Cloud Service Engine

API Reference

Issue 01

Date 2023-11-07

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

1 Before You Start

1.1 Overview

Cloud Service Engine (CSE) is a cloud middleware used for microservice applications. It supports ServiceComb engines contributed to Apache and open-source enhanced Nacos engines. You can also use other cloud services to quickly build a cloud-native microservice system, implementing quick development and high-availability O&M of microservice applications.

1.2 API Calling

Cloud Service Engine (CSE) supports Representational State Transfer (REST) APIs, allowing you to call APIs using HTTPS. For details about API calling, see **Calling APIs**.

1.3 Service Center Address

The service center address is the request address for calling APIs. Different microservice engine instances have different service center addresses. For details, see **Obtaining the Service Center Address of a ServiceComb Engine**.

1.4 Constraints

- The number of resources that you can create is determined by your quota. For details, **Quotas**.
- For more constraints, see API description.

1.5 Concepts

- Account

  An account is created upon successful registration with the cloud platform. The account has full access permissions for all of its cloud services and

resources. It can be used to reset user passwords and grant user permissions. The account is a payment entity and should not be used directly to perform routine management. For security purposes, create Identity and Access Management (IAM) users and grant them permissions for routine management.

- User

  An IAM user is created using an account to use cloud services. Each IAM user has their own identity credentials (password and access keys).

  To view an account ID and user ID, go to the **My Credentials** page. The account name, username, and password will be required for API authentication.

- Region

  Regions are divided based on geographical location and network latency. Public services, such as Elastic Cloud Server (ECS), Elastic Volume Service (EVS), Object Storage Service (OBS), Virtual Private Cloud (VPC), Elastic IP (EIP), and Image Management Service (IMS), are shared within the same region. Regions are classified into universal and dedicated regions. A universal region provides universal cloud services for common tenants. A dedicated region provides specific services for specific tenants.

  For details, see **Region and AZ**.

- Availability zone

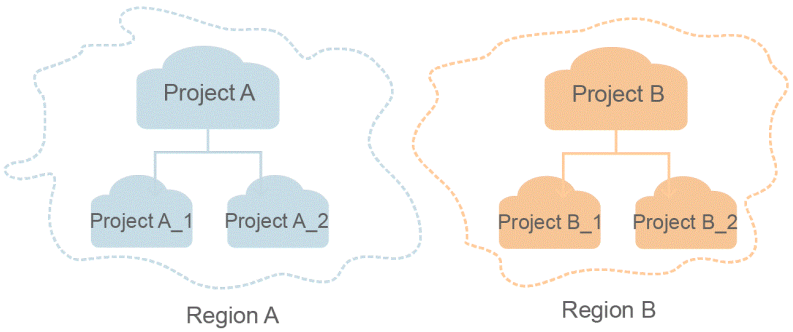
  An availability zone (AZ) comprises of one or more physical data centers equipped with independent ventilation, fire, water, and electricity facilities. Compute, network, storage, and other resources in an AZ are logically divided into multiple clusters. AZs within a region are connected using high-speed optical fibers to support cross-AZ high-availability systems.

- Project

  Projects group and isolate resources (including compute, storage, and network resources) across physical regions. A default project is provided for each region, and subprojects can be created under each default project. Users can be granted permissions to access all resources in a specific project. If you need more refined access control, create subprojects under a default project and create resources in subprojects. Then you can assign users the permissions required to access only the resources in the specific subprojects.

  To view a project ID, go to the **My Credentials** page.

Figure 1-1 Project isolation model



- Enterprise project

  Enterprise projects group and manage resources across regions. Resources in different enterprise projects are logically isolated. An enterprise project can contain resources of multiple regions, and resources can be added to or removed from enterprise projects.

  For details about how to obtain the enterprise project ID and features, see ***Enterprise Management User Guide***.

2 API Overview

CSE provides open APIs for microservice development, helping you quickly migrate services to the cloud at low costs and enabling efficient running of microservice applications.

For details about the APIs, see the following table.

Table 2-1 Overview

Type	Subtype	Description
CSE APIs	**Dynamic Configuration**	APIs related to configuration import to and export from the configuration management center.
	Engine Management	APIs related to microservice engine creation, query, and deletion.
	Microservice Governance	APIs related to governance policy creation, modification, query, and deletion.
Nacos APIs	**Nacos API**	APIs related to namespace creation, modification, query, and deletion.
ServiceComb-native APIs	**Authentication**	APIs related to microservice identity authentication.
	Microservice	APIs related to microservice management.
	Schema	APIs related to microservice contract query and upload.
	Microservice Instance	APIs related to microservice instance registry and heartbeat reporting.
	Dependency	APIs related to microservice dependency query.

Type	Subtype	Description
	Configuration Management	APIs related to configuration management of the configuration center.

NOTICE

APIs in **Microservice**, **Schema**, **Microservice Instance**, and **Dependency** are available only in ME-Riyadh, CN-Hong Kong, and AP-Singapore.

3 Calling APIs

3.1 Making an API Request

This section describes the structure of a REST API request, and calls the IAM API for **obtaining a user token** as an example. The obtained token can then be used to authenticate the calling of other APIs.

Request URI

A request URI is in the following format:

{URI-scheme} :// {Endpoint} / {resource-path} ? {query-string}

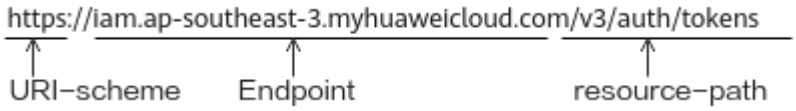
Although a request URI is included in the request header, most programming languages or frameworks require passing the request URI separately.

- **URI-scheme**: Protocol used to transmit requests. All APIs use **HTTPS**.
- **Endpoint**: Domain name or IP address of the server bearing the REST service. The endpoint varies between services in different regions. It can be obtained from **Regions and Endpoints**. For example, the endpoint of IAM in the AP-Singapore region is iam.ap-southeast-3.myhwcloud.com. When obtaining an account token of the microservice engine instance, the endpoint is set to the service center address.
- **resource-path**: Access path of an API for performing a specified operation. Obtain the value from the URI of an API. For example, the **resource-path** of the API for obtaining a user token is **/v3/auth/tokens**.
- **query-string**: Query parameter, which is optional. Ensure that a question mark (?) is included before each query parameter that is in the format of "Parameter name=Parameter value". For example, **? limit=10** means to query up to 10 pieces of data.

For example, if you want to obtain the token of IAM in the AP-Singapore region, use the endpoint (iam.ap-southeast-3.myhuaweicloud.com) of the AP-Singapore region, find resource-path (/v3/auth/tokens) in the URI of **Obtaining a User Token** and combine it as follows:

```
https://iam.ap-southeast-3.myhuaweicloud.com/v3/auth/tokens
```


Figure 3-1 Example URI



NOTE

To simplify the URI display, each API is provided only with a resource-path and a request method. The **URI-scheme** of all APIs is **HTTPS**, and the endpoints of all APIs in the same region are identical.

Request Methods

The HTTP protocol defines the following request methods that can be used to send a request to the server:

- **GET**: requests the server to return specified resources.
- **PUT**: requests the server to update specified resources.
- **POST**: requests the server to add resources or perform special operations.
- **DELETE**: requests the server to delete specified resources, for example, an object.
- **HEAD**: requests a server resource header.
- **PATCH**: requests the server to update partial content of a specified resource. If the resource does not exist, a new resource can be created using the PATCH method.

For example, in the case of the API used to **obtain a user token**, the request method is **POST**. The request is as follows:

```
POST https://iam.ap-southeast-3.myhuaweicloud.com/v3/auth/tokens
```

Request Header

You can also add additional header fields to a request, such as the fields required by a specified URI or HTTP method. For example, to request for the authentication information, add **Content-Type**, which specifies the request body type.

Common request header fields are as follows:

- **Content-Type**: request body type or format. This field is mandatory and its default value is **application/json**. Other values of this field will be provided for specific APIs if any.
- **X-Auth-Token**: A user token only for token-based API authentication. The user token is a response to the API used to **obtain a user token**. This API is the only one that does not require authentication.
- **Authorization**: account token of the microservice engine instance. This field is mandatory when microservice engine security authentication is used. An account token is a response to the API for **Obtaining the User Token of an Exclusive ServiceComb Engine**. Only this API does not require authentication.

The API used to **obtain a user token** does not require authentication. Therefore, only the **Content-Type** field needs to be added to requests for calling the API. An example of such requests is as follows:


```
POST https://iam.ap-southeast-3.myhuaweicloud.com/v3/auth/tokens
Content-Type: application/json
```

Request Body

The body of a request is often sent in a structured format as specified in **Content-Type**. The request body transfers content other than the request header. If the request body contains Chinese characters, set **Content-type** to **utf-8**, for example, Content-Type: application/json;utf-8.

The request body varies between APIs. Some APIs do not require the request body, such as the APIs requested using the GET and DELETE methods.

In the case of the API used to **obtain a user token**, the request parameters and parameter description can be obtained from the API request. The following provides an example request with the body included. Replace *username*, *domainname*, ********* (login password), and *xxxxxxxxxxxxxxxxx* (project name) with the actual values. To learn how to obtain a project ID, see **Regions and Endpoints**.

NOTE

The **scope** parameter specifies where a token takes effect. You can set **scope** to an account or a project under an account. In the following example, the token takes effect only for the resources in a specified project. For more information about this API, see **Obtaining a User Token**

```
POST https://iam.ap-southeast-3.myhuaweicloud.com/v3/auth/tokens
Content-Type: application/json

{
    "auth": {
        "identity": {
            "methods": [
                "password"
            ],
            "password": {
                "user": {
                    "name": "username",
                    "password": "********",
                    "domain": {
                        "name": "domainname"
                    }
                }
            }
        },
        "scope": {
            "project": {
                "name": "xxxxxxxxxxxxxxxxxx"
            }
        }
    }
}
```

If all data required for the API request is available, you can send the request to call the API through curl or coding. In the response to the API used to **obtain a user token**, **x-subject-token** is the desired user token. This token can then be used to authenticate the calling of other APIs.

3.2 Authentication

You can use either of the following authentication methods to call APIs:

- Token-based authentication: Requests are authenticated using a token.
- AK/SK-based authentication: Requests are encrypted using an AK/SK pair

Token-based Authentication

NOTE

The validity period of a token is 24 hours. When using a token for authentication, cache it to prevent frequently calling the Identity and Access Management (IAM) API for obtaining a user token.

A token specifies temporary permissions in a computer system. During token-based API authentication, the token is added to requests to get permissions for calling the API.

In **Making an API Request**, the process of calling the API used to **obtain a user token** is described.

When **calling an API to obtain a user token**, you must set **auth.scope** in the request body to **project**.

```
{
    "auth": {
        "identity": {
            "methods": [
                "password"
            ],
            "password": {
                "user": {
                    "name": "username",
                    "password": "********",
                    "domain": {
                        "name": "domainname"
                    }
                }
            }
        },
        "scope": {
            "project": {
                "name": "xxxxxxxx"
            }
        }
    }
}
```

After a token is obtained, the **X-Auth-Token** header field must be added to requests to specify the token when calling other APIs. For example, if the token is **ABCDEFJ....**, **X-Auth-Token: ABCDEFJ....** can be added to a request as follows:

```
GET https://iam.ap-southeast-3.myhuaweicloud.com/v3/auth/projects
Content-Type: application/json
X-Auth-Token: ABCDEFJ....
```

AK/SK-based Authentication

NOTE

AK/SK authentication supports API requests with a body not larger than 12 MB. For API requests with a larger body, token authentication is recommended.

In AK/SK-based authentication, AK/SK is used to sign requests and the signature is then added to the requests for authentication.

- AK: access key ID. It is a unique identifier associated with a secret access key and is used in conjunction with a secret access key to sign requests cryptographically.
- SK: secret access key. It is used together with an AK to identify a sender who initiates a request and to cryptographically sign requests, preventing the request from being modified.

In AK/SK-based authentication, you can sign requests using an AK/SK based on the signature algorithm or using the signing SDK. For details about how to sign requests and use the signing SDK, see **API Request Signing Guide**.

NOTICE

Unlike SDKs provided by services, the signature SDK is only for signing.

3.3 Response

After sending a request, you will receive a response, including a status code, response header, and response body.

Status Code

A status code is a group of digits, ranging from 1xx to 5xx. It indicates the status of a request. For more information, see **Status Codes**.

Response Header

Similar to a request, a response also has a header, for example, **Content-Type**.

Response Body

The body of a response is often returned in a structured format as specified in **Content-Type**. The response body transfers content except the response header.

If an error occurs during API calling, an error code and a message will be displayed. The following shows an error response body.

```
{
  "error_code": "SVCSTG.0111",
  "error_msg": "xxxxxxxxx"
}
```

In the response body, **error_code** is an error code, and **error_msg** provides information about the error.

4 Examples

4.1 Registry and Discovery

Scenarios

This section describes how to register and discover microservices by calling APIs. For details about how to call APIs, see **Calling APIs**.

Process

1. Register the my-provider microservice.
2. Register the my-provider microservice instance.
3. Register the my-consumer microservice.
4. The my-consumer microservice discovers the my-provider microservice instance.

NOTE

The preceding microservice names are examples.

Prerequisites

Obtain the registry center address of the ServiceComb engine instance by referring to **Service Center Address**. The following uses **{cse_endpoint}** as an example.

Procedure

Step 1 Register the my-provider microservice.

Call the **Creating Static Information for a Microservice** API. The following is a request example:

```
curl -k -H "x-domain-name:default" -XPOST "https://{cse_endpoint}/v4/default/registry/microservices" -d '{
    "service": {
        "serviceName": "my-provider",
        "appId": "default",
        "version": "1.0.0",
        "description": "test",
```



```
        "level": "MIDDLE",
        "status": "UP"
    }
}'
```

Response:

```
{"serviceId":"918282e8562dc5fdc9a8dcd4d1baabb492190aa4"}
```

serviceId will be replaced with **{providerServiceId}** in the following example.

Step 2 Register the my-provider microservice instance.

Call the **Registering a Microservice Instance** API. The validity period of the instance is 1 hour. The instance is automatically brought offline when the validity period expires. Assume that the listening address of the my-provider microservice instance is **127.0.0.1:8080**. The following shows an example request.

```
curl -k -H "x-domain-name:default" -XPOST "https://{cse_endpoint}/v4/default/registry/microservices/
{providerServiceId}/instances" -d '{
    "instance": {
        "hostName": "test",
        "endpoints": [
            "rest:127.0.0.1:8080"
        ],
        "status": "UP",
        "healthCheck": {
            "mode": "push",
            "interval": 900,
            "times": 3
        }
    }
}'
```

Response:

```
{"instanceId":"2be605a095ed11eabcbe0255ac100fa3"}
```

Step 3 Register the my-consumer microservice.

Call the **Creating Static Information for a Microservice** API. The following is a request example:

```
curl -k -H "x-domain-name:default" -XPOST "https://{cse_endpoint}/v4/default/registry/microservices" -d '{
    "service": {
        "serviceName": "my-consumer",
        "appId": "default",
        "version": "1.0.0",
        "description": "test",
        "level": "MIDDLE",
        "status": "UP"
    }
}'
```

Response:

```
{"serviceId":"9db248934c31fc754d6e922b48ede4a5c004d3c1"}
```

serviceId will be replaced with **{consumerServiceId}** in the following example.

Step 4 The my-consumer microservice discovers the my-provider microservice instance.

Call the **Querying a Microservice Instance by Filter Criteria** API. The my-consumer microservice queries information about the my-provider microservice instance from the service center based on its own service ID. The following shows an example request.


```
curl -k -H "x-domain-name:default" -H "X-ConsumerId:{consumerServiceId}" -XGET "https://
{cse_endpoint}/v4/default/registry/instances?appId=default&serviceName=my-provider&version=0.0.0%2B"
```

Response:

```
{
   "instances": [
      {
         "instanceId": "2be605a095ed11eabcbe0255ac100fa3",
         "serviceId": "918282e8562dc5fdc9a8dcd4d1baabb492190aa4",
         "endpoints": [
            "rest:127.0.0.1:8080"
         ],
         "hostName": "test",
         "status": "UP",
         "healthCheck": {
            "mode": "push",
            "interval": 150,
            "times": 3
         },
         "timestamp": "1589465646",
         "modTimestamp": "1589465646",
         "version": "1.0.0"
      }
   ]
}
```

In actual practice, the my-consumer microservice can obtain the address of the my-provider microservice instance from the **endpoint** field in the instance query result and initiate service invoking.

Alternatively, choose **Service Catalog** on the CSE console to view the service registry information.

----End

5 CSE API

5.1 API Calling

CSE provides REST APIs, allowing you to call APIs using HTTPS.

To call the CSE APIs of a microservice engine, see **Calling APIs**.

5.2 Dynamic Configuration

5.2.1 Importing Configurations

Function

This API is used to import configurations to the configuration management center.

NOTE

This API applies to ServiceComb engine 2.x.

URL

POST /v1/{project_id}/kie/file

Table 5-1 Path parameter

Parameter	Mandatory	Type	Description
project_id	Yes	String	Tenant sub-project ID, which must be unique. The value contains 1 to 64 characters. See **Obtaining a Project ID**.

Table 5-2 Query parameters

Parameter	Mandatory	Type	Description
override	Yes	String	**force**: Forcible import will overwrite duplicate items. **skip**: Duplicate items will be skipped. **abort**: If duplicate items are found, the import stops.
label	No	String	To import a specified label, enter a value in the format of {Label key}:{Label value}. If this parameter is empty, the label of the body is imported.

Request

Table 5-3 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token.
x-engine-id	Yes	String	ID of the engine whose configuration is to be uploaded. See **Querying the Microservice Engine List**.
X-Enterprise-Project-ID	No	String	Enterprise project ID. If this parameter is not set, the default enterprise project is **default** and the ID is **0**.

Table 5-4 Request body parameter

Parameter	Mandatory	Type	Description

upload_file	Yes	File	File to be imported. The file name format is fileName.json.
			File format:
			{
			"data": [
			{
			"key": "keyItem1",
			"value": "valueItem1",
			"labels":{
			"1": "1"
			},
			"status": "enabled",
			"value_type": "text"
			}
			]
			}

Response

Status code: 200

Table 5-5 Response body parameters

Parameter	Type	Description
success	List<**KVDoc**>	List of configuration items that are successfully imported.
failure	List<**DocFailedOfUpload**>	List of configuration items that fail to be imported.

Table 5-6 KVDoc

Parameter	Type	Description
id	String	Configuration ID.
key	String	Configuration key.
value	String	Configuration value.
value_type	String	Type of the configuration value.
status	String	Configuration status.
create_time	Integer	Time when the configuration is created.
update_time	Integer	Time when the configuration is updated.
create_revision	Integer	Version number of the created configuration.

Parameter	Type	Description
update_revision	Integer	Version number of the updated configuration.
labels	Map<String,String>	Configuration labels.

Table 5-7 DocFailedOfUpload

Parameter	Type	Description
key	String	Key of the configuration item that fails to be imported.
labels	Map<String,String>	Labels of the configuration item that fails to be imported.
error_code	String	Error code.
error_message	String	Error message.

Status code: 400

Table 5-8 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_message	String	Error message.
detail	String	Location details.

Status code: 500

Table 5-9 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_message	String	Error message.
detail	String	Location details.

Example Request

Import a configuration file. If the same configurations exist, these configurations will be overwritten based on the body label.

```
POST https://{endpoint}/v1/{project_id}/kie/file/override=force

----WebKitFormBoundary7MA4YWxkTrZu0gW
Content-Disposition: form-data; name="upload_file"; filename="custom_1663124521493.json"
Content-Type: application/json
----WebKitFormBoundary7MA4YWxkTrZu0gW
```

Example Response

Successfully queried.

```
{
   "success": [
      {
         "id": "7e2d346e-d907-4871-bf51-a17fc7e52ad4",
         "key": "2",
         "value": "2",
         "value_type": "text",
         "create_revision": 3,
         "update_revision": 3,
         "status": "enabled",
         "create_time": 1636374809,
         "update_time": 1636374809,
         "labels": {
            "2": "2",
            "environment": "production"
         }
      },
      {
         "id": "907b3891-c691-4ae0-816f-e91eba705e28",
         "key": "1",
         "value": "1",
         "value_type": "text",
         "create_revision": 4,
         "update_revision": 4,
         "status": "enabled",
         "create_time": 1636374809,
         "update_time": 1636374809,
         "labels": {
            "1": "1",
            "environment": "production"
         }
      }
   ],
   "failure": []
}
```

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **CSE Error Codes**.

5.2.2 Exporting Configurations

Function

This API is used to export configurations from the configuration management center.

NOTE

This API applies to ServiceComb engine 2.x.

URI

POST /v1/{project_id}/kie/download

Table 5-10 Path parameter

Parameter	Mandatory	Type	Description
project_id	Yes	String	Tenant sub-project ID, which must be unique. The value contains 1 to 64 characters. See **Obtaining a Project ID**.

Table 5-11 Query parameters

Parameter	Mandatory	Type	Description
label	No	String	Exports data filtered by label. The format is {Label key}:{Label value}.
match	No	String	Matching option of the filtered items. Value **exact** indicates exact matching, including the same number of labels. If the value is null, inclusive matching is used.

Request

Table 5-12 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token.
x-engine-id	Yes	String	ID of the engine whose configuration is to be downloaded. See **Querying the Microservice Engine List**.
X-Enterprise-Project-ID	No	String	Enterprise project ID. If this parameter is not set, the default enterprise project is **default** and the ID is **0**.

Table 5-13 Request body parameter

Parameter	Mandatory	Type	Description
ids	Yes	List<String>	ID list of the configuration items to be exported. See **Querying the Configuration List**. When **ids** is empty, this API is called to export all configuration items.

Response

Status code: 200

Table 5-14 Response body parameter

Parameter	Mandatory	Type	Description
metadata	No	**Object**	Additional information about a configuration item.
data	Yes	List<**KVCreateBody**>	List of configuration items to be created.

Table 5-15 KVCreateBody

Parameter	Mandatory	Type	Description
id	Yes	String	Configuration ID.

Parameter	Mandatory	Type	Description
key	Yes	String	Configuration key.
value	No	String	Configuration value.
labels	No	Map<String,String>	Configuration labels.
status	No	String	Configuration status.
value_type	No	String	Type of the configuration value.

Table 5-16 Metadata

Parameter	Mandatory	Type	Description
version	No	String	Version information.
annotations	No	Object	Additional information about a configuration item.

Status code: 400

Table 5-17 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_message	String	Error message.
detail	String	Location details.

Status code: 500

Table 5-18 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_message	String	Error message.
detail	String	Location details.

Example Request

Export the configuration whose ID is **0bdfddb8-c4f9-4a88-8360-07a8a616a804**.

```
POST https://{endpoint}/v1/{project_id}/kie/download

{
   "ids":[
      "0bdfddb8-c4f9-4a88-8360-07a8a616a804"
   ]
}
```

Example Response

Successfully queried.

```
{
   "metadata": {
      "version": "2.0.0"
   },
   "data": [
      {
         "id": "0bdfddb8-c4f9-4a88-8360-07a8a616a804",
         "key": "2",
         "labels": {
            "2": "2"
         },
         "value": "2",
         "status": "enabled",
         "value_type": "text"
      },
      {
         "id": "34befef9-7f3a-4394-85f9-142622a11b1d",
         "key": "1",

         "labels": {
            "1": "1"
         },
         "value": "1",
         "status": "enabled",
         "value_type": "text"
      }
   ]
}
```

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **CSE Error Codes**.

5.3 Engine Management

5.3.1 Querying Flavors Supported by a Microservice Engine

Function

This API is used to query the list of flavors supported by a microservice engine.

URI

GET /v2/{project_id}/enginemgr/flavors

Table 5-19 Path parameter

Parameter	Mandatory	Type	Description
project_id	Yes	String	Tenant sub-project ID, which must be unique. The value contains 1 to 64 characters. See **Obtaining a Project ID**.

Table 5-20 query parameter

Parameter	Mandatory	Type	Description
specType	No	String	• To query a ServiceComb engine, set it to **CSE2**. • To query a Nacos engine, set it to **Nacos**.

Request

Table 5-21 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token.
X-Enterprise-Project-ID	No	String	Enterprise project ID. If this parameter is not set, the default enterprise project is **default** and the ID is **0**.

Response

Status code: 200

Table 5-22 Response body parameters

Parameter	Type	Description
total	Integer	Total number of flavors of a microservice engine.
data	Array of **Flavor** objects	Flavor details of a microservice engine.

Table 5-23 Flavor

Parameter	Type	Description
flavor	String	Flavor of a microservice engine.
description	String	Flavor description of a microservice engine.

Status code: 400

Table 5-24 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 5-25 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Query the list of flavors supported by a microservice engine.

```
GET /v2/{project_id}/enginemgr/flavors
```

Example Response

```
{
  "total": 0,
  "data": [
    {
      "flavor": "string",
      "description": "string"
    }
  ]
}
```

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **CSE Error Codes**.

5.3.2 Querying the Microservice Engine List

Function

This API is used to query the microservice engine list.

URI

GET /v2/{project_id}/enginemgr/engines

Table 5-26 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Unique ID of a tenant's sub-project. The value contains 1 to 64 characters.

Table 5-27 Query Parameters

Parameter	Mandatory	Type	Description
offset	Yes	Integer	Offset.
limit	Yes	String	Number of items displayed per page.

Request Parameters

Table 5-28 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token.
X-Enterprise-Project-ID	No	String	Default enterprise project: default. Default ID: 0.
Content-Type	Yes	String	Enter **application/json;charset=UTF-8**.
Accept	Yes	String	Enter **application/json**.

Response Parameters

Status code: 200

Table 5-29 Response body parameters

Parameter	Type	Description
total	Integer	Total number of exclusive microservice engines that meet the search criteria.
data	Array of **EngineSimpleInfo** objects	Details about an exclusive microservice engine.

Table 5-30 EngineSimpleInfo

Parameter	Type	Description
id	String	ID of an exclusive microservice engine.
name	String	Engine name.
enterpriseProjectId	String	ID of the enterprise project to which a microservice engine belongs.

Parameter	Type	Description
enterpriseProjectName	String	Name of the enterprise project to which a microservice engine belongs.
type	String	Microservice engine type. **CSE** indicates an exclusive one, and **CSE_Share** indicates a professional one. Enumeration values: • **CSE** • **CSE_Share**
description	String	Description of an exclusive microservice engine.
flavor	String	Flavor of an exclusive microservice engine. Enumeration values: • **cse.s1.small2** • **cse.s1.medium2** • **cse.s1.large2** • **cse.s1.xlarge2**
payment	String	Billing mode of an exclusive microservice engine. **0** indicates yearly/monthly, **1** indicates pay-per-use, and **2** indicates free of charge.
authType	String	Authentication mode of a microservice engine. RBAC indicates security authentication, and NONE indicates no authentication.
status	String	Current status of an exclusive microservice engine. Enumeration values: • **Creating** • **Available** • **Unavailable** • **Deleting** • **Deleted** • **Upgrading** • **Modifying** • **CreateFailed** • **DeleteFailed** • **UpgradeFailed** • **ModifyFailed** • **Freezed**
externalAddress	String	Address for accessing a microservice engine in a VPC on the tenant side.

Parameter	Type	Description
serviceEndpoint	Map<String,EntrypointItem>	Address for accessing a microservice engine component in a VPC on the tenant side.
publicAddress	String	Public network access address of a microservice engine. You need to enable public network access.
publicServiceEndpoint	Map<String,EntrypointItem>	Public network access address of a microservice engine component. You need to enable public network access.
totalInstance	Integer	Total number of instances supported by a microservice engine.
usedInstance	Integer	Total number of used instances.
availableInstance	Integer	Total number of available instances.
version	String	Current version of an exclusive microservice engine.
latestVersion	String	Latest version of a microservice engine.
createTime	Long	Creation time of a microservice engine.
dueTo	Long	Expiration time of a microservice engine.
latestJobId	Integer	ID of the latest job of a microservice engine.
engineAdditionalActions	Array of strings	Additional operations allowed by a microservice engine. Enumeration values: • **ForceDelete** • **Rollback** • **Retry**
specType	String	Deployment type of a microservice engine. Enumeration values: • **CCE** • **CSE** • **SpringCloud**
reference	**EngineReference** object	Additional information about a microservice engine.

Table 5-31 EntrypointItem

Parameter	Type	Description
masterEntrypoint	String	Primary IPv4 address for accessing an exclusive microservice engine component.
masterEntrypointIpv6	String	Primary IPv6 address for accessing an exclusive microservice engine component.
slaveEntrypoint	String	Secondary IPv4 address for accessing an exclusive microservice engine component.
slaveEntrypointIpv6	String	Secondary IPv4 address for accessing an exclusive microservice engine component.
type	String	Type of an exclusive microservice engine component. Enumeration values: • **REGISTRY** • **SERVICE**

Table 5-32 EngineReference

Parameter	Type	Description
vpc	String	VPC name.
azList	Array of strings	List of AZs for microservice engine deployment.
networkId	String	Subnet network ID of a microservice engine.
subnetCidr	String	IPv4 subnet division of a microservice engine.
subnetCidrV6	String	IPv6 subnet division of a microservice engine.
subnetGateway	String	Subnet gateway of a microservice engine.
publicIpId	String	Public IP address ID of a microservice engine. You need to enable public network access.
serviceLimit	Integer	Total number of microservices supported by a package.
instanceLimit	Integer	Total number of instances supported by a package.
inputs	Map<String,String>	Additional parameter of an exclusive microservice engine.

Status code: 400

Table 5-33 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Status code: 500

Table 5-34 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
GET https://{endpoint}/v2/{project_id}/enginemgr/engines
```

Example Responses

Status code: 200

OK

```
{
  "total" : 0,
  "data" : [ {
    "id" : "891bf21a-4024-4f47-b38c-bd259ca8f10a",
    "name" : "test",
    "enterpriseProjectId" : "0",
    "enterpriseProjectName" : "default",
    "type" : "CSE",
    "description" : "string",
    "flavor" : "cse.s1.medium2",
    "payment" : "0",
    "authType" : "NONE",
    "status" : "Available",
    "externalAddress" : "192.168.0.169",
    "serviceEndpoint" : {
      "kie" : {
        "masterEntrypoint" : "https://192.168.0.169:30110",
        "masterEntrypointIpv6" : "https://[2407:c080:11f0:11:b11d:675c:97ab:65f6]:30110",
        "slaveEntrypoint" : "string",
        "slaveEntrypointIpv6" : "string",
        "type" : "REGISTRY"
      }
    },
    "publicAddress" : "",
    "publicServiceEndpoint" : {
      "kie" : {
        "masterEntrypoint" : "",
```



```
      "masterEntrypointIpv6" : "",
      "slaveEntrypoint" : "",
      "slaveEntrypointIpv6" : "",
      "type" : "REGISTRY"
    }
  },
  "totalInstance" : 200,
  "usedInstance" : 0,
  "availableInstance" : 200,
  "version" : "2.3.1",
  "latestVersion" : "2.3.4",
  "createTime" : 1635576800332,
  "dueTo" : 4102415999000,
  "latestJobId" : 12339,
  "engineAdditionalActions" : [ "Retry" ],
  "specType" : "CSE2",
  "reference" : {
    "vpc" : "vpc-test",
    "azList" : [ "string" ],
    "networkId" : "88550801-e892-4f8e-b21b-f7147f604f69",
    "subnetCidr" : "192.168.0.0/24",
    "subnetCidrV6" : "2407:c080:11f0:11::/64",
    "subnetGateway" : "192.168.0.2",
    "publicIpId" : "",
    "serviceLimit" : 200,
    "instanceLimit" : 200,
    "inputs" : {
      "is_arm_cluster" : "true",
      "nodeFlavor" : "s6.large.2"
    }
  }
 } ]
}
```


Status Codes

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Codes

See **CSE Error Codes**.

5.3.3 Creating an Exclusive Microservice Engine

Function

This API is used to create an exclusive microservice engine.

URI

POST /v2/{project_id}/enginemgr/engines

Table 5-35 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Unique ID of a tenant's sub-project. The value contains 1 to 64 characters.

Request Parameters

Table 5-36 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token.
X-Enterprise-Project-ID	No	String	Default enterprise project: default. Default ID: 0. Default: **0**
Content-Type	Yes	String	Enter **application/json;charset=UTF-8**.
Accept	Yes	String	Enter **application/json**.

Table 5-37 Request body parameters

Parameter	Mandatory	Type	Description
name	Yes	String	Name of an exclusive microservice engine.
description	No	String	Description of an exclusive microservice engine. Maximum: **255**
payment	Yes	String	Billing mode of an exclusive microservice engine. **0** indicates yearly/monthly, **1** indicates pay-per-use, and **2** indicates free of charge. Maximum: **24**

Parameter	Mandatory	Type	Description
flavor	Yes	String	Flavor of an exclusive microservice engine. Enumeration values: • **cse.s1.small2** • **cse.s1.medium2** • **cse.s1.large2** • **cse.s1.xlarge2**
azList	Yes	Array of strings	List of AZs at the current region.
authType	Yes	String	Authentication mode of a microservice engine. RBAC indicates security authentication, and NONE indicates no authentication. Enumeration values: • **RBAC** • **NONE**
vpc	Yes	String	VPC name. The value can contain up to 64 characters.
vpcId	No	String	VPC ID. The value can contain up to 64 characters.
networkId	Yes	String	Subnet ID of a microservice engine. The value can contain up to 64 characters.
subnetCidr	Yes	String	Subnet division of a microservice engine.The value can contain up to 24 characters.
publicIpId	No	String	ID of the public network address for accessing a microservice engine. The value can contain up to 64 characters.
auth_cred	No	**EngineRbacPwd** object	This parameter is mandatory when RBAC authentication is selected. It contains the authentication information of the engine.

Parameter	Mandatory	Type	Description
specType	Yes	String	Deployment type of a microservice engine. Default: **CSE** Enumeration values: • **CCE** • **CSE** • **SpringCloud**
inputs	No	Map<String,String>	Additional parameter of an exclusive microservice engine.

Table 5-38 EngineRbacPwd

Parameter	Mandatory	Type	Description
pwd	No	String	Password of the default user **root** of an exclusive microservice engine with security authentication enabled.

Response Parameters

Status code: 200

Table 5-39 Response body parameters

Parameter	Type	Description
id	String	ID of the exclusive microservice engine to be created.
name	String	Name of the exclusive microservice engine to be created.
jobId	Integer	ID of a microservice engine job.

Status code: 400

Table 5-40 Response body parameters

Parameter	Type	Description
error_code	String	Error code.

Parameter	Type	Description
error_msg	String	Error message.
detail	String	Location details.

Status code: 500

Table 5-41 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
POST https://{endpoint}/v2/{project_id}/enginemgr/engines

{
  "name" : "string",
  "description" : "string",
  "payment" : "string",
  "flavor" : "cse.s1.small2",
  "azList" : [ "string" ],
  "authType" : "NONE",
  "vpc" : "string",
  "networkId" : "string",
  "subnetCidr" : "string",
  "publicIpId" : "string",
  "specType" : "string",
  "inputs" : {
    "nodeFlavor" : "string"
  }
}
```

Example Responses

Status code: 200

OK

```
{
  "id" : "891bf21a-4024-4f47-b38c-bd259ca8f10a",
  "name" : "test",
  "jobId" : 17655
}
```


Status Codes

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Codes

See **CSE Error Codes**.

5.3.4 Querying Details About a Microservice Engine

Function

This API is used to query details about a microservice engine.

URI

GET /v2/{project_id}/enginemgr/engines/{engine_id}

Table 5-42 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Tenant sub-project ID, which must be unique. The value contains 1 to 64 characters. See **Obtaining a Project ID**.
engine_id	Yes	String	ID of a microservice engine. See **Querying the Microservice Engine List**.

Request

Table 5-43 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token.

Parameter	Mandatory	Type	Description
X-Enterprise-Project-ID	No	String	Enterprise project ID. If this parameter is not set, the default enterprise project is **default** and the ID is **0**.

Response

Status code: 200

Table 5-44 Response body parameters

Parameter	Type	Description
id	String	ID of a microservice engine.
name	String	Name of a microservice engine.
projectId	String	ID of the project to which a microservice engine belongs.
enterpriseProjectId	String	ID of the enterprise project to which a microservice engine belongs.
enterpriseProjectName	String	Name of the enterprise project to which a microservice engine belongs.
type	String	Engine type. • CSE • NACOS
description	String	Description of a microservice engine.
beDefault	Boolean	Whether the microservice engine is the default engine.

Parameter	Type	Description
flavor	String	Microservice engine flavor. • cse.s1.small2: HA 100-instance ServiceComb engine • cse.s1.medium2: HA 200-instance ServiceComb engine • cse.s1.large2: HA 500-instance ServiceComb engine • cse.s1.xlarge2: HA 2000-instance ServiceComb engine • cse.nacos2.c1.large.10-10: 500 to 500 Nacos engines • cse.nacos2.c1.xlarge.11-80: 550 to 4000 Nacos engines • cse.nacos2.c1.2xlarge.81-100: 4050 to 5000 Nacos engines
payment	String	Billing mode of a microservice engine. **1** indicates pay-per-use, and **2** indicates free of charge.
authType	String	Authentication mode of a microservice engine. **RBAC** indicates security authentication, and **NONE** indicates no authentication.
status	String	Current status of a microservice engine.
cceSpec	**Object**	CCE specification of a microservice engine.
externalEntrypoint	**Object**	Access address of a microservice engine.
publicAddress	String	Public network access address of a microservice engine. You need to enable public network access.
version	String	Current version of a microservice engine.
latestVersion	String	Latest version of a microservice engine.
createTime	Integer	Creation time of a microservice engine.
createUser	String	Creator of a microservice engine.
latestJobId	Integer	ID of the latest job of a microservice engine.

Parameter	Type	Description
engineAdditionalActions	Array of String	Additional operations allowed by a microservice engine. Delete ForceDelete Upgrade Modify Retry
specType	String	Deployment type of a microservice engine. • For a ServiceComb engine, the fixed value is **CSE2**. • For a Nacos engine, the fixed value is **NACOS2**.
reference	**Object**	Additional information about a microservice engine.
vmIds	Array of String	ID list of VMs used by a microservice engine on the resource tenant side.

Table 5-45 EngineReference

Parameter	Type	Description
vpc	String	VPC name.
vpcId	String	VPC ID. The value can contain up to 64 characters.
azList	Array of String	List of AZs for microservice engine deployment.
networkId	String	Subnet network ID of a microservice engine.
subnetCidr	String	IPv4 subnet division of a microservice engine.
subnetCidrV6	String	IPv6 subnet division of a microservice engine.
subnetGateway	String	Subnet gateway of a microservice engine.
publicIpId	String	Public IP address ID of a microservice engine. You need to enable public network access.
serviceLimit	Integer	Total number of microservices supported by a package.
instanceLimit	Integer	Total number of instances supported by a package.

Parameter	Type	Description
inputs	Map <String,String>	Additional parameter of a microservice engine.

Table 5-46 Spec

Parameter	Type	Description
id	Integer	CCE specification ID of a microservice engine.
engineId	String	ID of a microservice engine.
specType	String	CCE cluster deployment type of a microservice engine.
cluster	String	CCE cluster information about a microservice engine. Currently, the value is **null**.
clusterId	String	CCE cluster ID of a microservice engine.
clusterNodes	**Object**	CCE node list of a microservice engine.
flavor	String	CCE cluster flavor of a microservice engine.
region	String	Region where the CCE cluster of a microservice engine is located.
version	String	CCE cluster version of a microservice engine.
extendParam	String	Additional parameter of the CCE cluster of a microservice engine.

Table 5-47 SpecClusterNode

Parameter	Type	Description
clusterNodes	Array of **ClusterNode** objects	CCE node information of a microservice engine.

Table 5-48 ClusterNode

Parameter	Type	Description
id	String	CCE node ID of a microservice engine.
az	String	AZ to which the CCE cluster of a microservice engine belongs.

Parameter	Type	Description
ip	String	CCE node IP of a microservice engine.
label	String	CCE node label of a microservice engine.
status	String	CCE node label of a microservice engine.

Table 5-49 EngineExternalEntrypoint

Parameter	Type	Description
externalAddress	String	Address for accessing a microservice engine in a VPC on the tenant side.
publicAddress	String	Public network access address of a microservice engine. You need to enable public network access.
serviceEndpoint	**Object**	Address for accessing a microservice engine component in a VPC on the tenant side.
publicServiceEndpoint	**Object**	Public network access address of a microservice engine component. You need to enable public network access.

Table 5-50 Endpoint

Parameter	Type	Description
kie	**Object**	Endpoint information of the configuration center.
serviceCenter	**Object**	Endpoint information of the service center.

Table 5-51 EntryPoint

Parameter	Type	Description
masterEntrypoint	String	Primary IPv4 address for accessing a microservice engine component in a VPC.
masterEntrypointIpv6	String	Primary IPv6 address for accessing microservice engine component in a VPC.
slaveEntrypoint	String	Secondary IPv4 address for accessing a microservice engine component in a VPC.

Parameter	Type	Description
slaveEntrypointIpv6	String	Secondary IPv6 address for accessing a microservice engine component in a VPC.
type	String	Microservice engine component type. CSE NACOS

Status code: 400

Table 5-52 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 5-53 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Query details about an exclusive engine with specified **engine_id**.

```
GET /v2/{project_id}/enginemgr/engines/{engine_id}
```

Example Response

```
{
  "id": "891bf21a-4024-4f47-b38c-bd259ca8f10a",
  "name": "test",
  "description": "",
  "authType": "RBAC",
  "flavor": "cse.s1.medium2",
  "payment": "0",
  "version": "2.3.1",
  "latestVersion": "2.3.3",
  "status": "Creating",
  "beDefault": true,
```



```
"createUser": "test",
"createTime": 1635576800332,
"cceSpec": {
    "id": 7465,
    "engineId": "891bf21a-4024-4f47-b38c-bd259ca8f10a",
    "specType": "CCE",
    "cluster": null,
    "clusterId": "41115a6f-912f-11eb-9af9-0255ac100188",
    "clusterNodes": {
        "clusterNodes": [{
            "id": "c13aaf5c-2192-421c-8e03-522e2b9a06b5",
            "az": "test",
            "ip": "172.31.25.277",
            "label": "test",
            "status": "Active"
        }]
    },
    "flavor": null,
    "region": "test",
    "version": "",
    "extendParam": ""
},
"externalEntrypoint": {
    "externalAddress": "192.168.0.169",
    "publicAddress": "",
    "serviceEndpoint": {
        "kie": {
            "masterEntrypoint": "https://192.168.0.169:30110",
            "masterEntrypointIpv6": "https://[2407:c080:11f0:11:b11d:675c:97ab:65f6]:30110",
            "slaveEntrypoint": null,
            "slaveEntrypointIpv6": null,
            "type": "REGISTRY"
        },
                "serviceCenter": {
                        "masterEntrypoint": "https://192.168.0.169:30100",
                        "masterEntrypointIpv6": "https://[2407:c080:11f0:11:b11d:675c:97ab:65f6]:30100",
                        "slaveEntrypoint": null,
                        "slaveEntrypointIpv6": null,
                        "type": "REGISTRY"
                 }
    },
    "publicServiceEndpoint": {
        "kie": {
            "masterEntrypoint": "https://192.168.0.169:30110",
            "masterEntrypointIpv6": null,
            "slaveEntrypoint": null,
            "slaveEntrypointIpv6": null,
            "type": "REGISTRY"
        },
                "serviceCenter": {
                        "masterEntrypoint": "https://192.168.0.169:30100",
            "masterEntrypointIpv6": null,
            "slaveEntrypoint": null,
            "slaveEntrypointIpv6": null,
            "type": "REGISTRY"
                    }
    }
},
"reference": {
    "vpc": "vpc-test",
            "vpcId": "09902850-9454-4715-9764-018f0c3701hy",
    "azList": ["test"],
    "networkId": "88550801-e892-4f8e-b21b-f7147f604f69",
    "subnetCidr": "192.168.0.0/24",
    "subnetCidrV6": "2407:c080:11f0:11::/64",
    "subnetGateway": "192.168.0.2",
    "publicIpId": null,
    "serviceLimit": 200,
    "instanceLimit": 200,
```



```
        "inputs": {
            "is_arm_cluster": "false",
            "nodeFlavor": "s6.large.2"
        }
    },
    "latestJobId": 12339,
    "enterpriseProjectId": "0",
    "enterpriseProjectName": "default",
    "engineAdditionalActions": ["Retry"],
    "specType": "CSE2",
    "type": "CSE",
    "projectId": "string",
    "vmIds": [""]
}
```


Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **Error Code**.

5.3.5 Deleting a Microservice Engine

Function

This API is used to delete a microservice engine.

URI

DELETE /v2/{project_id}/enginemgr/engines/{engine_id}

Table 5-54 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Tenant sub-project ID, which must be unique. The value contains 1 to 64 characters. See **Obtaining a Project ID**.
engine_id	Yes	String	ID of a microservice engine.

Request

Table 5-55 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token.
X-Enterprise-Project-ID	No	String	Enterprise project ID. If this parameter is not set, the default enterprise project is **default** and the ID is **0**.

Response

Status code: 200

Table 5-56 Response body parameters

Parameter	Type	Description
id	String	ID of a microservice engine.
name	String	Name of a microservice engine.
jobId	Integer	ID of a microservice engine job.

Status code: 400

Table 5-57 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 5-58 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.

Parameter	Type	Description
detail	String	Location details.

Example Request

Delete an engine with specified **engine_id**.

```
DELETE /v2/{project_id}/enginemgr/engines/{engine_id}
```

Example Response

```
{
  "id": "891bf21a-4024-4f47-b38c-bd259ca8f10a",
  "name": "test",
  "jobId": 7256
}
```

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **CSE Error Codes**.

5.3.6 Querying Details About a Microservice Engine Job

Function

This API is used to query details about a microservice engine job.

URI

GET /v2/{project_id}/enginemgr/engines/{engine_id}/jobs/{job_id}

Table 5-59 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Unique ID of a tenant's sub-project. The value contains 1 to 64 characters.

Parameter	Mandatory	Type	Description
engine_id	Yes	String	ID of an exclusive microservice engine.
job_id	Yes	String	Job ID.

Request Parameters

Table 5-60 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token.
X-Enterprise-Project-ID	No	String	Default enterprise project: default. Default ID: 0.
Content-Type	Yes	String	Enter **application/json;charset=UTF-8**.
Accept	Yes	String	Enter **application/json**.

Response Parameters

Status code: 200

Table 5-61 Response body parameters

Parameter	Type	Description
id	Integer	Job ID.
engineId	String	ID of an exclusive microservice engine.
type	String	Job type. Enumeration values: • **Create** • **Delete** • **Upgrade** • **Modify**
description	String	Job description.

Parameter	Type	Description
status	String	Job status. Enumeration values: • **Init** • **Executing** • **Error** • **Timeout** • **Finished**
scheduling	Integer	Whether a job is being executed. 0: no; 1: yes.
createUser	String	Creator of a job.
startTime	Long	Start time of a job.
endTime	Long	End time of a job.
context	String	Job execution context.
tasks	Array of **TaskSteps** objects	Job phases.

Table 5-62 TaskSteps

Parameter	Type	Description
taskName	String	Name of a phase.
taskNames	Array of strings	List of procedures contained in the current phase.
status	String	Status of a phase. Enumeration values: • **Init** • **Executing** • **Error** • **Timeout** • **Finished**
startTime	Long	Start time of a phase.
endTime	Long	End time of a phase.
taskExecutorBrief	**TaskExecutorBrief** object	Job metadata.
tasks	Array of **Task** objects	Procedure.

Table 5-63 Task

Parameter	Type	Description
jobId	Integer	ID of the job to which the sub-job belongs.
id	Long	Sub-job ID, which is in UUID format.
type	String	Sub-job type. Enumeration values: • **Create** • **Delete** • **Upgrade** • **Modify**
assigned	String	Executor of a sub-job.
taskName	String	Sub-job name.
engineName	String	Name of the engine to which the sub-job belongs.
taskOrder	Integer	Sequence in which sub-jobs are executed, in ascending order.
status	String	Sub-job status. Enumeration values: • **Init** • **Executing** • **Error** • **Timeout** • **Finished**
startTime	Long	Start time of a sub-job.
endTime	Long	End time of a sub-job.
createTime	Long	Creation time of a sub-job.
updateTime	Long	Update time of a sub-job.
timeout	Integer	Whether a sub-job times out.
log	String	Sub-job details, which are auxiliary information generated during the execution.
output	String	Sub-job output information. Default: **{}**
taskExecutorBrief	**TaskExecutorBrief** object	Job metadata.

Table 5-64 TaskExecutorBrief

Parameter	Type	Description
duration	Long	Sub-job duration.
description	String	Sub-job description.

Status code: 400

Table 5-65 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Status code: 500

Table 5-66 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
GET https://{endpoint}/v2/{project_id}/enginemgr/engines/{engine_id}/jobs/{job_id}
```

Example Responses

Status code: 200

OK

```
{
  "id" : 12339,
  "engineId" : "ad76972c-a743-4770-859c-c98a680f4d98",
  "type" : "Create",
  "description" : "",
  "status" : "Finished",
  "scheduling" : 0,
  "createUser" : "test",
  "startTime" : 1636540095901,
  "endTime" : 1636540460230,
  "context" : "string",
  "tasks" : [ {
```



```
  "taskName" : "resourcePrepare",
  "taskNames" : [ "TenantTaskExecutor" ],
  "status" : "Finished",
  "startTime" : 1636540100216,
  "endTime" : 1636540110215,
  "taskExecutorBrief" : {
    "duration" : 6000,
    "description" : "Resource preparation"
  },
  "tasks" : [ {
    "jobId" : 12339,
    "id" : 89117,
    "type" : "Create",
    "assigned" : "string",
    "taskName" : "TenantTaskExecutor",
    "engineName" : "test",
    "taskOrder" : 0,
    "status" : "Init",
    "startTime" : 1636540100216,
    "endTime" : 1636540110215,
    "createTime" : 1636540095910,
    "updateTime" : 1636540110230,
    "timeout" : -1,
    "log" : "string",
    "output" : "{}",
    "taskExecutorBrief" : {
      "duration" : 6000,
      "description" : "Tenant processing"
    }
  } ]
 } ]
}
```


Status Codes

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Codes

See **CSE Error Codes**.

5.3.7 Retrying an Exclusive ServiceComb Engine

Function

This API is used to retry an exclusive ServiceComb engine.

URI

PUT /v2/{project_id}/enginemgr/engines/{engine_id}/actions

Table 5-67 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Tenant sub-project ID, which must be unique. The value contains 1 to 64 characters. See **Obtaining a Project ID**.
engine_id	Yes	String	ID of an exclusive ServiceComb engine.

Request

Table 5-68 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token.
X-Enterprise-Project-ID	No	String	Enterprise project ID. If this parameter is not set, the default enterprise project is **default** and the ID is **0**.

Table 5-69 Request body parameter

Parameter	Mandatory	Type	Description
action	Yes	String	Fixed value: Retry.

Response

Status code: 200

Table 5-70 Response body parameters

Parameter	Type	Description
id	String	ID of the exclusive ServiceComb engine.
name	String	Name of the exclusive ServiceComb engine.
jobId	Integer	ID of the job executed by the exclusive ServiceComb engine.

Status code: 400

Table 5-71 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.

Status code: 500

Table 5-72 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.

Example Request

Retry an engine with specified **engine_id**.

```
{
    "action": "Retry"
}
```

Example Response

Status code: 200

Response structure for retrying an exclusive ServiceComb engine.

```
{
    "id": "4a1bc63d-4719-4bd3-b1ad-5ee590b05e87",
    "name": "cse-gve7lg",
    "jobId": 18638
}
```

Status Code

Status Code	Description
200	ServiceComb
400	Bad Request
500	Internal Server Error

Error Code

See **CSE Error Codes**.

5.3.8 Upgrading an Exclusive ServiceComb Engine

Function

This API is used to upgrade an exclusive ServiceComb engine.

URI

PUT /v2/{project_id}/enginemgr/engines/{engine_id}/upgrade

Table 5-73 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Tenant sub-project ID, which must be unique. The value contains 1 to 64 characters. See **Obtaining a Project ID**.
engine_id	Yes	String	ID of an exclusive ServiceComb engine. See **Querying the Microservice Engine List**.

Request

Table 5-74 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token.
X-Enterprise-Project-ID	No	String	Enterprise project ID. If this parameter is not set, the default enterprise project is **default** and the ID is **0**.

Table 5-75 Request body parameter

Parameter	Mandatory	Type	Description
version	Yes	String	Target version number.

Response

Status code: 200

Table 5-76 Response body parameters

Parameter	Type	Description
id	String	ID of the exclusive ServiceComb engine.
name	String	Name of the exclusive ServiceComb engine.
jobId	Integer	ID of the job executed by the exclusive ServiceComb engine.

Status code: 400

Table 5-77 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.

Status code: 500

Table 5-78 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.

Example Request

Upgrade an engine with specified **engine_id** to 2.4.0.

```
{
    "version":"2.4.0"
}
```

Example Response

Status code: 200

Response structure for upgrading an exclusive ServiceComb engine.

```
{
  "id" : "891bf21a-4024-4f47-b38c-bd259ca8f10a",
  "name" : "test",
  "jobId" : 17655
}
```


Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **CSE Error Codes**.

5.3.9 Changing Microservice Engine Specifications

Function

This API is used to change microservice engine specifications.

URI

PUT /v2/{project_id}/enginemgr/engines/{engine_id}/resize

Table 5-79 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.
engine_id	Yes	String	Engine ID.

Request Parameters

Table 5-80 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	IAM-Token.
X-Enterprise-Project-ID	No	String	Default enterprise project: default. Default ID: 0. Default: **0**
Content-Type	Yes	String	Enter **application/json;charset=UTF-8**.
Accept	Yes	String	Enter **application/json**.

Table 5-81 Request body parameters

Parameter	Mandatory	Type	Description
flavor	No	String	New flavor.
inputs	No	Map<String,String>	New configuration, which overwrites the **input** parameter of the component.

Response Parameters

Status code: 200

Table 5-82 Response body parameters

Parameter	Type	Description
id	String	Engine ID.
name	String	Engine name.
job_id	Integer	jobID.

Status code: 400

Table 5-83 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Status code: 500

Table 5-84 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
PUT https://{endpoint}/v2/{project_id}/enginemgr/engines/{engine_id}/resize

{
  "flavor" : "string",
  "inputs" : {
    "additionalProp1" : "string",
    "additionalProp2" : "string",
    "additionalProp3" : "string"
  }
}
```

Example Responses

Status code: 200

OK

```
{
  "id" : "string",
  "name" : "string",
  "job_id" : 0
}
```

Status Codes

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Codes

See **CSE Error Codes**..

5.3.10 Updating the Configuration of an Exclusive Microservice Engine

Function

This API is used to update the configuration of an exclusive microservice engine

URI

PUT /v2/{project_id}/enginemgr/engines/{engine_id}/config

Table 5-85 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.
engine_id	Yes	String	ID of the engine to be updated.

Request Parameters

Table 5-86 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	IAM-Token.
X-Enterprise-Project-ID	No	String	Default enterprise project: default. Default ID: 0. Default: **0**

Table 5-87 Request body parameters

Parameter	Mandatory	Type	Description
authType	Yes	String	Security authentication type. Value: NONE or RBAC.

Response Parameters

Status code: 200

Table 5-88 Response body parameters

Parameter	Type	Description
id	String	ID of the exclusive microservice engine to be created.
name	String	Name of the exclusive microservice engine to be created.
jobId	Integer	ID of an exclusive microservice engine job.

Status code: 400

Table 5-89 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Status code: 500

Table 5-90 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
PUT https://{endpoint}/v2/{project_id}/enginemgr/engines/{engine_id}/config

{
  "authType" : "RBAC"
}
```

Example Responses

Status code: 200

OK

```
{
  "id" : "string",
  "name" : "string",
  "jobId" : 0
}
```

Status Codes

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Codes

See **CSE Error Codes**..

5.4 Microservice Governance

5.4.1 Querying the Governance Policy List

Function

This API is used to query the governance policy list.

URI

GET /v3/{project_id}/govern/governance/display

Table 5-91 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Unique ID of a tenant's sub-project. The value contains 1 to 64 characters.

Table 5-92 Query Parameters

Parameter	Mandatory	Type	Description
environment	Yes	String	Environment. Value **all** indicates to query all environments.
app	No	String	Application.

Request Parameters

Table 5-93 Request header parameters

Parameter	Mandatory	Type	Description
Content-Type	Yes	String	Enter **application/json;charset=UTF-8**.
X-Auth-Token	Yes	String	User token.
x-engine-id	Yes	String	Instance ID of an exclusive microservice engine.

Parameter	Mandatory	Type	Description
X-Enterprise-Project-ID	Yes	String	Enterprise project ID.

Response Parameters

Status code: 200

Table 5-94 Response body parameters

Parameter	Type	Description
[items]	Array of **GovPolicyDetail** objects	OK

Table 5-95 GovPolicyDetail

Parameter	Type	Description
matchGroup	**CreateBussinessScene** object	Traffic definition.
policies	Array of **policies** objects	Governance policy definition.

Table 5-96 CreateBussinessScene

Parameter	Type	Description
name	String	Traffic name.
status	String	Enabling status. Value: enabled or disabled. Default: **enabled**
selector	**GovSelector** object	Governance policy delivery scope.
spec	**spec** object	Traffic feature description.

Table 5-97 spec

Parameter	Type	Description
alias	String	Feature name.
matches	Array of **matches** objects	Matching condition definition.

Table 5-98 matches

Parameter	Type	Description
name	String	Condition name.
apiPath	Object	Matched **PATH**.
headers	Object	Matched **Headers**.
method	Array of strings	List of matched **Method**.
serviceName	String	Matched microservice name.

Table 5-99 policies

Parameter	Type	Description
id	String	Governance policy ID.
name	String	Governance policy name.
kind	String	Governance kind. Value: retry, rate-limiting, loadbalance, circuit-breaker, instance-isolation, fault-injection, or bulkhead.
status	String	Enabling status. Value: enabled or disabled.
selector	**GovSelector** object	Governance policy delivery scope.
spec	Object	Governance policy definition content.

Table 5-100 GovSelector

Parameter	Type	Description
environment	String	Environment.
app	String	Application.

Parameter	Type	Description
service	String	Optional. Governance is delivered to a microservice.

Status code: 400

Table 5-101 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
GET https://{endpoint}/v3/{project_id}/govern/governance/display
```

Example Responses

Status code: 200

OK

```
[ {
  "matchGroup" : {
    "name" : "",
    "status" : "enabled",
    "selector" : {
      "environment" : "string",
      "app" : "string",
      "service" : "string"
    },
    "spec" : {
      "alias" : "string",
      "matches" : [ {
        "name" : "string",
        "apiPath" : { },
        "headers" : { },
        "method" : [ "string" ],
        "serviceName" : "string"
      } ]
    }
  },
  "policies" : [ {
    "id" : "string",
    "name" : "string",
    "kind" : "string",
    "status" : "string",
    "selector" : {
      "environment" : "string",
      "app" : "string",
      "service" : "string"
    },
    "spec" : { }
```



```
 }]
}]
```

Status Codes

Status Code	Description
200	OK
400	Bad Request

Error Codes

See **CSE Error Codes**.

5.4.2 Creating a Dark Launch Policy

Function

This API is used to create a dark launch policy.

URI

PUT /v3/{project_id}/govern/route-rule/microservices/{service_name}

Table 5-102 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Unique ID of a tenant's sub-project. The value contains 1 to 64 characters.
service_name	Yes	String	Microservice name.

Table 5-103 Query Parameters

Parameter	Mandatory	Type	Description
environment	No	String	Environment. If this parameter is left blank,**<empty>** is used.
app_id	No	String	Application. If this parameter is left blank, **default** is used.

Request Parameters

Table 5-104 Request header parameters

Parameter	Mandatory	Type	Description
Content-Type	Yes	String	Enter **application/json;charset=UTF-8**.
X-Auth-Token	Yes	String	User token.
x-engine-id	Yes	String	Instance ID of an exclusive microservice engine.
X-Enterprise-Project-ID	Yes	String	Enterprise project ID.

Table 5-105 Request body parameters

Parameter	Mandatory	Type	Description
[items]	Yes	Array of **CreateRules** objects	Struct of the request for creating a dark launch policy.

Table 5-106 CreateRules

Parameter	Mandatory	Type	Description
precedence	No	Integer	Priority. A larger value indicates a higher priority.
match	No	**CreateMatch** object	Request matching rule. The value ranges from 0 to N. If this parameter is left empty, the request is matched.
route	No	Array of **CreateRoute** objects	Routing rule list.

Table 5-107 CreateMatch

Parameter	Mandatory	Type	Description
headers	No	**headers** object	Matched **Headers**.

Table 5-108 headers

Parameter	Mandatory	Type	Description
<header>	No	**<header>** object	Rule for matching **Headers**.

Table 5-109 <header>

Parameter	Mandatory	Type	Description
exact	No	String	Exactly matched value.
caseInsensitive	No	Boolean	Case insensitive or not.

Table 5-110 CreateRoute

Parameter	Mandatory	Type	Description
name	No	String	Rule name.
weight	No	Integer	Weight.
tags	No	**tags** object	List of matched instance tags.

Table 5-111 tags

Parameter	Mandatory	Type	Description
<tag>	No	String	Instance tag. Instances that meet the tag conditions are placed in this group.

Response Parameters

Status code: 200

Table 5-112 Response body parameters

Parameter	Type	Description
result	String	Result message.

Status code: 400

Table 5-113 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
PUT https://{endpoint}/v3/{project_id}/govern/route-rule/microservices/{service_name}

[ {
  "precedence" : 1,
  "match" : {
    "headers" : {
      "<header>" : {
        "exact" : "test",
        "caseInsensitive" : false
      }
    }
  },
  "route" : [ {
    "name" : "rule_1",
    "weight" : 20,
    "tags" : {
      "<tag>" : "1.0.0"
    }
  } ]
} ]
```

Example Responses

Status code: 200

OK

```
{
  "result" : "string"
}
```

Status Codes

Status Code	Description
200	OK
400	Bad Request

Error Codes

See **Error Codes**.

5.4.3 Querying a Dark Launch Rule of a Microservice

Function

This API is used to query a dark launch rule of a microservice.

URI

GET /v3/{project_id}/govern/route-rule/microservices/{service_name}

Table 5-114 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Unique ID of a tenant's sub-project. The value contains 1 to 64 characters.
service_name	Yes	String	Microservice name.

Table 5-115 Query Parameters

Parameter	Mandatory	Type	Description
environment	No	String	Environment. If this parameter is left blank,**<empty>** is used.
app_id	No	String	Application. If this parameter is left blank, **default** is used.

Request Parameters

Table 5-116 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token.
x-engine-id	Yes	String	Instance ID of an exclusive microservice engine.
X-Enterprise-Project-ID	Yes	String	Enterprise project ID.

Response Parameters

Status code: 200

Table 5-117 Response body parameters

Parameter	Type	Description
result	String	Result message.

Status code: 400

Table 5-118 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
GET https://{endpoint}/v3/{project_id}/govern/route-rule/microservices/{service_name}
```

Example Responses

Status code: 200

OK

```
{
  "result" : "string"
}
```

Status Codes

Status Code	Description
200	OK
400	Bad Request

Error Codes

See **CSE Error Codes**.

5.4.4 Deleting a Dark Launch Policy

Function

This API is used to delete a dark launch policy.

URI

DELETE /v3/{project_id}/govern/route-rule/microservices/{service_name}

Table 5-119 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Unique ID of a tenant's sub-project. The value contains 1 to 64 characters.
service_name	Yes	String	Microservice name.

Table 5-120 Query Parameters

Parameter	Mandatory	Type	Description
environment	No	String	Environment. If this parameter is left blank,**<empty>** is used.
app_id	No	String	Application. If this parameter is left blank, **default** is used.

Request Parameters

Table 5-121 Request header parameters

Parameter	Mandatory	Type	Description
Content-Type	Yes	String	Enter **application/json;charset=UTF-8**.
X-Auth-Token	Yes	String	User token.
x-engine-id	Yes	String	Instance ID of an exclusive microservice engine.
X-Enterprise-Project-ID	Yes	String	Enterprise project ID.

Response Parameters

Status code: 200

Table 5-122 Response body parameters

Parameter	Type	Description
result	String	Result message.

Status code: 400

Table 5-123 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
DELETE https://{endpoint}/v3/{project_id}/govern/route-rule/microservices/{service_name}
```

Example Responses

Status code: 200

OK

```
{
  "Result" : "string"
}
```

Status Codes

Status Code	Description
200	OK
400	Bad Request

Error Codes

See **CSE Error Codes**.

5.4.5 Changing a Governance Policy

Function

This API is used to change a governance policy.

URI

PUT /v3/{project_id}/govern/governance/{kind}/{policy_id}

Table 5-124 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Unique ID of a tenant's sub-project. The value contains 1 to 64 characters.
kind	Yes	String	Governance policy kind.
policy_id	Yes	String	Governance policy ID.

Request Parameters

Table 5-125 Request header parameters

Parameter	Mandatory	Type	Description
Content-Type	Yes	String	Enter **application/json;charset=UTF-8**.
X-Auth-Token	Yes	String	User token.
x-engine-id	Yes	String	Instance ID of an exclusive microservice engine.
X-Enterprise-Project-ID	Yes	String	Enterprise project ID.
x-environment	No	String	Environment.

Table 5-126 Request body parameters

Parameter	Mandatory	Type	Description
name	No	String	Governance policy name.
selector	No	**GovSelector** object	Governance policy delivery scope.
spec	No	Object	Governance policy definition content.

Table 5-127 GovSelector

Parameter	Mandatory	Type	Description
environment	No	String	Environment.
app	No	String	Application.
service	No	String	Optional. Governance is delivered to a microservice.

Response Parameters

Status code: 200

Table 5-128 Response body parameters

Parameter	Type	Description
result	String	Result message.

Status code: 400

Table 5-129 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
PUT https://{endpoint}/v3/{project_id}/govern/governance/{kind}/{policy_id}

{
  "name" : "string",
  "selector" : {
    "environment" : "string",
    "app" : "string",
    "service" : "string"
  },
  "spec" : { }
}
```

Example Responses

Status code: 200

OK


```
{
  "result" : "string"
}
```

Status Codes

Status Code	Description
200	OK
400	Bad Request

Error Codes

See **CSE Error Codes**.

5.4.6 Deleting a Governance Policy

Function

This API is used to delete a governance policy.

URI

DELETE /v3/{project_id}/govern/governance/{kind}/{policy_id}

Table 5-130 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Unique ID of a tenant's sub-project. The value contains 1 to 64 characters.
kind	Yes	String	Governance policy kind.
policy_id	Yes	String	Governance policy ID.

Request Parameters

Table 5-131 Request header parameters

Parameter	Mandatory	Type	Description
Content-Type	Yes	String	Enter **application/json;charset=UTF-8**.
X-Auth-Token	Yes	String	User token.

Parameter	Mandatory	Type	Description
x-engine-id	Yes	String	Instance ID of an exclusive microservice engine.
X-Enterprise-Project-ID	Yes	String	Enterprise project ID.
x-environment	No	String	Environment.

Response Parameters

Status code: 200

Table 5-132 Response body parameters

Parameter	Type	Description
result	String	Result message.

Status code: 400

Table 5-133 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
DELETE https://{endpoint}/v3/{project_id}/govern/governance/{kind}/{policy_id}
```

Example Responses

Status code: 200

OK

```
{
  "Result" : "string"
}
```


Status Codes

Status Code	Description
200	OK
400	Bad Request

Error Codes

See **CSE Error Codes**.

5.4.7 Querying Governance Policy Details

Function

This API is used to query governance policy details.

URI

GET /v3/{project_id}/govern/governance/{kind}/{policy_id}

Table 5-134 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Unique ID of a tenant's sub-project. The value contains 1 to 64 characters.
kind	Yes	String	Governance policy kind.
policy_id	Yes	String	Governance policy ID.

Request Parameters

Table 5-135 Request header parameters

Parameter	Mandatory	Type	Description
Content-Type	Yes	String	Enter **application/json;charset=UTF-8**.
X-Auth-Token	Yes	String	User token.
x-engine-id	Yes	String	Instance ID of an exclusive microservice engine.

Parameter	Mandatory	Type	Description
X-Enterprise-Project-ID	Yes	String	Enterprise project ID.
x-environment	No	String	Environment.

Response Parameters

Status code: 200

Table 5-136 Response body parameters

Parameter	Type	Description
name	String	Governance policy name.
selector	GovSelector object	Governance policy delivery scope.
spec	Object	Governance policy definition content.

Table 5-137 GovSelector

Parameter	Type	Description
environment	String	Environment.
app	String	Application.
service	String	Optional. Governance is delivered to a microservice.

Status code: 400

Table 5-138 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
GET https://{endpoint}/v3/{project_id}/govern/governance/{kind}/{policy_id}
```


Example Responses

Status code: 200

OK

```
{
  "name" : "string",
  "selector" : {
    "environment" : "string",
    "app" : "string",
    "service" : "string"
  },
  "spec" : { }
}
```

Status Codes

Status Code	Description
200	OK
400	Bad Request

Error Codes

See **CSE Error Codes**.

5.4.8 Creating a Governance Policy

Function

This API is used to create a governance policy.

URI

POST /v3/{project_id}/govern/governance/{kind}

Table 5-139 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Unique ID of a tenant's sub-project. The value contains 1 to 64 characters.
kind	Yes	String	Governance policy kind.

Request Parameters

Table 5-140 Request header parameters

Parameter	Mandatory	Type	Description
Content-Type	Yes	String	Enter **application/json;charset=UTF-8**.
X-Auth-Token	Yes	String	User token.
x-engine-id	Yes	String	Instance ID of an exclusive microservice engine.
X-Enterprise-Project-ID	Yes	String	Enterprise project ID.
x-environment	No	String	Environment.

Table 5-141 Request body parameters

Parameter	Mandatory	Type	Description
name	No	String	Governance policy name.
selector	No	**GovSelector** object	Governance policy delivery scope.
spec	No	Object	Governance policy definition content.

Table 5-142 GovSelector

Parameter	Mandatory	Type	Description
environment	No	String	Environment.
app	No	String	Application.
service	No	String	Optional. Governance is delivered to a microservice.

Response Parameters

Status code: 200

Table 5-143 Response body parameters

Parameter	Type	Description
result	String	Result message.

Status code: 400

Table 5-144 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
POST https://{endpoint}/v3/{project_id}/govern/governance/{kind}

{
  "name" : "string",
  "selector" : {
    "environment" : "string",
    "app" : "string",
    "service" : "string"
  },
  "spec" : { }
}
```

Example Responses

Status code: 200

OK

```
{
  "result" : "string"
}
```

Status Codes

Status Code	Description
200	OK
400	Bad Request

Error Codes

See **CSE Error Codes**.

5.4.9 Querying the Governance Policy List of a Specified Kind

Function

This API is used to query the governance policy list of a specified kind.

URI

GET /v3/{project_id}/govern/governance/{kind}

Table 5-145 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Unique ID of a tenant's sub-project. The value contains 1 to 64 characters.
kind	Yes	String	Governance policy kind.

Request Parameters

Table 5-146 Request header parameters

Parameter	Mandatory	Type	Description
Content-Type	Yes	String	Enter **application/json;charset=UTF-8**.
X-Auth-Token	Yes	String	User token.
x-engine-id	Yes	String	Instance ID of an exclusive microservice engine.
X-Enterprise-Project-ID	Yes	String	Enterprise project ID.
x-environment	No	String	Environment.

Response Parameters

Status code: 200

Table 5-147 Response body parameters

Parameter	Type	Description
[items]	Array of **GovPolicyDetail** objects	OK

Table 5-148 GovPolicyDetail

Parameter	Type	Description
matchGroup	**CreateBussinessScene** object	Traffic definition.
policies	Array of **policies** objects	Governance policy definition.

Table 5-149 CreateBussinessScene

Parameter	Type	Description
name	String	Traffic name.
status	String	Enabling status. Value: enabled or disabled. Default: **enabled**
selector	**GovSelector** object	Governance policy delivery scope.
spec	**spec** object	Traffic feature description.

Table 5-150 spec

Parameter	Type	Description
alias	String	Feature name.
matches	Array of **matches** objects	Matching condition definition.

Table 5-151 matches

Parameter	Type	Description
name	String	Condition name.
apiPath	Object	Matched **PATH**.
headers	Object	Matched **Headers**.
method	Array of strings	List of matched **Method**.
serviceName	String	Matched microservice name.

Table 5-152 policies

Parameter	Type	Description
id	String	Governance policy ID.
name	String	Governance policy name.
kind	String	Governance kind. Value: retry, rate-limiting, loadbalance, circuit-breaker, instance-isolation, fault-injection, or bulkhead.
status	String	Enabling status. Value: enabled or disabled.
selector	**GovSelector** object	Governance policy delivery scope.
spec	Object	Governance policy definition content.

Table 5-153 GovSelector

Parameter	Type	Description
environment	String	Environment.
app	String	Application.
service	String	Optional. Governance is delivered to a microservice.

Status code: 400

Table 5-154 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
GET https://{endpoint}/v3/{project_id}/govern/governance/{kind}
```

Example Responses

Status code: 200

OK

```
[ {
  "matchGroup" : {
    "name" : "",
    "status" : "enabled",
    "selector" : {
      "environment" : "string",
      "app" : "string",
      "service" : "string"
    },
    "spec" : {
      "alias" : "string",
      "matches" : [ {
        "name" : "string",
        "apiPath" : { },
        "headers" : { },
        "method" : [ "string" ],
        "serviceName" : "string"
      } ]
    }
  },
  "policies" : [ {
    "id" : "string",
    "name" : "string",
    "kind" : "string",
    "status" : "string",
    "selector" : {
      "environment" : "string",
      "app" : "string",
      "service" : "string"
    },
    "spec" : { }
  } ]
} ]
```

Status Codes

Status Code	Description
200	OK

Status Code	Description
400	Bad Request

Error Codes

See **CSE Error Codes**.

6 Nacos API

6.1 Querying the nacos Namespace

Function

This API is used to query the nacos namespace.

URI

GET /v1/{project_id}/nacos/v1/console/namespaces

Table 6-1 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Unique ID of a tenant's sub-project. The value contains 1 to 64 characters.

Table 6-2 Query Parameters

Parameter	Mandatory	Type	Description
offset	Yes	Integer	Offset. The value starts from **0**.
limit	Yes	Integer	Page size. Value **0** indicates that all results are displayed on one page.

Request Parameters

Table 6-3 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	IAM-Token.
x-engine-id	Yes	String	Instance ID of an exclusive microservice engine.
X-Enterprise-Project-ID	Yes	String	Enterprise project ID.

Response Parameters

Status code: 400

Table 6-4 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
GET https://{endpoint}/v1/{project_id}/nacos/v1/console/namespaces
```

Example Responses

None

Status Codes

Status Code	Description
200	OK
400	Bad Request

Error Codes

See **CSE Error Codes**.

6.2 Creating the nacos Namespace

Function

This API is used to create the nacos namespace.

URI

POST /v1/{project_id}/nacos/v1/console/namespaces

Table 6-5 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Unique ID of a tenant's sub-project. The value contains 1 to 64 characters.

Table 6-6 Query Parameters

Parameter	Mandatory	Type	Description
custom_namespace_id	Yes	String	Namespace ID. The value can contain a maximum of 128 characters, including uppercase letters, lowercase letters, digits, hyphens (-), and underscores (_).
namespace_name	Yes	String	Namespace name. The value can contain a maximum of 128 characters, excluding @, #, $, %, ^, &, and *.
namespace_desc	No	String	Namespace description. The value can contain a maximum of 256 characters.

Request Parameters

Table 6-7 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	IAM-Token.
x-engine-id	Yes	String	Instance ID of an exclusive microservice engine.

Parameter	Mandatory	Type	Description
X-Enterprise-Project-ID	Yes	String	Enterprise project ID.

Response Parameters

Status code: 400

Table 6-8 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
POST https://{endpoint}/v1/{project_id}/nacos/v1/console/namespaces
```

Example Responses

None

Status Codes

Status Code	Description
200	OK
400	Bad Request

Error Codes

See **CSE Error Codes**.

6.3 Updating the nacos Namespace

Function

This API is used to update the nacos namespace.

URI

PUT /v1/{project_id}/nacos/v1/console/namespaces

Table 6-9 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Unique ID of a tenant's sub-project. The value contains 1 to 64 characters.

Table 6-10 Query Parameters

Parameter	Mandatory	Type	Description
namespace	Yes	String	Namespace ID.
namespace_show_name	Yes	String	Namespace name. The value can contain a maximum of 128 characters, excluding @, #, $, %, ^, &, and *.
namespace_desc	Yes	String	Namespace description. The value can contain a maximum of 256 characters.

Request Parameters

Table 6-11 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	IAM-Token.
x-engine-id	Yes	String	Instance ID of an exclusive microservice engine.
X-Enterprise-Project-ID	Yes	String	Enterprise project ID.

Response Parameters

Status code: 400

Table 6-12 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
PUT https://{endpoint}/v1/{project_id}/nacos/v1/console/namespaces
```

Example Responses

None

Status Codes

Status Code	Description
200	OK
400	Bad Request

Error Codes

See **CSE Error Codes**.

6.4 Deleting the nacos Namespace

Function

This API is used to delete the nacos namespace.

URI

DELETE /v1/{project_id}/nacos/v1/console/namespaces

Table 6-13 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Unique ID of a tenant's sub-project. The value contains 1 to 64 characters.

Table 6-14 Query Parameters

Parameter	Mandatory	Type	Description
namespace_id	Yes	String	Namespace ID.

Request Parameters

Table 6-15 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	IAM-Token.
x-engine-id	Yes	String	Instance ID of an exclusive microservice engine.
X-Enterprise-Project-ID	Yes	String	Enterprise project ID.

Response Parameters

Status code: 400

Table 6-16 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_msg	String	Error message.
detail	String	Location details.

Example Requests

```
DELETE https://{endpoint}/v1/{project_id}/nacos/v1/console/namespaces
```

Example Responses

None

Status Codes

Status Code	Description
200	OK

Status Code	Description
400	Bad Request

Error Codes

See **CSE Error Codes**.

7 ServiceComb API

7.1 API Calling

CSE provides REST APIs, allowing you to call APIs using HTTPS.

NOTICE

APIs in **Microservice**, **Schema**, **Microservice Instance**, and **Dependency** are available only in ME-Riyadh, CN-Hong Kong, and AP-Singapore.

To call the ServiceComb APIs of an exclusive ServiceComb engine, do as follows:

1. Log in to CSE.
2. Choose **Exclusive ServiceComb Engines**.
3. Click the target ServiceComb engine.

   When calling the APIs in **Authentication**, **Microservice**, **Schema**, **Microservice Instance**, and **Dependency**, view or click to copy the service center address of the engine.

   When calling the APIs in **Configuration Management**, view or click to copy the configuration center address of the engine.
4. Call the API by referring to **Calling APIs**. In **Request URI**, replace **{Endpoint}** with the obtained service center address.

7.2 Authentication

7.2.1 Obtaining the User Token of an Exclusive ServiceComb Engine

Function

This API is intended only for exclusive ServiceComb engines with security authentication enabled.

Before accessing APIs of an exclusive ServiceComb engine with security authentication enabled, you need to call this API to obtain a user token. This token is the unique credential for accessing the APIs.

The token must be contained in the request header when an access request is sent. The format is as follows:

```
Authorization:Bearer{Token}
```

Restrictions

None

URI

POST /v4/token

Request

Table 7-1 Request body parameters

Parameter	Mandatory	Type	Description
name	Yes	String	Account name.
password	Yes	String	Password of the account.

Response

Status code: 200

Table 7-2 Response body parameter

Parameter	Type	Description
token	String	User token, which is valid for 12 hours.

Status code: 401

Table 7-3 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-4 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

```
POST https://{Service center address}/v4/token

{
    "name": "root",
    "password": "********"
}
```

Example Response

Status code: 200

OK

```
{
  "token" :
"****bGciOiJSUzUxMiIsInR5cCI6IkpXVCJ9.eyJhY2NvdW50IjoicmV9vdCIsImV4cCI6MTY1MDU5MTcwMSwicm9sZXMiOlsiYWRtaW4iXX0.WKwNAjaYMMCSjNX0qCGCeyh13FJRzLousxoXlThdkMwkH-pXEmG51_SguH0LlHOZoIc8gNJq-ilQg4bxTo1s0pnQZIS3wma0qvE-MzaYnFguTuHM7rxD7eZdwnbUe3dhnw9xRqR1hcd-lTuBbLoL9fbED4U_63IoEDyBCJl9D_l0F86uGzpUysCvC-t6MrJHgi7miUaO7ZZQmSAUNhmbEoN8IIVp-QtP_cWNWtWaFO-eoQrmCT2FdlYiB9MCuELr9-5EGM_mFLPgs6E4fyIGiGHy7IwoGUKOCW5w6Jb0l-2JxeUe3eOl5Md5kzOIAE_EYUATxCbJ5GmgpSSJf*****"
}
```

Status Code

Status Code	Description
200	OK
401	Unauthorized
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.3 Microservice

7.3.1 Querying Information About a Microservice

Function

This API is used to query the definition information about a microservice based on **service_id**.

Restrictions

None

URI

GET /v4/{project_id}/registry/microservices/{service_id}

Table 7-5 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.
service_id	Yes	String	Microservice ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^.*$. See **Querying Information About All Microservices**.

Request

Table 7-6 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Response

Status code: 200

Table 7-7 Response body parameter

Parameter	Type	Description
service	**MicroService** object	Microservice information.

Table 7-8 MicroService

Parameter	Type	Description
serviceId	String	Microservice ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^.*$
environment	String	Microservice environment. Value: development, testing, acceptance, or production.

Parameter	Type	Description
appId	String	Application ID, which must be unique. The value contains 1 to 160 characters. Regular expression: ^[a-zA-Z0-9]$\|^[a-zA-Z0-9][a-zA-Z0-9_-.][a-zA-Z0-9]$
serviceName	String	Microservice name, which must be unique in an application. The value contains 1 to 128 characters. Regular expression: ^[a-zA-Z0-9]$\|^[a-zA-Z0-9][a-zA-Z0-9_-.][a-zA-Z0-9]$
version	String	Microservice version. The value contains 1 to 64 characters. Regular expression: ^[0-9]$\|^[0-9]+(.[0-9]+)$
description	String	Microservice description. The value contains a maximum of 256 characters.
level	String	Microservice level. Value: FRONT, MIDDLE, or BACK.
registerBy	String	Microservice registration mode. Value: SDK, PLATFORM, SIDECAR, or UNKNOWN.
schemas	Array of strings	Microservice schema content. The value must be 1 to 160 bytes long. Only digits, letters, underscores (_), hyphens (-), and periods (.) are allowed. An array contains a maximum of 100 schemas.
status	String	Microservice status. Value: UP or DOWN. Default value: UP. Value: • UP • DOWN
timestamp	String	Microservice registration time.
modTimestamp	String	Latest modification time (UTC).
framework	**Framework** object	Development framework.
paths	Array of **ServicePath** objects	Service path.

Table 7-9 Framework

Parameter	Type	Description
name	String	Microservice development framework. Default value: UNKNOWN.
version	String	Version of the microservice development framework.

Table 7-10 ServicePath

Parameter	Type	Description
Path	String	Route address.
Property	Object	Extended attribute. You can customize a key and value. The value must be at least 1 byte long.

Status code: 400

Table 7-11 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-12 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Query details about the microservice whose ID is **819706e21b7173306797d19922ce4231441c17c5**.


```
GET https://{endpoint}/v4/{project_id}/registry/microservices/
819706e21b7173306797d19922ce4231441c17c5
```

Example Response

Status code: 200

Successfully queried.

```
{
   "service": {
      "serviceId": "819706e21b7173306797d19922ce4231441c17c5",
      "appId": "default",
      "serviceName": "SERVICECENTER",
      "version": "2.4.8",
      "level": "BACK",
      "schemas": [
         "servicecenter.grpc.api.ServiceCtrl",
         "servicecenter.grpc.api.ServiceInstanceCtrl"
      ],
      "status": "UP",
      "timestamp": "1616426688",
      "modTimestamp": "1616426688",
      "environment": "development"
   }
}
```

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.3.2 Deleting Definition Information About a Microservice

Function

This API is used to delete definition and related information about a microservice, and deregister all instances of the microservice.

Restrictions

None

URI

DELETE /v4/{project_id}/registry/microservices/{service_id}

Table 7-13 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.
service_id	Yes	String	Microservice ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^.*$. See **Querying Information About All Microservices**.

Table 7-14 Query parameter

Parameter	Mandatory	Type	Description
force	No	Boolean	Whether to forcibly delete information about a microservice. true: forcible deletion; false: non-forcible deletion. If you choose forcible deletion, all service instances are automatically deregistered and related service dependencies are deleted. If this parameter is not transferred, the service cannot be deleted when it has an instance. Default value: false.

Request

Table 7-15 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Response

Status code: 400

Table 7-16 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-17 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Delete the microservice whose ID is **e0f0da073f2c91e8979a89ff2d7c69t6**.

```
DELETE https://{endpoint}/v4/d9f4da085f2c11e8959a00ff2d7c69b7/registry/microservices/
e0f0da073f2c91e8979a89ff2d7c69t6
```

Example Response

None

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.3.3 Querying Information About All Microservices

Function

This API is used to query definition information about microservices that meet filter criteria.

Restrictions

None

URI

GET /v4/{project_id}/registry/microservices

Table 7-18 Path parameter

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.

Request

Table 7-19 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Response

Status code: 200

Table 7-20 Response body parameter

Parameter	Type	Description
services	Array of **MicroService** objects	Microservice list.

Table 7-21 MicroService

Parameter	Type	Description
serviceId	String	Microservice ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^.*$

Parameter	Type	Description
environment	String	Microservice environment. Value: development, testing, acceptance, or production. You can use the API for uploading schemas in batches to add or modify an existing schema only when the value is **development**, **testing**, or **acceptance**. Default value: development.
appId	String	Application ID, which must be unique. The value contains 1 to 160 characters. Regular expression: ^[a-zA-Z0-9]$\|^[a-zA-Z0-9][a-zA-Z0-9_-.][a-zA-Z0-9]$
serviceName	String	Microservice name, which must be unique in an application. The value contains 1 to 128 characters. Regular expression: ^[a-zA-Z0-9]$\|^[a-zA-Z0-9][a-zA-Z0-9_-.][a-zA-Z0-9]$
version	String	Microservice version. The value contains 1 to 64 characters. Regular expression: ^[0-9]$\|^[0-9]+(.[0-9]+)$
description	String	Microservice description. The value contains a maximum of 256 characters.
level	String	Microservice level. Value: FRONT, MIDDLE, or BACK.
registerBy	String	Microservice registration mode. Value: SDK, PLATFORM, SIDECAR, or UNKNOWN.
schemas	Array of strings	Microservice schema content. The value must be 1 to 160 bytes long. Only digits, letters, underscores (_), hyphens (-), and periods (.) are allowed. An array contains a maximum of 100 schemas.
status	String	Microservice status. Value: UP or DOWN. Default value: UP.
timestamp	String	Microservice registration time.
modTimestamp	String	Latest modification time (UTC).
framework	**Framework** object	Development framework.
paths	Array of **ServicePath** objects	Service path.

Table 7-22 Framework

Parameter	Type	Description
name	String	Microservice development framework. Default value: UNKNOWN.
version	String	Version of the microservice development framework.

Table 7-23 ServicePath

Parameter	Type	Description
Path	String	Route address.
Property	Object	Extended attribute. You can customize a key and value. The value must be at least 1 byte long.

Status code: 400

Table 7-24 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-25 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Query information about all registered microservices.

```
GET https://{endpoint}/v4/{project_id}/registry/microservices
```


Example Response

Status code: 200

Successfully queried.


```
{
   "services": [
      {
         "serviceId": "8aed80ea052ac04a64dfc79c24f2170224d074f5",
         "appId": "default",
         "serviceName": "test",
         "version": "1.0.0",
         "description": "this is a test",
         "level": "BACK",
         "status": "UP",
         "timestamp": "1650543950",
         "modTimestamp": "1650543950"
      },
      {
         "serviceId": "dcc6c1073eab3cadb47cea2e1a874b7883b02a63",
         "appId": "test",
         "serviceName": "test1",
         "version": "1.0.0",
         "level": "BACK",
         "status": "UP",
         "timestamp": "1650544223",
         "modTimestamp": "1650544223"
      }
   ]
}
```


Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.3.4 Creating Static Information for a Microservice

Function

This API is used to create static information for a microservice before registering a microservice instance. The registered instance is associated with the static information based on **serviceId**. One service corresponds to multiple instances.

serviceId can be customized. If **serviceId** is not customized, the system generates a random service ID.

Restrictions

None

URI

POST /v4/{project_id}/registry/microservices

Table 7-26 Path parameter

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.

Request

Table 7-27 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Table 7-28 Request body parameters

Parameter	Mandatory	Type	Description
service	Yes	**MicroService** object	Microservice information.
rules	No	Array of **Rule** objects	Blacklist and whitelist.

Parameter	Mandatory	Type	Description
instances	No	Array of **MicroServiceInstance** objects	Instance information.
tags	No	Object	Extended attribute. You can customize a key and value. The value must be at least 1 byte long.

Table 7-29 MicroService

Parameter	Mandatory	Type	Description
serviceId	No	String	Microservice ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^.*$
environment	No	String	Microservice environment. Value: development, testing, acceptance, or production. You can use the API for uploading schemas in batches to add or modify an existing schema only when the value is **development**, **testing**, or **acceptance**. Default value: development.
appId	No	String	Application ID, which must be unique. The value contains 1 to 160 characters. Regular expression: ^[a-zA-Z0-9]$\|^[a-zA-Z0-9][a-zA-Z0-9_-.][a-zA-Z0-9]$
serviceName	Yes	String	Microservice name, which must be unique in an application. The value contains 1 to 128 characters. Regular expression: ^[a-zA-Z0-9]$\|^[a-zA-Z0-9][a-zA-Z0-9_-.][a-zA-Z0-9]$
version	No	String	Microservice version. The value contains 1 to 64 characters. Regular expression: ^[0-9]$\|^[0-9]+(.[0-9]+)$

Parameter	Mandatory	Type	Description
description	No	String	Microservice description. The value contains a maximum of 256 characters.
level	No	String	Microservice level. Value: FRONT, MIDDLE, or BACK.
registerBy	No	String	Microservice registration mode. Value: SDK, PLATFORM, SIDECAR, or UNKNOWN.
schemas	No	Array of strings	Foreign key ID of a microservice access schema. The array length supports a maximum of 100 schemas.
status	No	String	Microservice status. Value: UP or DOWN. Default value: UP.
timestamp	No	String	Microservice registration time.
modTimestamp	No	String	Latest modification time (UTC).
framework	No	**Framework** object	Development framework.
paths	No	Array of **ServicePath** objects	Service path.

Table 7-30 Framework

Parameter	Mandatory	Type	Description
name	No	String	Microservice development framework. Default value: **UNKNOWN**.
version	No	String	Version of the microservice development framework.

Table 7-31 ServicePath

Parameter	Mandatory	Type	Description
Path	No	String	Route address.

Parameter	Mandatory	Type	Description
Property	No	Object	Extended attribute. You can customize a key and value. The value must be at least 1 byte long.

Table 7-32 Rule

Parameter	Mandatory	Type	Description
ruleId	No	String	Customized rule ID.
ruleType	No	String	Rule type. Value: WHITE or BLACK.
attribute	No	String	If the value starts with **tag_*xxx***, the attributes are filtered by **Tag**. Otherwise, the attributes are filtered by **serviceId**, **AppId**, **ServiceName**, **Version**, **Description**, **Level**, or **Status**.
pattern	No	String	Matching rule. The value is a regular expression containing 1 to 64 characters.
description	No	String	Rule description.
timestamp	No	String	Time when a rule is created. This parameter is used only when you query rules.
modTimestamp	No	String	Update time.

Table 7-33 MicroServiceInstance

Parameter	Mandatory	Type	Description
instanceId	No	String	Instance ID, which must be unique. The instance ID is generated by the service center.
serviceId	No	String	Microservice ID, which must be unique. During instance creation, use the service ID in the URL instead of the service ID here.

Parameter	Mandatory	Type	Description
version	No	String	Microservice version.
hostName	Yes	String	Host information.
endpoints	Yes	Array of strings	Access address information.
status	No	String	Instance status. Value: UP, DOWN, STARTING, or OUTOFSERVICE. Default value: UP.
properties	No	Object	Extended attribute. You can customize a key and value. The value must be at least 1 byte long.
healthCheck	No	**HealthCheck** object	Health check information.
dataCenterInfo	No	**DataCenterInfo** object	Data center information.
timestamp	No	String	Time when an instance is created, which is automatically generated.
modTimestamp	No	String	Update time.

Table 7-34 HealthCheck

Parameter	Mandatory	Type	Description
mode	Yes	String	Heartbeat mode. Value: push or pull.
port	No	Integer	Port.
interval	Yes	Integer	Heartbeat interval. Unit: s. If the value is less than 5s, the registration is performed at an interval of 5s.
times	Yes	Integer	Maximum retries.

Table 7-35 DataCenterInfo

Parameter	Mandatory	Type	Description
name	Yes	String	Region name.

Parameter	Mandatory	Type	Description
region	Yes	String	Region.
availableZone	Yes	String	AZ.

Response

Status code: 200

Table 7-36 Response body parameter

Parameter	Type	Description
serviceId	String	Microservice ID, which must be unique.

Status code: 400

Table 7-37 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-38 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Create static information about a microservice: microservice name is **test**, microservice version is **1.0.0**, rule type is whitelist, host information of the instance is **instanceTest**, and access address is **rest:127.0.0.1:8080**.

```
POST https://{endpoint}/v4/{project_id}/registry/microservices
```



```
{
    "service": {
        "appId": "default",
        "serviceName": "test",
        "version": "1.0.0",
        "description": "this is a test"
    },
    "rules": [{
        "ruleType": "WHITE",
        "attribute": "tag_123",
        "pattern": "aaa"
    }],
    "instances": [{
        "hostName": "instanceTest",
        "endpoints": ["rest:127.0.0.1:8080"]
    }],
    "tags": {
        "test_tag1": "test_tag1",
        "test_tag2": "test_tag2",
        "test_tag3": "test_tag3"
    }
}
```

Example Response

Status code: 200

Successfully created. **serviceId** indicates the ID generated for the microservice. Subsequent operations are operated based on the microservice ID.

```
{
  "serviceId" : "8aed80ea052ac04a64dfc79c24f2170224d074f5"
}
```

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.3.5 Deleting Static Information About Microservices in Batches

Function

This API is used to delete the definitions and related information about microservices in batches, and deregister all instances of the microservices.

Restrictions

None

URI

DELETE /v4/{project_id}/registry/microservices

Table 7-39 Path parameter

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.

Table 7-40 Query parameter

Parameter	Mandatory	Type	Description
force	No	Boolean	Whether to forcibly delete static information about microservices. true: forcible deletion; false: non-forcible deletion. If you choose forcible deletion, all service instances are automatically deregistered and related service dependencies are deleted. If this parameter is not transferred, the service cannot be deleted when it has an instance. Default value: false.

Request

Table 7-41 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Table 7-42 Request body parameter

Parameter	Mandatory	Type	Description
serviceIds	Yes	Array of strings	Service ID list. See **Querying Information About All Microservices**.

Response

Status code: 200

Table 7-43 Response body parameter

Parameter	Type	Description
services	Array of **DelServicesRspInfo** objects	List of services to be deleted.

Table 7-44 DelServicesRspInfo

Parameter	Type	Description
serviceId	String	Microservice ID.

Status code: 400

Table 7-45 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-46 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Delete the static information about the microservices whose IDs are **id1** and **id2**.

```
DELETE https://{endpoint}/v4/{project_id}/registry/microservices

{
  "serviceIds" : [ "id1", "id2"]
}
```

Example Response

Status code: 200

Successfully deleted.

```
{
  "services" : [ {
    "serviceId" : "id1"
  },
 {
    "serviceId" : "id2"
  }
```



```
]
}
```

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.3.6 Modifying Extended Attributes of a Microservice

Function

This API is used to modify static information about a microservice. To update some fields in the static information, input all static information (including the fields that do not need to be updated) in JSON format.

Restrictions

None

URI

PUT /v4/{project_id}/registry/microservices/{service_id}/properties

Table 7-47 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.
service_id	Yes	String	Microservice ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^.*$. See **Querying Information About All Microservices**.

Request

Table 7-48 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Table 7-49 Request body parameter

Parameter	Mandatory	Type	Description
properties	No	Object	Extended attribute. You can customize a key and value. The value must be at least 1 byte long.

Response

Status code: 400

Table 7-50 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-51 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Modify extended information about a microservice with specified **service_id**.

```
PUT https://{endpoint}/v4/{project_id}/registry/microservices/{service_id}/properties

{
  "properties" : {
    "a" : "test"
  }
}
```

Example Response

None

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.3.7 Querying the Unique Service or Schema ID of a Microservice

Function

This API is used to query the unique service or schema ID of a microservice based on filter criteria.

URI

GET /v4/{project_id}/registry/existence

Table 7-52 Path parameter

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.

Table 7-53 Query parameters

Parameter	Mandatory	Type	Description
type	Yes	String	Request type. • microservice • schema **NOTE** • If **type** is set to **microservice**, **env**, **appId**, **serviceName**, and **version** are mandatory. • If **type** is set to **schema**, **serviceId** and **schemaId** are mandatory.
env	No	String	Microservice environment. Value: development, testing, acceptance, or production.
appId	Yes	String	Application ID, which needs to be transferred when the resource type is **microservice**. The value contains 1 to 160 characters. Regular expression: ^[a-zA-Z0-9]$\|^[a-zA-Z0-9][a-zA-Z0-9_-.][a-zA-Z0-9]$
serviceName	Yes	String	Microservice name, which needs to be transferred when the resource type is **microservice**. The value contains 1 to 128 characters. Regular expression: ^[a-zA-Z0-9]$\|^[a-zA-Z0-9][a-zA-Z0-9_-.][a-zA-Z0-9]$
version	Yes	String	Microservice version, which needs to be transferred when the resource type is **microservice**. The value contains 1 to 64 characters. Regular expression: ^[0-9]$\|^[0-9]+(.[0-9]+)$

Parameter	Mandatory	Type	Description
serviceId	Yes	String	Microservice ID, which needs to be transferred when the resource type is **schema**. The value contains 1 to 64 characters. Regular expression: ^.*$
schemaId	Yes	String	Schema ID, which needs to be transferred when the resource type is **schema**. The value contains 1 to 160 characters. Regular expression: ^[a-zA-Z0-9]{1,160}$\|^[a-zA-Z0-9][a-zA-Z0-9_-.]{0,158}[a-zA-Z0-9]$.

Request

Table 7-54 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Response

Status code: 200

Table 7-55 Response header parameter

Parameter	Type	Description
X-Schema-Summary	String	Summary of a microservice schema.

Table 7-56 Response body parameters

Parameter	Type	Description
serviceId	String	When a service is queried, a service ID is returned.
schemaId	String	When a schema is queried, a schema ID is returned.

Status code: 400

Table 7-57 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-58 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Query **serviceId** based on **microservice** as **type**, **default** as **appId**, **service** as **serviceName**, and **1.0.0** as **version**.

```
GET https://{endpoint}/v4/{project_id}/registry/existence?
type=microservice&appId=default&serviceName=service&version=1.0.0
```


Example Response

Status code: 200

Successfully queried. **serviceId** or **schemaId** is returned.

```
{
  "serviceId" : "8aed80ea052ac04a64dfc79c24f2170224d074f5"
}
```

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.4 Schema

7.4.1 Querying a Microservice Schema

Function

Query a microservice schema based on **service_id** and **schema_id**.

Restrictions

None

URI

GET /v4/{project_id}/registry/microservices/{service_id}/schemas/{schema_id}

Table 7-59 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.

Parameter	Mandatory	Type	Description
service_id	Yes	String	Microservice ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^.*$. See **Querying Information About All Microservices**.
schema_id	Yes	String	Schema ID. See **Querying All Schema Information About a Microservice**.

Request

Table 7-60 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Response

Status code: 200

Table 7-61 Response header parameter

Parameter	Type	Description
X-Schema-Summary	String	Schema summary.

Table 7-62 Response body parameter

Parameter	Type	Description
schema	String	Schema content.

Status code: 400

Table 7-63 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-64 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Query details about a schema with specified **service_id** and **schema_id**.

```
GET https://{endpoint}/v4/{project_id}/registry/microservices/{service_id}/schemas/{schema_id}
```

Example Response

Status code: 200

Successfully queried. If the summary exists, the value of **X-Schema-Summary** in the header is the summary of the schema.

```
{
  "schema" : "---\nswagger: \"2.0\"\ninfo:\n  version: \"1.0.0\"\n  title: \"swagger definition for com.service.provider.controller.ProviderImpl\"\n  x-java-interface: \"cse.gen.springmvc.provider.provider.ProviderImplIntf\"\nbasePath: \"/provider\"\nconsumes:\n- \"application/json\"\nproduces:\n- \"application/json\"\npaths:\n  /helloworld:\n    get:\n      operationId: \"helloworld\"\n      produces:\n      - \"application/json\"\n      parameters:\n      - name: \"name\"\n        in: \"query\"\n        required: true\n        type: \"string\"\n      responses:\n        200:\n          description: \"response of 200\"\n          schema:\n            type: \"string\"\n"
}
```


Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.4.2 Modifying a Microservice Schema

Function

Modify a microservice schema based on **schema_id**.

Restrictions

In versions earlier than 2.3.33, schema overwriting is not supported in production environment. In version 2.3.33 and later, the environment variable **SCHEMA_EDITABLE** can be used to support schema overwriting.

URI

PUT /v4/{project_id}/registry/microservices/{service_id}/schemas/{schema_id}

Table 7-65 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.
service_id	Yes	String	Microservice ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^.*$. See **Querying Information About All Microservices**.

Parameter	Mandatory	Type	Description
schema_id	Yes	String	Microservice schema ID, which must be unique. The value contains 1 to 160 characters. Regular expression: ^[a-zA-Z0-9]{1,160}$\|^[a-zA-Z0-9][a-zA-Z0-9_-.]{0,158}[a-zA-Z0-9]$. See **Querying All Schema Information About a Microservice**.

Request

Table 7-66 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Table 7-67 Request body parameters

Parameter	Mandatory	Type	Description
schema	Yes	String	Schema content.
summary	No	String	Schema summary.

Response

Status code: 400

Table 7-68 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-69 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Modify the content and summary of a schema with specified **service_id** and **schema_id**.

```
PUT https://{endpoint}/v4/{project_id}/registry/microservices/{service_id}/schemas/{schema_id}

{
  "schema": "---\nswagger: \"2.0\"\ninfo:\n  version: \"1.0.0\"\n  title: \"swagger definition for com.service.provider.controller.ProviderImpl\"\n  x-java-interface: \"cse.gen.springmvc.provider.provider.ProviderImplIntf\"\nbasePath:  \"/provider\"\nconsumes:\n- \"application/json\"\nproduces:\n- \"application/json\"\npaths:\n  /helloworld:\n    get:\n      operationId: \"helloworld\"\n      produces:\n      - \"application/json\"\n      parameters:\n      - name: \"name\"\n        in: \"query\"\n        required: true\n        type: \"string\"\n      responses:\n        200:\n          description: \"response of 200\"\n          schema:\n            type: \"string\"\n",
  "summary": "test"
}
```

Example Response

None

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.4.3 Querying All Schema Information About a Microservice

Function

Query all schema information (including **schemaId** and **summary**) about a microservice.

URI

GET /v4/{project_id}/registry/microservices/{service_id}/schemas

Table 7-70 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.
service_id	Yes	String	Microservice ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^.*$. See **Querying Information About All Microservices**.

Table 7-71 Query parameter

Parameter	Mandatory	Type	Description
withSchema	No	Integer	Whether to query the schema content. Default value: 0. 0: Only **schema_id** and **summary** are displayed. 1: **schema_id**, **summary**, and **schema** are displayed.

Request

Table 7-72 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Response

Status code: 200

Table 7-73 Response body parameter

Parameter	Type	Description
schemas	Array of **Schema** objects	Schema list.

Table 7-74 Schema

Parameter	Type	Description
schemaId	String	Schema ID.
schema	String	Schema content.
summary	String	Schema summary.

Status code: 400

Table 7-75 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-76 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Query the schema of a microservice with specified **service_id**, including **schema_id**, **summary**, and **schema**.

```
GET /v4/{project_id}/registry/microservices/{service_id}/schemas?withSchema=1
```

Example Response

Status code: 200

Successfully queried.

```
{
    "schemas": [{
        "schemaId": "xxxxmvc",
        "schema": "---\nswagger: \"2.0\"\ninfo:\n  version: \"1.0.0\"\n  title: \"swagger definition for com.service.provider.controller.ProviderImpl\"\n  x-java-interface: \"cse.gen.springmvc.provider.provider.ProviderImplIntf\"\nbasePath: \"/provider\"\nconsumes:\n- \"application/json\"\nproduces:\n- \"application/json\"\npaths:\n  /helloworld:\n    get:\n      operationId: \"helloworld\"\n      produces:\n      - \"application/json\"\n      parameters:\n      - name: \"name\"\n        in: \"query\"\n        required: true\n        type: \"string\"\n      responses:\n        200:\n          description: \"response of 200\"\n          schema:\n            type: \"string\"\nxxxx",
        "summary": "abcda7b4072ef2d7a5fc9aefccf03e5548029ae31c6cd5fc29da7685d6d9e14adea3"
    }]
}
```

Status Code

Status Code	Description
200	OK

Status Code	Description
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.5 Microservice Instance

7.5.1 Registering a Microservice Instance

Function

This API is used to register a microservice instance after a microservice is created.

Information about the instance must be provided during registration.

instanceid can be customized. If it is customized, the new instance ID will overwrite the original one. If it is not customized, the system automatically generates an ID. If the endpoints are duplicate, the original ID is used.

URI

POST /v4/{project_id}/registry/microservices/{service_id}/instances

Table 7-77 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.
service_id	Yes	String	Microservice ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^.*$. See **Querying Information About All Microservices**.

Request

Table 7-78 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Table 7-79 Request body parameter

Parameter	Mandatory	Type	Description
instance	Yes	**MicroServiceInstance** object	Microservice instance information.

Table 7-80 MicroServiceInstance

Parameter	Mandatory	Type	Description
instanceId	No	String	Instance ID, which must be unique. The instance ID is generated by the service center.
serviceId	No	String	Microservice ID, which must be unique. During instance creation, use the microservice ID in the URL instead of the microservice ID specified here.
version	No	String	Microservice version.
hostName	Yes	String	Host information.

Parameter	Mandatory	Type	Description
endpoints	No	Array of strings	Access address information.
status	No	String	Instance status. Value: UP, DOWN, STARTING, or OUTOFSERVICE. Default value: UP.
properties	No	Object	Extended attribute. You can customize a key and value. The value must be at least 1 byte long.
healthCheck	No	**HealthCheck** object	Health check information.
dataCenterInfo	No	**DataCenterInfo** object	Data center information.
timestamp	No	String	Time when an instance is created, which is automatically generated.
modTimestamp	No	String	Update time.

Table 7-81 HealthCheck

Parameter	Mandatory	Type	Description
mode	Yes	String	Heartbeat mode. Value: push or pull.
port	No	Integer	Port.
interval	Yes	Integer	Heartbeat interval. Unit: s. If the value is less than 5s, the registration is performed at an interval of 5s.
times	Yes	Integer	Maximum retries.

Table 7-82 DataCenterInfo

Parameter	Mandatory	Type	Description
name	Yes	String	Region name.
region	Yes	String	Region.

Parameter	Mandatory	Type	Description
availableZone	Yes	String	AZ.

Response

Status code: 200

Table 7-83 Response body parameter

Parameter	Type	Description
instanceId	String	Instance ID.

Status code: 400

Table 7-84 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-85 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Register a microservice instance: host information is **localhost**, heartbeat mode is **push**, heartbeat interval is **30s**, region is **r1**, name is **dc**, and AZ is **az1**.

```
POST https://{endpoint}/v4/{project_id}/registry/microservices/{service_id}/instances

{
  "instance" : {
    "endpoints" : [ "grpc://127.0.1.312:9980", "rest://127.0.0.111:8081" ],
    "hostName" : "localhost",
```



```
  "status" : "UP",
  "properties" : {
    "_TAGS" : "A, B",
    "attr1" : "a",
    "nodeIP" : "127.0.0.1"
  },
  "dataCenterInfo" : {
    "name" : "dc",
    "region" : "r1",
    "availableZone" : "az1"
  },
  "healthCheck" : {
    "mode" : "push",
    "interval" : 30,
    "times" : 3
  }
 }
}
```

Example Response

Status code: 200

Successfully registered.

```
{
 "instanceId" : "8540bb8b693c4ad1a7fb6a756c415244"
}
```

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.5.2 Querying a Microservice Instance Based on service_id

Function

This API is used to query all instances of a microservice based on **service_id** after the instances are registered.

URI

GET /v4/{project_id}/registry/microservices/{service_id}/instances

Table 7-86 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.
service_id	Yes	String	Microservice ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^.*$. See **Querying Information About All Microservices**.

Table 7-87 Query parameter

Parameter	Mandatory	Type	Description
tags	No	String	Tag. When there are multiple tags, separate them using commas (,). Regular expression: ^[a-zA-Z][a-zA-Z0-9_-.]{0,63}$

Request

Table 7-88 Request header parameters

Parameter	Mandatory	Type	Description
X-ConsumerId	No	String	Microservice consumer ID, which must be unique.
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Response

Status code: 200

Table 7-89 Response body parameter

Parameter	Type	Description
instances	Array of **MicroServiceInstance** objects	Instance list.

Table 7-90 MicroServiceInstance

Parameter	Type	Description
instanceId	String	Instance ID, which must be unique. The instance ID is generated by the service center.
serviceId	String	Microservice ID, which must be unique. During instance creation, use the microservice ID in the URL instead of the microservice ID specified here.
version	String	Microservice version.
hostName	String	Host information.
endpoints	Array of strings	Access address information.
status	String	Instance status. Value: UP, DOWN, STARTING, or OUTOFSERVICE. Default value: UP.
properties	Object	Extended attribute. You can customize a key and value. The value must be at least 1 byte long.
healthCheck	**HealthCheck** object	Health check information.
dataCenterInfo	**DataCenterInfo** object	Data center information.
timestamp	String	Time when an instance is created, which is automatically generated.
modTimestamp	String	Update time.

Table 7-91 HealthCheck

Parameter	Type	Description
mode	String	Heartbeat mode. Value: push or pull.
port	Integer	Port.
interval	Integer	Heartbeat interval. Unit: s. If the value is less than 5s, the registration is performed at an interval of 5s.
times	Integer	Maximum retries.

Table 7-92 DataCenterInfo

Parameter	Type	Description
name	String	Region name.
region	String	Region.
availableZone	String	AZ.

Status code: 400

Table 7-93 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-94 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Query instance information about a microservice with specified **service_id**.

```
GET https://{endpoint}/v4/{project_id}/registry/microservices/{service_id}/instances
```

Example Response

Status code: 200

Successfully queried.

```
{
   "instances": [
      {
         "instanceId": "8540bb8b693c4ad1a7fb6a756c415244",
         "serviceId": "8aed80ea052ac04a64dfc79c24f2170224d074f5",
         "endpoints": [
            "rest:127.0.0.1:8080"
         ],
         "hostName": "hostNameTest",
         "status": "UP",
         "properties": {
            "engineID": "30c263e5-2eac-4da1-9c72-5abb9ac94550",
            "engineName": "cse-fkln1-HA"
         },
         "healthCheck": {
            "mode": "push",
            "interval": 30,
            "times": 3
         },
         "timestamp": "1650545035",
         "modTimestamp": "1650545035",
         "version": "1.0.0"
      }
   ]
}
```

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.5.3 Deregistering a Microservice Instance

Function

This API is used to deregister an instance based on **instance_id**.

URI

DELETE /v4/{project_id}/registry/microservices/{service_id}/instances/{instance_id}

Table 7-95 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.
service_id	Yes	String	Microservice instance ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^[A-Za-z0-9_.-]*$. See **Querying a Microservice Instance Based on service_id**.
instance_id	Yes	String	Microservice instance ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^[A-Za-z0-9_.-]*$. See **Querying a Microservice Instance Based on service_id**.

Request

Table 7-96 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Response

Status code: 400

Table 7-97 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-98 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Deregister an instance with specified **instance_id**.

```
DELETE https://{endpoint}/v4/{project_id}/registry/microservices/{service_id}/instances/{instance_id}
```

Example Response

None

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.5.4 Querying Details About a Microservice Instance

Function

This API is used to query details about an instance based on **service_id** and **instance_id** after the instance is registered.

URI

GET /v4/{project_id}/registry/microservices/{service_id}/instances/{instance_id}

Table 7-99 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.
service_id	Yes	String	Microservice ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^.*$. See **Querying Information About All Microservices**.
instance_id	Yes	String	Microservice instance ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^[A-Za-z0-9_.-]*$. See **Querying a Microservice Instance Based on service_id**.

Table 7-100 Query parameter

Parameter	Mandatory	Type	Description
tags	No	String	Tag. When there are multiple tags, separate them using commas (,). Regular expression: ^[a-zA-Z][a-zA-Z0-9_-.]{0,63}$

Request

Table 7-101 Request header parameters

Parameter	Mandatory	Type	Description
X-consumerId	No	String	Microservice consumer ID, which must be unique.
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Response

Status code: 200

Table 7-102 Response body parameter

Parameter	Type	Description
instance	**MicroServiceInstance** object	Microservice instance information.

Table 7-103 MicroServiceInstance

Parameter	Type	Description
instanceId	String	Instance ID, which must be unique. The instance ID is generated by the service center.

Parameter	Type	Description
serviceId	String	Microservice ID, which must be unique. During instance creation, use the microservice ID in the URL instead of the microservice ID specified here.
version	String	Microservice version.
hostName	String	Host information.
endpoints	Array of strings	Access address information.
status	String	Instance status. Value: UP, DOWN, STARTING, or OUTOFSERVICE. Default value: UP.
properties	Object	Extended attribute. You can customize a key and value. The value must be at least 1 byte long.
healthCheck	**HealthCheck** object	Health check information.
dataCenterInfo	**DataCenterInfo** object	Data center information.
timestamp	String	Time when an instance is created, which is automatically generated.
modTimestamp	String	Update time.

Table 7-104 HealthCheck

Parameter	Type	Description
mode	String	Heartbeat mode. Value: push or pull.
port	Integer	Port.
interval	Integer	Heartbeat interval. Unit: s. If the value is less than 5s, the registration is performed at an interval of 5s.
times	Integer	Maximum retries.

Table 7-105 DataCenterInfo

Parameter	Type	Description
name	String	Region name.

Parameter	Type	Description
region	String	Region.
availableZone	String	AZ.

Status code: 400

Table 7-106 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-107 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Query details about an instance with specified **service_id** and **instance_id**.

```
GET https://{endpoint}/v4/{project_id}/registry/microservices/{service_id}/instances/{instance_id}
```

Example Response

Status code: 200

Successfully queried.

```
{
    "instance": {
        "instanceId": "4994929d6b8044b29462d4f7daa33c4f",
        "serviceId": "8aed80ea052ac04a64dfc79c24f2170224d074f5",
        "endpoints": [
            "rest:127.0.0.1:8080"
        ],
        "hostName": "hostNameTest",
        "status": "UP",
        "properties": {
            "engineID": "30c263e5-2eac-4da1-9c72-5abb9ac94550",
```



```
            "engineName": "cse-fkln1-HA"
        },
        "healthCheck": {
            "mode": "push",
            "interval": 30,
            "times": 3
        },
        "timestamp": "1650545270",
        "modTimestamp": "1650545270",
        "version": "1.0.0"
    }
}
```


Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.5.5 Modifying the Extended Information About a Microservice Instance

Function

This API is used to add or update the extended information about a microservice instance based on **instance_id** after the instance is registered.

Restrictions

None

URI

PUT /v4/{project_id}/registry/microservices/{service_id}/instances/{instance_id}/properties

Table 7-108 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.

Parameter	Mandatory	Type	Description
service_id	Yes	String	Microservice ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^.*$. See **Querying Information About All Microservices**.
instance_id	Yes	String	Microservice instance ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^[A-Za-z0-9_.-]*$. See **Querying a Microservice Instance Based on service_id**.

Request

Table 7-109 Request header parameters

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.
X-consumerId	No	String	Microservice consumer ID, which must be unique.

Table 7-110 Request body parameter

Parameter	Mandatory	Type	Description
properties	No	Object	Extended attribute. You can customize a key and value. The value must be at least 1 byte long.

Response

Status code: 400

Table 7-111 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-112 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Update the extended information of an instance with specified **instance_id** to **"attr1": "b"**.

```
PUT https://{endpoint}/v4/{project_id}/registry/microservices/{service_id}/instances/{instance_id}/properties

{
  "properties" : {
    "attr1" : "b"
  }
}
```

Example Response

None

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.5.6 Modifying Status of a Microservice Instance

Function

This API is used to update the status of a microservice instance based on **instance_id** after the instance is registered.

Restrictions

The TESTING state is supported only in version 2.3.X.

URI

PUT /v4/{project_id}/registry/microservices/{service_id}/instances/{instance_id}/status

Table 7-113 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.
service_id	Yes	String	Microservice ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^.*$. See **Querying Information About All Microservices**.
instance_id	Yes	String	Microservice instance ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^[A-Za-z0-9_.-]*$. See **Querying a Microservice Instance Based on service_id**.

Table 7-114 Query parameter

Parameter	Mandatory	Type	Description
value	Yes	String	Instance status. UP: online; OUTOFSERVICE: off-hook; STARTING: starting; DOWN: offline; TESTING: dialing test.

Request

Table 7-115 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Response

Status code: 400

Table 7-116 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-117 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Update the status of an instance with specified **instance_id** to **UP**.

```
PUT https://{endpoint}/v4/{project_id}/registry/microservices/{service_id}/instances/{instance_id}/status?value=UP
```

Example Response

None

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.5.7 Sending Heartbeat Information

Function

This API is used to send heartbeat information. Service providers need to send heartbeat information to the service center, so that the center can check whether service instances are normal.

Restrictions

None

URI

PUT /v4/{project_id}/registry/microservices/{service_id}/instances/{instance_id}/heartbeat

Table 7-118 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.
service_id	Yes	String	Microservice ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^.*$. See **Querying Information About All Microservices**.
instance_id	Yes	String	Microservice instance ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^[A-Za-z0-9_.-]*$. See **Querying a Microservice Instance Based on service_id**.

Request

Table 7-119 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Response

Status code: 400

Table 7-120 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-121 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Send heartbeat information to check whether an instance with specified **instance_id** is healthy.

```
PUT https://{endpoint}/v4/{project_id}/registry/microservices/{service_id}/instances/{instance_id}/heartbeat
```

Example Response

None

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.5.8 Querying a Microservice Instance by Filter Criteria

Function

This API is used to query a microservice instance based on microservice field filter criteria after the instance is registered.

URI

GET /v4/{project_id}/registry/instances

Table 7-122 Path parameter

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.

Table 7-123 Query parameters

Parameter	Mandatory	Type	Description
appId	Yes	String	Application ID, which must be unique. The value contains 1 to 160 characters. Regular expression: ^[a-zA-Z0-9]$\|^[a-zA-Z0-9][a-zA-Z0-9_-.][a-zA-Z0-9]$
serviceName	Yes	String	Microservice name The value contains 1 to 128 characters. Regular expression: ^[a-zA-Z0-9]$\|^[a-zA-Z0-9][a-zA-Z0-9_-.][a-zA-Z0-9]$
tags	No	String	Tag. When there are multiple tags, separate them using commas (,). Regular expression: ^[a-zA-Z][a-zA-Z0-9_-.]{0,63}$
env	No	String	Environment information about the instance.

Request

Table 7-124 Request header parameters

Parameter	Mandatory	Type	Description
X-ConsumerId	No	String	Microservice consumer ID, which must be unique.
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Response

Status code: 200

Table 7-125 Response body parameter

Parameter	Type	Description
instances	Array of **MicroServiceInstance** objects	Instance list.

Table 7-126 MicroServiceInstance

Parameter	Type	Description
instanceId	String	Instance ID, which must be unique. The instance ID is generated by the service center.

Parameter	Type	Description
serviceId	String	Microservice ID, which must be unique. During instance creation, use the microservice ID in the URL instead of the microservice ID specified here.
version	String	Microservice version.
hostName	String	Host information.
endpoints	Array of strings	Access address information.
status	String	Instance status. Value: UP, DOWN, STARTING, or OUTOFSERVICE. Default value: UP.
properties	Object	Extended attribute. You can customize a key and value. The value must be at least 1 byte long.
healthCheck	**HealthCheck** object	Health check information.
dataCenterInfo	**DataCenterInfo** object	Data center information.
timestamp	String	Time when an instance is created, which is automatically generated.
modTimestamp	String	Update time.

Table 7-127 HealthCheck

Parameter	Type	Description
mode	String	Heartbeat mode. Value: push or pull.
port	Integer	Port.
interval	Integer	Heartbeat interval. Unit: s. If the value is less than 5s, the registration is performed at an interval of 5s.
times	Integer	Maximum retries.

Table 7-128 DataCenterInfo

Parameter	Type	Description
name	String	Region name.

Parameter	Type	Description
region	String	Region.
availableZone	String	AZ.

Status code: 400

Table 7-129 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-130 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Query the instance of the microservice whose application ID is **default** and microservice name is **test**.

```
GET https://{endpoint}/v4/{project_id}/registry/instances?appId=default&serviceName=test
```

Example Response

Status code: 200

Successfully queried.

```
{
  "instances": [
    {
      "instanceId": "4098483294bb42d8b3c27eee0c166c1c",
      "serviceId": "8aed80ea052ac04a64dfc79c24f2170224d074f5",
      "endpoints": [
        "rest:127.0.0.1:8080"
      ],
      "hostName": "hostNameTest",
      "status": "UP",
```



```
        "properties": {
            "engineID": "30c263e5-2eac-4da1-9c72-5abb9ac94550",
            "engineName": "cse-fkln1-HA"
        },
        "healthCheck": {
            "mode": "push",
            "interval": 30,
            "times": 3
        },
        "timestamp": "1650545431",
        "modTimestamp": "1650545442",
        "version": "1.0.0"
    }
  ]
}
```

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.5.9 Querying Microservice Instances in Batches

Function

This API is used to query microservice instances in batches based on microservice field filter criteria after the instances are registered.

URI

POST /v4/{project_id}/registry/instances/action

Table 7-131 Path parameter

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.

Table 7-132 Query parameter

Parameter	Mandatory	Type	Description
type	Yes	String	Operation type. Currently, only **query** is supported.

Request

Table 7-133 Request header parameters

Parameter	Mandatory	Type	Description
X-consumerId	No	String	Microservice consumer ID, which must be unique.
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Table 7-134 Request body parameters

Parameter	Mandatory	Type	Description
services	At least one of the two parameters must be specified.	Array of **FindService** objects	Structure for querying service information.
instances		Array of **FindInstance** objects	Structure for querying instance information.

Table 7-135 FindService

Parameter	Mandatory	Type	Description
service	Yes	**DependencyKey** object	Dependency item.
rev	No	String	Version number of the client cache, which determines the difference between the local cache and the microservice in the service center. • Input **0** for the first time. • If the input **rev** is the same as that of the current microservice, **notModified** is returned. If they are different, **updated** is returned. • If **rev** is required in the next request, use the value of **rev** in **updated** in the response body.

Table 7-136 DependencyKey

Parameter	Mandatory	Type	Description
environment	Yes	String	Environment. Value: development, testing, acceptance, or production.
appId	Yes	String	Application ID, which must be unique.
serviceName	Yes	String	Microservice name

Table 7-137 FindInstance

Parameter	Mandatory	Type	Description
instance	No	**HeartbeatSetElement** object	Request struct of a microservice instance.

Parameter	Mandatory	Type	Description
rev	No	String	Version number cached by the client. Version number of the client cache, which determines the difference between the local cache and the microservice instance in the service center. • Input **0** for the first time. • If the input **rev** is the same as that of the current microservice instance, **notModified** is returned. If they are different, **updated** is returned. • If **rev** is required in the next request, use the value of **rev** in **updated** in the response body.

Table 7-138 HeartbeatSetElement

Parameter	Mandatory	Type	Description
serviceId	No	String	Microservice ID.
instanceId	No	String	Microservice instance ID.

Response

Status code: 200

Table 7-139 Response body parameters

Parameter	Type	Description
services	**BatchFindResult** object	Batch query result structure.
instances	**BatchFindResult** object	Batch query result structure.

Table 7-140 BatchFindResult

Parameter	Type	Description
failed	Array of **FindFailedResult** objects	Query failure result structure.

Parameter	Type	Description
notModified	Array of integers	Index set corresponding to the request array.
updated	Array of **FindResult** objects	Query result structure list.

Table 7-141 FindFailedResult

Parameter	Type	Description
indexes	Array of integers	Index set corresponding to the request array.
error	**Error** object	Error struct.

Table 7-142 Error

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Table 7-143 FindResult

Parameter	Type	Description
index	Integer	Index corresponding to the request array.
rev	String	Versions returned by the server. If the versions are the same as the versions cached on the client, the **instances** parameter is left empty.
instances	Array of **MicroServiceInstance** objects	Instance list.

Table 7-144 MicroServiceInstance

Parameter	Type	Description
instanceId	String	Instance ID, which must be unique. The instance ID is generated by the service center.
serviceId	String	Microservice ID, which must be unique. During instance creation, use the microservice ID in the URL instead of the microservice ID specified here.
version	String	Microservice version.
hostName	String	Host information.
endpoints	Array of strings	Access address information.
status	String	Instance status. Value: UP, DOWN, STARTING, or OUTOFSERVICE. Default value: UP.
properties	Object	Extended attribute. You can customize a key and value. The value must be at least 1 byte long.
healthCheck	**HealthCheck** object	Health check information.
dataCenterInfo	**DataCenterInfo** object	Data center information.
timestamp	String	Time when an instance is created, which is automatically generated.
modTimestamp	String	Update time.

Table 7-145 HealthCheck

Parameter	Type	Description
mode	String	Heartbeat mode. Value: push or pull.
port	Integer	Port.
interval	Integer	Heartbeat interval. Unit: s. If the value is less than 5s, the registration is performed at an interval of 5s.
times	Integer	Maximum retries.

Table 7-146 DataCenterInfo

Parameter	Type	Description
name	String	Region name.
region	String	Region.
availableZone	String	AZ.

Status code: 400

Table 7-147 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-148 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Query all instances of the microservice whose application ID is **default** and microservice name is **test**.

```
POST https://{endpoint}/v4/{project_id}/registry/instances/action?type=query
{
  "services": [
    {
      "service": {
        "environment": "",
        "appId": "default",
        "serviceName": "test"
      },
      "rev": "0"
    }
  ]
}
```


Example Response

Status code: 200

Successfully queried.


```
{
   "services": {
      "updated": [
         {
            "index": 0,
            "rev": "0feb784798bca7b2fb4de8351578c4437b516c4b",
            "instances": [
               {
                  "instanceId": "79cdaf47cacf43a5b2b4185527da2255",
                  "serviceId": "8aed80ea052ac04a64dfc79c24f2170224d074f5",
                  "endpoints": [
                     "rest:127.0.0.1:8080"
                  ],
                  "hostName": "hostNameTest",
                  "status": "UP",
                  "properties": {
                     "engineID": "30c263e5-2eac-4da1-9c72-5abb9ac94550",
                     "engineName": "cse-fkln1-HA"
                  },
                  "healthCheck": {
                     "mode": "push",
                     "interval": 30,
                     "times": 3
                  },
                  "timestamp": "1650546183",
                  "modTimestamp": "1650546183",
                  "version": "1.0.0"
               }
            ]
         }
      ]
   }
}
```


Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.6 Dependency

7.6.1 Querying All Providers of a Microservice

Function

This API is used to query all providers of a microservice based on **consumer_id**. For example, if service A depends on services B and C, you can query the information about services B and C based on service A ID.

URI

GET /v4/{project_id}/registry/microservices/{consumer_id}/providers

Table 7-149 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.
consumer_id	Yes	String	Consumer service ID.

Table 7-150 Query parameters

Parameter	Mandatory	Type	Description
noSelf	No	Integer	Whether to return the self-dependency relationship. 0: no; 1: yes.
sameDomain	No	Integer	Whether to return the shared-service relationship. 0: no; 1: yes.

Request

Table 7-151 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Response

Status code: 200

Table 7-152 Response body parameter

Parameter	Type	Description
providers	**MicroService** object	Microservice information.

Table 7-153 Microservice

Parameter	Type	Description
serviceId	String	Microservice ID, which must be unique. The value contains 1 to 64 characters. Regular expression: ^.*$
environment	String	Microservice environment. Value: development, testing, acceptance, or production. You can use the API for uploading schemas in batches to add or modify an existing schema only when the value is **development**, **testing**, or **acceptance**. Default value: development.

Parameter	Type	Description
appId	String	Application ID, which must be unique. The value contains 1 to 160 characters. Regular expression: ^[a-zA-Z0-9]$\|^[a-zA-Z0-9][a-zA-Z0-9_-.][a-zA-Z0-9]$
serviceName	String	Microservice name, which must be unique in an application. The value contains 1 to 128 characters. Regular expression: ^[a-zA-Z0-9]$\|^[a-zA-Z0-9][a-zA-Z0-9_-.][a-zA-Z0-9]$
version	String	Microservice version. The value contains 1 to 64 characters. Regular expression: ^[0-9]$\|^[0-9]+(.[0-9]+)$
description	String	Microservice description. The value contains a maximum of 256 characters.
level	String	Microservice level. Value: FRONT, MIDDLE, or BACK.
registerBy	String	Microservice registration mode. Value: SDK, PLATFORM, SIDECAR, or UNKNOWN.
schemas	Array of strings	Microservice schema content. The value must be 1 to 160 bytes long. Only digits, letters, underscores (_), hyphens (-), and periods (.) are allowed. An array contains a maximum of 100 schemas.
status	String	Microservice status. Value: UP or DOWN. Default value: UP.
timestamp	String	Microservice registration time.
modTimestamp	String	Latest modification time (UTC).
framework	**Framework** object	Development framework.
paths	Array of **ServicePath** objects	Service path.
properties	Object	Extended attribute. You can customize a key and value.

Table 7-154 Framework

Parameter	Type	Description
name	String	Microservice development framework. Default value: UNKNOWN.
version	String	Version of the microservice development framework.

Table 7-155 ServicePath

Parameter	Type	Description
Path	String	Route address.
Property	Object	Extended attribute. You can customize a key and value. The value must be at least 1 byte long.

Status code: 400

Table 7-156 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-157 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Query the providers of a consumer service with specified **consumer_id**.

```
GET https://{endpoint}/v4/{project_id}/registry/microservices/{consumer_id}/providers
```


Example Response

Status code: 200

Successfully queried. **serviceId** or **schemaId** is returned.


```
{
   "providers": [
      {
         "serviceId": "8aed80ea052ac04a64dfc79c24f2170224d074f5",
         "appId": "default",
         "serviceName": "test",
         "version": "1.0.0",
         "description": "this is a test",
         "level": "BACK",
         "status": "UP",
         "properties": {
            "test": "test"
         },
         "timestamp": "1650543950",
         "modTimestamp": "1650544411"
      }
   ]
}
```


Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.7 Configuration Management

7.7.1 Creating a Configuration

Function

This API is used to create a configuration.

URI

POST /v1/{project_id}/kie/kv

Table 7-158 Path parameter

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.

Request

Table 7-159 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Table 7-160 Request body parameters

Parameter	Mandatory	Type	Description
key	Yes	String	Configuration key.
value	No	String	Configuration value.
labels	No	Map<String,String>	Configuration labels.
status	No	String	Configuration status.
value_type	No	String	Type of the configuration value.

Response

Status code: 200

Table 7-161 Response body parameter

Parameter	Type	Description
kie	**SingleKieInfo**	Configuration information.

Table 7-162 SingleKieInfo

Parameter	Type	Description
id	String	Configuration ID.
key	String	Configuration key.
value	String	Configuration value.
value_type	String	Type of the configuration value.
status	String	Configuration status.
create_time	String	Time when the configuration is created.
update_time	String	Time when the configuration is updated.
labels	Map<String,String>	Configuration labels.
create_revision	Integer	Created version number of the configuration.
update_revision	Integer	Updated version number of the configuration.

Status code: 400

Table 7-163 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_message	String	Error message.

Status code: 500

Table 7-164 Response body parameters

Parameter	Type	Description
error_code	String	Error code.

Parameter	Type	Description
error_message	String	Error message.

Example Request

Create a configuration whose label is **"key1": "value1"** and **"key2": "value2"**.

```
POST https://{endpoint}/v1/{project_id}/kie/kv
{
    "key": "String",
    "value": "String",
    "labels": {
         "key1": "value1",
         "key2": "value2",
    },
    "status": "String",
    "value_type": "String"
}
```

Example Response

Status code: 200

Successfully created. The configuration information is returned.

```
{
    "id": "8a9e6a5d-8d65-48fb-a40c-5150c8479da8",
    "key": "string",
    "labels": {
        "key1": "value1",
        "key2": "value2"
    },
    "value": "string",
    "value_type": "text",
    "status": "enabled",
    "create_time": 1623139038,
    "update_time": 1623139038,
    "create_revision": 13,
    "update_revision": 13,
}
```

Status Code

Status Code	Description
200	OK
400	Bad Request
409	Conflict
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.7.2 Modifying a Configuration

Function

This API is used to modify a configuration.

URI

PUT /v1/{project_id}/kie/kv/{kv_id}

Table 7-165 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.
kv_id	Yes	String	ID of the configuration to be modified. See **Querying the Configuration List**.

Request

Table 7-166 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Table 7-167 Request body parameters

Parameter	Mandatory	Type	Description
value	Yes	String	Configuration value.
status	No	String	Configuration status.

Response

Status code: 200

Table 7-168 Response body parameter

Parameter	Type	Description
kie	SingleKieInfo	Configuration information.

Status code: 400

Table 7-169 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_message	String	Error message.

Status code: 500

Table 7-170 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_message	String	Error message.

Example Request

Modify the value of the configuration whose ID is **8a9e6a5d-8d65-48fb-a40c-5150c8479da8** to **this is a test** and set the configuration status to **enabled**.

```
PUT https://{endpoint}/v1/{project_id}/kie/kv/8a9e6a5d-8d65-48fb-a40c-5150c8479da8

{
   "value": "this is a test",
```



```
    "status": "enabled"
}
```

Example Response

Status code: 200

Successfully modified. The configuration information is returned.

```
{
    "id": "8a9e6a5d-8d65-48fb-a40c-5150c8479da8",
    "key": "string",
    "labels": {
        "key1": "value1",
        "key2": "value2"
    },
    "value": "this is a test",
    "value_type": "text",
    "status": "enabled",
    "create_time": 1623139038,
    "update_time": 1623139038,
    "create_revision": 13,
    "update_revision": 13,
}
```

Status Code

Status Code	Description
200	OK
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.7.3 Querying a Configuration

Function

This API is used to query a configuration in the configuration center.

URI

GET /v1/{project_id}/kie/kv/{kv_id}

Table 7-171 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.

Parameter	Mandatory	Type	Description
kv_id	Yes	String	ID of the configuration to be queried. See **Querying the Configuration List**.

Request

Table 7-172 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Response

Status code: 200

Table 7-173 Response body parameter

Parameter	Type	Description
id	String	Configuration ID.
key	String	Configuration key.
value	String	Configuration value.
value_type	String	Type of the configuration value.
status	String	Configuration status.
create_time	String	Time when the configuration is created.

Parameter	Type	Description
update_time	String	Time when the configuration is updated.
labels	Map<String,String>	Configuration labels.
create_revision	Integer	Created version number of the configuration.
update_revision	Integer	Updated version number of the configuration.

Status code: 400

Table 7-174 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_message	String	Error message.

Status code: 404

Table 7-175 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_message	String	Error message.

Status code: 500

Table 7-176 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_message	String	Error message.

Example Request

Query the configuration whose configuration ID is **8a9e6a5d-8d65-48fb-a40c-5150c847**.

```
GET https://{endpoint}/v1/{project_id}/kie/kv/8a9e6a5d-8d65-48fb-a40c-5150c8479da8
```

Example Response

Status code: 200.

Successfully queried.

```
{
   "id": "8a9e6a5d-8d65-48fb-a40c-5150c8479da8",
   "key": "string",
   "labels": {
      "key1": "value1",
      "key2": "value2"
   },
   "value": "string",
   "value_type": "text",
   "status": "enabled",
   "create_time": 1623139038,
   "update_time": 1623139038,
   "create_revision": 13,
   "update_revision": 13
}
```

Status Code

Status Code	Description
200	OK
400	Bad Request
404	Not Found
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.7.4 Querying the Configuration List

Function

This API is used to query the configuration list in the configuration center.

URI

GET /v1/{project_id}/kie/kv

Table 7-177 Path parameter

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.

Request

Table 7-178 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Table 7-179 Query parameters

Parameter	Mandatory	Type	Description
label	No	String	Exports data filtered by label. The format is {Label key}: {Label value}.
match	No	String	Matching option of the filtered items. Value **exact** indicates exact matching, including the same number of labels. If the value is null, inclusive matching is used.

Parameter	Mandatory	Type	Description
revision	No	Integer	Configuration version number This parameter is used to determine the difference between the local cache configuration and the configuration in the configuration center. • Input **0** for the first time. • If the input **revision** value is the same as the current **revision** value, status code 304 is returned, but no body is returned. If they are inconsistent, both **revision** values are returned. • If **revision** is required in the next request, use the value of **update_revision** in the response body.

Response

Status code: 200

Table 7-180 Response body parameters

Parameter	Type	Description
total	Integer	Number of query results.
data	List<SingleKieInfo>	List of queried kie.

Table 7-181 SingleKieInfo

Parameter	Type	Description
id	String	Configuration ID.
key	String	Configuration key.
value	String	Configuration value.
value_type	String	Type of the configuration value.
status	String	Configuration status.
create_time	String	Time when the configuration is created.

Parameter	Type	Description
update_time	String	Time when the configuration is updated.
labels	Map<String,String>	Configuration labels.
create_revision	Integer	Created version number of the configuration.
update_revision	Integer	Updated version number of the configuration.

Status code: 304

None

Status code: 400

Table 7-182 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_message	String	Error message.

Status code: 500

Table 7-183 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_message	String	Error message.

Example Request

Query the configuration list.

```
GET https://{endpoint}/v1/{project_id}/kie/kv
```

Example Response

Status code: 200

Successfully queried.


```
{
   "total": 1,
   "data":[{
      "id": "8a9e6a5d-8d65-48fb-a40c-5150c8479da8",
      "key": "string",
      "labels": {
         "key1": "value1",
         "key2": "value2"
      },
      "value": "string",
      "value_type": "text",
      "status": "enabled",
      "create_time": 1623139038,
      "update_time": 1623139038,
      "create_revision": 13,
      "update_revision": 13
   }]
}
```


Status Code

Status Code	Description
200	OK
304	Not Modified
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.7.5 Deleting a Configuration

Function

This API is used to delete a configuration.

URI

DELETE /v1/{project_id}/kie/kv/{kv_id}

Table 7-184 Path parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.
kv_id	Yes	String	ID of the configuration to be deleted. See **Querying the Configuration List**.

Request

Table 7-185 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Response

Status code: 200

Successfully deleted.

Status code: 404

Table 7-186 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_message	String	Error message.

Status code: 400

Table 7-187 Response body parameters

Parameter	Type	Description
error_code	String	Error code.

Parameter	Type	Description
error_message	String	Error message.

Status code: 500

Table 7-188 Response body parameters

Parameter	Type	Description
error_code	String	Error code.
error_message	String	Error message.

Example Request

Delete the configuration whose configuration ID is **8a9e6a5d-8d65-48fb-a40c-5150c8479da8**.

```
DELETE https://{endpoint}/v1/{project_id}/kie/kv/8a9e6a5d-8d65-48fb-a40c-5150c8479da8
```

Example Response

Status code: 204

Successfully deleted.

Status Code

Status Code	Description
200	No Content
400	Bad Request
404	Not Found
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

7.7.6 Deleting All Configurations

Function

This API is used to delete all configurations.

URI

DELETE /v1/{project_id}/kie/kv

Table 7-189 Path parameter

Parameter	Mandatory	Type	Description
project_id	Yes	String	Fixed value: default.

Request

Table 7-190 Request header parameter

Parameter	Mandatory	Type	Description
Authorization	No	String	This parameter is mandatory if security authentication is enabled for an exclusive ServiceComb engine. Otherwise, this parameter is not required. The token of an exclusive ServiceComb engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive ServiceComb Engine**.

Table 7-191 Request body parameter

Parameter	Mandatory	Type	Description
ids	Yes	List<String>	ID list of the configurations to be deleted. See **Querying the Configuration List**.

Response

Status code: 204

Successfully deleted.

Status code: 404

Table 7-192 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Status code: 500

Table 7-193 Response body parameters

Parameter	Type	Description
errorCode	String	Error code.
errorMessage	String	Error message.
detail	String	Location details.

Example Request

Delete all configurations.

```
DELETE https://{endpoint}/v1/{project_id}/kie/kv

{
   "ids":[
      "id1",
      "id2"
   ]
}
```

Example Response

Status code: 204

Successfully deleted.

Status Code

Status Code	Description
204	No Content
400	Bad Request
500	Internal Server Error

Error Code

See **ServiceComb Error Codes**.

8 Appendixes

8.1 Status Codes

Table 8-1 describes the status codes.

Table 8-1 Status codes

Status Code	Message	Description
200	-	OK
204	No Content	OK
400	Bad Request	Invalid request. The client should not repeat the request without modifications.
401	Unauthorized	The authentication information is incorrect or invalid.
404	Not Found	The requested resource cannot be found. The client should not repeat the request without modifications.
409	Conflict	The resource already exists.
422	Unprocessable Entity	The request was well-formed but was unable to be followed due to semantic errors.
500	Internal Server Error	The server is able to receive the request but it could not understand the request.

8.2 Error Code

8.2.1 CSE Error Codes

If an error code starting with **APIGW** is returned after you call an API, rectify the fault by referring to the instructions provided in **Error Codes**.

Table 8-2 Engine management error codes

Status Code	Error Code	Error Message	Description	Measure
400	SVCSTG.00501134	Bad Request	Invalid engine type.	The engine type field is the **specType** field in the request query. Set it to a valid value (for example, **CSE2**) and try again. For example, to query the specifications supported by CSE, set **specType** to **CSE2**.
400	SVCSTG.00501111	Bad Request	Duplicate engine instance name.	Change the engine name.
400	SVCSTG.00501112	Bad Request	Insufficient engine instance quota.	Contact technical support engineers to increase the engine instance quota or delete unnecessary engine instances.

Status Code	Error Code	Error Message	Description	Measure
400	SVCSTG.0050 1133	Bad Request	Too many engines at the site.	Contact technical support engineers.
400	SVCSTG.0050 1103	Bad Request	Empty or invalid billing mode.	Input the billing mode parameter of the engine and check whether the user can create such an engine.
400	SVCSTG.0050 1104	Bad Request	Incorrect engine instance flavor format.	Invalid flavor. Enter a valid flavor.
400	SVCSTG.0050 1105	Bad Request	Incorrect AZ format.	Specify a valid AZ in the request parameter. The AZ field is a string array and cannot be empty. The number of AZ name levels ranges from 1 to 3.
400	SVCSTG.0050 1117	Bad Request	Engine cannot be deleted.	Wait until the engine state changes to Available, Unavailable, or Creation failed, and then try again.
400	SVCSTG.0050 1152	Bad Request	Incorrect task ID format.	Use the job ID returned by the APIs for creating and deleting an engine.

Status Code	Error Code	Error Message	Description	Measure
400	SVCSTG.00501153	Bad Request	Failed to find the task.	Use the job ID returned by the APIs for creating and deleting an engine.
400	SVCSTG.00501149	Bad Request	Abnormal engine node.	Try again later or contact technical support engineers.
400	SVCSTG.00501116	Bad Request	Failed to find the engine instance.	Check whether the input engine ID is correct. Use the username and password of the tenant to log in to CSE and check whether the current tenant has an engine with the corresponding ID.

Status Code	Error Code	Error Message	Description	Measure
400	SVCSTG.00501201	Bad Request	Incorrect format of the peer engine instance name.	The peer engine instance name must contain 3 to 24 characters, including letters, digits, and hyphens (-), and must start with a letter but cannot end with a hyphen (-). The name cannot be **default** and must be different from the current engine instance name.
400	SVCSTG.00501202	Bad Request	The peer engine must be an HA engine.	The engine must be an HA engine. Enter the engine flavor again.
400	SVCSTG.00501203	Bad Request	The peer engine must be in a single AZ.	The AZs must be single AZs. Enter the AZ list again.

Status Code	Error Code	Error Message	Description	Measure
500	SVCSTG.0050 0400	Internal Server Error	Incorrect internal request parameter of the service.	Check the error message. If the parameter is invalid, check whether the input argument is correct. For other errors, contact technical support engineers.

Table 8-3 Dynamic configuration error codes

Status Code	Error Code	Error Message	Description	Measure
400	SVCSTG.0040 0649	Bad Request	Too long key or value.	The configuration item contains 1 to 2048 characters, including digits, letters, and special characters (.-_:[]). The value can contain 1 to 131072 characters, including digits, letters, and special characters ('~!@#$%^&*()_\{}|:"<>?`-[],./=).
400	SVCSTG.0040 0650	Bad Request	Invalid configuration file.	Upload a valid file.

Status Code	Error Code	Error Message	Description	Measure
400	SVCSTG.0040 0651	Bad Request	The file is empty or in an incorrect format.	The uploaded file is empty or not in JSON format.
400	SVCSTG.0040 0652	Bad Request	The key or value is empty or is not a string.	The uploaded key or value is null or is not a string.
400	SVCSTG.0040 0653	Bad Request	Invalid label or body.	Enter a valid body or label in the query parameter.
400	SVCSTG.0040 0100	Bad Request	Invalid parameter.	Change parameters as prompted.
404	SVCSTG.0040 0106	Not Found	The resource does not exist.	Enter valid search criteria.
409	SVCSTG.0040 0107	Conflict	The resource already exists.	Do not create the same record.
500	SVCSTG.0030 0605	Failed to connect the configuration center to ETCD.	Internal Server Error	Contact technical support engineers.
400	SVCSTG.0030 0401	Invalid token	Unauthorized	Enter a correct token.

8.2.2 ServiceComb Error Codes

If an error code starting with **APIGW** is returned after you call an API, rectify the fault by referring to the instructions provided in **Error Codes**.

Category	Status Code	Error Code	Error Message	Description	Measure
Common error codes	400	400001	Bad Request	Invalid parameter.	Change parameters as prompted.
	404	404001	Not Found	The resource does not exist.	Enter valid search criteria.
	409	409001	Conflict	The resource already exists.	Do not create the same record.
	500	500003	Internal Server Error	Internal server error.	Contact technical support engineers.
Microservices	400	400002	Bad Request	The service is unhealthy.	Try again later or contact technical support engineers.
		400010	Bad Request	The service already exists.	Modify the service ID or microservice description in the body of the request for creating a microservice.
		400011	Bad Request	No available backend instance.	Try again later or contact technical support engineers.
		400012	Bad Request	The service does not exist.	Enter a valid service ID.

Category	Status Code	Error Code	Error Message	Description	Measure
		400013	Bad Request	The microservice cannot be deleted because instances have been deployed.	Take the instance offline and then delete the microservice. Alternatively, forcibly delete the microservice by setting the query parameter **force** to **true** in the URL.
		400014	Bad Request	The schema ID does not exist.	Enter a valid schema ID.
		400015	Bad Request	The schema cannot be modified.	The schema has been registered and cannot be modified.
		400016	Bad Request	The schema does not exist.	Register the schema first.
		400017	Bad Request	The instance does not exist.	Enter a valid instance ID.
		400018	Bad Request	The tag does not exist.	This error code is usually generated for a query API, indicating that the tag does not exist. The service performs subsequent processing based on the returned value.
		400019	Bad Request	The rule already exists.	The rule is repeatedly created. Generally, this error can be ignored.

Category	Status Code	Error Code	Error Message	Description	Measure
		400020	Bad Request	Blacklist and whitelist error.	Modify the parameter value based on the error message.
		400021	Bad Request	The rule cannot be modified.	You can modify the microservice information only after changing the version number.
		400022	Bad Request	The rule does not exist.	This error code is usually generated for a query API, indicating that the rule does not exist. The service performs subsequent processing based on the returned value.
		400023	Bad Request	The microservice cannot be deleted because it is the dependent service of other microservices.	You can forcibly delete microservices by setting the query parameter **force** to **true** in the URL.
		400024	Bad Request	Insufficient permissions.	Use a proper account to perform operations.
		400025	Bad Request	The port already exists.	Check whether the port is occupied by another instance.

Category	Status Code	Error Code	Error Message	Description	Measure
		400026	Bad Request	The microservice instance does not exist.	Enter a correct version number or range.
		400100	Bad Request	Insufficient quota.	The quotas of resources such as microservices, instances, or schemas are insufficient. Delete some resources and create again.
	401	401204	Unauthorized	Unauthorized.	This parameter is mandatory if security authentication is enabled for the microservice engine. Otherwise, this parameter is not required. The token of the microservice engine with security authentication enabled is in the following format: `Authorization:Bearer {Token}` For details about how to obtain the token, see **Obtaining the User Token of an Exclusive Microservice Engine**.
		401201	Unauthorized	Unauthorized.	Enter a valid authorization.

Category	Status Code	Error Code	Error Message	Description	Measure
	403	403001	Forbidden	Insufficient permissions.	Use a proper account to perform operations.
	500	500011	Internal Server Error	The registry service is unavailable.	Contact technical support engineers.
		500101	Internal Server Error	No quota.	Try again later or contact technical support engineers.
		500605	N/A	Failed to connect to etcd of the configuration center.	Try again later or contact technical support engineers.
Authentication	401	401202	Unauthorized	Incorrect account name or password.	Enter the correct account name and password.

8.3 Obtaining a Project ID

Obtaining a Project ID on the Console

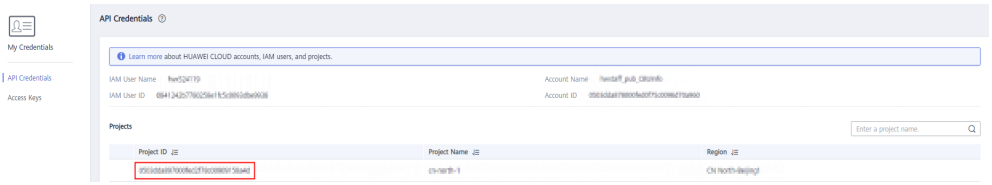
A project ID is required for some URLs when an API is called. To obtain an ID, do as follows:

Step 1 Log in to CSE.

Step 2 Move the cursor to the username in the upper right corner and select **My Credentials** from the drop-down list.

Step 3 On the **Project List** tab, obtain the required project ID in the project list.

Figure 8-1 Viewing a project ID



----End

Obtaining a Project ID by Calling an API

You can also call the API for **querying project information based on the specified criteria** to obtain a project ID.

The API for obtaining a project ID is **GET https://*{Endpoint}*/v3/projects**, where *{Endpoint}* indicates the IAM endpoint. You can obtain the IAM endpoint from **Regions and Endpoints**. For details about API authentication, see **Authentication**.

The following is an example response. *id* indicates a project ID.

```
{
    "projects": [
        {
            "domain_id": "65382450e8f64ac0870cd180d14e684b",
            "is_domain": false,
            "parent_id": "65382450e8f64ac0870cd180d14e684b",
            "name": "region_id",
            "description": "",
            "links": {
                "next": null,
                "previous": null,
                "self": "https://www.example.com/v3/projects/a4a5d4098fb4474fa22cd05f897d6b99"
            },
            "id": "a4a5d4098fb4474fa22cd05f897d6b99",
            "enabled": true
        }
    ],
    "links": {
        "next": null,
        "previous": null,
        "self": "https://www.example.com/v3/projects"
    }
}
```


9 Change History

Released On	Change History
2021-11-30	This issue is the second official release. The API sections are divided into CSE API and ServiceComb API sections.
2021-09-20	This issue is the first official release.

