

CodeArts

Best Practices

Issue 02
Date 2023-05-08



Copyright © Huawei Technologies Co., Ltd. 2023. 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 Technologies Co., Ltd.

Trademarks and Permissions



HUAWEI and other Huawei trademarks are trademarks 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 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.

Security Declaration

Vulnerability

Huawei's regulations on product vulnerability management are subject to "Vul. Response Process". For details about the policy, see the following website:<https://www.huawei.com/en/psirt/vul-response-process>
For enterprise customers who need to obtain vulnerability information, visit:<https://securitybulletin.huawei.com/enterprise/en/security-advisory>

Contents

1 Huawei E2E (HE2E) DevOps Practice.....	1
1.1 Overview.....	1
1.2 Planning Resources.....	5
1.3 Process.....	5
1.4 Procedure.....	7
1.4.1 Preparations.....	7
1.4.2 Step 1: Managing Project Plans.....	9
1.4.3 Step 2: Managing Project Configurations.....	13
1.4.4 Step 3: Writing Code.....	15
1.4.5 Step 4: Checking Code.....	19
1.4.6 Step 5: Building an Application.....	20
1.4.7 Step 6: Deploying an Application (CCE).....	26
1.4.8 Step 6: Deploying an Application (ECS).....	30
1.4.9 Step 7: Managing Project Testing.....	34
1.4.10 Step 8: Configuring a Pipeline for CD.....	39
1.4.11 Releasing Resources.....	42

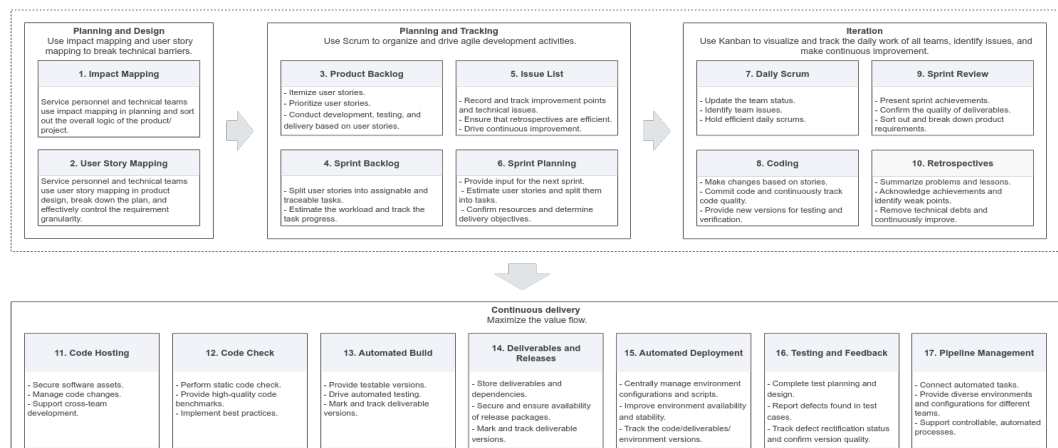
1 Huawei E2E (HE2E) DevOps Practice

1.1 Overview

Background

Huawei's end-to-end (HE2E) DevOps implementation framework is an operable, feasible, and agile development methodology developed based on years of R&D experience and industry-leading practices, as shown in [Figure 1-1](#).

Figure 1-1 HE2E DevOps implementation framework



- **Planning and Design**

Steps 1 and 2 in the diagram represent the process of product planning between service personnel (even customers) and technicians to sort out the overall product logic, implementing product planning and design, and controlling the requirement granularity and breakdown.

- Software development solves issues and delivers value, not simply provides functions. The impact map is used to identify user requirements and root causes.
- During microservice design, we put objectives and requirements in user stories to facilitate information exchanges among customers, service

personnel, and developers. If you view only separate requirement items, you will not think from the entire solution's perspective. User stories focus on scenarios, sort out and display stages and activities in a tree structure. In this way, you will view both requirement items and overall requirement scenarios.

- Planning, Tracing, and Sprint

Steps 3 to 10 are the main management practices in a Scrum framework process.

- Scrum defines a relatively complete framework for agile process management. CodeArts well integrates the Scrum framework with daily activities of development teams. Main process deliverables include the product backlog, sprint backlog, potential deliverable product increments, and issue list. Core team activities include Sprint planning meetings, daily Scrums, Sprint reviews, Sprint retrospective meetings, and daily team updates.

- ContainerOps

Start from step 11 and enter the engineering practice in a CI/CD process.

- CI/CD is based on code configuration management. It covers not only traditional security control of code assets, concurrent development, and version and baseline management, but also reflects team collaboration and communication.
- The pipeline connects code check (or static scanning), automated build, automated testing in all stages, and automated deployment.
- CI/CD also covers continuous artifact management and environment management at different levels, including development, test, quasi-production, and production environments.
- The CI/CD pipeline manages stages, environments, activities, entry and pass quality gates, and input and output artifacts in each stage.

Application Scenarios

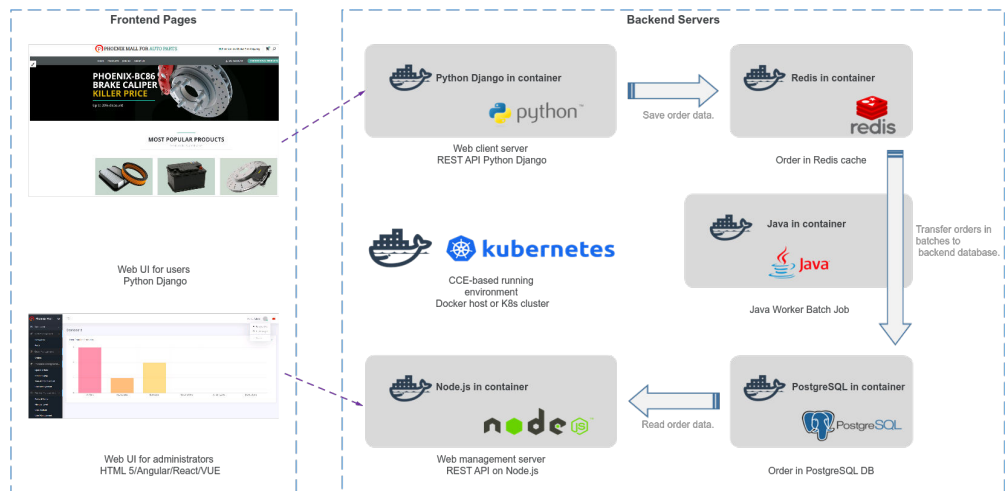
We will use the sample code for an auto part e-mall named Phoenix Mall and a DevOps full-process sample project to describe how to use CodeArts to implement the HE2E DevOps framework. This solution is applicable to agile/Scrum R&D projects.

Solution Architecture

- Phoenix Mall Architecture

Figure 1-2 shows the architecture of the Phoenix Mall sample project.

Figure 1-2 Technical architecture of Phoenix Mall



The sample project consists of five microservice components that can be independently developed, tested, and deployed, as shown in [Table 1-1](#).

Table 1-1 Microservice components of Phoenix Mall

Microservice Component	Description
Web client server (corresponding to the Vote function in the sample code)	<ul style="list-style-type: none"> Service logic: Users can use a browser to access the web UI of this service. When a user clicks Like on a specific offering, the service saves the record of the selected offering in the Redis cache. Technology stack: Python and Flask frameworks Application server: Gunicorn
Web management server (corresponding to the Result function in the sample code)	<ul style="list-style-type: none"> Service logic: Users can use a browser to access the web UI of this service. The statistics about Like clicked by users on the UI are dynamically displayed. The data is obtained from the PostgreSQL database. Technology stack: Node.js and Express frameworks Application server: server.js

Microservice Component	Description
Background order batch processing program (corresponding to the Worker function in the sample code)	<ul style="list-style-type: none"> Service logic: This service is a background process. It monitors item records in the Redis cache, obtains new records, and saves them in the PostgreSQL database so that the management UI can extract data for statistics display. Technology stack: .net core or Java (This service provides two technology stacks to implement the same function. You can modify the configuration and select one as the runtime process.)
Order cache	<ul style="list-style-type: none"> Service logic: Persists data for the client UI. Stack: Redis
Order database	<ul style="list-style-type: none"> Service logic: Persists data for the management UI. Stack: PostgreSQL

- Composition of the DevOps Full-Process Sample Project
This project uses Scrum and presets some service templates. Products and services involved in this project.

Table 1-2 List of involved products/services

Service	Description	
CodeArts	Req	Presets three planned and completed Sprints, project module settings, and several statistical reports.
	Repo	Presets the code repository phoenix-sample for storing sample code.
	Check	Presets four tasks. For details, see Step 4: Checking Code .
	Build	Presets five tasks. For details, see Step 5: Building an Application .
	Artifact	Stores software packages generated by build tasks.
	Deploy	Presets three applications. For details, see Step 6: Deploying an Application (CCE) .
	TestPlan	Presets more than 10 test cases in the test case library.
	Pipeline	Presets five pipelines. For details, see Step 8: Configuring a Pipeline for CD .

Service		Description
Other components and services	Identity and Access Management (IAM)	Manages accounts.
	SoftWare Repository for Container (SWR)	Stores Docker images generated by build tasks.
	Cloud Container Engine (CCE)	Deploys software packages, which is different from ECS-based deployment.
	Elastic Cloud Server (ECS)	Deploys software packages, which is different from CCE-based deployment.

Advantages

- We provide a one-stop cloud DevOps platform to manage the entire software development process, to address R&D pain points such as frequent requirement changes, complex development and test environments, difficult multi-version maintenance, and failure to effectively monitor the progress and quality.
- This service provides visualized and customizable pipeline services for CD, doubling the software rollout speed.

1.2 Planning Resources

The following table lists the resources required for completing this practice. The practice may take 2 to 3 hours.

Table 1-3 Resource planning

Resource Name	Quantity
CodeArts	Buy the basic edition and basic test package.
Cloud Container Engine (CCE)	1
Elastic Cloud Server (ECS)	1

1.3 Process

This document describes the operation process of the HE2E DevOps practice.

Figure 1-3 HE2E DevOps practice process

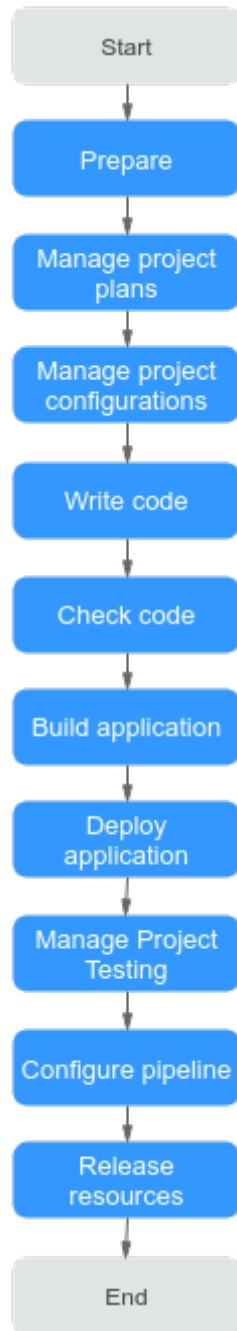


Table 1-4 HE2E DevOps practice process description

Step	Description
Prepare	Complete preparations before the practice, including creating a project and adding project members.
Manage project plans	Complete the overall planning of the project, including project and Sprint requirements.

Step	Description
Manage project configurations	Customize the work item change notification and status transfer modes based on project requirements.
Write code	Use branches to compile code, including create branches, commit code, and merge branches.
Check code	Perform static scanning on the code and optimize the code based on fix suggestions to improve the code quality.
Build application	Build environment images and compile and package code into software packages.
Deploy application	Install and run the built environment image and software package in the environment. This document provides deployment methods in two environments: Cloud Container Engine (CCE) and Elastic Cloud Server (ECS).
Manage Project Testing	Create test plans for Sprints, design test cases, and execute test cases as planned.
Configure pipeline	Orchestrate jobs such as code check, build, and deployment into a pipeline. When code is updated, the pipeline is automatically triggered for CD.
Release resources	After completing the practice, release resources for CodeArts, CCE, and so on.

1.4 Procedure

1.4.1 Preparations

Before performing a specific task, you need to complete the following preparations:


Prerequisites

You have **purchased CodeArts** (together with the basic edition package and CloudTest basic package).

Creating a Project

Before starting the practice, Sarah creates a project.

Step 1 **Log in to the CodeArts console.**

Step 2 Click  and select a region.

Step 3 Clicking **Access Service**.

Step 4 Click **Create Project**, and select **DevOps Full-Process Sample Project**.

Step 5 Enter the project name **Phoenix Mall** and click **OK**. The project is created.

----End

Adding Project Members

The product owner Sarah creates accounts for team members and adds them to the project.

This sample project involves four project roles. To facilitate introduction, each role in this document corresponds to a person, as shown in [Table 1-5](#).

Table 1-5 Project role list

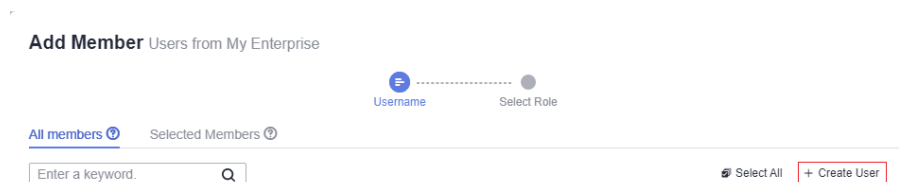
Project Member	Project Role	Responsibility
Sarah	Product owner (project creator)	Be responsible for the overall product planning and product team setup.
Maggie	Project manager	Manage project delivery plans.
Chris	Developer	Develop, compile, deploy, and verify project code.
Billy	Tester	Write and execute test cases.

Step 1 Go to the **Phoenix** project. Choose **Settings > General Settings > Members** from the left navigation pane.

Step 2 Click **Add Member** above the project member list and choose **Users from My Enterprise** from the drop-down list.

Step 3 In the dialog box that is displayed, click **Create User**. The **Users** page is displayed.

Figure 1-4 Adding members



Step 4 Click **Create User** and name them **Maggie**, **Chris**, and **Billy** in sequence.

Step 5 Return to the CodeArts page, refresh the browser, click **Add Member** above the member list, and choose **Users from My Enterprise**. Select members **Maggie**, **Chris**, and **Billy**, and click **Next**.

Step 6 Click the **Role** drop-down list in each row, select **Project manager** for Maggie, **Developer** for Chris, and **Tester** for Billy, and then click **Save**.

----End

1.4.2 Step 1: Managing Project Plans

CodeArts Req provides simple and efficient team collaboration services, including multi-project management, agile iteration, and task management.

This sample project uses the Scrum mode for iterative development. Each Sprint lasts for two weeks. The Phoenix Mall version has been developed in the first three Sprints, and Sprint 4 is being planned.

According to the project plan, time-limited discount and group buying activity management functions need to be implemented in Sprint 4.

Due to business and market changes, store network query is added as an urgent requirement. Therefore, this function will be developed in Sprint 4.

This section describes how Sarah (product owner) and Maggie (project manager) manage requirements and Sprints, and track the project progress.

Managing Requirement Planning

Project requirements are managed using mind maps, which present work items in the hierarchical structure of "Epic > Feature > Story > Task". These work item types are described in [Table 1-6](#).

Table 1-6 Work item types

Type	Description
Epic	An important strategic measure of a company. For example, Phoenix Mall in this sample project is a key strategic measure related to the survival of the company.
Feature	Function valuable to customers. You can use features to meet customer requirements. For example, the store network query function in Phoenix Mall is continuously delivered in multiple Sprints.
Story	Breakdown of a function into user scenarios. A story can be completed in a Sprint.
Task	Breakdown of a user story. Preparing environments and test cases can be tasks for a story.

Step 1 Create a work item for the new requirement.

The store network query function is a new requirement. Therefore, the product owner Sarah needs to add it to the requirement planning view.

1. Go to the **Phoenix** project, choose **Work** from the left navigation pane, and click the **Plans** tab.

2. Go to the Phenix Mall mind map.

 **NOTE**

If the **Plans** tab page is empty, create a mind map.

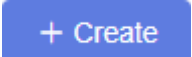

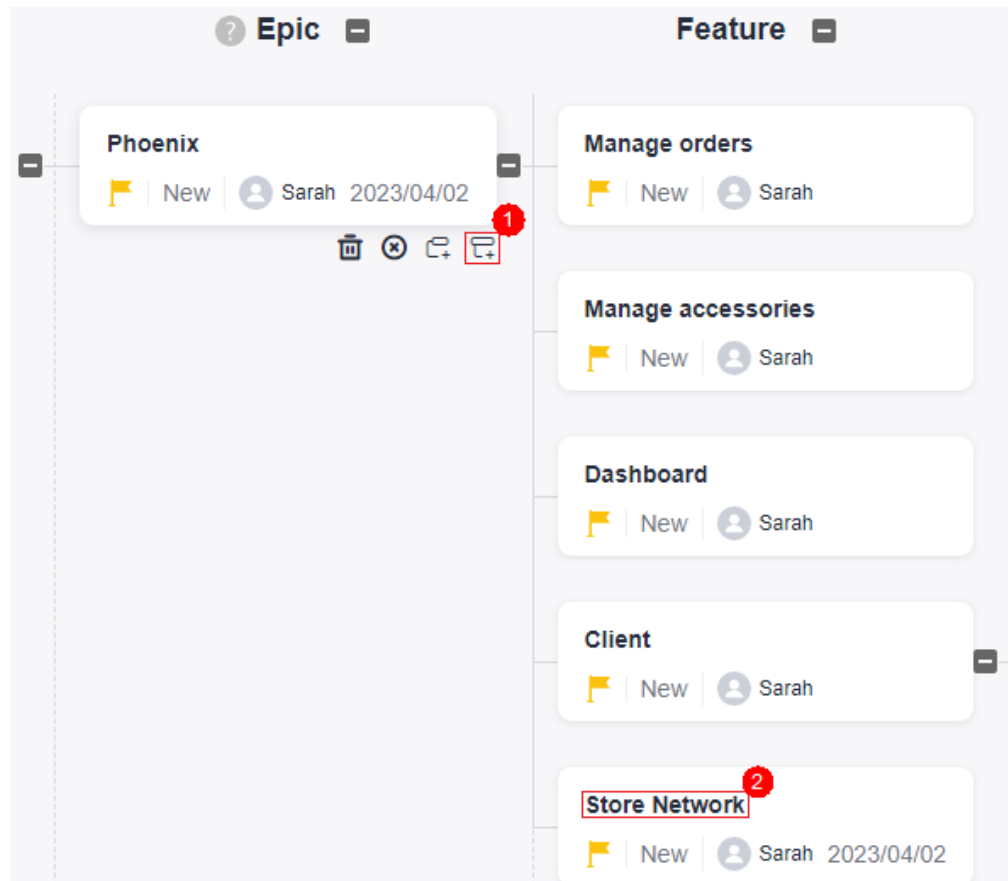
1. Click  and choose **Mind Map** from the drop-down list.
In the dialog box that is displayed, enter the name **Requirement_Planning**, and click **OK**. The mind map details page is displayed.
 2. Click **Add Epic**. In the dialog box that is displayed, select **Phoenix** and click **OK**.
3. Create a feature **Store Network**.
 - a. Click  under Epic **Phoenix**.
 - b. Enter the name **Store Network** and press **Enter** to save the settings.

Figure 1-5 Creating a feature



4. Use the same method to add a story **User can query network of all stores** to the feature **Store Network**.

Step 2 Edit the story.

1. Click Story **to query all store network** and edit story information by referring to the following table.

Table 1-7 Story configurations

Configuration Item	Suggestion
Description	Enter As a user, I want to query all stores so that I can select a proper store to obtain the service.
Priority	Select High .
Severity	Select Critical .

2. Prepare a local Excel file for **Store Network**. For details about the file content, see the following table.

Table 1-8 Store network list

Branch Name	Branch Address
Branch A	123 meters to the departure floor, Terminal 1, Airport E
Branch B	No. 456, Street G, Area F
Branch C	No. 789, Street J, Area H
Branch D	West side of Building K, Avenue L, Area K

3. Return to the story editing page, find **Click to select a file, or drag and drop a file**, choose **Upload** from the drop-down list, and upload the list file to the work item as an attachment.
4. Click **Save**. The story details are edited.

----End

Managing Sprint Planning

Before a Sprint starts, the project manager Maggie organizes a plan meeting to add the new story to the Sprint, breaks down the story into tasks, and assigns the tasks to developers.

This section describes how to plan Sprint 4.

Step 1 Create a Sprint.


1. Go to the **Phoenix** project, choose **Work** from the left navigation pane, and click the **Sprints** tab.
2. Click  next to **Sprint** in the upper left corner of the page. In the dialog box that is displayed, configure Sprints by following [Table 1-9](#). Click **OK**.

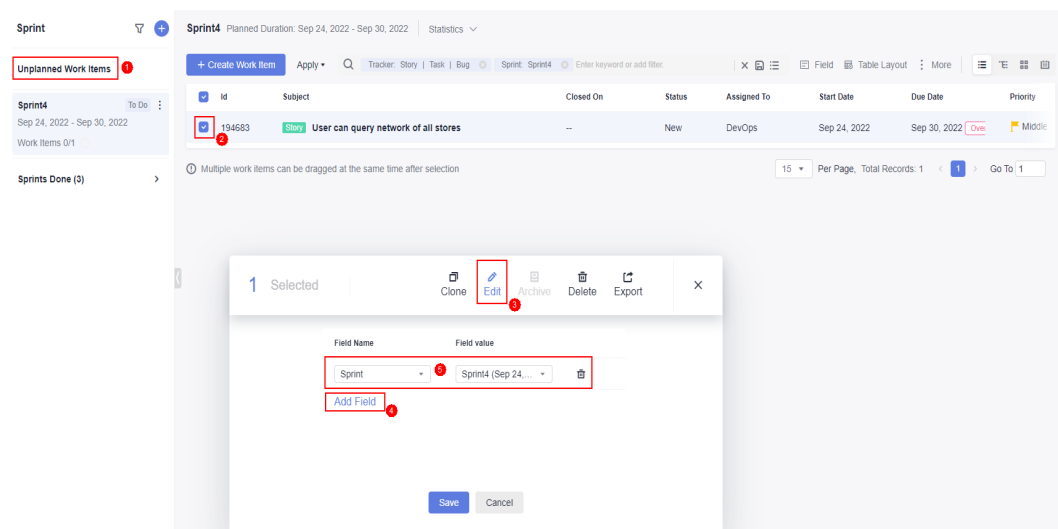
Table 1-9 Sprint information configurations

Configuration Item	Suggestion
Sprint Name	Enter Sprint4 .
Planned Duration	Set the duration to 2 weeks.

Step 2 Plan the Sprint.

1. From the left navigation pane, choose **Unplanned Work Items**.
2. Select the following three stories in the list as planned:
 - User can query network of all stores
 - Admin can add group buying activities
 - Admin can add time-limited discounts
3. Click **Edit** at the bottom of the page.
4. Click **Add Field**.
5. Choose **Sprint** from the **Field Name** drop-down list box, select **Sprint4** from the **Field Value** drop-down list box, and click **Save**.

Figure 1-6 Planning the Sprint



Step 3 Assign the stories.

1. Choose **Sprint4** from the left navigation pane.
2. Select all stories and set **Assign To** to **Chris** following [Planning Sprint](#).

Step 4 Break down the stories.

1. Find the story **User can query network of all stores**. Click the story name.
2. In the right pane of the page, click the **Child Work Items** tab.
3. Click **Fast Create Child**. Enter the title **Frontend display-add store network menu**, assign it to **Chris**, and click **OK**.

4. Use the same method to add the task **Background management-Add store network management and maintenance module**.


----End

Monitoring and Tracking Project Status

- Daily Stand-ups to Track Task Progress

After the Sprint starts, the project team communicates the current progress of each work item through daily stand-up meetings and updates the status.

You can view the status of work items in a Sprint in the card mode.

Go to the **Sprints** tab page and click  to switch to the card mode. This page displays work item cards in each status. You can drag a work item card to update its status.

- Review Meeting to Accept Results

Before the expected end time of the Sprint, the project team holds a review meeting to present work achievements of the current Sprint.

The **Sprints** tab page provides Sprint statistics and charts. The team can easily collect statistics on the progress of the current Sprint, including the requirement completion status, Sprint burndown chart, and workload.

Go to the **Sprints** tab page and click **Statistics** to display the progress view.

1.4.3 Step 2: Managing Project Configurations

Managing Project Notifications

The project manager Maggie wants team members to be notified when assigned a task (work item) so that the members can handle the task (work item) in time.


Step 1 Go to the **Phoenix Mall** project, and choose **Settings > Project Settings > Notifications** from the navigation pane.

Step 2 View default settings of the sample project displayed on the page.

We will keep default settings unchanged because they can meet requirements. If you need, modify the settings, which will be automatically saved.

Step 3 Verify the result.

When the project manager is done with breaking down the story, the developer Chris will receive the following two types of notifications.

- Direct messages: After Chris logs in to the homepage, he will view a number in the upper right corner. He can click  to view the notification.
- Email: The project member also receives an email if an email address has been configured for the corresponding user and the **Email Notifications** option has been enabled on the **This Account Settings** page.

 **NOTE**

All members can set whether to receive email notifications. To enable email notification, perform the following steps:

1. Click the username in the upper right corner of the page and choose **This Account Settings** from the drop-down list. The **Notifications** page is displayed by default.
2. Find **Email Notifications** on the page and click **Enable**. You can click **Edit Settings** to change the email address.

----End

Customizing a Project Workflow

In the Sprint review meeting, the team demonstrates the product to the product owner and presents the test report. The product owner confirms whether the story is complete. However, the current story status does not show that the test is complete. Therefore, the tester suggests adding a status **Accepting**.

The project manager Maggie performs the following operations to add a status to a story.

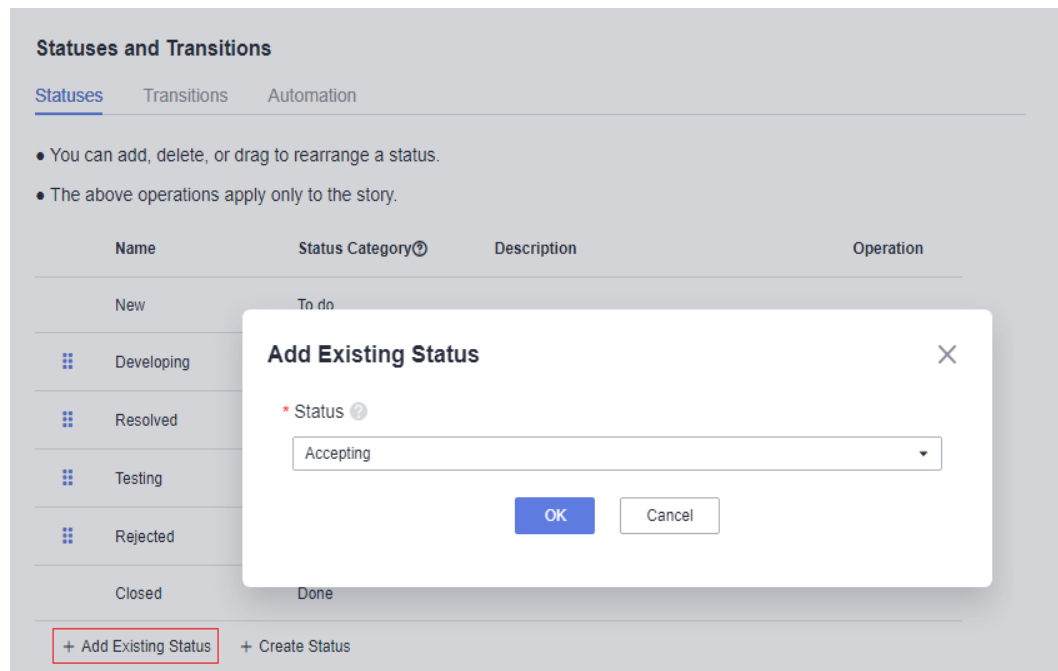
- Step 1** Go to the **Phoenix Mall** project, and choose **Settings > Project Settings** from the navigation pane.
- Step 2** Choose **Common Statuses** from the middle navigation pane. View the default work item statuses of the sample project.
- Step 3** Click **Add Status**. In the dialog box that is displayed, edit the status information by referring to [Table 1-10](#), and click **Add**.

Table 1-10 Status configurations

Configuration Item	Suggestion
Status	Enter Accepting .
Status Category	Select Doing .

- Step 4** Choose **Stories > Statuses and Transitions** from the middle navigation pane. View the default story statuses of the sample project.
- Step 5** Click **Add Existing Status**. In the displayed dialog box, select **Accepting** and click **OK**.

Figure 1-7 Adding a story status



Step 6 Drag and place **Accepting** below **Testing**.

Step 7 Verify the configuration result.

1. Choose **Work** from the left navigation pane, and click the **Work Items** tab.
2. Click any story name in the list to view story details.
3. Click the value in the **Status** column. In the drop-down list, you can see that the **Accepting** option is displayed.

----End

1.4.4 Step 3: Writing Code

CodeArts Repo provides Git-based online code management service, including code cloning/submission and branch management.

This section uses store network query as an example to describe how to manage and develop source code. Store network query is a high-priority story.

In this sample project, code is developed in a branch. Developer Chris creates a branch in a code repository, develops code, and submits a merge request. Project manager Maggie approves the request and then merges it to the master branch.

Using Branches

A branch is a tool for parallel feature development. Branches enable you to diverge from the main line of development and continue to do work without messing with that main line.

When a code repository is created, there is a default branch named **master**, that is, the main line. To ensure the stable running of Phoenix, a stable and continuously available master is required. The project manager suggests "function

branches + request merges". Each function branch must be reviewed by other members in the team before merge.


Step 1 Set master as a protected branch (this operation is performed by project manager Maggie in this document).

1. Go to the **Phoenix Mall** project, choose **Code > Repo** from the navigation pane, and locate the code repository **phoenix-sample**.
2. Go to the code repository by clicking its name and click the **Settings** tab. Choose **Policy Settings > Protected Branches** from the navigation pane.
3. Click **Create Protected Branch**. Complete the settings by referring to the following table and click **OK**.

Table 1-11 Creating a protected branch

Configuration Item	Suggestion
Branch	Select master .
Push	Configure this parameter as required. In this example, the default configurations are used.
Merge	Configure this parameter as required. In this example, the default configurations are used.
Members	Select Push and Merge as required, and select a member from the drop-down list. In this example, the default configurations are used.

 **NOTE**

If the protected branch **master** already exists on the page, click  and modify the branch configuration as required.

Step 2 Create a function branch (this operation is performed by developer Chris in this document).

1. Go to the **Phoenix** project. Choose **Code > Repo**. Find the **phoenix-sample** in the repository.
2. Click the repository name to access the code repository. On the **Code** tab, click **Branches**.
3. Click **Create** and set the branch information by referring to [Table 1-12](#), and click **OK**.

Table 1-12 Creating a branch

Configuration Item	Suggestion
Based On	Select master .
Branch	Enter Feature-Store .
Work Items to Associate	Select User can query network of all stores .

----End

Modifying and Committing Code

The store network query function is divided into frontend display and backend management tasks during **sprint planning**. This section uses **Frontend display - Add store network menu** as an example to describe how to modify and commit code.

Step 1 Choose **Work** from the navigation pane, and click the **Sprints** tab.


In Sprint4, find the task **Frontend display - Add store network menu**, and change the task status to **Developing**.

Step 2 Choose **Code > Repo**, and find the **phoenix-sample** repository.

Step 3 Go to the code repository by clicking its name and click the **Code** tab.

Step 4 Click **master** above the file list and select the branch **Feature-Store** from the drop-down list.

Step 5 Find **vote/templates/store-network.html** in the file list and open it.

Step 6 Click , add the store address specified in the story, enter the commit message **Store list added** in the text box at the bottom of the page, and click **OK**.

```
<ul>
  <li>Branch A: 123 meters to the departure floor, Terminal 1, Airport E</li>
  <li>Branch B: No. 456, Street G, Area F</li>
  <li>Branch C: No. 789, Street J, Area H</li>
  <li>Branch D: West side of Building K, Avenue L, Area K</li>
</ul>
```

Step 7 Open and edit the **/vote/templates/index.html** file in the same way.

Add the menu **Store Network** in line 179, enter the commit message **fix #xxxxxx Frontend display - Add store network menu**, and click **OK**.

In the preceding information, **#xxxxxx** indicates the ID of the task for **Frontend display - Add store network menu**, which is obtained from the work item list.

```
<li class="nav-item"> <a href="store-network" class="nav-link">Store network</a> </li>
```

Step 8 Choose **Work > Sprints** from the navigation pane. In Sprint 4, find the task **Frontend display - Add store network menu**.

- Click the task name. The task status automatically changes to **Resolved** on the details page.

- Click the **Associated** tab. Under **Code Commit Records**, a record is displayed with the commit message configured in the [previous step](#).

Figure 1-8 Code commit record

Branch	Commit Message	Committed By	Committed At
Feature-Store	2ef0caf6 - fix #2081315 Frontend Display-Add Store Network Menu	Chris	Apr 01, 2023 16:39:10 GMT+08:00

----End

Reviewing Code and Merging Branches

Step 1 A developer initiates a merge request.

After coding, developer Chris initiates a merge request to merge their function branch into the master.

1. Go to the code repository, click the **Merge Requests** tab, and click **New**.
2. Set the source branch to **Feature-Store** and the target branch to **master**, and click **Next**.
3. Edit the merge request details by referring to [Table 1-13](#).

Table 1-13 Merge request configurations

Configuration Item	Suggestion
Title	Enter Add store list .
Mergers	Click . In the dialog box that is displayed, select Maggie and click OK .
Approvers	Click . In the dialog box that is displayed, select Maggie and click OK .

4. Click **Create Merge Request** to complete the creation.

Step 2 The project manager reviews the code and merges the commit request.

In this document, project manager Maggie is both the reviewer and merger. Maggie reviews the request content and merges the request.

1. Go to the code repository, click the **Merge Requests** tab, and find the merge request created by developer Chris.
2. Click the request to view details.
3. Leave comments on the page. Click **Approve** in the **Approval Gate** to complete the approval.
4. Click **Merge** to merge the branch into **master**.

 NOTE

If you have selected **Delete source branch after merge** when creating a merge request, the branch Feature-Store will be deleted after the branch merge is complete.

----End

1.4.5 Step 4: Checking Code

CodeArts Check provides cloud-based code quality management, static code check (quality and style), security check, issue fixing suggestions, and trend analysis.

As Phoenix becomes increasingly larger, more issues have occurred, involving high fixing costs. However, no unified coding standards are available. The project manager suggests some basic standards, continuous static code scanning, and issue fixing within Sprints.

This section describes how developer Chris scans static code and identify and fix issues for different technology stacks.

Introduction to Preset Tasks

The sample project has four preset code check tasks.

Table 1-14 Preset tasks

Preset Task	Description
phoenix-codecheck-worker	Checks the task corresponding to the Worker function code.
phoenix-codecheck-result	Checks the task corresponding to the Result function code.
phoenix-codecheck-vote	Checks the task corresponding to the Vote function code.
phoenix-sample-javas	Checks the task corresponding to the JavaScript function code.

 NOTE

For details about Vote, Result, and Worker, see [Solution Architecture](#).

This section uses the **phoenix-codecheck-worker** task as an example.

Configuring and Executing a Task

For comprehensive checks, developers can add some simple configurations (for example, a Python check rule set) to the preset code check task.

Step 1 Edit a task.


1. Go to the **Phoenix** project and choose **Code > Check**. The preset four tasks are displayed.

2. Find the **phoenix-codecheck-worker** task in the list.
3. Click the task name to go to the details page and click the **Settings** tab.
4. In the navigation pane, choose **Rule Sets**. The default language of a rule set is Java.
5. Add the Python language check rule set.

- a. Click  next to **Languages Included** to refresh the language list.

 **NOTE**

If Python is displayed on the page, skip this step.

- b. Click  to enable the Python language.
- c. In the dialog box that is displayed, click **OK**.

Step 2 Execute the task.

1. Click **Start Check** to start the task.
2. If **Success** is displayed on the page, the task is successfully executed.
If the task fails, check and fix errors based on the message displayed on the page.

----End

Viewing the Code Check Result

CodeArts Check collects check results and provides fix suggestions for detected issues. Optimize the project code based on the suggestions.

Step 1 On the task details page, click the **Overview** tab to view the result statistics.

Step 2 Click the **Issues** tab to view the issue list.

Click **Help** in the question box to view fixing suggestions. You can find the corresponding file and code location in the code repository as required and optimize the code based on the fix suggestions.

----End

1.4.6 Step 5: Building an Application

CodeArts Build provides a hybrid language build platform with simple configurations, supports one-click task creation, configuration, and execution, and automates activities such as code obtaining, building, and packaging.

This section describes how Chris, a developer, builds environment images, compiles and packages code into software packages, and triggers automatic building through code changes to implement continuous integration.

Introduction to Preset Tasks

There are five build tasks preset in the sample project.

Table 1-15 Preset tasks

Preset Task	Description
phoenix-sample-ci	Basic build task
phoenix-sample-ci-test	Task for building available images in the test environment
phoenix-sample-ci-worker	Task for creating a Worker function image
phoenix-sample-ci-result	Task for creating a Result function image
phoenix-sample-ci-vote	Task for creating a Vote function image

 **NOTE**

For details about Vote, Result, and Worker, see [Solution Architecture](#).

This section uses the **phoenix-sample-ci** task as an example. The following table shows the steps involved in this task.

Table 1-16 Build Actions

Build Actions	Description
Create Vote image and push it to SWR	Find Dockerfile in the working directory ./vote and Dockerfile directory ./Dockerfile , create a Vote image based on Dockerfile , and push the image to SWR.
Create Result image and push it to SWR	Find ./Dockerfile in the working directory ./result and Dockerfile directory ./Dockerfile , create and push the Result image based on Dockerfile , and push the image to SWR.
Install Worker dependency using Maven	Use Maven to install the dependency required by the Worker function.
Create Worker image and push it to SWR	Find Dockerfile.j2 in the working directory ./worker and Dockerfile directory Dockerfile.j2 , create the Worker image based on Dockerfile , and push the image to SWR.
Generating Postgres and Redis Dockerfile	Run the shell command to generate Dockerfile for creating Postgres (database) and Redis (cache) images.
Create Postgres image and push it to SWR	Create a Postgres image based on Dockerfile generated in Create Postgres and Redis Dockerfile and push the image to SWR.

Build Actions	Description
Create Redis image and push it to SWR	Create a Redis image based on Dockerfile generated in Create Postgres and Redis Dockerfile and push the image to SWR.
Replace image of Docker-Compose deployment file	To ensure that the correct image can be pulled when the image is deployed on ECS, run the shell command to perform the following operations: Run the sed command to replace the parameters in the docker-compose-standalone.yml file with the dockerServer , dockerOrg , and BUILDNUMBER parameters of the build task in sequence. Run the tar command to compress the docker-compose-standalone.yml file into the docker-stack.tar.gz file. Compress the files required for deployment so that the files can be uploaded and archived in subsequent steps.
Replacing the Image Version of the Kubernetes Deployment File	To ensure that the correct image can be pulled when the image is deployed on CCE, run the shell command to perform the following operations: Run the sed command to replace the docker-server and docker-org parameters in all the files whose names end with deployment in the kompose directory with the dockerServer and dockerOrg parameters of the build task. Run the sed command to replace image-version in the result-deployment.yaml , vote-deployment.yaml , and worker-deployment.yaml files with BUILDNUMBER .
Uploading the Kubernetes Deployment File to the Software Release Library	Upload all .yaml files to the software release library for archiving.
Uploading the docker-compose Deployment File to the Software Release Library	Upload the compressed docker-stack.tar.gz (build package directory) to the software release repo for archiving. Name the package docker-stack to manage versions of the software package.

 NOTE

During project deployment, failures often occur due to environment inconsistency. For example, after the JDK in the R&D debugging environment is upgraded, the JDK is not marked in the environment list. As a result, the production environment is not upgraded, causing failures. To avoid problems caused by environment inconsistency, Docker is used to package microservice applications and environments into images to ensure that each environment (development and commissioning environment, test environment, QA environment, and production environment) is consistent.

Configuring SWR

In this example, environment images are stored in SWR. Configure SWR first.

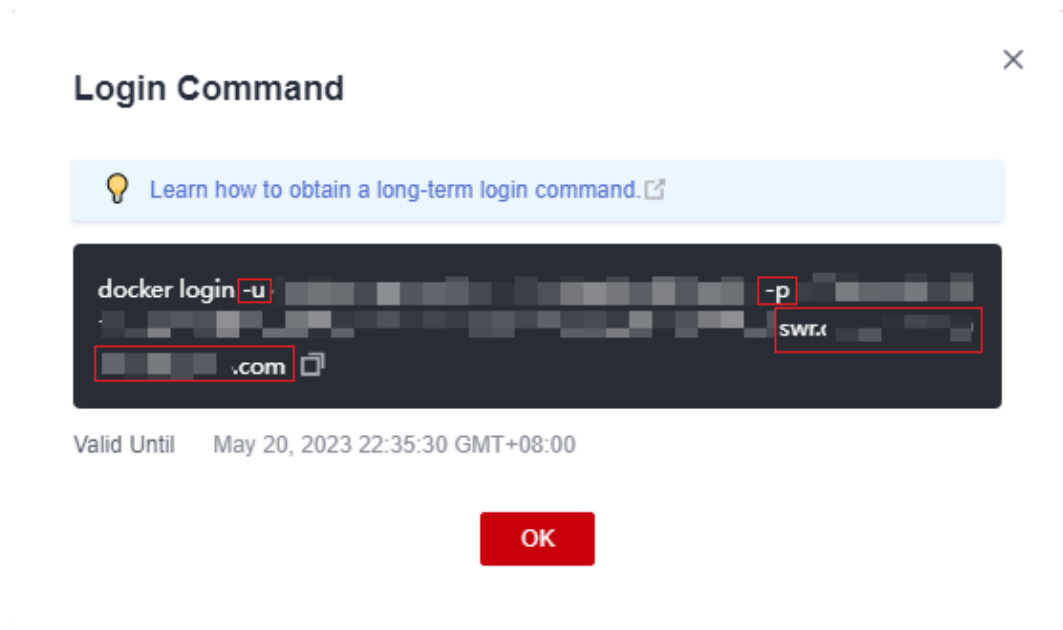
Step 1 Log in to the SWR console.

Check the region list in the upper left corner of the page and ensure that the region is the same as the region where the build task is compiled. If the regions are different, select the same region.

Step 2 Click **Generate Login Command**. The login command is displayed in a dialog box, where,

- The character string following **-u** is the user name.
- The character string following **-p** is the password.
- The last character string is the address of the SWR server, which is the value of **dockerServer** for configuring and executing tasks later.

Figure 1-9 Generating a login command



NOTE

The login instruction generated here is a temporary login instruction with a validity period of 24 hours. If a long-term valid login command is required, see [Obtaining a Long-Term Valid Login Command](#).

Step 3 Click **Create Organization**. In the displayed dialog box, enter **phoenix** as the organization name and click **OK**. (The organization name must be globally unique. If a message is displayed indicating that the organization already exists, enter another name.)

The organization name is the value of **dockerOrg**, which will be used for configuring and executing tasks later.

----End

Configuring and Executing a Task

Step 1 Configure a task.

1. Go to the **Phoenix Mall** project and choose **CICD > Build**. The built-in build task of the sample project is displayed on the page.
2. Find the **phoenix-sample-ci** task in the list. Click ******* and choose **Edit** from the drop-down list.
3. Click the **Parameters** tab, and set parameters by referring to [Table 1-17](#).



Table 1-17 Setting Parameters

Name	Default Value
codeBranch	master
dockerOrg	Organization created in SWR. For this example, enter phoenix .
version	1.0.0
dockerServer	SWR server address obtained from SWR.

NOTE

Ensure that the values of **dockerOrg** and **dockerServer** are correct. Otherwise, the task fails.

Step 2 Click **Save and Run**. In the dialog box that is displayed, click **OK** and start the build task.

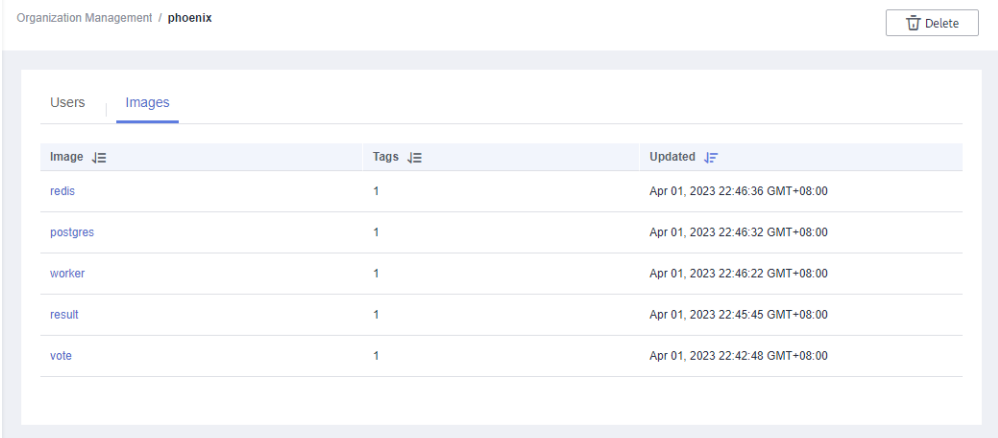
When  is displayed, the task is successfully executed. Record the character string starting with # (for example,  **#20230401.1**).

If the build fails, rectify the fault based on the failed action information and error information in logs.

Step 3 Check the release file.

1. Choose **Artifact** in the navigation pane and click the **Release Repos** tab.
2. In the repository named after the project, you can find the **docker-stack** and **phoenix-sample-ci** folders.
 - In the **docker-stack** folder, you can find the folder named after the character string recorded in [Step 2](#). In this folder, you can find the release file **docker-stack.tar.gz**.
 - In the **phoenix-sample-ci/1.0.0** folder, you can find 10 archived **.yaml** files.
3. Go to SWR. In the navigation pane, choose **Organizations**, and click the organization with the same name as the value of the build task parameter **dockerOrg**.

Click the **Images** tab. You can find five images (redis, postgres, worker, result and vote) in the list.

Figure 1-10 Viewing images

Organization Management / phoenix Delete

Users | Images

Image	Tags	Updated
redis	1	Apr 01, 2023 22:46:36 GMT+08:00
postgres	1	Apr 01, 2023 22:46:32 GMT+08:00
worker	1	Apr 01, 2023 22:46:22 GMT+08:00
result	1	Apr 01, 2023 22:45:45 GMT+08:00
vote	1	Apr 01, 2023 22:42:48 GMT+08:00

4. Click the names of the five images in sequence to go to the image details page. View the image version on the **Tags** tab page.
 - The Redis image version is alpine.
 - The image version of postgres is 9.4.
 - The image versions of worker, result, and vote are the same as those recorded in [Step 2](#).

----End

Setting Code Submission to Trigger Automatic Compilation

After the following configuration, the application build task can be automatically triggered after the code is changed for continuous project integration.

Step 1 On the details page of the task **phoenix-sample-ci**, click **Modify** in the upper right corner of the page.

Step 2 Click the **Schedule** tab.

Step 3 Enable **Run upon Code Commit**. When is displayed, click **Save**.

The default value of the codeBranch parameter on the parameter setting page is **master**. Therefore, the build is automatically triggered when the master code changes.


Step 4 Verify the configuration result: Modify the project code and submit it to master to check whether the build task is automatically executed.

----End

Configuring Scheduled Execution

To prevent a code bug from entering the production environment and ensure that the application is always in a deployable state, the team recommends continuous verification of the application.

You can perform the following operations to execute the build task at a scheduled time:

- Step 1** On the details page of the task **phoenix-sample-ci**, click **Modify** in the upper right corner of the page.
 - Step 2** Click the **Schedule** tab.
 - Step 3** Enable **Scheduled Execution**. After  is displayed, and click **Save**.
In this document, select **All**. The execution time is **12:00**.
 - Step 4** Verify the configuration result: Check whether the build task is automatically executed at the configured time. We will not elaborate on the details in this section.
- End

1.4.7 Step 6: Deploying an Application (CCE)

CodeArts Deploy provides visualized and automated deployment. Various deployment actions are provided to help you formulate a standard deployment process, reduce deployment costs, and improve release efficiency.

To deliver software more quickly and stably, the development team needs some self-service deployment service capabilities to reduce some subsequent maintenance work.

This section describes how Chris deploys a release package on CCE. For details about ECS-based deployment, see [Step 6: Deploying an Application \(ECS\)](#).

Preset Applications

There are three deployment applications preset in the sample project.

Table 1-18 Presetting applications

Presetting Applications	Application Description
phoenix-cd-cce	Application deployed on CCE
phoenix-sample-standalone	Application deployed on ECS
phoenix-sample-predeploy	Application for which you install dependency tools on ECS

This section uses the **phoenix-cd-cce** application as an example.

Buying and Configuring CCE

In this document, Cloud Container Engine (CCE) is used.

[Buy a CCE cluster](#) on the console.

For details about the mandatory configurations of clusters and nodes, see [Table 1-19](#) and [Table 1-20](#). You can select the configurations that are not listed in the table based on the site requirements.

Table 1-19 Buying a CCE cluster

Category	Configuration Item	Suggestion
Basic Settings	Billing Mode	Select Pay-per-use .
	Cluster Version	Select a version as required. You are advised to select the latest version.
Network Settings	Network Model	Select Tunnel network .
	VPC	Select an existing VPC. If no proper VPC is available in the list, click Create VPC .
	Master Node Subnet	Select an existing subnet. If no proper subnet is available in the list, click Create Subnet .
	Container CIDR Block	Click Auto select .

Table 1-20 Configuring a node

Category	Configuration Item	Suggestion
Compute Settings	Billing Mode	Select Pay-per-use .
	Node Type	Select Elastic Cloud Server (VM) .
	Specifications	Select General-purpose with 2 vCPUs and 8 GiB memory or higher.
	Container Engine	Select Docker .
	OS	Click Public image and select an Euler image.
	Node Name	Enter a custom name.
	Login Mode	Select Password .

Category	Configuration Item	Suggestion
	Password	Enter a password.
Network Settings	Node IP	Select Random .
	EIP	Select Auto create .

Configuring and Executing an Application

Deploy the `.yaml` files generated in [Step 5: Building an Application](#) in the CCE cluster one by one.

Step 1 Configure the application.

1. Go to the **Phoenix Mall** project and choose **CICD > Deploy**. The built-in deployment applications of the sample project are displayed on the page.
2. Find application **phoenix-cd-cce**. Click ******* and choose **Edit** from the drop-down list.
3. On the **Deployment Actions** tab page, complete the following configurations in each action.

Table 1-21 Configuring deployment actions

Configuration Item	Suggestion
Cluster Name	Use the cluster name set when buying a CCE cluster.
Namespace	In this document, select default .

4. Click the **Parameters** tab and set parameters.


Table 1-22 Parameters


Parameter	Example Value
ci_task_name	Enter phoenix-sample-ci .
version	Use the value of version of the phoenix-sample-ci task.

5. Click **Save**.

Step 2 Go to the CCE console. Locate the target cluster, click , click the **Deployments** tab, and verify that no record exists in the list.

If there are records in the list, select all records, click **Delete**, select all resource release options, and click **Yes** to clear the records in the list.

Step 3 Return to the application list page, click  in the row of the **phoenix-cd-cce** application, and click **OK** in the dialog box that is displayed to start deployment.

If  is displayed on the page, the deployment is successful. If the deployment fails, rectify the fault based on the failed action information and error information in logs.

Step 4 Verify the deployment result.


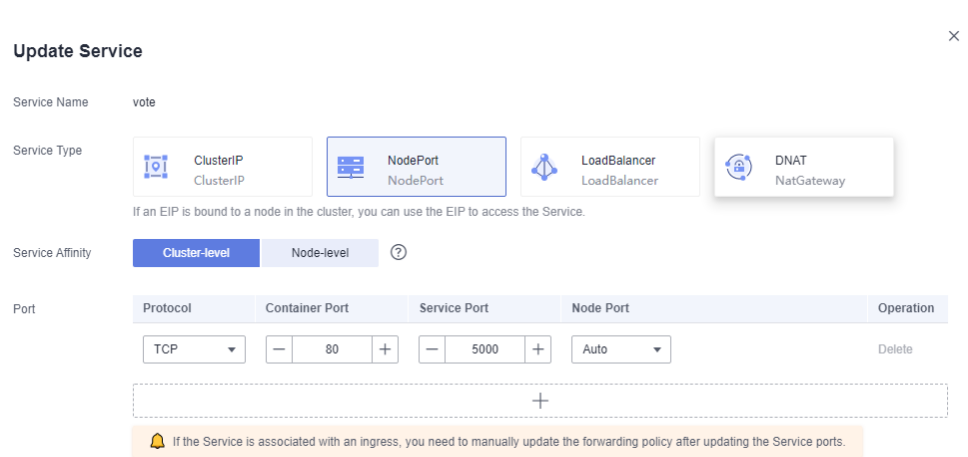
1. Go to the CCE console.
2. Locate the target cluster, click , and click the **Deployments** tab.
Five records are displayed on the page. All the records are in the **Running** state.
3. Click **vote** to go to the details page. On the **Access Mode** tab page, choose **More > Update**.
Set the parameters by referring to [Table 1-23](#), and click **OK**.

Table 1-23 Updating a service

Parameter	Example Value
Service Type	Select NodePort .
Service Affinity	Select Cluster-level .
Service Port	Enter 5000 .
Container Port	Enter 80 .
Node Port	Set the port number based on the site requirements. In this document, this parameter is set to Auto .

Figure 1-11 Updating a service



4. Return to the list. Record the port number with a dotted line below the text in the **Port/Protocol** column.

Figure 1-12 Workload access mode



5. Open a new browser page and enter **http://ip: port number** in the address box. The page is displayed successfully.
If **ip** is set to the elastic IP address bound to the node in **Buying and Configuring CCE**. The port number is the port number recorded in **Step 4.4**.
6. Return to the **Deployments** page and update **result** (the service port is **5001**) by referring to **Step 4.3**.
After the creation is successful, enter the node IP address and service port number in the address box of a new browser. The page shows a success message.

----End

1.4.8 Step 6: Deploying an Application (ECS)

This section uses **phoenix-sample-standalone** as an example to describe how to deploy the release package to a host. For CCE-based deployment details, see **Step 6: Deploying an Application (CCE)**.

Purchasing and Configuring an ECS

Elastic Cloud Server (ECS) is used in this document. You can also use your own Linux host (running Ubuntu 16.04 OS).

Step 1 Buy an ECS.

The following table lists mandatory configurations. You can also select other configurations as you need.

Table 1-24 Configuring an ECS purchase

Category	Configur ation Item	Suggestion
Configure Basic Settings	Billing Mode	Select Pay-per-use .
	CPU Architect ure	Select x86 . If this configuration item does not exist, ignore it.
	Specificat ions	Select General computing with 2 vCPUs and 8 GiB memory or higher.
	Image	Choose Public image > Ubuntu > Ubuntu > 16.04 Server 64bit .
Configure Network	EIP	Select Auto assign .

Category	Configuration Item	Suggestion
	Bandwidth Size	Select Bandwidth .
Configure Advanced Settings	Login Mode	Select Password .
	Password	Enter a custom password.

Step 2 Configure security group rules.

Use ports 5000 and 5001 to verify the sample project. Therefore, add an inbound rule that allows access to ports 5000 and 5001.

The procedure is as follows:

1. Log in to the ECS list page, locate the ECS purchased in step [Step 1](#), and click the ECS name.
2. Click the **Security Group** tab and add an inbound rule in which **Protocol** is set to **TCP** and **Port Range** is set to **5000-5001** by referring to [Configuring Security Group Rules](#).

----End

Adding a Target Host to the Project

Before deploying applications to ECSs, add the target hosts to the basic resources of the project.

Step 1 Go to the **Phoenix Mall** project and choose **Settings > General > Basic Resources** from the navigation pane.

Step 2 Click **Create Host Cluster**, enter the name **hosts**, set the OS to **Linux**, and click **Save**.

Step 3 Click **Add Target Host**. In the dialog box that is displayed, configure the following information, agree to the statement, and click **OK**.

Table 1-25 Adding a host

Configuration Item	Suggestion
Host Name	Enter a custom host name. For easy identification, use the name set when buying an ECS.
IP	Enter the EIP generated when buying the ECS.
Username	Enter root .
Password	Enter the password set when buying the ECS.

Configuration Item	Suggestion
SSH Port	Enter 22 .

Step 4 A host record is displayed on the page. If **Succeed** is displayed in the **Verification Result** column, the host is added successfully.

If the host fails to be added, check the host configuration based on the failure details.

----End

Installing Dependency Tools on ECS

The sample program depends on Docker and Docker-Compose, which must be installed on the target ECS.

Step 1 Go to the **Phoenix Mall** project, choose **CICD > Deploy**, and find the **phoenix-sample-predeploy** application in the list.

Step 2 Click ******* and choose **Edit** from the drop-down list.

Step 3 Click the **Environment Management** tab and configure the host environment.

1. Click **Create Environment**, enter the environment name **phoenix-hostgroup**, set **Resource Type** to **Hosts** and **OS** to **Linux**, and click **Save**.
2. After a new environment record is added to the list, click the environment name. In the window that is displayed, click the **Resources** tab.
3. Click **Import Host**. In the dialog box that is displayed, select the created host cluster from the drop-down list, select the host from the list, and click **Import**.

NOTE

If you do not have permission to create environments, contact the administrator to grant you permissions on the permission management page of the application.

Step 4 On the **Deployment Actions** tab page, edit the actions of the application.

Click **Install Docker** and select **phoenix-hostgroup** from the **Environment** drop-down list. If a dialog box is displayed confirming whether you want to change the environment of the subsequent actions to **phoenix-hostgroup**, click **OK**.

Step 5 Click **Save & Deploy** to start the deployment task.

If a message is displayed indicating successful deployment, the task is successfully executed.

Step 6 Log in to the ECS and run the following commands to check whether the dependency tools are successfully installed:

- Check the Docker image version.
`docker -v`
- Check the Docker-Compose version.
`docker-compose -v`

If information similar to that in **Figure 1-13** is displayed, the installation is successful.

Figure 1-13 Checking the Docker and Docker-Compose versions

```
root@ecs-he2e:~# docker -v
Docker version 18.09.0, build 4d60db4
root@ecs-he2e:~# docker-compose -v
docker-compose version 1.17.1, build 6d101fb
root@ecs-he2e:~#
```

----End

Configuring and Executing an Application

During deployment, configure the ECS in the environment list of the application and set the build task **phoenix-sample-ci** as the deployment source.

- Step 1** Go to the **Phoenix Mall** project, choose **CICD > Deploy**, and find the **phoenix-sample-standalone** application in the list.
- Step 2** Click ******* and choose **Edit** from the drop-down list.
- Step 3** Click the **Environment Management** tab and configure the host environment.
 1. Click **Create Environment**, enter the environment name **phoenix-hostgroup**, set **Resource Type** to **Hosts** and **OS** to **Linux**, and click **Save**.
 2. After a new environment record is added to the list, click the environment name. In the window that is displayed, click the **Resources** tab.
 3. Click **Import Host**. In the dialog box that is displayed, select the created host cluster from the drop-down list, select the host from the list, and click **Import**.
 4. Close the window after a message is displayed, indicating that the import is successful.
- Step 4** On the **Deployment Actions** tab page, edit the actions of the application.
 1. Click **Select Deployment Source**. Set the deployment source by referring to [Table 1-26](#).

Table 1-26 Configuring the deployment source

Configuration Item	Suggestion
Source	Select Build task
Build Task	Select phoenix-sample-ci .

- 2. Retain the default settings in actions **Decompress File** and **Run Shell Commands**.
- Step 5** Click the **Parameters** tab and set parameters based on the SWR login command.

You can obtain the login command from the console. For details, see [Configuring SWR](#).
- Step 6** Click **Save & Deploy** and start deployment.

If a message is displayed indicating that the deployment is successful, continue with the next step. If the deployment fails, rectify the fault based on the failed action information and error information in logs.

Step 7 Verify the deployment result.

Open a browser, enter **http://ECS IP address:5000** in the address box, and press **Enter**.

You can view the **Store network** menu in the navigation bar.

Enter **http://ECS IP address:5001**. The page displays a success message.

----End

1.4.9 Step 7: Managing Project Testing

CodeArts TestPlan provides a one-stop cloud test platform that integrates DevOps agile test concepts, helping efficiently manage test activities and ensure high-quality product delivery.

This section describes how Billy manages the test period of a project, including creating and executing test cases and tracking the test progress.

Creating a Sprint Test Plan

After the requirements (stories) to be implemented in Sprint 4 are determined (that is, [Step 1: Managing Project Plans](#) is complete), a tester can write test cases when developing code.

Step 1 Create a test plan.

1. Go to the **Phoenix Mall** project and choose **Testing > Testing Plan**.
2. Click **Create** and configure test plan information.
 - a. Basic information: Configure the following information and click **Next**.

Table 1-27 Basic information about a test plan

Sub-Configuration Item	Suggestion
Name	Enter Sprint 4 .
Processor	Select Billy .
Plan Period	You are advised to use the same period as Sprint 4 created in Req.
Associated Sprint	Select Sprint 4 .

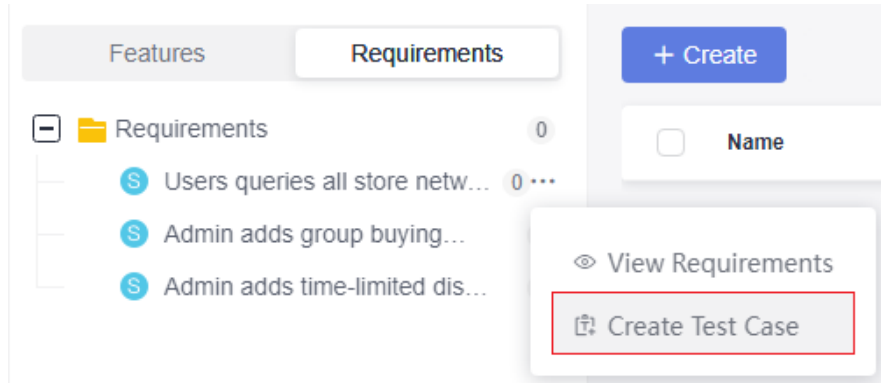
- b. Advanced configuration: Select **Manual Test**. Confirm that the requirements in the list are the same as those of **Sprint 4** in Req, and then click **Save and Use**.

- Return to the Testing Plan page. You can find the newly created test plan **Sprint 4** in the list. The status of the test plan is **New**.

Step 2 Design test cases.

- In the test plan **Sprint 4**, click **Design**.
- Expand the **Requirements** on the left of the page and find the Story **User can query network of all stores**.
Click **...** and choose **Create Test Case**.

Figure 1-14 Creating a test case



- Enter **Store Network Query**, edit Test Procedure and Expected Results by referring to [Table 1-28](#), and click **Save**.

Table 1-28 Test steps

Steps	Expected Results
Opening the homepage of Phoenix Mall	The page is displayed.
Clicking the Store Network menu	The Store Network page is displayed. Province filtering is available on the page, and information about recommended stores is displayed at the bottom of the page.
Selecting City A	The store information list of city A is displayed.

- Create test cases for the other two stories in the same way.
- Choose Navigation **Testing > Testing Plan** to return to the test plan list. In the list, the status of the test plan **Sprint4** is **Designing**.

----End

Executing a Test Plan

After developing story codes and deploying applications on the test environment (completing [Step 6: Deploying an Application \(CCE\)](#) or [Step 6: Deploying an Application \(ECS\)](#)), developers can set the story status to **Resolved** and set the story handler to the tester.

In this case, the test personnel can execute the test cases corresponding to the story.


This section uses the store network query function as an example to describe how to execute test cases and how to report bug when test case execution fails.

Step 1 In the **Phoenix Mall** project, click Navigation **Work > Sprint**.

In Sprint 4, find the story **User can query network of all stores** and change the story status to **Testing**.

Step 2 Go to the **Testing > Testing Case** page and select **Sprint 4** in the upper part of the page.

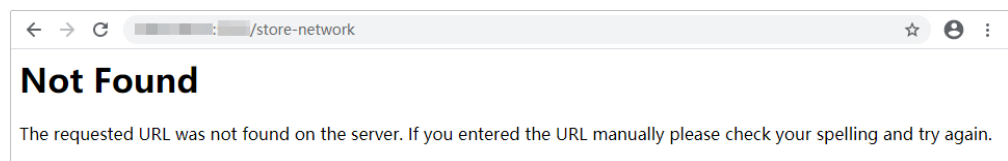
Step 3 In the list, click the case **Store Network Query**, change the status to **Testing**, and click **Save**.

Step 4 Click the **Manual Test** tab and click  in the row where **Store Network Query** is located. The **Execute** window is displayed on the right.

Step 5 Execute the steps one by one in the test environment.

- If the operation is successful, go to **Step 6** to continue the operation.
- The execution fails. For example, when you perform step 2, the page fails to be redirected and 404 is displayed. The system switches to **Step 7** to continue the operation.

Figure 1-15 Page display failure



Step 6 Return to the test case execution window and record the execution result.

1. In the table, set Actual Result of all steps to **Successful**.
2. In the upper part of the table, set the test case result to **Successful**.
3. Select **set the case status to Completed**.
4. Click **Save** in the upper right corner of the page.

Figure 1-16 Successful test case execution

Execute: Store Network Query

Save 4 ⋮ 🗑️ ✕

No prerequisites.

Setting Result

* Expected Result

Successful 2 Set the case status to Completed. 3

No.	Step	Expected Result	Actual Result
1	Open the home page of Phoenix Mall.	The page is properly displayed.	Successful 2 Click to enter the actual result.
2	Click Store Network.	The Store Network page is displayed. Province filtering is available on the page, and information about recommended stores is displayed at the bottom of the page.	Successful 2 Click to enter the actual result. 1
3	Set Province to A City.	The store information list in A City is displayed.	Successful 2 Click to enter the actual result.

The status of the test case automatically changes to **Completed**. Go to [Step 13](#) to continue the operation.

Step 7 Return to the test case execution window and record the execution result.

1. In the table, set the actual result of step 1 to **Successful**.
2. In the table, set the actual result of step 2 to **Failed** and enter the content **Redirection failure. 404 is displayed on the page**.
3. In the upper part of the table, set the test case result to **Failed**.
4. Click **Save** in the upper right corner of the page.

Figure 1-17 Failed test case execution

Execute: Store Network Query

Save ⋮ 🗑️ ✕

No prerequisites.

Setting Result

* Expected Result

Failed 3 Set the case status to Completed.

No.	Step	Expected Result	Actual Result
1	Open the home page of Phoenix Mall.	The page is properly displayed.	Successful 1 Click to enter the actual result.
2	Click Store Network.	The Store Network page is displayed. Province filtering is available on the page, and information about recommended stores is displayed at the bottom of the page.	Failed 2 Redirection failure. 404 is displayed on the page.
3	Set Province to A City.	The store information list in A City is displayed.	--Select-- Click to enter the actual result.


- Step 8** Click  in the upper right corner of the page and choose **Create Defect**. The **Create work item** page is displayed.
- Step 9** At the end of the text box in the lower left corner of the page, you can see the reproduction procedure for automatically set in the defect.
Edit defect details by referring to [Table 1-29](#) and click **Save**. The work item list page is displayed.

Table 1-29 Defect Details Configuration

Configuratio n Item	Suggestion
Title	Enter Store Network 404 .
Assigned To	Select Chris .
Sprint	Select Sprint 4 .

- Step 10** In the work item list, find Bug **Store Network 404**. Click the name and click the **Associated** tab. You can find the test case **Store Network Query** by expanding **Associate with Test Case**.
- Step 11** You can click the ID of an associated case to go to the case details page.
Click the **Defect List** tab. A defect record is displayed, that is, the defect created in [Step 9](#).
- Step 12** After the development personnel fix the defect and verify the defect, set the case result by referring to [Step 6](#) and set the corresponding defect status to **Closed**.
- Step 13** Execute other test cases.
- Step 14** When the status of all cases is **Completed**, choose **Testing > Testing Plan** to return to the test plan list where you can find the status of Sprint 4 of the test plan is **Completed**.

----End

Track test plans.

- View a quality report.
Through the quality report, the team can intuitively view the current progress of the test plan, including the requirement coverage rate, defects, case pass rate, and case completion rate.
On the **Testing > Testing Plan** page, click **Report** in the card of **Sprint 4** to view the quality report.
- Customize reports.
In addition to built-in quality reports, teams can customize statistical reports as needed.
The following describes how to customize a statistical report by taking test case execution statistics as an example.

- a. On the **Quality Report** page, click in the blank area in the lower part of the page and click **Add Report**. In the dialog box that is displayed, select **Custom Report**.
- b. Edit the report information by referring to [Table 1-30](#) and click **Save**.

Table 1-30 Report configuration

Configurati on Item	Suggestion
Report Title	Enter Test Case Execution Statistics .
Data Type	Select Test Cases .
Analysis Dimension	Select Result .

- c. The **Quality Report** page is displayed, and you can find the new report at the bottom of the page.

1.4.10 Step 8: Configuring a Pipeline for CD

CodeArts Pipeline provides a visualized and customizable software pipeline for automatic delivery and supports multiple task types, such as code check, build, and deployment tasks.

As the project progresses, each stage (build, release, and deployment) becomes more and more standardized. However, each stage is relatively independent and is semi-finished and cannot deliver business value directly. Only by effectively connecting each stage to form a complete continuous delivery (CD) pipeline can we truly improve the efficiency and quality of software release and continuously create business value.

This section describes how Chris, a developer, connects code check, building, and deployment for CD.

Introduction to Preset Pipelines

There are five pipeline tasks preset in the sample project. You can view and use them as needed.

Table 1-31 Preset pipeline tasks

Preset Pipeline Task	Description
phoenix-workflow	A basic pipeline task
phoenix-workflow-test	Pipeline task corresponding to the test environment
phoenix-workflow-work	Pipeline task corresponding to the Worker function

Preset Pipeline Task	Description
phoenix-workflow-result	Pipeline task corresponding to the Result function
phoenix-workflow-vote	Pipeline task corresponding to the Vote function

 NOTE

- For details about Vote, Result, and Worker, see [Solution Architecture](#).

Configuring and Executing a Pipeline

A pipeline usually consists of multiple stages. You can add multiple jobs to each stage.

Step 1 Configure a pipeline.



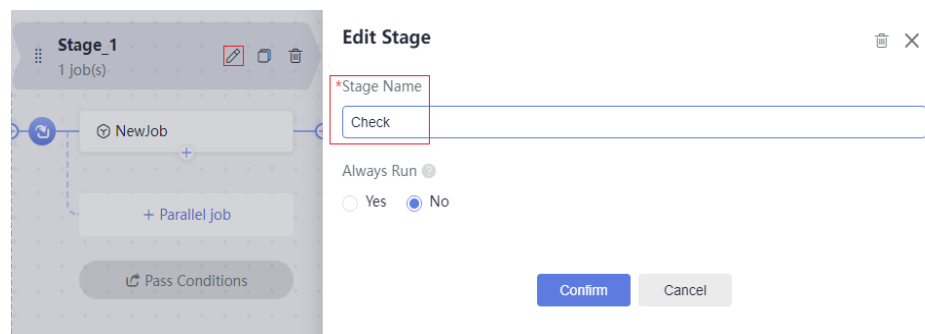
1. Go to the **Phoenix Mall** project and choose **CICD > Pipeline**.
2. Find pipeline **phoenix-workflow**. Click ******* and click **Edit**.
3. Add a code check stage.
 - a. Click  between the pipeline source and build to add a stage.
 - b. Click  next to **Stage_1**. In the **Edit Stage** window, enter the stage name **Check** and click **Confirm**.

Figure 1-18 Editing the stage name



- c. Click **NewJob**.
In the **NewJob** window, click **Add** next to the **Check** extension.
- d. Select the **phoenix-codecheck-worker** task and click **OK**.

 NOTE

The check task has three modes. This procedure uses the default mode **Full**. You can change the mode as required.

- **Full**: All files in the code repository are scanned.
- **Incremental (last commit)**: Incremental check is performed based on the latest commit file.
- **Incremental (last success)**: Incremental check is performed based on the changed files since the latest access control was passed.


4. Configure a deployment task.

Click the deployment task name, select the associated build task **phoenix-sample-ci**, and check the values of configuration items.


- The configurations of task **phoenix-sample-standalone** must be the same as those on the **Parameters** page of the task with the same name in Deploy.
- The configurations of task **phoenix-cd-cce** must be the same as those on the **Parameters** page of the task with the same name in Deploy.


 NOTE

Two deployment tasks are added in this example. If you selected only one deployment mode in preceding steps, keep the corresponding task and delete the other one.

Step 2 Go to the CCE console. Locate the target cluster, click , click the **Deployments** tab, and verify that no record exists in the list.

If there are records in the list, select all records, click **Delete**, select all options, and click **Yes** to clear the records in the list.

Step 3 Return to the pipeline list page. Click  in the row where **phoenix-sample-pipeline** is located, and click **Run** in the window that is displayed to start the pipeline.


If  is displayed on the page, the task is successfully executed.

If the task fails to be executed, check the failure cause in the failed task. You can open the step details page to view the task logs and rectify the faults based on the logs.

----End

Configuring Pass Conditions

To control the code quality, the code must be scanned and the number of errors must be within a reasonable range before being released. By adding quality access control, you can effectively automate the control process.

Step 1 On the **phoenix-sample-pipeline** details page, click  in the upper right corner and choose **Edit** from the drop-down list.

Step 2 In the **Check** stage, click **Pass Conditions**.

Step 3 In the **Pass Conditions** dialog box, click **Add** next to **Pass-Conditions-of-Standard-Policies**.

Step 4 Select **System Policy** and click **OK**.

Step 5 Click **Save and Run**.

If the number of check issues does not reach the threshold, the pipeline task fails to be executed.

 **NOTE**

For details about how to manage pass conditions, see [Rules and Policies](#).

----End

Configuring Code Changes to Automatically Trigger Pipelines

Through the following configuration, code changes can automatically trigger pipeline execution, implementing continuous project delivery.

Step 1 On the **phoenix-sample-pipeline** details page, click **in the upper right corner to edit**.

Step 2 Click the **Schedule** tab, select **Run upon Code Commit** in the Event Triggering Directory area, select master from the Branch Filter drop-down list box, and click **Save**.

Step 3 Verify the configuration result: Modify the code and push it to the master branch to check whether the pipeline task is automatically executed.

----End

1.4.11 Releasing Resources

To avoid unnecessary expenses, Sarah can release the following resources after completing the sample project experience.

NOTICE

Released resources cannot be recovered. Exercise caution when performing these operations.

Deleting a Project

A project is the basis for using CodeArts services. Deleting a project will delete all data (including work items, files, code repositories, software packages, and build task) in the project.

Step 1 Choose **Settings > General > Basic Information**.

Step 2 In the displayed dialog box, enter the project name and click **Delete Project**.

----End

Deleting Hosts

When purchasing an ECS, configure EIPs and EVS disks. When deleting an ECS, you need to delete the EIP and the disks attached to the ECS.

Step 1 Log in to ECS.


Step 2 Locate the ECS to be deleted in the list, click **More** and choose **Delete** from the drop-down list.

Step 3 In the dialog box that is displayed, select all options and click **Yes**.

----End

Deleting a Cluster

Step 1 Log in to CCE.

Step 2 Find the cluster to be deleted and click .

Step 3 In the dialog box that is displayed, select all options and click **OK**.

----End

Deleting Organizations and Images

Step 1 Delete organizations and images.

1. Log in to SWR.
2. On the **My Images** page, select the images created in this example, click **Delete**, and then confirm the deletion as prompted.
3. On the **Organizations** page, click the name of the organization to be deleted, click **Delete**, and then confirm the deletion as prompted.

----End