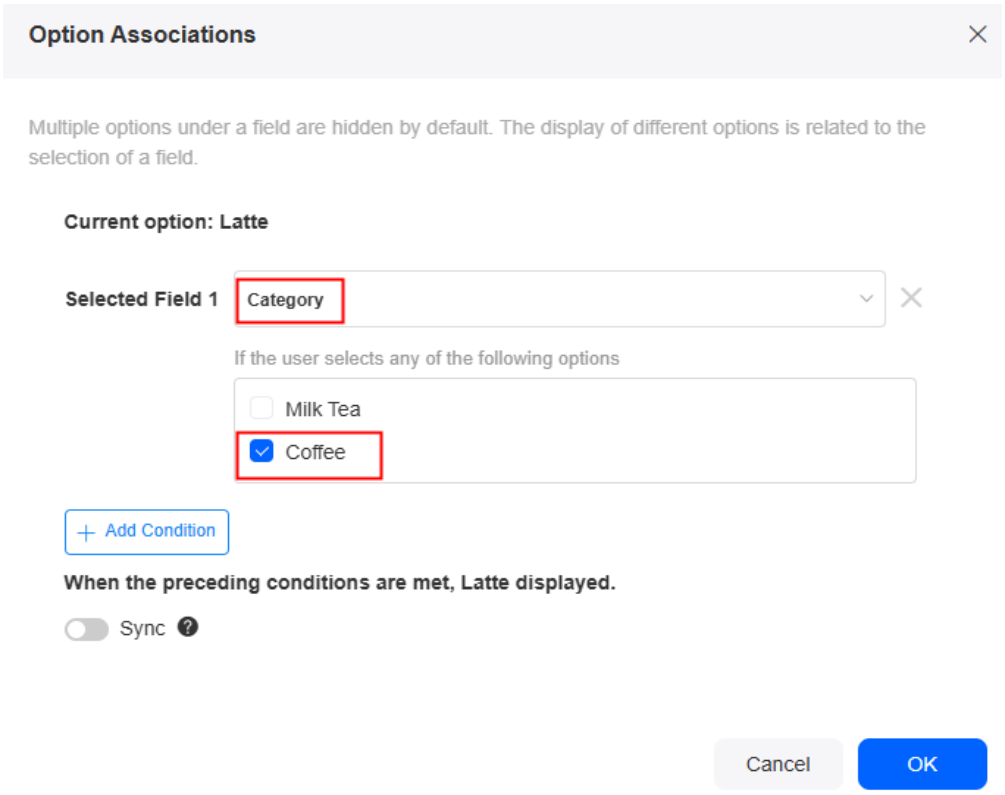
Figure 11-6 Adding option associations



3. Select the value of the associated field, for example, **Coffee** under **Category**, and click **OK**.

   When this option is selected, the current option "Latte" is displayed on the page.

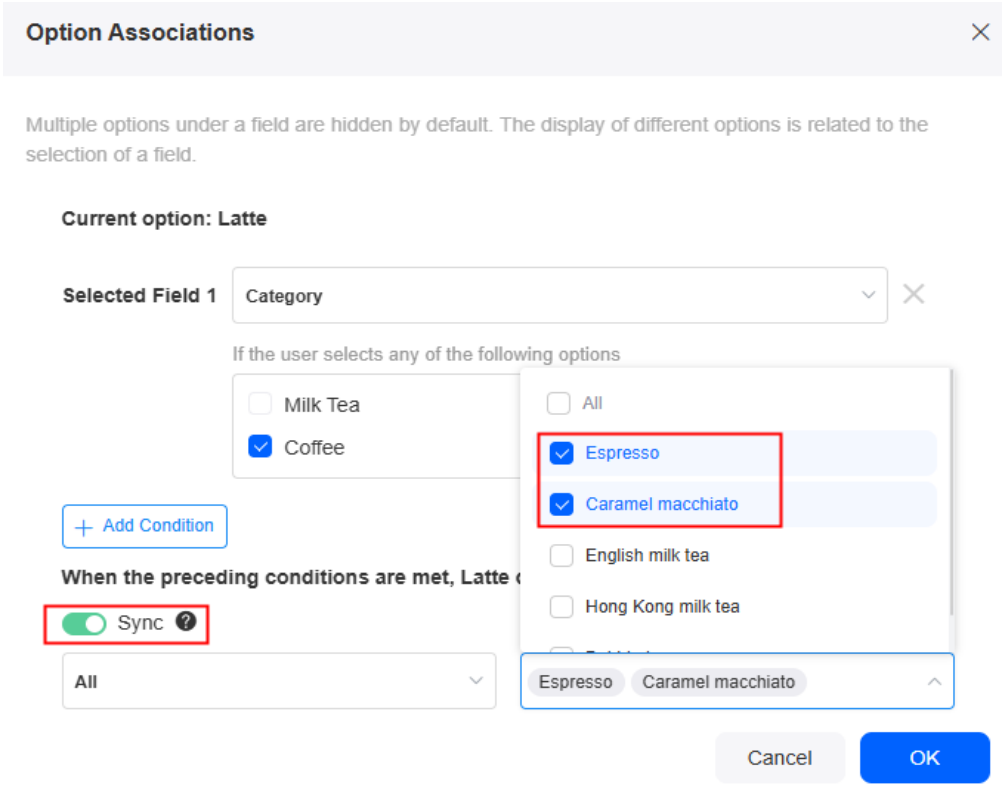
Figure 11-7 Setting associated options



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

4. Enable **Sync** to associate other options with the coffee category, as shown in **Step 5.3**. For example, associate espresso and caramel macchiato with the coffee category.

Figure 11-8 Associating other options



After the synchronization is successful, return to the survey setting page. The values of **Set associations** are changed to **Modify**. Click **Modify** next to **Espresso**. Its category is changed to **Coffee**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 11-9 Returning to Survey Settings

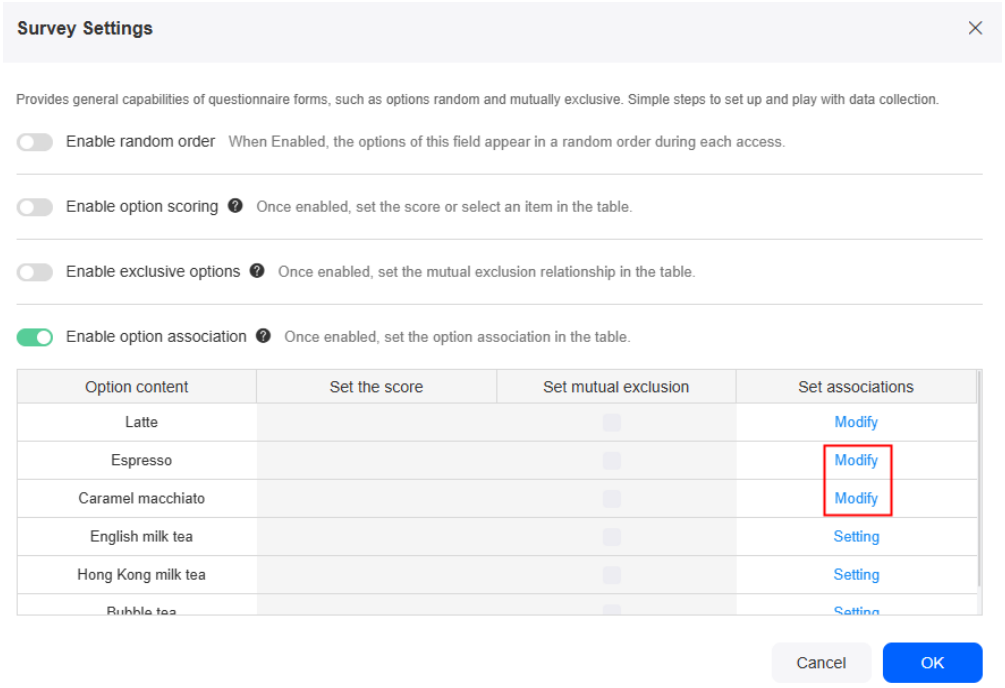
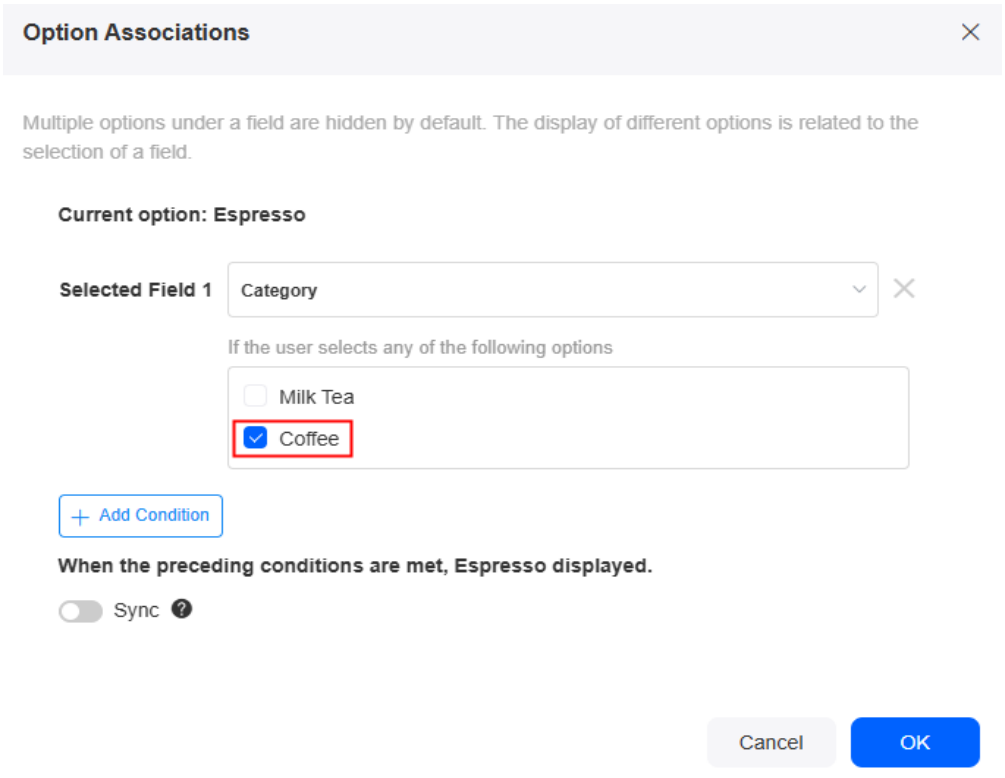


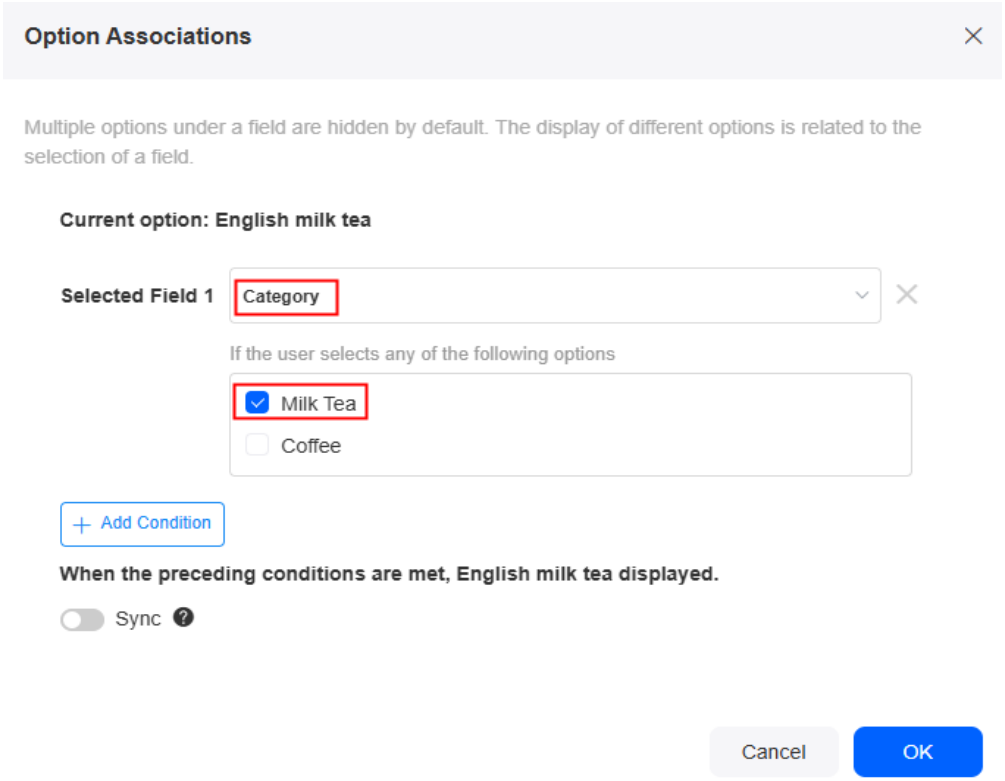
Figure 11-10 Viewing the modified category



5. On the survey setting page, click **Setting** next to **Hong Kong milk tea** and set its category to milk tea.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

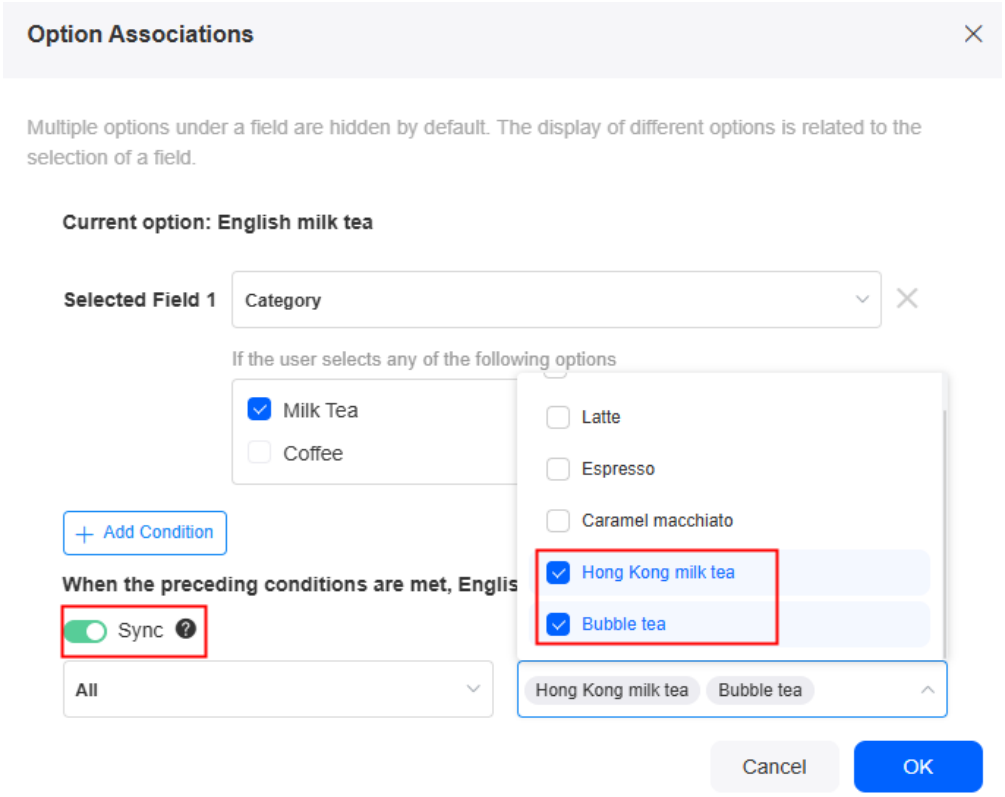
Figure 11-11 Setting option associations



6. Enable **Sync** to add **Hong Kong milk tea** and **Bubble tea** to the milk tea category.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 11-12 Setting the Hong Kong milk tea and bubble tea categories

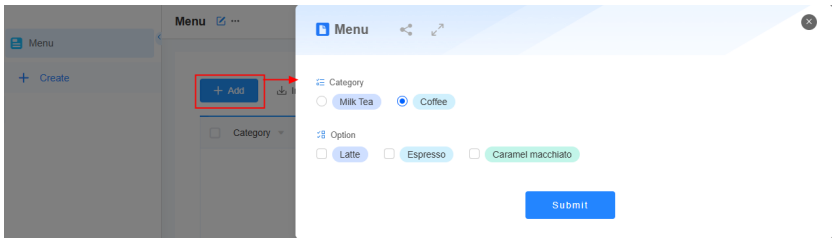


7. Click **OK** to return to the survey setting page.
8. On the survey setting page, click the confirm button to return to the form design page.

Step 6 Click **Save**, return to the application development page, share the form, and view the final effect.

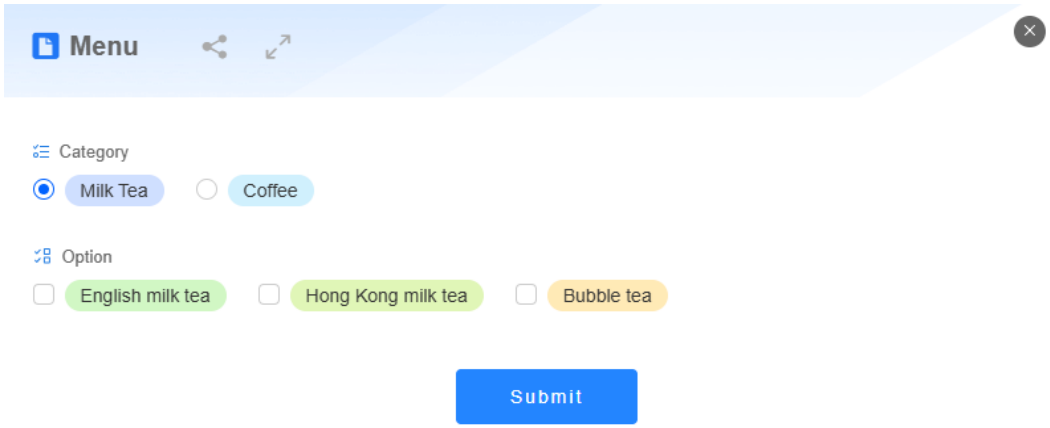
When you click coffee, **Figure 11-13** is displayed. When you click milk tea, **Figure 11-14** is displayed. The effect meets the expectation.

Figure 11-13 Viewing the final effect (coffee)



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 11-14 Viewing the final effect (milk tea)



----End

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.