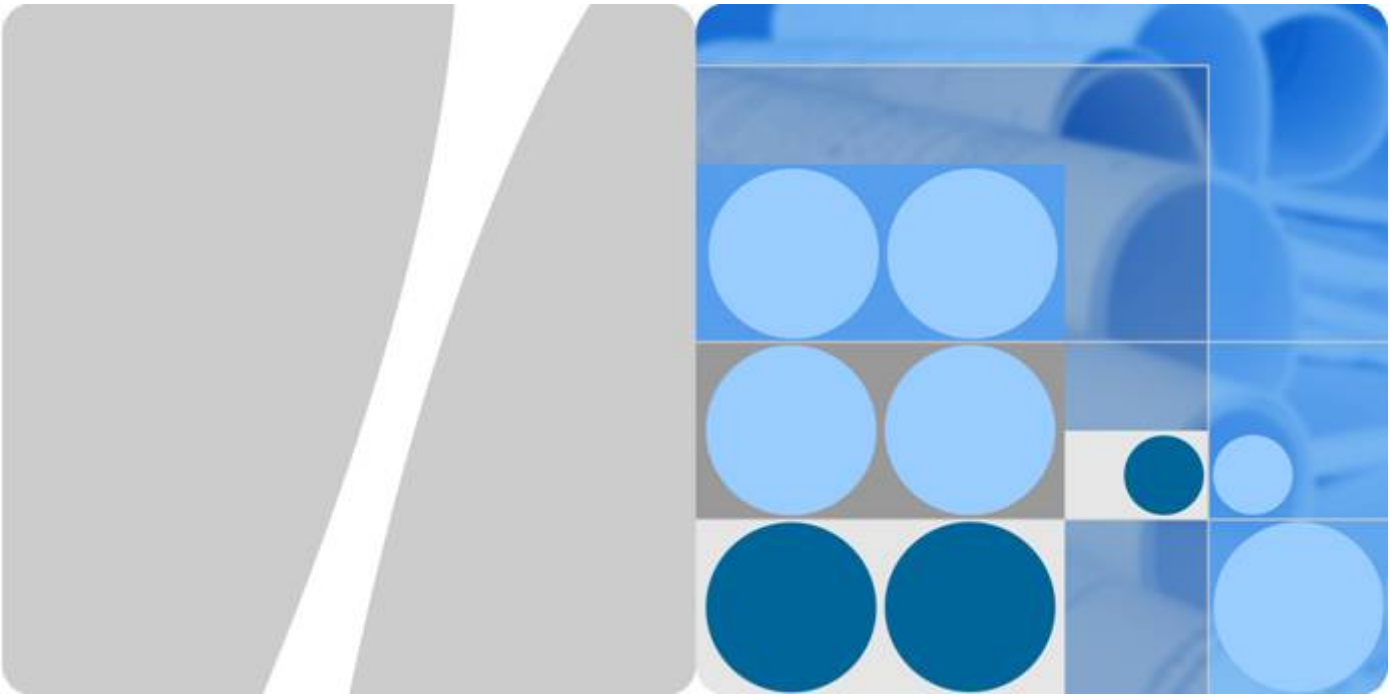


ServiceStage

API Reference

Date **2023-06-25**

Contents

1 Before You Start

1.1 Overview

ServiceStage is an application management and O&M platform that lets you deploy, roll out, monitor, and maintain applications all in one place. Java, Go, PHP, Node.js, Python, Docker, and Tomcat are supported. Web applications, microservice applications such as Apache ServiceComb, Spring Cloud, Dubbo, and service mesh, and common applications make it easier to migrate enterprise applications to the cloud.

ServiceStage provides the following self-developed APIs:

- Application management APIs
- Git repository access APIs
- Cloud Service Engine (CSE) APIs

1.2 API Calling

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

1.3 Endpoints

An endpoint is the **request address** for calling an API. Endpoints vary depending on services and regions. For the endpoints of all services, see **the system administrator**.

1.4 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.

- User

  A user is created in Identity and Access Management (IAM) to use cloud services. Each user has its own identity credentials (password and access keys).

  The account name, username, and password will be required for API authentication.

- Region

  Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

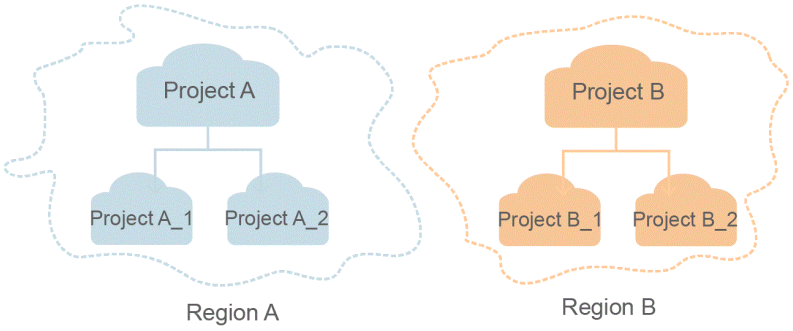
- AZ

  AZs are physically isolated locations in a region, but are interconnected through an internal network for enhanced application availability.

- 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.

  Figure 1-1 Project isolating model



2 API Overview

ServiceStage provides open APIs for developers and partners to develop, deploy, host, operate, and manage applications, helping them effectively innovate services at low costs and shorten the application rollout period.

For details about the APIs, see the following table.

Table 2-1 Overview

Type	Subtype	Description
Application management APIs	**Meta APIs**	APIs related to application templates, runtimes, and application resource specifications.
	Environment APIs	APIs related to application environments.
	Application APIs	APIs related to applications.
	Component APIs	APIs related to application components.
	Instance APIs	APIs related to application component instances.
	Job APIs	APIs related to jobs.
Git repository access APIs	**Git Repository Access APIs**	APIs related to Git repository authorization, namespaces, projects, branches, tags, commits, hooks, repository file directories, and contents.

Type	Subtype	Description
Cloud Service Engine (CSE) APIs	CSE APIs	APIs related to CSE.

Meta APIs

Table 2-2 Meta APIs

API	Description
Obtaining All Supported Runtimes of Application Components	Obtain all supported runtimes of application components.
Obtaining All Supported Flavors of Application Resources	Obtain all supported flavors of application resources.

Environment APIs

Table 2-3 Environment APIs

API	Description
Creating an Environment	Create an environment.
Obtaining All Environments	Obtain all created environments.
Modifying Environment Information	Modify environment information based on the environment ID.
Deleting an Environment Based on the Environment ID	Delete an environment based on the environment ID.
Obtaining Environment Details Based on the Environment ID	Obtain environment details based on the environment ID.
Modifying Environment Resources	Modify environment resources.

Application APIs

Table 2-4 Application APIs

API	Description
Creating an Application	Create an application.
Obtaining All Applications	Obtain all created applications.

API	Description
Modifying Application Information	Modify application information based on the application ID.
Deleting an Application Based on the Application ID	Delete an application based on the application ID.
Obtaining Application Details Based on the Application ID	Obtain application details based on the application ID.
Adding or Modifying Application Configurations	Add or modify application configurations.
Deleting Application Configurations	Delete application configurations.
Obtaining Application Configurations	Obtain application configurations.

Component APIs

Table 2-5 Component APIs

API	Description
Creating an Application Component	Create an application component.
Obtaining All Components of an Application	Obtain all components of an application.
Modifying Component Information Based on the Component ID	Modify component information based on the component ID.
Deleting an Application Component Based on the Component ID	Delete a component based on the component ID.
Obtaining Application Component Information Based on the Component ID	Obtain component information based on the component ID.

Instance APIs

Table 2-6 Instance APIs

API	Description
Creating an Application Component Instance	Create an application component instance.
Obtaining All Instances of a Component	Obtain all instances of a component.

API	Description
Querying the Operations Performed on a Component Instance	Perform operations on a component instance.
Modifying an Application Component Instance	Modify an application component instance.
Deleting an Application Component Instance	Delete an application component instance.
Querying Instance Details Based on the Instance ID	Query instance details based on the instance ID.
Obtaining Component Instance Snapshots	Obtain component instance snapshots.

Job APIs

Table 2-7 Job APIs

API	Description
Obtaining Job Details	Obtain job details.

Git Repository Access APIs

Table 2-8 Git repository access APIs

API	Description
Obtaining a Git Repository Authorization List	Obtain a Git repository authorization list.
Obtaining an Authorization Redirection URL	Obtain an authorization redirection URL.
Creating OAuth Authorization	Create OAuth authorization.
Creating Private Token Authorization	Create private token authorization.
Creating Password Authorization	Create password authorization for a Git repository.
Deleting Repository Authorization	Delete repository authorization based on the name.

API	Description
Obtaining a Repository Namespace	Obtain a repository namespace.
Obtaining Repository Information Based on the Clone URL	Obtain repository information based on the clone URL.
Obtaining All Projects in a Namespace	Obtain all projects in a namespace.
Creating a Software Repository Project	Create a software repository project.
Obtaining a Project Branch	Obtain a project branch.
Obtaining a Project Tag	Obtain a project tag.
Creating a Project Tag	Create a project tag.
Deleting a Project Tag	Delete a project tag.
Obtaining Project Commits	Obtain the latest ten project commits.
Obtaining a Project Hook	Obtain a project hook.
Creating a Project Hook	Create a project hook.
Deleting a Project Hook	Delete a project hook.
Obtaining a Repository File Directory	Obtain a repository file directory.
Obtaining Repository File Contents	Obtain repository file contents.
Creating a Repository File	Create a repository file.
Modifying Repository File Contents	Modify repository file contents.
Deleting a Repository File	Delete a repository file.

CSE APIs

Table 2-9 CSE APIs

API	Description
Querying Static Information About a Microservice	Query static information about a microservice based on the service ID.
Querying Static Information About All Microservices	Query static information about microservices that meet filter criteria.
Creating Static Information for a Microservice	Create static information for a microservice before registering a microservice instance. The registered instance is associated with the static information based on the **serviceId** field. One service corresponds to multiple instances. The **serviceId** field can be customized. If **serviceId** is not customized, the system generates a random service ID.
Modifying Static Information About a Microservice	Modify static information about a microservice. To update some fields in the static information, you need to input all static information (including the fields that do not need to be updated) in JSON format.
Querying a Microservice Schema	Query a microservice schema based on the service ID and schema ID.
Modifying a Microservice Schema	Modify a microservice schema based on the schema ID.
Creating a Dependency Between Services	Create a dependency between services. For **consumer**, **version** must be a confirmed version and **serviceName** cannot be *. **consumer** must be an existing service while **provider** can be a service that has not been created.
Querying All Providers of a Microservice	Query all providers of a microservice based on the consumer ID.
Querying the Unique Service or Schema ID of a Microservice	Query the unique service or schema ID of a microservice based on filter criteria.

API	Description
Registering a Microservice Instance	Register a microservice instance after a microservice is created. Information about the instance must be provided during registration. The instance ID can be customized. If it is customized, all contents are overwritten upon re-registration. If it is not customized, the system automatically generates an ID. If the endpoint content is duplicate, the original ID is used.
Querying All Instances of a Microservice Based on the Service ID	Query all instances of a microservice based on the service ID after the instances are registered.
Deregistering a Microservice Instance	Deregister an instance based on the instance ID after the instance is registered.
Querying Details About a Microservice Instance	Query details about an instance based on the service ID and instance ID after the instance is registered.
Modifying the Extended Information About a Microservice Instance	Add or update the extended information about a microservice instance based on the instance ID after the instance is registered.
Changing the Status of a Microservice Instance	Change the status of a microservice instance based on the instance ID after the instance is registered.
Sending Heartbeat Information	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.
Querying a Microservice Instance by Filter Criteria	Query a microservice instance based on microservice version rules or field filter criteria after the instance is registered.
Querying Configurations	Query configurations in the configuration center.
Deleting Static Information About a Microservice	Delete the definition and related information about a microservice, and deregister all instances of the microservice.

API	Description
Deleting Static Information About Microservices in Batches	Delete the definitions and related information about microservices in batches, and deregister all instances of the microservices.
Querying Microservice Instances in Batches	Query microservice instances in batches based on microservice version rules or field filter criteria after the instances are registered.
Querying All Schema Information About a Microservice	Query all schema information (including **schemaId** and **summary**) about a microservice.

3 Calling APIs

3.1 Making an API Request

This section describes the structure of a Representational State Transfer (REST) API request, and uses the Identity and Access Management (IAM) API for **obtaining a user token** as an example to demonstrate how to call an API. 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 the request URI to be transmitted 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 **the system administrator**.
- **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** indicates that a maximum of 10 data records will be displayed.

NOTE

To simplify the URI display in this document, 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 HTTPS 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.{Endpoint}/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 HTTPS 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**: specifies the 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**: specifies 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.
- **X-Project-ID**: specifies a subproject ID. This parameter is optional and can be used in multi-project scenarios.
- **X-Domain-ID**: specifies an account ID.

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.{Endpoint}/v3/auth/tokens
Content-Type: application/json
```

Request Body

A request body is generally sent in a structured format. It corresponds to **Content-Type** in the request header and transfers content except the request header.

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 **the system administrator**.

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.{Endpoint}/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, postman, 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

Requests for calling an API can be authenticated using:

- Token-based authentication: Requests are authenticated using a token.

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 used to obtain a user token.

A token specifies temporary permissions in a computer system. During API authentication using a token, 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.

A project-level token is required for calling ServiceStage APIs. When calling an API to **obtain a user token**, set **project** in **auth.scope** in the request body, as shown in the following figure.

```
{
   "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.{Endpoint}/v3/auth/projects
Content-Type: application/json
X-Auth-Token: ABCDEFJ....
```

3.3 Response

Status Code

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

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**.

For example, if status code 201 is returned for calling the API used to **obtain a user token**, the request is successful.

Response Header

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

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.

Response Body

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

The following is part of the response body for the API used to **obtain a user token**.

```
{
    "token": {
        "expires_at": "2019-02-13T06:52:13.855000Z",
        "methods": [
            "password"
        ],
        "catalog": [
            {
                "endpoints": [
                    {
                        "region_id": "xx-xxxxxx-1",
.....
```

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": "FGS.0111",
  "error_msg": "xxxxxxxxx"
}
```

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

4 Application Management APIs

4.1 Meta

4.1.1 Obtaining All Supported Runtimes of Application Components

Function

This API is used to obtain all supported runtimes of application components.

URI

GET /v2/{project_id}/cas/metadata/runtimes

Table 4-1 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.

Request

Request parameters

None

Response

Response parameters

Table 4-2 Response parameters

Parameter	Type	Description
runtimes	Array of objects	Runtime parameters. See **Table 4-3**.

Table 4-3 runtimes parameters

Parameter	Type	Description
type_name	String	Type.
display_name	String	Display name.
container_default_port	Integer	Default container port.
type_desc	String	Type description.

Example

Example request

None

Example response

```
{
   "runtimes": [
      {
         "type_name": "Java8",
         "display_name": "Java 8",
         "container_default_port": 8080,
         "type_desc": "Java 8 runtime server"
      }
   ]
}
```

Status Code

Table 4-4 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.1.2 Obtaining All Supported Flavors of Application Resources

Function

This API is used to obtain all supported flavors of application resources.

URI

GET /v2/{project_id}/cas/metadata/flavors

Table 4-5 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.

Request

None

Response

Table 4-6 Response parameters

Parameter	Type	Description
flavors	Array of objects	Flavor parameters. See **Table 4-7**.

Table 4-7 flavors parameters

Parameter	Type	Description
flavor_id	String	Flavor ID.
storage_size	String	Storage size.
num_cpu	String	CPU limit.
num_cpu_init	String	Initial CPU value.

Parameter	Type	Description
memory_size	String	Memory limit.
memory_size_init	String	Initial memory value.
label	String	Label.
custom	boolean	Whether resource specifications are customized.

Example

Example request

None

Example response

```
{
   "flavors": [
      {
         "flavor_id": "MICRO-5G:0.5C:1G",
         "storage_size": "5G",
         "num_cpu": "500m",
         "num_cpu_init": "200m",
         "memory_size": "1Gi",
         "memory_size_init": "200Mi",
         "label": "Micro: 5G Storage, 0.5 CPU, 1G Memory",
         "custom": false
      }
   ]
}
```

Status Code

Table 4-8 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.2 Environment

4.2.1 Creating an Environment

Function

An environment is a collection of compute, storage, and network resources used for deploying and running an application. ServiceStage enables you to add multiple Elastic Load Balance (ELB) and Distributed Cache Service (DCS) instances to Cloud Container Engine (CCE) clusters in the same Virtual Private Cloud (VPC) to set up an environment, such as the development environment, test environment, production-like environment, and production environment. The resources within an environment can be networked together. Managing resources and deploying services by environment simplifies O&M.

This API is used to create an environment.

URI

POST /v2/{project_id}/cas/environments

Table 4-9 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.

Request

Table 4-10 Request parameters

Parameter	Type	Mandatory	Description
name	String	Yes	Environment name. The value contains 2 to 64 characters consisting of letters, digits, hyphens (-), and underscores (_). It starts with a letter and ends with a letter or digit.
alias	String	No	Environment alias. The value can contain up to 64 characters.

Parameter	Type	Mandatory	Description
description	String	No	Environment description. The value can contain up to 128 characters.
enterprise_project_id	String	No	Enterprise project ID.
charge_mode	String	No	Fixed value: provided.
vpc_id	String	Yes	VPC ID.
base_resources	Array of objects	Yes	Basic resources. See **Table 4-11**.
optional_resources	Array of objects	No	Optional resources. See **Table 4-11**.

Table 4-11 resource parameters

Parameter	Type	Mandatory	Description
id	String	Yes	Resource ID.
name	String	No	Resource name.
type	String	Yes	Basic resources: cce, and ecs. Optional resources: dcs, elb, and other services.

Response

Table 4-12 Response parameters

Parameter	Type	Description
id	String	Environment ID.
name	String	Environment name.
alias	String	Environment alias.
description	String	Environment description.
project_id	String	Project ID.
enterprise_project_id	String	Enterprise project ID.

Parameter	Type	Description
charge_mode	String	Billing mode.
vpc_id	String	VPC ID.
base_resources	Array of objects	Basic resources. See **Table 4-13**.
optional_resources	Array of objects	Optional resources. See **Table 4-13**.
creator	String	Creator.
create_time	Integer	Creation time.
update_time	Integer	Update time.

Table 4-13 resource

Parameter	Type	Description
id	String	Resource ID.
name	String	Resource name.
type	String	Basic resources: cce, and ecs. Optional resources: dcs, elb, and other services.

Example

Example request

```
{
   "name": "development-env",
   "description": "",
   "charge_mode": "provided",
   "vpc_id": "29d55020-ae0e-4a18-871c-93e6976ee7bd",
   "base_resources": [
      {
         "id": "523498f1-36c4-11eb-ae36-0255ac1000c2",
         "type": "cce",
         "name": "cce-test"
      }
   ],
   "optional_resources": [
      {
         "id": "9963306a-791e-468c-9306-cf80a9d00298",
         "type": "elb",
         "name": "elb-2dd2"
      }
   ]
}
```

Example response

```
{
   "id": "00078e9d-a61c-476e-ac63-a10c9cb2638e",
```

```
  "name": "development-env",
  "alias": null,
  "description": "",
  "project_id": "e7d2e9c589e5445e808a8ff0d1235aca",
  "enterprise_project_id": "0",
  "charge_mode": "provided",
  "vpc_id": "29d55020-ae0e-4a18-871c-93e6976ee7bd",
  "base_resources": [
     {
        "id": "523498f1-36c4-11eb-ae36-0255ac1000c2",
        "type": "cce",
        "name": "cce-test"
     }
  ],
  "optional_resources": [
     {
        "id": "9963306a-791e-468c-9306-cf80a9d00298",
        "type": "elb",
        "name": "elb-2dd2"
     }
  ],
  "creator": "ss-test",
  "create_time": 1610418873730,
  "update_time": 1610418873730
}
```


Status Code

Table 4-14 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.2.2 Obtaining All Environments

Function

This API is used to obtain all created environments.

URI

GET /v2/{project_id}/cas/environments

Table 4-15 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.

Table 4-16 Query parameters

Parameter	Type	Mandatory	Description
limit	Integer	No	Number of records to be displayed. The value is **1000** or ranges from 0 to 100. If the specified value is not within the range, value **10** is assigned. In the non-pagination scenario, the value is **1000**. In the pagination scenario, the value ranges from 0 to 100.
offset	Integer	No	Offset, which indicates the result after the Nth data record is queried.
order_by	String	No	Sorting field. By default, query results are sorted by creation time. The following enumerated values are supported: create_time, name, and update_time. If the transferred value is not within the supported enumerated value range, the default sorting field is used.
order	String	No	Descending or ascending order. Default value: desc.

Request

None

Response

Table 4-17 Response parameters

Parameter	Type	Description
count	Integer	Total number of environments.

Parameter	Type	Description
environments	Array of objects	Environment parameters. See **Table 4-18**.

Table 4-18 environments parameters

Parameter	Type	Description
id	String	Environment ID.
name	String	Environment name.
alias	String	Environment alias.
description	String	Environment description.
project_id	String	Project ID.
enterprise_project_id	String	Enterprise project ID.
charge_mode	String	Billing mode.
vpc_id	String	VPC ID.
base_resources	Array of objects	Basic resources. See **Table 4-19**.
optional_resources	Array of objects	Optional resources. See **Table 4-19**.
creator	String	Creator.
create_time	Integer	Creation time.
update_time	Integer	Update time.

Table 4-19 resource parameters

Parameter	Type	Description
id	String	Resource ID.
name	String	Resource name.
type	String	Basic resources: cce, and ecs. Optional resources: dcs, elb, and other services.

Example

Example request

None

Example response


```
{
   "environments": [
      {
         "id": "00078e9d-a61c-476e-ac63-a10c9cb2638e",
         "name": "development-env",
         "alias": null,
         "description": "",
         "project_id": "e7d2e9c589e5445e808a8ff0d1235aca",
         "enterprise_project_id": "0",
         "charge_mode": "provided",
         "vpc_id": "29d55020-ae0e-4a18-871c-93e6976ee7bd",
         "base_resources": [
            {
               "id": "523498f1-36c4-11eb-ae36-0255ac1000c2",
               "type": "cce",
               "name": "cce-test"
            }
         ],
         "optional_resources": [
            {
               "id": "9963306a-791e-468c-9306-cf80a9d00298",
               "type": "elb",
               "name": "elb-2dd2"
            }
         ],
         "creator": "ss-test",
         "create_time": 1610418873730,
         "update_time": 1610418873730
      }
   ],
   "count": 1
}
```


Status Code

Table 4-20 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.2.3 Modifying Environment Information

Function

This API is used to modify environment information based on the environment ID.

URI

PUT /v2/{project_id}/cas/environments/{environment_id}

Table 4-21 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
environment_id	String	Yes	Environment ID.

Request

Table 4-22 Request parameters

Parameter	Type	Mandatory	Description
name	String	No	Environment name. The value contains 2 to 64 characters consisting of letters, digits, hyphens (-), and underscores (_). It starts with a letter and ends with a letter or digit.
alias	String	No	Environment alias. The value can contain up to 64 characters.
description	String	No	Environment description. The value can contain up to 128 characters.

Response

Table 4-23 Response parameters

Parameter	Type	Description
id	String	Environment ID.
name	String	Environment name.
alias	String	Environment alias.
description	String	Environment description.

Parameter	Type	Description
project_id	String	Project ID.
enterprise_project_id	String	Enterprise project ID.
charge_mode	String	Fixed value: provided.
vpc_id	String	VPC ID.
base_resources	Array of objects	Basic resources. See **Table 4-24**.
optional_resources	Array of objects	Optional resources. See **Table 4-24**.
creator	String	Creator.
create_time	Integer	Creation time.
update_time	Integer	Update time.

Table 4-24 resources parameters

Parameter	Type	Description
id	String	Resource ID.
name	String	Resource name.
type	String	Basic resources: cce, and ecs. Optional resources: dcs, elb, and other services.

Example

Example request

```
{
   "name": "development-env2",
   "description": "here is description"
}
```

Example response

```
{
   "id": "00078e9d-a61c-476e-ac63-a10c9cb2638e",
   "name": "development-env2",
   "alias": null,
   "description": "here is description",
   "project_id": "e7d2e9c589e5445e808a8ff0d1235aca",
   "enterprise_project_id": "0",
   "charge_mode": "provided",
   "vpc_id": "29d55020-ae0e-4a18-871c-93e6976ee7bd",
   "base_resources": [
      {
         "id": "523498f1-36c4-11eb-ae36-0255ac1000c2",
         "type": "cce",
```

```
            "name": "cce-test"
        }
    ],
    "optional_resources": [
        {
            "id": "9963306a-791e-468c-9306-cf80a9d00298",
            "type": "elb",
            "name": "elb-2dd2"
        }
    ],
    "creator": "ss-test",
    "create_time": 1610418873730,
    "update_time": 1610420992462
}
```


Status Code

Table 4-25 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.2.4 Deleting an Environment Based on the Environment ID

Function

This API is used to delete an environment based on the environment ID.

URI

DELETE /v2/{project_id}/cas/environments/{environment_id}

Table 4-26 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
environment_id	String	Yes	Environment ID.

Request

None

Response

None

Example

Example request

None

Example response

None

Status Code

Table 4-27 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.2.5 Obtaining Environment Details Based on the Environment ID

Function

This API is used to obtain environment details based on the environment ID.

URI

GET /v2/{project_id}/cas/environments/{environment_id}

Table 4-28 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
environment_id	String	Yes	Environment ID.

Request

None

Response

Table 4-29 Response parameters

Parameter	Type	Description
id	String	Environment ID.
name	String	Environment name.
alias	String	Environment alias.
description	String	Environment description.
project_id	String	Project ID.
enterprise_project_id	String	Enterprise project ID.
charge_mode	String	Fixed value: provided.
vpc_id	String	VPC ID.
base_resources	Array of objects	Basic resources. See **Table 4-30**.
optional_resources	Array of objects	Optional resources. See **Table 4-30**.
creator	String	Creator.
create_time	Integer	Creation time.
update_time	Integer	Update time.

Table 4-30 resources parameters

Parameter	Type	Description
id	String	Resource ID.
name	String	Resource name.
type	String	Basic resources: cce, and ecs. Optional resources: dcs, elb, and other services.

Example

Example request

None

Example response

```
{
   "id": "ea011e01-2eb5-453f-87bf-874e4a855abe",
   "name": "dev-env",
   "alias": null,
   "description": "develop environment",
   "project_id": "bf8523d898b64e4eb956e3be3555ca16",
   "enterprise_project_id": "0",
   "charge_mode": "provided",
   "vpc_id": "234241234124xxvasf2342xxxxxxxxxx",
   "base_resources": [
      {
         "id": "211112333sd332w23322332",
         "type": "cce"
      }
   ],
   "optional_resources": null,
   "creator": "string",
   "create_time": 1578984198394,
   "update_time": 1578984198394
}
```

Status Code

Table 4-31 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.2.6 Modifying Environment Resources

Function

This API is used to modify environment resources.

URI

PATCH /v2/{project_id}/cas/environments/{environment_id}/resources

Table 4-32 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's ID.
environment_id	String	Yes	Environment ID.

Request

Table 4-33 Request parameters

Parameter	Type	Mandatory	Description
add_base_resources	Array of objects	No	Basic resources to be added. See **Table 4-34**.
add_optional_resources	Array of objects	No	Optional resources to be added. See **Table 4-35**.
remove_resources	Array of objects	No	Resources to be removed. See **Table 4-36**.

Table 4-34 add_base_resources parameters

Parameter	Type	Mandatory	Description
id	String	Yes	Resource ID.
name	String	No	Resource name.

Parameter	Type	Mandatory	Description
type	String	Yes	Basic resources: cce, and ecs.

Table 4-35 add_optional_resources parameters

Parameter	Type	Mandatory	Description
id	String	Yes	Resource ID.
name	String	No	Resource name.
type	String	Yes	Optional resources: dcs, elb, and other services.

Table 4-36 remove_resources parameters

Parameter	Type	Mandatory	Description
id	String	Yes	Resource ID.
type	String	Yes	Basic resources: cce, and ecs. Optional resources: dcs, elb, and other services.

Response

Table 4-37 Parameters

Parameter	Type	Description
id	String	Environment ID.
name	String	Environment name.
alias	String	Environment alias.
description	String	Environment description.
project_id	String	Project ID.
enterprise_project_id	String	Enterprise project ID.
charge_mode	String	Fixed value: provided.
vpc_id	String	VPC ID.

Parameter	Type	Description
base_resources	Array of objects	Basic resources. See **Table 4-38**.
optional_resources	Array of objects	Optional resources. See **Table 4-38**.
creator	String	Creator.
create_time	Integer	Creation time.
update_time	Integer	Update time.

Table 4-38 schemas parameters

Parameter	Type	Description
id	String	Resource ID.
name	String	Resource name.
type	String	Basic resources: cce, and ecs. Optional resources: dcs, elb, and other services.

Example

Example request

```
{
   "add_base_resources": [
      {
         "id": "ed2f3420-7031-4d93-b92b-e360cd4acf9e",
         "type": "ecs",
         "name": "ecs-9876"
      }
   ],
   "add_optional_resources": [
      {
         "id": "default",
         "type": "cse",
         "name": "Cloud Service Engine"
      }
   ],
   "remove_resources": [
      {
         "id": "523498f1-36c4-11eb-ae36-0255ac1000c2",
         "type": "cce"
      }
   ]
}
```

Example response

```
{
   "id": "00078e9d-a61c-476e-ac63-a10c9cb2638e",
   "name": "development-env2",
   "alias": null,
   "description": "here is description",
```

```
  "project_id": "e7d2e9c589e5445e808a8ff0d1235aca",
  "enterprise_project_id": "0",
  "charge_mode": "provided",
  "vpc_id": "29d55020-ae0e-4a18-871c-93e6976ee7bd",
  "base_resources": [
     {
        "id": "ed2f3420-7031-4d93-b92b-e360cd4acf9e",
        "type": "ecs",
        "name": "ecs-9876"
     }
  ],
  "optional_resources": [
     {
        "id": "9963306a-791e-468c-9306-cf80a9d00298",
        "type": "elb",
        "name": "elb-2dd2"
     },
     {
        "id": "default",
        "type": "cse",
        "name": "Cloud Service Engine"
     }
  ],
  "creator": "ss-test",
  "create_time": 1610418873730,
  "update_time": 1610420992462
}
```


Status Code

Table 4-39 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.3 Application

4.3.1 Creating an Application

Function

An application is a service system with complete functions and consists of one or more components related to features.

This API is used to create an application.

URI

POST /v2/{project_id}/cas/applications

Table 4-40 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.

Request

Table 4-41 Request parameters

Parameter	Type	Mandatory	Description
name	String	Yes	Application name. The value contains 2 to 64 characters consisting of letters, digits, hyphens (-), and underscores (_). It starts with a letter and ends with a letter or digit.
description	String	No	Application description. The value can contain up to 128 characters.
enterprise_project_id	String	No	Enterprise project ID. Default value: 0.

Response

Table 4-42 Response parameters

Parameter	Type	Description
id	String	Application ID.
name	String	Application name.
description	String	Application description.
creator	String	Creator.
project_id	String	Project ID.

Parameter	Type	Description
enterprise_project_id	String	Enterprise project ID.
create_time	Integer	Creation time.
update_time	Integer	Update time.
unified_model	String	Whether to enable the unified model.

Example

Example request

```
{
    "name": "app-xpmtii",
    "description": "test"
}
```

Example response

```
{
    "id": "e5213b62-0e3c-476f-9960-3e4108787350",
    "name": "app-xpmtii",
    "description": "test",
    "project_id": "e7d2e9c589e5445e808a8ff0d1235aca",
    "enterprise_project_id": "0",
    "creator": "ss-test",
    "create_time": 1610432385245,
    "update_time": 1610432385245,
    "unified_model": null
}
```

Status Code

Table 4-43 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.3.2 Obtaining All Applications

Function

This API is used to obtain all created applications.

URI

GET /v2/{project_id}/cas/applications

Table 4-44 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Project ID.

Table 4-45 Query parameters

Parameter	Type	Mandatory	Description
limit	String	No	Number of records to be displayed. The value is **1000** or ranges from 0 to 100. If the specified value is not within the range, value **10** is assigned. In the non-pagination scenario, the value is **1000**. In the pagination scenario, the value ranges from 0 to 100.
offset	String	No	Offset, which indicates the result after the Nth data record is queried.
order_by	String	No	Sorting field. By default, query results are sorted by creation time. The following enumerated values are supported: create_time, name, and update_time. If the transferred value is not within the supported enumerated value range, the default sorting field is used.
order	String	No	Descending or ascending order. Default value: desc.

Request

None

Response

Table 4-46 Response parameters

Parameter	Type	Description
count	Integer	Total number of applications.
applications	Array of objects	Application information. See **Table 4-47**.

Table 4-47 applications parameters

Parameter	Type	Description
id	String	Application ID.
name	String	Application name.
description	String	Application description.
creator	String	Creator.
project_id	String	Project ID.
enterprise_project_id	String	Enterprise project ID.
create_time	Integer	Creation time.
update_time	Integer	Update time.
unified_model	String	Whether to enable the unified model.
component_count	Integer	Number of components.

Example

Example request

None

Example response

```
{
    "count": 1,
    "applications": [
        {
            "id": "e5213b62-0e3c-476f-9960-3e4108787350",
            "name": "app-xpmtii",
            "description": "test",
            "project_id": "e7d2e9c589e5445e808a8ff0d1235aca",
            "enterprise_project_id": "0",
            "creator": "ss-test",
            "create_time": 1610432385245,
            "update_time": 1610432385245,
```

```
            "unified_model": null,
            "component_count": 0
        }
    ]
}
```

Status Code

Table 4-48 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.3.3 Modifying Application Information

Function

This API is used to modify application information based on the application ID.

URI

PUT /v2/{project_id}/cas/applications/{application_id}

Table 4-49 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.

Request

Table 4-50 Request parameters

Parameter	Type	Mandatory	Description
name	String	No	Application name. The value contains 2 to 64 characters consisting of letters, digits, hyphens (-), and underscores (_). It starts with a letter and ends with a letter or digit.
description	String	No	Application description. The value can contain up to 128 characters.

Response

Table 4-51 Response parameters

Parameter	Type	Description
id	String	Application ID.
name	String	Application name.
description	String	Application description.
creator	String	Creator.
project_id	String	Project ID.
enterprise_project_id	String	Enterprise project ID.
create_time	Integer	Creation time.
update_time	Integer	Update time.
unified_model	String	Whether to enable the unified model.

Example

Example request

```
{
    "name": "app-test",
    "description": "test"
}
```

Example response

```
{
    "id": "e5213b62-0e3c-476f-9960-3e4108787350",
    "name": "app-test",
    "description": "test",
    "project_id": "e7d2e9c589e5445e808a8ff0d1235aca",
    "enterprise_project_id": "0",
    "creator": "ss-test",
    "create_time": 1610432385245,
    "update_time": 1610433070875,
    "unified_model": null
}
```


Status Code

Table 4-52 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.3.4 Deleting an Application Based on the Application ID

Function

This API is used to delete an application based on the application ID.

URI

DELETE /v2/{project_id}/cas/applications/{application_id}

Table 4-53 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.

Request

None

Response

None

Example

Example request

None

Example response

None

Status Code

Table 4-54 Status codes

Status Code	Description
204	OK
400	Bad Request
404	Not Found
409	Conflict
500	Internal Server Error

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.3.5 Obtaining Application Details Based on the Application ID

Function

This API is used to obtain application details based on the application ID.

URI

GET /v2/{project_id}/cas/applications/{application_id}

Table 4-55 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.

Request

None

Response

Table 4-56 Response parameters

Parameter	Type	Description
id	String	Application ID.
name	String	Application name.
description	String	Application description.
creator	String	Creator.
project_id	String	Project ID.
enterprise_project_id	String	Enterprise project ID.
create_time	Integer	Creation time.
update_time	Integer	Update time.
unified_model	String	Whether to support enterprise projects.
component_count	Integer	Number of components.

Example

Example request

None

Example response

```
{
    "id": "e5213b62-0e3c-476f-9960-3e4108787350",
    "name": "app-test",
    "description": "test",
    "project_id": "e7d2e9c589e5445e808a8ff0d1235aca",
```

```
    "enterprise_project_id": "0",
    "creator": "ss-test",
    "create_time": 1610432385245,
    "update_time": 1610433070875,
    "unified_model": null,
    "component_count": 0
}
```


Status Code

Table 4-57 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.3.6 Adding or Modifying Application Configurations

Function

This API is used to add or modify application configurations.

Application configurations refer to common environment variables of an application. Components deployed in the application inherit these environment variables.

URI

PUT /v2/{project_id}/cas/applications/{application_id}/configuration

Table 4-58 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.

Request

Table 4-59 Request parameters

Parameter	Type	Mandatory	Description
environment_id	String	Yes	Environment ID.
configuration	Object	Yes	Application configurations, such as public environment variables. See **Table 4-60**.

Table 4-60 configuration parameter

Parameter	Type	Mandatory	Description
env	Array of objects	Yes	Environment variables of the application. See **Table 4-61**. If the names of multiple environment variables are the same, only the last environment variable takes effect.

Table 4-61 env parameters

Parameter	Type	Mandatory	Description
name	String	Yes	Name of the environment variable. The value contains 1 to 64 characters consisting of letters, digits, underscores (_), hyphens (-), and dots (.), and cannot start with a digit.
value	String	Yes	Value of the environment variable.

Response

Table 4-62 Response parameters

Parameter	Type	Description
application_id	String	Application ID.

Parameter	Type	Description
environment_id	String	Environment ID.
configuration	Object	Application configurations. See **Table 4-63**.

Table 4-63 configuration parameter

Parameter	Type	Description
env	Array of objects	Environment variables. See **Table 4-64**.

Table 4-64 env parameters

Parameter	Type	Description
name	String	Name of the environment variable.
value	String	Value of the environment variable.

Example

Example request

```
{
   "environment_id": "0fa75dfe-0d32-4bc1-848b-8008cf3f2567",
   "configuration": {
      "env": [
         {
            "name": "app-env",
            "value": "env"
         }
      ]
   }
}
```

Example response

```
{
   "application_id": "06cf2fda-af45-44b4-9e22-6294c4804515",
   "environment_id": "0fa75dfe-0d32-4bc1-848b-8008cf3f2567",
   "configuration": {
      "env": [
         {
            "name": "app-env",
            "value": "env"
         }
      ]
   }
}
```

Status Code

Table 4-65 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.3.7 Deleting Application Configurations

Function

This API is used to delete application configurations.

URI

DELETE /v2/{project_id}/cas/applications/{application_id}/configuration

Table 4-66 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.

Table 4-67 Query parameters

Parameter	Type	Mandatory	Description
environment_id	String	Yes	Environment ID.

Request

None

Response

None

Example

Example request

None

Example response

None

Status Code

Table 4-68 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.3.8 Obtaining Application Configurations

Function

This API is used to obtain application configurations.

URI

GET /v2/{project_id}/cas/applications/{application_id}/configuration

Table 4-69 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.

Table 4-70 Query parameters

Parameter	Type	Mandatory	Description
environment_id	String	No	Environment ID. If this parameter is not specified, all environments are queried.

Request

None

Response

Table 4-71 Response parameters

Parameter	Type	Description
configuration	Array of objects	Configuration information. See **Table 4-72**.

Table 4-72 Application configuration parameters

Parameter	Type	Description
application_id	String	Application ID.
environment_id	String	Environment ID.
configuration	Object	Application configurations. See **Table 4-73**.

Table 4-73 configuration parameters

Parameter	Type	Description
env	Array of objects	Environment variables of the application component. See **Table 4-74**.

Table 4-74 env parameters

Parameter	Type	Description
name	String	Name of the environment variable.
value	String	Value of the environment variable.

Example

Example request

None

Example response

```
{
   "configuration": [
      {
         "application_id": "e5213b62-0e3c-476f-9960-3e4108787350",
         "environment_id": "00078e9d-a61c-476e-ac63-a10c9cb2638e",
         "configuration": {
            "env": [
               {
                  "name": "env",
                  "value": "value1"
               }
            ]
         }
      }
   ]
}
```

Status Code

Table 4-75 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.4 Component

4.4.1 Creating an Application Component

Function

A component implements a service feature of an application. It is in the form of code or software packages and can be deployed independently.

This API is used to create an application component.

URI

POST /v2/{project_id}/cas/applications/{application_id}/components

Table 4-76 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.

Request

Table 4-77 Request parameters

Parameter	Type	Mandatory	Description
name	String	Yes	Application component name. The value contains 2 to 64 characters consisting of letters, digits, hyphens (-), and underscores (_). It starts with a letter and ends with a letter or digit.
runtime	String	Yes	Runtime. The value can be obtained from **type_name** returned by the API in **Obtaining All Supported Runtimes of Application Components**.

Parameter	Type	Mandatory	Description
category	String	Yes	Application component type. Example: Webapp, MicroService, or Common.
sub_category	String	No	Application component sub-type. Webapp sub-types include Web. MicroService sub-types include Java Chassis, Go Chassis, Mesher, Spring Cloud, and Dubbo. Common sub-type can be empty.
description	String	No	Description. The value can contain up to 128 characters.
source	Object	No	Source of the code or software package. See **Table 4-78**.
build	Object	No	Component build. See **Table 4-82**.

Table 4-78 source parameters

Parameter	Type	Mandatory	Description
kind	String	No	Type. Option: source code or artifact software package.
spec	Object	No	For details about the source code, see **Table 4-79**. For details about the artifact software package, see **Table 4-80**.

Table 4-79 code spec parameters

Parameter	Type	Mandatory	Description
repo_type	String	Yes	Code repository. Value: GitHub, GitLab, Gitee, or Bitbucket.
repo_url	String	Yes	Code repository URL. Example: https://github.com/example/demo.git.

Parameter	Type	Mandatory	Description
repo_ref	String	No	Code branch or tag. Default value: master.
repo_auth	String	Yes	Authorization name, which can be obtained from the authorization list.

Table 4-80 artifact spec parameters

Parameter	Type	Mandatory	Description
storage	String	Yes	Storage mode. Value: obs.
type	String	Yes	Type. Value: package.
url	String	Yes	Software package address.
auth	String	No	Authentication mode. Value: iam or none. Default value: iam.
properties	Object	No	Other attributes of the software package. You need to add these attributes only when you set **storage** to **obs**. See **Table 4-81**.

Table 4-81 artifact spec properties parameters

Parameter	Type	Mandatory	Description
endpoint	String	No	OBS endpoint address. Example: https://obs.region_id.external_domain_name.com.
bucket	String	No	Name of the OBS bucket where the software package is stored.
key	String	No	Object in the OBS bucket, which is usually the name of the software package. If there is a folder, the path of the folder must be added. Example: test.jar or demo/test.jar.

Table 4-82 build parameters

Parameter	Type	Mandatory	Description
parameters	Map<String, Object>	No	See **Table 4-83**.

Table 4-83 parameters description

Parameter	Type	Mandatory	Description
build_cmd	String	No	Compilation command. By default: 1. When **build.sh** exists in the **root** directory, the command is **./build.sh**. 2. When **build.sh** does not exist in the **root** directory, the command varies depending on the OS. Example: Java and Tomcat: mvn clean package Nodejs: npm build
dockerfile_path	String	No	Address of the Docker file. By default, the Docker file is in the root directory (./).
artifact_namespace	String	No	Build archive organization. Default value: cas_{project_id}.
cluster_id	String	No	ID of the cluster to be built.
node_label_selector	Map<String, String>	No	**key** indicates the key of the tag, and **value** indicates the value of the tag.

Response

Table 4-84 Response parameters

Parameter	Type	Description
id	String	Application component ID.
name	String	Application component name.
status	Integer	Value: 0 or 1. 0: Normal. 1: Being deleted.

Parameter	Type	Description
runtime	String	Runtime.
category	String	Application component type. Example: Webapp, MicroService, or Common.
sub_category	String	Application component sub-type. Webapp sub-types include Web. MicroService sub-types include Java Chassis, Go Chassis, Mesher, Spring Cloud, and Dubbo. Common sub-type can be empty.
description	String	Description.
project_id	String	Project ID.
application_id	String	Application ID.
source	Object	Source of the code or software package. See **Table 4-85**.
build	Object	Build. See **Table 4-89**.
pipeline_ids	ID list	Pipeline ID list. A maximum of 10 pipeline IDs are supported.
create_time	Integer	Creation time.
update_time	Integer	Update time.
creator	String	Creator.

Table 4-85 source parameters

Parameter	Type	Description
kind	String	Type. Option: source code or artifact software package.
spec	Object	For details about the source code, see **Table 4-86**. For details about the artifact software package, see **Table 4-87**.

Table 4-86 code spec parameters

Parameter	Type	Description
repo_type	String	Code repository. Value: GitHub, GitLab, Gitee, or Bitbucket.

Parameter	Type	Description
repo_url	String	Code repository URL. Example: https://github.com/example/demo.git.
repo_ref	String	Code branch or tag. Default value: master.
repo_auth	String	Authorization name, which can be obtained from the authorization list.

Table 4-87 artifact spec parameters

Parameter	Type	Description
storage	String	Storage mode.
type	String	Type.
url	String	Address of the software package or source code.
auth	String	Authentication mode.
properties	Object	Other attributes of the software package. You need to add these attributes only when you set **storage** to **obs**. See **Table 4-88**.

Table 4-88 artifact spec properties parameters

Parameter	Type	Mandatory	Description
endpoint	String	No	OBS endpoint address. Example: https://obs.region_id.external_domain_name.com.
bucket	String	No	Name of the OBS bucket where the software package is stored.
key	String	No	Object in the OBS bucket, which is usually the name of the software package. If there is a folder, the path of the folder must be added. Example: test.jar or demo/test.jar.

Table 4-89 build parameters

Parameter	Type	Description
ID	String	Type.

Parameter	Type	Description
parameters	Map<String, Object>	See **Table 4-90**.

Table 4-90 parameters description

Parameter	Type	Description
build_cmd	String	Compilation command.
dockerfile_path	String	Address of the Docker file.
artifact_namespace	String	Build archive organization.
cluster_id	String	ID of the cluster to be built.
node_label_selector	Map<String, String>	**key** indicates the key of the tag, and **value** indicates the value of the tag.

Example

Example request

```
{
   "name": "mycomponent",
   "runtime": "Java8",
   "category": "MicroService",
   "sub_category": "Java Chassis",
   "description": "",
   "build": {
      "parameters": {
         "artifact_namespace": "ns"
      }
   },
      "source": {
      "kind": "artifact",
      "spec": {
         "storage": "obs",
         "type": "package",
         "url": "obs://myapp/demo.jar",
         "properties": {
            "bucket": "myapp",
            "key": "demo.jar",
            "endpoint": "https://obs.region_id.external_domain_name.com"
         }
      }
   }
}
```

Example response

```
{
   "id": "384eb8d4-c193-4d84-9558-6fda2366b536",
   "name": "mycomponent",
   "runtime": "Java8",
   "category": "MicroService",
   "sub_category": "Java Chassis",
   "description": "",
```

```
  "project_id": "384eb8d4-c193-4d84-9558-6fda23698536",
  "application_id": "a8f7eed5-0aa0-4251-9723-c9119a6bf56d",
     "source": {
     "kind": "artifact",
     "spec": {
         "storage": "obs",
         "type": "package",
         "url": "obs://myapp/demo.jar",
         "properties": {
             "bucket": "myapp",
             "key": "demo.jar",
             "endpoint": "https://obs.region_id.external_domain_name.com"
         }
     }
  },
  "build": {
     "id": "w3dpv7p0t1vpxvey5hjb22iuwxway1vupwx0nae1",
     "parameters": {
         "artifact_namespace": "ns",
         "use_public_cluster": true
     }
  },
  "pipeline_ids": null,
  "status": 0,
  "creator": "xxx",
  "create_time": 1610333934288,
  "update_time": 1610333934288
}
```


Status Code

Table 4-91 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.4.2 Obtaining All Components of an Application

Function

This API is used to obtain all components of an application.

URI

GET /v2/{project_id}/cas/applications/{application_id}/components

Table 4-92 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.

Table 4-93 Query parameters

Parameter	Type	Mandatory	Description
limit	String	No	Number of records to be displayed. The value is **1000** or ranges from 0 to 100. If the specified value is not within the range, value **10** is assigned. In the non-pagination scenario, the value is **1000**. In the pagination scenario, the value ranges from 0 to 100.
offset	String	No	Offset.
order_by	String	No	Sorting field. By default, query results are sorted by creation time. The following enumerated values are supported: create_time, name, and update_time.
order	String	No	Descending or ascending order. Default value: desc.

Request

None

Response

Table 4-94 Response parameters

Parameter	Type	Description
count	Integer	Total number of components.
components	Array of objects	Component parameters. See **Table 4-95**.

Table 4-95 components parameters

Parameter	Type	Description
id	String	Application component ID.
name	String	Application component name.
status	Integer	Value: 0 or 1. 0: Normal. 1: Being deleted.
runtime	String	Runtime.
category	String	Application component type. Example: Webapp, MicroService, or Common.
sub_category	String	Application component sub-type. Webapp sub-types include Web. MicroService sub-types include Java Chassis, Go Chassis, Mesher, Spring Cloud, and Dubbo. Common sub-type can be empty.
application_id	String	Application ID.
source	Object	Source of the code or software package. See **Table 4-96**.
description	String	Description.
creator	String	Creator.
create_time	Integer	Creation time.
update_time	Integer	Update time.

Table 4-96 source parameters

Parameter	Type	Description
kind	String	Type. Option: source code or artifact software package.
spec	Object	For details about the source code, see **Table 4-97**. For details about the artifact software package, see **Table 4-98**.

Table 4-97 code spec parameters

Parameter	Type	Description
repo_type	String	Code repository. Value: GitHub, GitLab, Gitee, or Bitbucket.
repo_url	String	Code repository URL. Example: https://github.com/example/demo.git.
repo_ref	String	Code branch or tag. Default value: master.
repo_auth	String	Authorization name, which can be obtained from the authorization list.

Table 4-98 artifact spec parameters

Parameter	Type	Description
storage	String	Storage mode.
type	String	Type.
url	String	Software package address.
auth	String	Authentication mode.
properties	Object	Other attributes of the software package. You need to add these attributes only when you set **storage** to **obs**. See **Table 4-99**.

Table 4-99 artifact spec properties parameters

Parameter	Type	Description
endpoint	String	OBS endpoint address. Example: https://obs.region_name.external_domain_name.com.
bucket	String	Name of the OBS bucket where the software package is stored.
key	String	Object in the OBS bucket, which is usually the name of the software package. If there is a folder, the path of the folder must be added. Example: test.jar or demo/test.jar.

Example

Example request

None

Example response

```
{
   "components": [
     {
   "id": "384eb8d4-c193-4d84-9558-6fda2366b536",
   "name": "mycomponent",
   "runtime": "Java8",
   "category": "MicroService",
   "sub_category": "Java Chassis",
   "description": "",
   "project_id": "384eb8d4-c193-4d84-9558-6fda23698536",
   "application_id": "a8f7eed5-0aa0-4251-9723-c9119a6bf56d",
      "source": {
      "kind": "artifact",
      "spec": {
         "storage": "obs",
         "type": "package",
         "url": "obs://myapp/demo.jar",
         "properties": {
            "bucket": "myapp",
            "key": "demo.jar",
            "endpoint": "https://obs.region_id.external_domain_name.com"
         }
      }
   },
   "build": {
      "id": "w3dpv7p0t1vpxvey5hjb22iuwxway1vupwx0nae1",
      "parameters": {
         "artifact_namespace": "ns",
         "use_public_cluster": true
      }
   },
   "pipeline_ids": null,
   "status": 0,
   "creator": "xxx",
   "create_time": 1610333934288,
   "update_time": 1610333934288
     }
   ],
   "count": 1
}
```


Status Code

Table 4-100 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.4.3 Modifying Component Information Based on the Component ID

Function

This API is used to modify component information based on the component ID.

URI

PUT /v2/{project_id}/cas/applications/{application_id}/components/{component_id}

Table 4-101 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.
component_id	String	Yes	Component ID.

Request

Table 4-102 Request parameters

Parameter	Type	Mandatory	Description
name	String	No	Application component name. The value contains 2 to 64 characters consisting of letters, digits, hyphens (-), and underscores (_). It starts with a letter and ends with a letter or digit.
description	String	No	Description. The value can contain up to 128 characters.
source	Object	No	Source of the code or software package. See **Table 4-103**.
build	Object	No	Build. See **Table 4-107**.

Table 4-103 source parameters

Parameter	Type	Mandatory	Description
kind	String	Yes	Type. Option: source code or artifact software package.
spec	Object	Yes	For details about the source code, see **Table 4-104**. For details about the artifact software package, see **Table 4-105**.

Table 4-104 code spec parameters

Parameter	Type	Mandatory	Description
repo_type	String	Yes	Code repository. Value: GitHub, GitLab, Gitee, or Bitbucket.
repo_url	String	Yes	Code repository URL. Example: https://github.com/example/demo.git.
repo_ref	String	No	Code branch or tag. Default value: master.
repo_auth	String	Yes	Authorization name, which can be obtained from the authorization list.

Table 4-105 artifact spec parameters

Parameter	Type	Mandatory	Description
storage	String	Yes	Storage mode. Value: obs.
type	String	Yes	Type. Value: package.
url	String	Yes	Address of the software package or source code.
auth	String	No	Authentication mode. Value: iam or none. Default value: iam.

Parameter	Type	Mandatory	Description
properties	Object	No	Other attributes of the software package. You need to add these attributes only when you set **storage** to **obs**. For details, see **Table 4-106**.

Table 4-106 artifact spec properties parameters

Parameter	Type	Mandatory	Description
endpoint	String	No	OBS endpoint address. Example: https://obs.region_id.external_domain_name.com.
bucket	String	No	Name of the OBS bucket where the software package is stored.
key	String	No	Object in the OBS bucket, which is usually the name of the software package. If there is a folder, the path of the folder must be added. Example: test.jar or demo/test.jar.

Table 4-107 build parameters

Parameter	Type	Mandatory	Description
parameters	Map<String, Object>	No	See **Table 4-108**.

Table 4-108 parameters description

Parameter	Type	Mandatory	Description
build_cmd	String	No	Compilation command. By default: 1. When **build.sh** exists in the **root** directory, the command is **./build.sh**. 2. When **build.sh** does not exist in the **root** directory, the command varies depending on the OS. Example: ● Java and Tomcat: mvn clean package ● Nodejs: npm build
dockerfile_path	String	No	Address of the Docker file. By default, the Docker file is in the root directory (./).
artifact_namespace	String	No	Build archive organization. Default value: cas_{project_id}.
cluster_id	String	No	ID of the cluster to be built.
node_label_selector	Map<String, String>	No	**key** indicates the key of the tag, and **value** indicates the value of the tag.

Response

Table 4-109 Response parameters

Parameter	Type	Description
id	String	Application component ID.
name	String	Application component name.
status	Integer	Value: 0 or 1. 0: Normal. 1: Being deleted.
runtime	String	Runtime.
category	String	Application component type. Example: Webapp, MicroService, or Common.

Parameter	Type	Description
sub_category	String	Application component sub-type. Webapp sub-types include Web. MicroService sub-types include Java Chassis, Go Chassis, Mesher, Spring Cloud, and Dubbo. Common sub-type can be empty.
description	String	Description.
pipeline_ids	Array<String>	Pipeline ID list. A maximum of 10 pipeline IDs are supported.
project_id	String	Project ID.
application_id	String	Application ID.
source	Object	Source of the code or software package. See **Table 4-110**.
build	Object	Build. See **Table 4-113**.
create_time	Integer	Creation time.
update_time	Integer	Update time.
creator	String	Creator.

Table 4-110 source parameters

Parameter	Type	Description
kind	String	Type.
spec	Object	See **Table 4-111**.

Table 4-111 spec parameters

Parameter	Type	Description
storage	String	Storage mode.
type	String	Type.
url	String	Address of the software package or source code.
auth	String	Authentication mode.
properties	Object	Other attributes of the software package. You need to add these attributes only when you set **storage** to **obs**. For details, see **Table 4-112**.

Table 4-112 artifact spec properties parameters

Parameter	Type	Description
endpoint	String	OBS endpoint address. Example: https://obs.region_id.external_domain_name.com.
bucket	String	Name of the OBS bucket where the software package is stored.
key	String	Object in the OBS bucket, which is usually the name of the software package. If there is a folder, the path of the folder must be added. Example: test.jar or demo/test.jar.

Table 4-113 build parameters

Parameter	Type	Description
ID	String	Type
parameters	Map<String, Object>	See **Table 4-114**.

Table 4-114 parameters description

Parameter	Type	Description
build_cmd	String	Compilation command.
dockerfile_path	String	Address of the Docker file.
artifact_namespace	String	Build archive organization.
cluster_id	String	ID of the cluster to be built.
node_label_selector	Map<String, String>	**key** indicates the key of the tag, and **value** indicates the value of the tag.

Example

Example request

```
{
   "name": "test-micro",
   "description": "",
   "source": {
      "kind": "artifact",
      "spec": {
         "storage": "obs",
         "type": "package",
         "url": "obs://test-soft/fusionweather-1.0.0.jar",
         "auth": "iam",
```

```
            "webUrl": "",
            "properties": {
                "bucket": "test-soft",
                "endpoint": "https://obs.region_id.development.com:443",
                "key": "fusionweather-1.0.0.jar"
            }
        }
    },
    "build": {
        "parameters": {
            "artifact_namespace": "ns",
            "use_public_cluster": false,
            "cluster_id": "523498f1-36c4-11eb-ae36-0255ac1000c2",
            "cluster_name": "cce-test",
            "cluster_type": "VirtualMachine"
        }
    },
    "pipeline_ids": null
}
```

Example response

```
{
    "id": "1cfdda6f-84cd-4ead-8e09-628fabf662e2",
    "name": "test-micro",
    "runtime": "Java8",
    "category": "MicroService",
    "sub_category": "Java Chassis",
    "description": "",
    "project_id": "e7d2e9c589e5445e808a8ff0d1235aca",
    "application_id": "4d92833a-fa05-4fc0-a761-e67620022310",
    "source": {
        "kind": "artifact",
        "spec": {
            "storage": "obs",
            "type": "package",
            "url": "obs://test-soft/fusionweather-1.0.0.jar",
            "auth": "iam",
            "webUrl": "",
            "properties": {
                "bucket": "test-soft",
                "endpoint": "https://obs.region_id.development.com:443",
                "key": "fusionweather-1.0.0.jar"
            }
        }
    },
    "build": {
        "id": "zpdkj42b3rnko8bmyojzt1ng828ubychw13j8flv",
        "parameters": {
            "artifact_namespace": "ns",
            "use_public_cluster": false,
            "cluster_id": "523498f1-36c4-11eb-ae36-0255ac1000c2",
            "cluster_name": "cce-test",
            "cluster_type": "VirtualMachine"
        }
    },
    "pipeline_ids": null,
    "status": 0,
    "creator": "ss-test",
    "create_time": 1610331760105,
    "update_time": 1610519881943
}
```

Status Code

Table 4-115 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.4.4 Deleting an Application Component Based on the Component ID

Function

This API is used to delete an application component based on the component ID.

URI

DELETE /v2/{project_id}/cas/applications/{application_id}/components/{component_id}

Table 4-116 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.
component_id	String	Yes	Component ID.

Table 4-117 Query parameters

Parameter	Type	Mandatory	Description
force	boolean	No	Whether to forcibly delete the application component.

Request

None

Response

None

Example

Example request

None

Example response

None

Status Code

Table 4-118 Status codes

Status Code	Description
204	OK
400	Bad Request
404	Not Found
409	Conflict
500	Internal Server Error

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.4.5 Obtaining Application Component Information Based on the Component ID

Function

This API is used to obtain application component information based on the component ID.

URI

GET /v2/{project_id}/cas/applications/{application_id}/components/{component_id}

Table 4-119 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.
component_id	String	Yes	Component ID.

Request

None

Response

Table 4-120 Response parameters

Parameter	Type	Description
id	String	Application component ID.
name	String	Application component name.
status	Integer	Value: 0 or 1. 0: Normal. 1: Being deleted.
runtime	String	Runtime.
category	String	Application component type. Example: Webapp, MicroService, or Common.

Parameter	Type	Description
sub_category	String	Application component sub-type. Webapp sub-types include Web. MicroService sub-types include Java Chassis, Go Chassis, Mesher, Spring Cloud, and Dubbo. Common sub-type can be empty.
description	String	Description.
pipeline_ids	Array<String>	Pipeline ID list. A maximum of 10 pipeline IDs are supported.
project_id	String	Project ID.
application_id	String	Application ID.
source	Object	Source of the code or software package. See **Table 4-121**.
build	Object	Build. See **Table 4-124**.
create_time	Integer	Creation time.
update_time	Integer	Update time.
creator	String	Creator.

Table 4-121 source parameters

Parameter	Type	Description
kind	String	Type.
spec	Object	See **Table 4-122**.

Table 4-122 spec parameters

Parameter	Type	Description
storage	String	Storage mode.
type	String	Type.
url	String	Address of the software package or source code.
auth	String	Authentication mode.
properties	Object	Other attributes of the software package. You need to add these attributes only when you set **storage** to **obs**. For details, see **Table 4-123**.

Table 4-123 artifact spec properties parameters

Parameter	Type	Description
endpoint	String	OBS endpoint address. Example: https://obs.region_id.external_domain_name.com.
bucket	String	Name of the OBS bucket where the software package is stored.
key	String	Object in the OBS bucket, which is usually the name of the software package. If there is a folder, the path of the folder must be added. Example: test.jar or demo/test.jar.

Table 4-124 build parameters

Parameter	Type	Description
ID	String	Type.
parameters	Map<String,String>	See **Table 4-125**.

Table 4-125 parameters description

Parameter	Type	Description
build_cmd	String	Compilation command.
dockerfile_path	String	Address of the Docker file.
artifact_namespace	String	Build archive organization.
cluster_id	String	ID of the cluster to be built.
node_label_selector	Map<String, String>	**key** indicates the key of the tag, and **value** indicates the value of the tag.

Example

Example request

None

Example response

```
{
    "id": "1cfdda6f-84cd-4ead-8e09-628fabf662e2",
    "name": "test-micro",
    "runtime": "Java8",
    "category": "MicroService",
```

```
    "sub_category": "Java Chassis",
    "description": "",
    "project_id": "e7d2e9c589e5445e808a8ff0d1235aca",
    "application_id": "4d92833a-fa05-4fc0-a761-e67620022310",
    "source": {
        "kind": "artifact",
        "spec": {
            "storage": "obs",
            "type": "package",
            "url": "obs://test-soft/fusionweather-1.0.0.jar",
            "auth": "iam",
            "webUrl": "",
            "properties": {
                "bucket": "test-soft",
                "endpoint": "https://obs.region_id.development.com:443",
                "key": "fusionweather-1.0.0.jar"
            }
        }
    },
    "build": {
        "id": "zpdkj42b3rnko8bmyojzt1ng828ubychw13j8flv",
        "parameters": {
            "artifact_namespace": "ns",
            "use_public_cluster": false,
            "cluster_id": "523498f1-36c4-11eb-ae36-0255ac1000c2",
            "cluster_name": "cce-test",
            "cluster_type": "VirtualMachine"
        }
    },
    "pipeline_ids": null,
    "status": 0,
    "creator": "ss-test",
    "create_time": 1610331760105,
    "update_time": 1610519881943
}
```


Status Code

Table 4-126 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.5 Instance

4.5.1 Creating an Application Component Instance

Function

This API is used to create an application component instance.

URI

POST /v2/{project_id}/cas/applications/{application_id}/components/{component_id}/instances

Table 4-127 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.
component_id	String	Yes	Component ID.

Request

Table 4-128 Request parameters

Parameter	Type	Mandatory	Description
name	String	Yes	Component instance name. The value contains 2 to 63 characters consisting of lowercase letters, digits, and hyphens (-). It starts with a lowercase letter and ends with a lowercase letter or digit.
environment_id	String	Yes	Environment ID.
replica	Integer	Yes	Number of instance replicas.

Parameter	Type	Mandatory	Description
flavor_id	String	Yes	Resource specifications, which can be obtained by using the API in **Obtaining All Supported Flavors of Application Resources**. If you need to customize resource specifications, the format is as follows: CUSTOM-xxG:xxC-xxC:xxGi-xxGi. Where: • xxG: storage capacity allocated to a component instance. It is a reserved field. You can set it to a fixed number. • xxC-xxC: the maximum and minimum number of CPU cores allocated to a component instance. • xxGi-xxGi: the maximum and minimum memory allocated to a component instance. For example, **CUSTOM-10G:0.5C-0.25C:1.6Gi-0.8Gi** indicates that the maximum number of CPU cores allocated to a component instance is 0.5, the minimum number of CPU cores is 0.25, the maximum memory is 1.6 Gi, and the minimum memory is 0.8 Gi.
artifacts	Map<String, Object>	Yes	Artifact. **key** indicates the component name. In the Docker container scenario, **key** indicates the container name. See **Table 4-145**.
version	String	Yes	Application component version that meets version semantics. Example: 1.0.0.
configuration	Object	No	Configuration parameters, such as environment variables, deployment configurations, and O&M monitoring. By default, this parameter is left blank. See **Table 4-129**.
description	String	No	Description. The value can contain up to 128 characters.
external_accesses	Array of objects	No	External network access. See **Table 4-146**.
refer_resources	Array of objects	Yes	Deployed resources. See **Table 4-147**.

Table 4-129 configuration parameters

Parameter	Type	Mandatory	Description
env	Array of objects	No	Environment variable. See **Table 4-130**.
storage	Array of objects	No	Data storage configuration. For details, see **Table 4-131**.
strategy	Object	No	Upgrade policy. For details, see **Table 4-132**.
lifecycle	Object	No	Lifecycle. For details, see **Table 4-133**.
scheduler	Object	No	Scheduling policy. For details, see **Table 4-134**.
probes	Object	No	Health check. For details, see **Table 4-135**.

Table 4-130 environment parameters

Parameter	Type	Mandatory	Description
name	String	Yes	Name of the environment variable. The value contains 1 to 64 characters consisting of letters, digits, underscores (_), hyphens (-), and dots (.), and cannot start with a digit.
value	String	Yes	Environment variable value.

Table 4-131 storage parameters

Parameter	Type	Mandatory	Description
type	String	Yes	Storage type. Value: • HostPath: host path mounting. • EmptyDir: temporary directory mounting. • ConfigMap: configuration item mounting. • Secret: secret volume mounting. • PersistentVolumeClaim: cloud storage mounting.
parameters	Object	Yes	Storage parameter. For details, see **Table 4-144**.
mounts	Array of objects	Yes	Directory mounted to the container. For details, see **Table 4-143**.

Table 4-132 strategy parameter

Parameter	Type	Mandatory	Description
upgrade	String	No	Upgrade policy. Value: Recreate or RollingUpdate (default). The former indicates in-place upgrade while the latter indicates rolling upgrade.

Table 4-133 lifecycle parameters

Parameter	Type	Mandatory	Description
entrypoint	Object	No	Startup command. For details, see **Table 4-141**.
post-start	Object	No	Post-start processing. For details, see **Table 4-141**.
pre-stop	Object	No	Pre-stop processing. For details, see **Table 4-141**.

Table 4-134 scheduler parameters

Parameter	Type	Mandatory	Description
affinity	Object	No	Affinity. For details, see **Table 4-140**.
anti-affinity	Object	No	Anti-affinity. For details, see **Table 4-140**.

Table 4-135 probes parameters

Parameter	Type	Mandatory	Description
livenessProbe	Object	No	Component liveness probe. For details, see **Table 4-136**.
readinessProbe	Object	No	Component service probe. For details, see **Table 4-136**.

Table 4-136 probes_object parameters

Parameter	Type	Mandatory	Description
type	String	Yes	Value: http, tcp, or command. The check methods are HTTP request check, TCP port check, and command execution check, respectively.
parameters	Object	Yes	Parameters. • If **type** is set to **http**, see **Table 4-137**. • If **type** is set to **tcp**, see **Table 4-139**. • If **type** is set to **command**, see **Table 4-138**.
delay	Integer	No	Interval between the startup and detection.
timeout	Integer	No	Detection timeout interval.

Table 4-137 probes_param_http parameters

Parameter	Type	Mandatory	Description
scheme	String	Yes	Value: HTTP or HTTPS.
host	String	No	Pod IP address (default). You can specify an IP address.
port	Integer	Yes	Port.
path	String	Yes	Request path.

Table 4-138 probes_param_command parameters

Parameter	Type	Mandatory	Description
command	Array of String	Yes	Command list.

Table 4-139 probes_param_tcp parameters

Parameter	Type	Mandatory	Description
port	Integer	Yes	Port

Table 4-140 scheduler_affinity parameters

Parameter	Type	Mandatory	Description
az	Array of String	No	AZ list.
node	Array of String	No	Node private IP address list.
application	Array of String	No	Component instance list.

Table 4-141 lifecycle_entrypoint parameters

Parameter	Type	Mandatory	Description
command	Array of String	Yes	Command that can be executed.
args	Array of String	Yes	Running parameters.

Table 4-142 lifecycle_process parameters

Parameter	Type	Mandatory	Description
type	String	Yes	Value: command or http. The former indicates command execution and the latter indicates HTTP requests.
parameters	Object	Yes	Post-start or pre-stop processing parameters. See **Table 4-143**.

Table 4-143 lifecycle_process_parameters parameters

Parameter	Type	Mandatory	Description
command	Array of String	Yes	Command parameters, for example, ["sleep", "1"]. This parameter is applicable to the **command** type.
host	String	No	Pod IP address (default). You can also specify an IP address. This parameter is applicable to the **http** type.
port	Integer	Yes	Port number. This parameter is applicable to the **http** type.
path	String	Yes	Request URL. This parameter is applicable to the **http** type.

Table 4-144 storage_parameters parameters

Parameter	Type	Mandatory	Description
path	String	No	Host path. This parameter is applicable to the **HostPath** storage type.
name	String	No	Name of a configuration item or secret. This parameter is applicable to the **ConfigMap** and **Secret** storage type.
claimName	String	No	PVC name. This parameter is applicable to the **PersistentVolumeClaim** storage type.

Table 4-145 artifact parameters

Parameter	Type	Mandatory	Description
storage	String	Yes	Storage mode. Value: swr or obs.
type	String	Yes	Type. Value: package (VM-based deployment) or image (container-based deployment).
url	String	Yes	Software package or image address. For a component deployed on a VM, this parameter is the software package address. For a component deployed based on a container, this parameter is the image address or **component name:v${index}**. The latter indicates that the component source code or the image automatically built using the software package will be used.
auth	String	Yes	Authentication mode. Value: iam or none. Default value: iam.
version	String	No	Version number.
properties	Map<String, Object>	No	Property information.

Table 4-146 external_accesses parameters

Parameter	Type	Mandatory	Description
protocol	String	No	Protocol. Value: http or https.
address	String	No	Access address.
forward_port	Integer	No	Port number.

Table 4-147 refer_resources parameters

Parameter	Type	Mandatory	Description
id	String	Yes	Resource ID. Note: If **type** is set to **ecs**, the value of this parameter must be **Default**.
type	String	Yes	Basic resources: cce, and ecs. Optional resources: dcs, elb, and other services.
refer_alias	String	No	Application alias, which is provided only in DCS scenario. Value: "distributed_session", "distributed_cache", "distributed_session, distributed_cache". Default value: "distributed_session, distributed_cache".
parameters	Map<String,Object>	No	Reference resource parameter. **NOTICE** • When **type** is set to **cce**, this parameter is mandatory. You need to specify the namespace of the cluster where the component is to be deployed. Example: {"namespace": "default"}. • When **type** is set to **ecs**, this parameter is mandatory. You need to specify the hosts where the component is to be deployed. Example: {"hosts": ["04d9f887-9860-4029-91d1-7d3102903a69", "04d9f887-9860-4029-91d1-7d3102903a70"]}}.

Response

Table 4-148 Response parameters

Parameter	Type	Description
job_id	String	Job ID, which is used to query information about the created job.
instance_id	String	Component instance ID.

Example

Example request

```
{
    "name": "component-instance-name",
    "environment_id": "6e763000-9128-4a9d-adea-34c42cc5344d",
    "flavor_id": "MICRO-5G:0.5C:1G",
    "replica": 1,
    "artifacts": {
        "container-name": {
            "storage": "swr",
            "type": "image",
            "url": "{IP}:20202/xxxxxxxx/junit-platformstest:1.0",
            "auth": "iam"
        }
    },
    "version": 1,
    "description": "instance desc",
    "configuration": {
        "env": [
            {
                "name": "log-level",
                "value": "warn"
            }
        ]
    },
    "refer_resources": [
        {
            "id": "b6862a62-d916-11e9-bdf1-0255ac101fd9",
            "type": "cce",
            "parameters": {
                "namespace": "default"
            }
        },
        {
            "id": "8c0a45cc-626f-4d65-8257-507ee059aa9a",
            "type": "elb"
        }
    ]
}
```

Example response

```
{
    "instance_id": "89f5baf5-efe4-4f12-9c0d-734d2af5a184",
    "job_id": "JOB66761060-f209-407c-a093-4df6f531b9dc"
}
```

Status Code

Table 4-149 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.5.2 Obtaining All Instances of a Component

Function

This API is used to obtain all instances of a component.

URI

GET /v2/{project_id}/cas/applications/{application_id}/components/{component_id}/instances

Table 4-150 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.
component_id	String	Yes	Component ID.

Table 4-151 Query parameters

Parameter	Type	Mandatory	Description
limit	Integer	No	Number of records to be displayed. The value is **1000** or ranges from 0 to 100. If the specified value is not within the range, value **10** is assigned. In the non-pagination scenario, the value is **1000**. In the pagination scenario, the value ranges from 0 to 100.
offset	Integer	No	Offset.
order_by	String	No	Sorting field. By default, query results are sorted by creation time. The following enumerated values are supported: create_time, name, version, and update_time.
order	String	No	Descending or ascending order. Default value: desc.

Request

None

Response

Table 4-152 Response parameters

Parameter	Type	Description
count	Integer	Total number of instances.
instances	Array of objects	Instance parameters. See **Table 4-153**.

Table 4-153 instance parameters

Parameter	Type	Description
id	String	Component instance ID.
application_id	String	Application ID.
component_id	String	Component ID.
name	String	Component instance name.

Parameter	Type	Description
environment_id	String	Component environment ID.
platform_type	String	Platform type. Value: cce or vmapp.
artifacts	Map<String, Object>	Artifact. **key** indicates the component name. In the Docker container scenario, **key** indicates the container name. See **Table 4-154**.
version	String	Component version.
create_time	Integer	Creation time.
update_time	Integer	Update time.
external_accesses	Array of objects	Access mode. See **Table 4-155**.
status_detail	Object	Status details. See **Table 4-156**.

Table 4-154 artifact parameters

Parameter	Type	Description
storage	String	Storage mode. Value: swr or obs.
type	String	Type. Value: package (VM-based deployment) or image (container-based deployment).
url	String	Software package or image address.
auth	String	Authentication mode. Value: iam or none. Default value: iam.
version	String	Version number.
properties	Map<String,String>	Property information.

Table 4-155 external_accesses parameters

Parameter	Type	Description
protocol	String	Protocol.
address	String	Access address.
forward_port	Integer	Port for listening to an application component process.

Parameter	Type	Description
type	String	Type.
status	String	Status.
create_time	Integer	Creation time.
update_time	Integer	Update time.

Table 4-156 status_detail parameters

Parameter	Type	Description
enterprise_project_id	String	Enterprise project ID.
status	String	Instance status.
available_replica	Integer	Number of normal instance replicas.
replica	Integer	Number of instance replicas.
fail_detail	String	Failure description.
last_job_id	String	Latest job ID.
last_job_status	String	Latest job status.

Example

Example request

None

Example response

```
{
   "instances": [
      {
         "id": "11eddb33-140b-4e51-b1e2-6ec265373ca3",
         "application_id": "4d92833a-fa05-4fc0-a761-e67620022310",
         "component_id": "1cfdda6f-84cd-4ead-8e09-628fabf662e2",
         "name": "test-micro-test-env-7iafjk",
         "environment_id": "61b81021-21d5-42f3-b80e-0b6bd10dbf7d",
         "platform_type": "cce",
         "version": "1.0.0",
         "artifacts": {
            "test-micro": {
               "storage": "swr",
               "type": "image",
               "url": "swr.region_id.development.com/ns/examples:v1",
               "auth": "iam",
               "version": "1.0.0",
               "properties": {}
            }
```

```
        },
        "create_time": 1610331819996,
        "update_time": 1610331830398,
        "status_detail": {
            "status": "RUNNING",
            "replica": 1,
            "available_replica": 1,
            "fail_detail": null,
            "last_job_id": "JOB7bc0366c-dc43-41f9-b65f-8d6078038488",
            "last_job_status": "SUCCEEDED",
            "enterprise_project_id": "0"
        }
    }
  ],
  "count": 1
}
```


Status Code

Table 4-157 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.5.3 Querying the Operations Performed on a Component Instance

Function

This API is used to query the operations performed on a component instance.

URI

POST /v2/{project_id}/cas/applications/{application_id}/components/{component_id}/instances/{instance_id}/action

Table 4-158 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.
component_id	String	Yes	Component ID.
instance_id	String	Yes	Component instance ID.

Request

Table 4-159 Request parameters

Parameter	Type	Mandatory	Description
action	String	Yes	Operation. Value: start, stop, restart, scale, or rollback.
parameters	Object	No	Operation parameter, which is mandatory when **action** is set to **scale** or **rollback**. See **Table 4-160**.

Table 4-160 parameters description

Parameter	Type	Mandatory	Description
replica	Integer	No	Number of instances, which is mandatory when **action** is set to **scale**.
hosts	Array	No	ECS ID list. This parameter specifies the ECS deployed during VM scaling, and is mandatory for scaling.
version	String	No	Version number, which is mandatory when **action** is set to **rollback**. You can obtain the version number by using the API in **Obtaining Component Instance Snapshots**.

Response

Table 4-161 Response parameters

Parameter	Type	Description
job_id	String	Job ID.

Example

Example request

```
{
  "action": "scale",
  "parameters": {
    "replica": 4
  }
}
```

Example response

```
{
    "job_id": "JOB66761060-f209-407c-a093-4df6f531b9dc"
}
```

Status Code

Table 4-162 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.5.4 Modifying an Application Component Instance

Function

This API is used to modify an application component instance.

URI

PUT /v2/{project_id}/cas/applications/{application_id}/components/{component_id}/instances/{instance_id}

Table 4-163 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.
component_id	String	Yes	Component ID.
instance_id	String	Yes	Component instance ID.

Request

Table 4-164 Request parameters

Parameter	Type	Mandatory	Description
version	String	Yes	Application component version that meets version semantics. Example: 1.0.1.

Parameter	Type	Mandatory	Description
flavor_id	String	No	Resource specifications, which can be obtained by using the API in **Obtaining All Supported Flavors of Application Resources**. If you need to customize resource specifications, the format is as follows: CUSTOM-xxG:xxC-xxC:xxGi-xxGi. Where: • xxG: storage capacity allocated to a component instance. It is a reserved field. You can set it to a fixed number. • xxC-xxC: the maximum and minimum number of CPU cores allocated to a component instance. • xxGi-xxGi: the maximum and minimum memory allocated to a component instance. For example, **CUSTOM-10G:0.5C-0.25C:1.6Gi-0.8Gi** indicates that the maximum number of CPU cores allocated to a component instance is 0.5, the minimum number of CPU cores is 0.25, the maximum memory is 1.6 Gi, and the minimum memory is 0.8 Gi.
artifacts	Map<String, Object>	No	Artifact. **key** indicates the component name. In the Docker container scenario, **key** indicates the container name. See **Table 4-165**.
configuration	Map<String, Object>	No	Configuration parameters, such as environment variables, deployment configurations, and O&M monitoring. By default, this parameter is left blank. See **Table 4-129** in **Creating an Application Component Instance**.
description	String	No	Description. The value can contain up to 128 characters.
external_accesses	Array of objects	No	Access mode. See **Table 4-166**.
refer_resources	Array of objects	No	Deployed resources. See **Table 4-167**.

Table 4-165 artifact parameters

Parameter	Type	Mandatory	Description
storage	String	Yes	Storage mode. Value: swr or obs.
type	String	Yes	Type. Value: package (VM-based deployment) or image (container-based deployment).
url	String	Yes	Software package or image address. For a component deployed on a VM, this parameter is the software package address. For a component deployed based on a container, this parameter is the image address or **component name:v${index}**. The latter indicates that the component source code or the image automatically built using the software package will be used.
auth	String	Yes	Authentication mode. Value: iam or none. Default value: iam.
version	String	No	Version number.
properties	Map<String, String>	No	Property information.

Table 4-166 external_accesses parameters

Parameter	Type	Mandatory	Description
protocol	String	No	Protocol. Value: http or https.
address	String	No	Access address. Example: www.example.com.
forward_port	Integer	No	Port for listening to an application component process.

Table 4-167 refer_resources parameters

Parameter	Type	Mandatory	Description
id	String	Yes	Resource ID.

Parameter	Type	Mandatory	Description
type	String	Yes	Resource type. Example: dcs.
refer_alias	String	No	Application alias. This parameter is available only when **type** is set to **dcs**. Value: "distributed_session", "distributed_cache", or "distributed_session, distributed_cache" (default).
parameters	Map<String, Object>	No	Reference resource parameter.

Response

Table 4-168 Response parameters

Parameter	Type	Description
job_id	String	Job ID, which is used to query information about the created job.

Example

Example request

```
{
    "description": "",
    "artifacts": {
        "test-micro": {
            "storage": "swr",
            "type": "image",
            "url": "swr.roma-dev-1.roma.development.com/ns1/examples:v2",
            "auth": "iam",
            "version": "1.0.0",
            "properties": {}
        }
    },
    "refer_resources": [
        {
            "id": "523498f1-36c4-11eb-ae36-0255ac1000c2",
            "type": "cce",
            "parameters": {
                "namespace": "default"
            }
        },
        {
            "id": "default",
            "type": "cse"
        }
    ],
    "version": "1.0.2"
}
```

Example response

```
{
    "job_id": "JOB66761060-f209-407c-a093-4df6f531b9dc"
}
```

Status Code

Table 4-169 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.5.5 Deleting an Application Component Instance

Function

This API is used to delete an application component instance.

URI

DELETE /v2/{project_id}/cas/applications/{application_id}/components/{component_id}/instances/{instance_id}

Table 4-170 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.
component_id	String	Yes	Component ID.
instance_id	String	Yes	Component instance ID.

Table 4-171 Query parameters

Parameter	Type	Mandatory	Description
force	boolean	No	Whether to forcibly delete the application component.

Request

None

Response

Table 4-172 Response parameters

Parameter	Type	Description
job_id	String	Job ID

Example

Example request

None

Example response

```
{
    "job_id": "JOB66761060-f209-407c-a093-4df6f531b9dc"
  }
```

Status Code

Table 4-173 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.5.6 Querying Instance Details Based on the Instance ID

Function

This API is used to query instance details based on the instance ID.

URI

GET /v2/{project_id}/cas/applications/{application_id}/components/{component_id}/instances/{instance_id}

Table 4-174 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.
component_id	String	Yes	Component ID.
instance_id	String	Yes	Component instance ID.

Request

None

Response

Table 4-175 Response parameters

Parameter	Type	Description
id	String	Component instance ID.
name	String	Component instance name.
environment_id	String	Component environment ID.
platform_type	String	Platform type. Value: cce or vmapp.
description	String	Instance description.
flavor_id	String	Resource specifications.
artifacts	Map<String, Object>	Artifact. **key** indicates the component name. In the Docker container scenario, **key** indicates the container name. See **Table 4-176**.

Parameter	Type	Description
version	String	Component version.
configuration	Map<String, Object>	Component configurations, such as environment variables.
creator	String	Creator.
create_time	Integer	Creation time.
update_time	Integer	Update time.
external_accesses	Array of objects	Access mode. See **Table 4-177**.
refer_resources	Array of objects	Deployed resources. See **Table 4-178**.
status_detail	Object	Status details. See **Table 4-179**.

Table 4-176 artifact parameters

Parameter	Type	Description
storage	String	Storage mode. Value: swr or obs.
type	String	Type. Value: package (VM-based deployment) or image (container-based deployment).
url	String	Software package or image address.
auth	String	Authentication mode. Value: iam or none. Default value: iam.
version	String	Version number.
properties	Map<String,String>	Property information.

Table 4-177 external_accesses parameters

Parameter	Type	Description
protocol	String	Protocol.
address	String	Access address.
forward_port	Integer	Port for listening to an application component process.
type	String	Type.
status	String	Status.

Parameter	Type	Description
create_time	Integer	Creation time.
update_time	Integer	Update time.

Table 4-178 refer_resources parameters

Parameter	Type	Description
id	String	Resource ID.
type	String	Basic resources: cce, and ecs. Optional resources: dcs, elb, and other services.
refer_alias	String	Application alias, which is provided only in DCS scenario. Value: "distributed_session", "distributed_cache", "distributed_session, distributed_cache". Default value: "distributed_session, distributed_cache".
parameters	Map<String,Object>	Reference resource parameter.

Table 4-179 status_detail parameters

Parameter	Type	Description
enterprise_project_id	String	Enterprise project ID.
status	String	Instance status.
available_replica	Integer	Number of normal instance replicas.
replica	Integer	Number of instance replicas.
fail_detail	String	Failure description.
last_job_id	String	Latest job ID.
last_job_status	String	Latest job status.

Example

Example request

None

Example response


```
{
    "id": "11eddb33-140b-4e51-b1e2-6ec265373ca3",
    "name": "test-micro-test-env-7iafjk",
    "environment_id": "61b81021-21d5-42f3-b80e-0b6bd10dbf7d",
    "platform_type": "cce",
    "flavor_id": "CUSTOM-10G:250m-250m:0.5Gi-0.5Gi",
    "artifacts": {
        "test-micro": {
            "storage": "swr",
            "type": "image",
            "url": "swr.region_id.development.com/ns/examples:v2",
            "auth": "iam",
            "version": "1.0.0",
            "properties": {}
        }
    },
    "version": "1.0.2",
    "description": "",
    "configuration": {
        "env": [
            {
                "name": "PAAS_PROJECT_ID",
                "value": "e7d2e9c589e5445e808a8ff0d1235aca",
                "internal": true
            }
        ],
        "lifecycle": {
            "post-start": null,
            "pre-stop": null
        },
        "scheduler": {
            "affinity": {
                "az": [],
                "node": [],
                "application": []
            },
            "anti-affinity": {
                "az": [],
                "node": [],
                "application": []
            },
            "isSynchronized": true
        },
        "strategy": {
            "upgrade": "RollingUpdate",
            "spec": {
                "maxSurge": 0,
                "maxUnavailable": 1
            }
        }
    },
    "creator": null,
    "create_time": 1610331819996,
    "update_time": 1610522676792,
    "external_accesses": null,
    "refer_resources": [
        {
            "id": "523498f1-36c4-11eb-ae36-0255ac1000c2",
            "type": "cce",
            "parameters": {
                "namespace": "default"
            }
        },
        {
            "id": "default",
            "type": "cse"
```

```
        }
    ],
    "status_detail": {
        "status": "RUNNING",
        "replica": 1,
        "available_replica": 1,
        "fail_detail": null,
        "last_job_id": "JOB16c33466-41e6-4c5b-a9ac-2059253a1e5c",
        "last_job_status": "SUCCEEDED",
        "enterprise_project_id": "0"
    }
}
```


Status Code

Table 4-180 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.5.7 Obtaining Component Instance Snapshots

Function

This API is used to obtain component instance snapshots.

Instance snapshots refer to the instance information about the source version. They can be used for version rollback.

URI

GET /v2/{project_id}/cas/applications/{application_id}/components/{component_id}/instances/{instance_id}/snapshots

Table 4-181 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
application_id	String	Yes	Application ID.

Parameter	Type	Mandatory	Description
component_id	String	Yes	Component ID.
instance_id	String	Yes	Component instance ID.

Table 4-182 Query parameters

Parameter	Type	Mandatory	Description
limit	Integer	No	Number of records to be displayed. The value is **1000** or ranges from 0 to 100. If the specified value is not within the range, value **10** is assigned. In the non-pagination scenario, the value is **1000**. In the pagination scenario, the value ranges from 0 to 100.
offset	Integer	No	Offset.
order_by	String	No	Sorting field. By default, query results are sorted by creation time.
order	String	No	Descending or ascending order. Default value: desc.

Request

None

Response

Table 4-183 Response parameters

Parameter	Type	Description
count	Integer	Number of snapshots.
snapshots	Array of objects	Snapshot parameters. See **Table 4-184**.

Table 4-184 snapshot parameters

Parameter	Type	Description
create_time	Integer	Creation time.

Parameter	Type	Description
description	String	Description.
instance_id	String	Component instance ID.
version	String	Version number.

Example

Example request

None

Example response

```
{
   "snapshots": [
      {
         "version": "1.0.0",
         "description": "{\"action\":\"New Version: 1.0.0\"}",
         "instance_id": "11eddb33-140b-4e51-b1e2-6ec265373ca3",
         "create_time": 1610331830841
      }
   ],
   "count": 1
}
```

Status Code

Table 4-185 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

4.6 Job

4.6.1 Obtaining Job Details

Function

This API is used to obtain job details.

URI

GET /v2/{project_id}/cas/jobs/{job_id}

Table 4-186 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
job_id	String	Yes	Job ID.

Table 4-187 Query parameters

Parameter	Type	Mandatory	Description
instance_id	String	No	Component instance ID.
limit	Integer	No	Number of specified records.
offset	Integer	No	Offset.
desc	String	No	Whether query results are displayed in descending order. Value: true (descending order) or false (ascending order).

Request

None

Response

Table 4-188 Parameters

Parameter	Type	Description
task_count	Integer	Number of tasks.
job	Object	Job parameters. See **Table 4-189**.

Parameter	Type	Description
tasks	Array<Object>	Task parameters. See **Table 4-190**.

Table 4-189 job parameters

Parameter	Type	Description
CREATED_BY	String	Creator.
EXECUTION_STATUS	String	Execution status. • SUBMITTED • SCHEDULED • RUNNING • FAILED • SUCCEEDED • CANCELLED • PARTIALLY_FAILED • ROLLBACKING • ROLLBACKED • UNKNOWN
JOB_DESC	String	Job description.
JOB_ID	String	Job ID.
JOB_NAME	String	Job name.
JOB_TYPE	String	Type. • PROVISION • DEPROVISION • DAILY_OPERATION • SYSTEM_INNER • UNKNOWN
ORDER_ID	String	Order ID.
PROJECT_ID	String	Tenant's project ID.
SERVICE_INSTANCE_ID	String	Instance ID.

Table 4-190 task parameters

Parameter	Type	Description
CREATED_AT	String	Creation time.

Parameter	Type	Description
LAST_HEALTH_CHECK	String	Health check time.
MESSAGES	String	Message.
OWNER_ID	String	Creator ID.
TASK_ID	String	Task ID.
TASK_INDEX	Integer	Task index.
TASK_NAME	String	Task name.
TASK_STATUS	String	Task status. • SUBMITTED • SCHEDULED • RUNNING • FAILED • SUCCEEDED • CANCELLED • PARTIALLY_FAILED • SKIPPED • ROLLBACKING • ROLLBACKED • UNKNOWN
TASK_TYPE	String	Task type. • kube • restful • sleep • echo • context • sub-job

Example

Example request

None

Example response

```
{
    "job": {
        "JOB_ID": "JOB474362ef-091d-429f-9b3e-c9246b37dee6",
        "JOB_NAME": "provision/provMain",
        "JOB_TYPE": "PROVISION",
        "SERVICE_INSTANCE_ID": "429a35a0-dee8-4648-86f6-d1f8bc74ba52",
        "PROJECT_ID": "063a3158b1a34710b36ad208b9497d00",
```

```
    "ORDER_ID": "",
    "JOB_DESC": "provision/provMain",
    "EXECUTION_STATUS": "FAILED",
    "CREATED_BY": "585cc5c16cda4954b28d4eb460f06261"
  },
  "tasks": [
    {
      "MESSAGES": "{}",
      "TASK_ID": "Task-bc64b46c-1688-4787-b7ff-87cb53b346ea",
      "TASK_TYPE": "echo",
      "TASK_INDEX": 1,
      "TASK_NAME": "SEPARATOR_PREPARE_RESOURCES",
      "TASK_STATUS": "SUCCEEDED",
      "LAST_HEALTH_CHECK": "2020-02-04T04:01:03.344Z",
      "CREATED_AT": "2020-02-04T04:01:03.344Z",
      "OWNER_ID": "cas-mgr-69bb87d7c-wpqwb:30114"
    },
    {
      "MESSAGES": "{}",
      "TASK_ID": "Task-15ec252f-bca0-401e-ae5b-798a639df92e",
      "TASK_TYPE": "echo",
      "TASK_INDEX": 4,
      "TASK_NAME": "SEPARATOR_BUILDING",
      "TASK_STATUS": "SUCCEEDED",
      "LAST_HEALTH_CHECK": "2020-02-04T04:01:03.346Z",
      "CREATED_AT": "2020-02-04T04:01:03.345Z",
      "OWNER_ID": "cas-mgr-69bb87d7c-wpqwb:30114"
    },
    {
      "MESSAGES": "{}",
      "TASK_ID": "Task-52a5d1ae-2c54-46ec-8134-53e7b6a45fb7",
      "TASK_TYPE": "restful",
      "TASK_INDEX": 6,
      "TASK_NAME": "TN00091",
      "TASK_STATUS": "SUCCEEDED",
      "LAST_HEALTH_CHECK": "2020-02-04T04:01:08.924Z",
      "CREATED_AT": "2020-02-04T04:01:03.346Z",
      "OWNER_ID": "cas-mgr-69bb87d7c-wpqwb:30114"
    },
    {
      "MESSAGES": "{}",
      "TASK_ID": "Task-a149b585-d31b-4c72-a665-d820efcb0bf9",
      "TASK_TYPE": "restful",
      "TASK_INDEX": 7,
      "TASK_NAME": "TN00094",
      "TASK_STATUS": "SUCCEEDED",
      "LAST_HEALTH_CHECK": "2020-02-04T04:01:08.961Z",
      "CREATED_AT": "2020-02-04T04:01:08.925Z",
      "OWNER_ID": "cas-mgr-69bb87d7c-wpqwb:30114"
    },
    {
      "MESSAGES": "{
 \"Message\" : \"ECC00011\",
 \"Detail\" : \"Building failed.\"
}",
      "TASK_ID": "Task-a72fe8d8-8cc1-42b5-a097-0cf9e29692a7",
      "TASK_TYPE": "restful",
      "TASK_INDEX": 9,
      "TASK_NAME": "TN00097",
      "TASK_STATUS": "FAILED",
      "LAST_HEALTH_CHECK": "2020-02-04T04:01:38.975Z",
      "CREATED_AT": "2020-02-04T04:01:08.962Z",
      "OWNER_ID": "cas-mgr-69bb87d7c-wpqwb:30114"
    }
  ],
  "task_count": 5
}
```

Status Code

Table 4-191 Status codes

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

Error Code

Error code must be in the format of SVCSTG.00100.[Error_ID]. Example: SVCSTG.00100400. See **Error Codes of Application Management APIs**.

5 Git Repository Access APIs

5.1 Obtaining a Git Repository Authorization List

Function

This API is used to obtain a Git repository authorization list.

URI

GET /v1/{project_id}/git/auths

Table 5-1 describes the parameters.

Table 5-1 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.

Request

Request parameters

None

Response

Response parameters

Table 5-2 Response parameters

Parameter	Type	Description
authorizations	Array<Object>	Authorization structure. For details, see Table 5-3.

Table 5-3 authorizations parameters

Parameter	Type	Description
name	String	Authorization name.
repo_type	String	Repository type. Value: github, gitlab, gitee, or bitbucket.
repo_host	String	Repository address.
repo_home	String	Repository homepage.
repo_user	String	Repository username.
avartar	String	Avatar.
token_type	String	Authorization mode.
create_time	Long	Creation time.
update_time	Long	Update time.
status	Integer	Status.

Example

Example request

None

Example response

```
{
  "authorizations": [
    {
      "name": "string",
      "repo_type": "string",
      "repo_host": "string",
      "repo_home": "string",
      "repo_user": "string",
      "avartar": "string",
      "token_type": "string",
      "create_time": 0,
      "update_time": 0,
      "status": 0
    }
  ]
}
```

Status Code

Table 5-4 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.2 Obtaining an Authorization Redirection URL

Function

This API is used to obtain an authorization redirection URL.

URI

GET /v1/{project_id}/git/auths/{repo_type}/redirect

Table 5-5 describes the parameters.

Table 5-5 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
repo_type	String	Yes	Repository type. Value: github, gitlab, gitee, or bitbucket.

Table 5-6 Query parameters

Parameter	Type	Mandatory	Description
tag	String	No	Site tag.

Request

Request parameters

None

Response

Response parameters

Table 5-7 Response parameters

Parameter	Type	Description
url	String	Authorization redirection URL.

Example

Example request

None

Example response

```
{
  "url": "string"
}
```

Status Code

Table 5-8 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.3 Creating OAuth Authorization

Function

This API is used to create OAuth authorization.

URI

POST /v1/{project_id}/git/auths/{repo_type}/oauth

Table 5-9 describes the parameters.

Table 5-9 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
repo_type	String	Yes	Repository type. Value: github, gitlab, gitee, or bitbucket.

Table 5-10 Query parameters

Parameter	Type	Mandatory	Description
tag	String	No	Site tag.

Request

Request parameters

Table 5-11 Request parameters

Parameter	Type	Mandatory	Description
name	String	Yes	Authorization name.
code	String	Yes	Query parameter in the redirection URL after Git repository authorization is performed.
state	String	Yes	One-off authentication code and random string received after Git repository authorization is performed.

NOTE

To obtain the values of **code** and **state**, perform the following steps:

1. Enter the URL obtained in **Obtaining an Authorization Redirection URL** in the address box of the browser and press **Enter**.
2. On the login page, enter your code repository account and password.

   If you have logged in to the code repository, skip this step.
3. On the login page of the console, enter the account name and password, and click **Log In**.

   If you have logged in to the console, skip this step.
4. After the system displays a message indicating that the authorization fails to be created, copy the URL in the address box. Example:

   https://console****.com/servicestage/oauth?repo_type=github&**code**=04bed2df009f3fc9***c&**state**=%242a%2410%24.uQYThj5yKFs51w9s9ajeuQ0NSGshPH7IUfXoZLPlz7J%2FoAJ8B***
5. Decode the obtained URL to obtain the decoded URL. Example:

   https://console.****.com/servicestage/oauth?repo_type=github&**code**=04bed2df009f3fc9***c&**state**=$2a$10$.uQYThj5yKFs51w9s9ajeuQ0NSGshPH7IUfXoZLPlz7J/oAJ8B***

   Obtain the values of **code** and **state**.

The value of the obtained **state** is a one-off authentication code and random string, which can be used only once. If you need to call this API again to create OAuth authorization, follow the steps in **Obtaining an Authorization Redirection URL** again to obtain the URL and obtain the values of **code** and **state**.

Response

Response parameters

Table 5-12 Response parameters

Parameter	Type	Description
authorization	Object	Authorization structure. For details, see **Table 5-13**.

Table 5-13 authorization parameters

Parameter	Type	Description
name	String	Authorization name.
repo_type	String	Repository type. Value: github, gitlab, gitee, or bitbucket.
repo_host	String	Repository address.
repo_home	String	Repository homepage.
repo_user	String	Repository username.
avartar	String	Avatar.

Parameter	Type	Description
token_type	String	Authorization mode.
create_time	Long	Creation time.
update_time	Long	Update time.
status	Integer	Status.

Example

Example request

```
{
  "name": "string",
  "code": "string",
  "state": "string"
}
```

Example response

```
{
  "authorization": {
    "name": "string",
    "repo_type": "string",
    "repo_host": "string",
    "repo_home": "string",
    "repo_user": "string",
    "avartar": "string",
    "token_type": "string",
    "create_time": 0,
    "update_time": 0,
    "status": 0
  }
}
```

Status Code

Table 5-14 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.4 Creating Private Token Authorization

Function

This API is used to create private token authorization.

URI

POST /v1/{project_id}/git/auths/{repo_type}/personal

Table 5-15 describes the parameters.

Table 5-15 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
repo_type	String	Yes	Repository type. Value: github, gitlab, or gitee.

Request

Request parameters

Table 5-16 Request parameters

Parameter	Type	Mandatory	Description
name	String	Yes	Authorization name.
token	String	Yes	Repository token.
host	String	No	Repository address.

Response

Response parameters

Table 5-17 Response parameters

Parameter	Type	Description
authorization	Object	Authorization structure. For details, see Table 5-18.

Table 5-18 authorization parameters

Parameter	Type	Description
name	String	Authorization name.
repo_type	String	Repository type. Value: github, gitlab, or gitee.
repo_host	String	Repository address.
repo_home	String	Repository homepage.
repo_user	String	Repository username.
avartar	String	Avatar.
token_type	String	Authorization mode.
create_time	Long	Creation time.
update_time	Long	Update time.
status	Integer	Status.

Example

Example request

```
{
  "name": "string",
  "token": "string",
  "host": "string"
}
```

Example response

```
{
  "authorization": {
    "name": "string",
    "repo_type": "string",
    "repo_host": "string",
    "repo_home": "string",
    "repo_user": "string",
    "avartar": "string",
    "token_type": "string",
    "create_time": 0,
    "update_time": 0,
    "status": 0
  }
}
```

Status Code

Table 5-19 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.5 Creating Password Authorization

Function

This API is used to create password authorization for a Git repository.

URI

POST /v1/{project_id}/git/auths/{repo_type}/password

Table 5-20 describes the parameters.

Table 5-20 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
repo_type	String	Yes	Repository type. Value: github, devcloud, or bitbucket.

Request

Request parameters

Table 5-21 Request parameters

Parameter	Type	Mandatory	Description
name	String	Yes	Authorization name.
user	String	Yes	Repository username.
password	String	Yes	Repository password.

Response

Response parameters

Table 5-22 Response parameters

Parameter	Type	Description
authorization	Object	Authorization structure. For details, see **Table 5-23**.

Table 5-23 authorization parameters

Parameter	Type	Description
name	String	Authorization name.
repo_type	String	Repository type. Value: github, devcloud, or bitbucket.
repo_host	String	Repository address.
repo_home	String	Repository homepage.
repo_user	String	Repository username.
avartar	String	Avatar.
token_type	String	Authorization mode.
create_time	Long	Creation time.
update_time	Long	Update time.
status	Integer	Status.

Example

Example request

```
{
  "name": "string",
  "user": "string",
  "password": "string"
}
```

Example response

```
{
  "authorization": {
    "name": "string",
    "repo_type": "string",
    "repo_host": "string",
    "repo_home": "string",
    "repo_user": "string",
    "avartar": "string",
    "token_type": "string",
    "create_time": 0,
    "update_time": 0,
    "status": 0
  }
}
```

Status Code

Table 5-24 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.6 Deleting Repository Authorization

Function

This API is used to delete repository authorization based on the name.

URI

DELETE /v1/{project_id}/git/auths/{name}

Table 5-25 describes the parameters.

Table 5-25 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.

Parameter	Type	Mandatory	Description
name	String	Yes	Authorization name.

Request

Request parameters

None

Response

Response parameters

None

Example

Example request

None

Example response

None

Status Code

Table 5-26 Status codes

Status Code	Description
204	OK
500	Internal Server Error

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.7 Obtaining a Repository Namespace

Function

This API is used to obtain a repository namespace.

URI

GET /v1/{project_id}/git/repos/namespaces

Table 5-27 describes the parameters.

Table 5-27 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.

Request

Request parameters

None

Response

Response parameters

Table 5-28 Response parameters

Parameter	Type	Description
namespaces	Array<Object>	Namespace structure. For details, see Table 5-29.

Table 5-29 namespaces parameters

Parameter	Type	Description
id	String	Namespace ID.
name	String	Namespace name.

Example

Example request

None

Example response

```
{
  "namespaces": [
    {
      "id": "string",
      "name": "string"
    }
  ]
}
```

Status Code

Table 5-30 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.8 Obtaining Repository Information Based on the Clone URL

Function

This API is used to obtain repository information based on the clone URL.

URI

GET /v1/{project_id}/git/repos/project-info

Table 5-31 describes the parameters.

Table 5-31 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.

Table 5-32 Query parameters

Parameter	Type	Mandatory	Description
clone_url	String	Yes	Clone URL of a repository.

Request

Request parameters

None

Response

Response parameters

Table 5-33 Response parameters

Parameter	Type	Description
namespace_id	String	Namespace ID.
namespace	String	Namespace.
project_id	String	Repository project ID.
project	String	Repository project.

Example

Example request

```
https://{Endpoint}/v1/7c6a62ddb2bf45a3920d8053d3cb8fbf/git/repos/project-info?clone_url=http://xxx.gitlab.com:8090/xxx/springcloud-xxbbon.git
```

Example response

```
{
  "project": "string"
  "namespace_id": "string",
  "namespace": "string",
  "project_id": "string",
  }
```

Status Code

Table 5-34 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.9 Obtaining All Projects in a Namespace

Function

This API is used to obtain all projects in a namespace.

URI

GET /v1/{project_id}/git/repos/{namespace}/projects

Table 5-35 describes the parameters.

Table 5-35 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
namespace	String	Yes	Namespace ID.

Request

Request parameters

None

Response

Response parameters

Table 5-36 Response parameters

Parameter	Type	Description
projects	Array<Object>	Project structure. For details, see Table 5-37.

Table 5-37 projects parameters

Parameter	Type	Description
id	String	Project ID.
name	String	Project name.
clone_url	String	Clone URL.

Example

Example request

None

Example response

```
{
  "projects": [
    {
```

```
    "id": "string",
    "name": "string",
    "clone_url": "string"
  }
 ]
}
```

Status Code

Table 5-38 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.10 Creating a Software Repository Project

Function

This API is used to create a software repository project.

URI

POST /v1/{project_id}/git/repos/{namespace}/projects

Table 5-39 describes the parameters.

Table 5-39 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
namespace	String	Yes	Namespace ID.

Request

Request parameters

Table 5-40 Request parameters

Parameter	Type	Mandatory	Description
name	String	Yes	Project name.

Response

Response parameters

Table 5-41 Response parameters

Parameter	Type	Description
id	String	Project ID.
name	String	Project name.
clone_url	String	Clone URL.

Example

Example request

None

Example response

```
{
  "id": "string",
  "name": "string",
  "clone_url": "string"
}
```

Status Code

Table 5-42 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.11 Obtaining a Project Branch

Function

This API is used to obtain a project branch.

URI

GET /v1/{project_id}/git/repos/{namespace}/{project}/branches

Table 5-43 describes the parameters.

Table 5-43 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
namespace	String	Yes	Namespace ID.
project	String	Yes	Repository project ID. If the value contains a slash (/), replace it with a colon (:).

Request

Request parameters

None

Response

Response parameters

Table 5-44 Response parameters

Parameter	Type	Description
branches	Array<String>	Project branch.

Example

Example request

None

Example response

```
{
  "branches": [
```

```
    "string"
  ]
}
```

Status Code

Table 5-45 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.12 Obtaining a Project Tag

Function

This API is used to obtain a project tag.

URI

GET /v1/{project_id}/git/repos/{namespace}/{project}/tags

Table 5-46 describes the parameters.

Table 5-46 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
namespace	String	Yes	Namespace ID or URL code name.
project	String	Yes	Repository project ID or URL code name. If the value contains a slash (/), replace it with a colon (:).

Request

Request parameters

None

Response

Response parameters

Table 5-47 Response parameters

Parameter	Type	Description
tags	Array<String>	Project tag.

Example

Example request

None

Example response

```
{
  "tags": [
    "string"
  ]
}
```

Status Code

Table 5-48 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.13 Creating a Project Tag

Function

This API is used to create a project tag.

URI

POST /v1/{project_id}/git/repos/{namespace}/{project}/tags

Table 5-49 describes the parameters.

Table 5-49 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
namespace	String	Yes	Namespace ID or URL code name.
project	String	Yes	Repository project ID or URL code name. If the value contains a slash (/), replace it with a colon (:).

Table 5-50 Query parameters

Parameter	Type	Mandatory	Description
ref	String	Yes	Branch name, tag name, or commit SHA value.

Request

Request parameters

Table 5-51 Request parameters

Parameter	Type	Mandatory	Description
name	String	Yes	Tag name.
description	String	Yes	Description.

Response

Response parameters

Table 5-52 Response parameters

Parameter	Type	Description
name	String	Project tag name.

Example

Example request

```
{
  "name": "string",
  "description": "string"
}
```

Example response

```
{
  "name": "string"
}
```

Status Code

Table 5-53 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.14 Deleting a Project Tag

Function

This API is used to delete a project tag.

URI

DELETE /v1/{project_id}/git/repos/{namespace}/{project}/tags/{tag_name}

Table 5-54 describes the parameters.

Table 5-54 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
namespace	String	Yes	Namespace ID or URL code name.

Parameter	Type	Mandatory	Description
project	String	Yes	Repository project ID or URL code name. If the value contains a slash (/), replace it with a colon (:).
tag_name	String	Yes	Tag name.

Request

Request parameters

None

Response

Response parameters

None

Example

Example request

None

Example response

None

Status Code

Table 5-55 Status codes

Status Code	Description
204	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.15 Obtaining Project Commits

Function

This API is used to obtain the latest ten project commits.

URI

GET /v1/{project_id}/git/repos/{namespace}/{project}/commits

Table 5-56 describes the parameters.

Table 5-56 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
namespace	String	Yes	Namespace ID or URL code name.
project	String	Yes	Repository project ID or URL code name. If the value contains a slash (/), replace it with a colon (:).

Table 5-57 Query parameters

Parameter	Type	Mandatory	Description
ref	String	No	Branch or tag name. If it is not provided, use the default branch.

Request

Request parameters

None

Response

Response parameters

Table 5-58 Response parameters

Parameter	Type	Description
commits	Array<Object>	Project commits structure. For details, see **Table 5-59**.

Table 5-59 commits parameters

Parameter	Type	Description
sha	String	Commit SHA value.
message	String	Commit description.
authored_date	String	Import time.

Example

Example request

None

Example response

```
{
  "commits": [
    {
      "sha": "string",
      "message": "string",
      "authored_date": "string"
    }
  ]
```

Status Code

Table 5-60 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.16 Obtaining a Project Hook

Function

This API is used to obtain a project hook.

URI

GET /v1/{project_id}/git/repos/{namespace}/{project}/hooks

Table 5-61 describes the parameters.

Table 5-61 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
namespace	String	Yes	Namespace ID or URL code name.
project	String	Yes	Repository project ID or URL code name. If the value contains a slash (/), replace it with a colon (:).

Request

Request parameters

None

Response

Response parameters

Table 5-62 Response parameters

Parameter	Type	Description
hooks	Array of objects	Project hooks. For details, see Table 5-63.

Table 5-63 hooks parameters

Parameter	Type	Description
id	String	Hook ID.

Parameter	Type	Description
type	String	Hook type.
callback_url	String	Callback URL.

Example

Example request

None

Example response

```
{
  "hooks": [
    {
      "id": "string",
      "type": "string",
      "callback_url": "string"
    }
  ]
}
```

Status Code

Table 5-64 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.17 Creating a Project Hook

Function

This API is used to create a project hook.

URI

POST /v1/{project_id}/git/repos/{namespace}/{project}/hooks

Table 5-65 describes the parameters.

Table 5-65 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
namespace	String	Yes	Namespace ID or URL code name.
project	String	Yes	Repository project ID or URL code name. If the value contains a slash (/), replace it with a colon (:).

Request

Request parameters

Table 5-66 Request parameters

Parameter	Type	Mandatory	Description
secret	String	Yes	Character string used to verify received payloads.
url	String	Yes	URL that is called back when a hook is triggered.

Response

Response parameters

Table 5-67 Response parameters

Parameter	Type	Description
id	String	Hook ID.
type	String	Hook type.
callback_url	String	Callback URL.

Example

Example request

```
{
  "secret": "string",
```

```
  "url": "string"
}
```

Example response

```
{
  "id": "string",
  "type": "string",
  "callback_url": "string"
}
```

Status Code

Table 5-68 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.18 Deleting a Project Hook

Function

This API is used to delete a project hook.

URI

DELETE /v1/{project_id}/git/repos/{namespace}/{project}/hooks/{hook_id}

Table 5-69 describes the parameters.

Table 5-69 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
namespace	String	Yes	Namespace ID or URL code name.
project	String	Yes	Repository project ID or URL code name. If the value contains a slash (/), replace it with a colon (:).
hook_id	String	Yes	Hook ID.

Request

Request parameters

None

Response

Response parameters

None

Example

Example request

None

Example response

None

Status Code

Table 5-70 Status codes

Status Code	Description
204	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.19 Obtaining a Repository File Directory

Function

This API is used to obtain a repository file directory.

URI

GET /v1/{project_id}/git/files/{namespace}/{project}/trees

Table 5-71 describes the parameters.

Table 5-71 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
namespace	String	Yes	Namespace ID or URL code name.
project	String	Yes	Repository project ID or URL code name. If the value contains a slash (/), replace it with a colon (:).

Table 5-72 Query parameters

Parameter	Type	Mandatory	Description
ref	String	Yes	Branch name, tag name, or commit SHA value.

Request

Request parameters

None

Response

Response parameters

Table 5-73 Response parameters

Parameter	Type	Description
paths	Array<String>	File path.

Example

Example request

None

Example response

```
{
  "paths": [
    "string"
  ]
}
```

Status Code

Table 5-74 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.20 Obtaining Repository File Contents

Function

This API is used to obtain repository file contents.

URI

GET /v1/{project_id}/git/files/{namespace}/{project}/{path}

Table 5-75 describes the parameters.

Table 5-75 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
namespace	String	Yes	Namespace ID or URL code name.
project	String	Yes	Repository project ID or URL code name. If the value contains a slash (/), replace it with a colon (:).
path	String	Yes	File path. If the value contains a slash (/), replace it with a colon (:).

Table 5-76 Query parameters

Parameter	Type	Mandatory	Description
ref	String	Yes	Branch name, tag name, or commit SHA value.

Request

Request parameters

None

Response

Response parameters

Table 5-77 Response parameters

Parameter	Type	Description
path	String	File path.
sha	String	Commit SHA value.
encoding	String	Encoding mode. Value: base64 or text/plain.
content	String	File contents.

Example

Example request

None

Example response

```
{
  "path": "string",
  "sha": "string",
  "encoding": "base64",
  "content": "string"
}
```

Status Code

Table 5-78 Status codes

Status Code	Description
200	OK

Status Code	Description
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.21 Creating a Repository File

Function

This API is used to create a repository file.

URI

POST /v1/{project_id}/git/files/{namespace}/{project}/{path}

Table 5-79 describes the parameters.

Table 5-79 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
namespace	String	Yes	Namespace ID or URL code name.
project	String	Yes	Repository project ID or URL code name. If the value contains a slash (/), replace it with a colon (:).
path	String	Yes	File path. If the value contains a slash (/), replace it with a colon (:).

Table 5-80 Query parameters

Parameter	Type	Mandatory	Description
ref	String	Yes	Branch name, tag name, or commit SHA value.

Request

Request parameters

Table 5-81 Request parameters

Parameter	Type	Mandatory	Description
message	String	Yes	Description.
content	String	Yes	File contents.

Response

Response parameters

Table 5-82 Response parameters

Parameter	Type	Description
path	String	File path.

Example

Example request

```
{
  "message": "string",
  "content": "string"
}
```

Example response

```
{
  "path": "string"
}
```

Status Code

Table 5-83 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.22 Modifying Repository File Contents

Function

This API is used to modify repository file contents.

URI

PUT /v1/{project_id}/git/files/{namespace}/{project}/{path}

Table 5-84 describes the parameters.

Table 5-84 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
namespace	String	Yes	Namespace ID or URL code name.
project	String	Yes	Repository project ID or URL code name. If the value contains a slash (/), replace it with a colon (:).
path	String	Yes	File path. If the value contains a slash (/), replace it with a colon (:).

Table 5-85 Query parameters

Parameter	Type	Mandatory	Description
ref	String	Yes	Branch name, tag name, or commit SHA value.

Request

Request parameters

Table 5-86 Request parameters

Parameter	Type	Mandatory	Description
message	String	Yes	Description.
content	String	Yes	File contents.
sha	String	Yes	Commit SHA value.

Response

Response parameters

Table 5-87 Response parameters

Parameter	Type	Description
path	String	File path.

Example

Example request

```
{
  "message": "string",
  "content": "string",
  "sha": "string"
}
```

Example response

```
{
  "path": "string"
}
```

Status Code

Table 5-88 Status codes

Status Code	Description
200	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

5.23 Deleting a Repository File

Function

This API is used to delete a repository file.

URI

DELETE /v1/{project_id}/git/files/{namespace}/{project}/{path}

Table 5-89 describes the parameters.

Table 5-89 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
namespace	String	Yes	Namespace ID or URL code name.
project	String	Yes	Repository project ID or URL code name. If the value contains a slash (/), replace it with a colon (:).
path	String	Yes	File path. If the value contains a slash (/), replace it with a colon (:).

Table 5-90 Query parameters

Parameter	Type	Mandatory	Description
ref	String	Yes	Branch name, tag name, or commit SHA value.
message	String	Yes	Commit information.
sha	String	Yes	Latest commit SHA value.

Request

Request parameters

None

Response

Response parameters

None

Example

Example request

None

Example response

None

Status Code

Table 5-91 Status codes

Status Code	Description
204	OK
400	Bad Request

Error Code

Error code must be in the format of SVCSTG.REPO.[Error_ID]. Example: SVCSTG.REPO.0401. See **ServiceStage Error Codes**.

6 CSE API

6.1 API Calling

ServiceStage provides Representational State Transfer (REST) APIs, allowing you to call APIs using HTTPS. For details about how to access APIs of the exclusive CSE, see **Obtaining the Connection Address of Service Center**.

6.2 Querying Static Information About a Microservice

Function

This API is used to query static information about a microservice based on the service ID.

URI

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

Table 6-1 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Fixed value: default.
serviceId	String	Yes	Microservice ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^.*$

Request

None

Response

Response parameters

Table 6-2 Parameter

Parameter	Type	Mandatory	Description
service	Object	Yes	Microservice struct. For details, see **MicroService**.

Example

Example request

```
GET /v4/default/registry/microservices/819706e21b7173306797d19922ce4231441c17c5
```

Example response

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

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.3 Querying Static Information About All Microservices

Function

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

URI

GET /v4/{project_id}/registry/microservices

Table 6-3 Path parameters

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

Request

None

Response

Response parameters

Table 6-4 describes the parameters.

Table 6-4 Parameter

Parameter	Type	Description
services	Array of objects	Microservice list. For details, see **MicroService**.

Example

Example request

```
GET /v4/default/registry/microservices
```

Example response

```
{
  "services": [
    {
      "serviceId": "string",
```

```
    "environment": "string",
    "appId": "string",
    "serviceName": "string",
    "version": "string",
    "description": "string",
    "level": "string",
    "registerBy": "string",
    "schemas": [
      "string"
    ],
    "status": "UP",
    "timestamp": "string",
    "modTimestamp": "string",
    "framework": {
      "name": "string",
      "version": "string"
    },
    "paths": [
      {
        "Path": "string",
        "Property": {
          "additionalProp1": "string",
          "additionalProp2": "string",
          "additionalProp3": "string"
        }
      }
    ]
  }
 ]
}
```


Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.4 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 the **serviceId** field. One service corresponds to multiple instances.

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

URI

POST /v4/{project_id}/registry/microservices

Table 6-5 Path parameters

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

Request

Table 6-6 Request body parameters

Parameter	Type	Mandatory	Description
service	Object	Yes	Microservice struct. For details, see **MicroService**.
rules	Array of objects	No	Blacklist and whitelist structs. For details, see **Rule**.
instances	Array of objects	No	Microservice instance struct. For details, see **MicroServiceInstance**.
tags	Map<String, String>	No	Microservice tag. You can customize a key and value.

Response

Response parameters

Table 6-7 describes the parameters.

Table 6-7 Parameter

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

Example

Example request

```
{
  "service": {
```

```
  "serviceId": "string",
  "environment": "string",
  "appId": "string",
  "serviceName": "string",
  "version": "string",
  "description": "string",
  "level": "string",
  "registerBy": "string",
  "schemas": [
    "string"
  ],
  "status": "UP",
  "timestamp": "string",
  "modTimestamp": "string",
  "framework": {
    "name": "string",
    "version": "string"
  },
  "paths": [
    {
      "Path": "string",
      "Property": {
        "additionalProp1": "string",
        "additionalProp2": "string",
        "additionalProp3": "string"
      }
    }
  ]
},
"rules": [
  {
    "ruleId": "string",
    "ruleType": "string",
    "attribute": "string",
    "pattern": "string",
    "description": "string",
    "timestamp": "string",
    "modTimestamp": "string"
  }
],
"instances": [
  {
    "instanceId": "string",
    "serviceId": "string",
    "version": "string",
    "hostName": "string",
    "endpoints": [
      "string"
    ],
    "status": "string",
    "healthCheck": {
      "mode": "string",
      "port": 0,
      "interval": 0,
      "times": 0
    },
    "dataCenterInfo": {
      "name": "string",
      "region": "string",
      "availableZone": "string"
    },
    "timestamp": "string",
    "modTimestamp": "string"
  }
],
"tags": {
  "additionalProp1": "string",
  "additionalProp2": "string",
  "additionalProp3": "string"
```

```
  }
}
```

Example response

```
{
  "serviceId": "string"
}
```

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.5 Modifying Static Information About a Microservice

Function

After you create static information about a microservice, you can update some fields of the microservice. However, you must transfer the struct of extended attributes for each update.

URI

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

Table 6-8 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Fixed value: default.
serviceId	String	Yes	Microservice ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^.*$

Request

Request parameters

Table 6-9 Request body parameters

Parameter	Type	Mandatory	Description
properties	Map<String, String>	No	Extended attribute. You can customize a key and value.

Response

Response parameters

None

Example

Example request

```
{
   "properties": {
      "attr1": "value1"
   }
}
```

Example response

None

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.6 Querying a Microservice Schema

Function

This API is used to query a microservice schema based on the service ID and schema ID.

URI

GET /v4/{project_id}/registry/microservices/{serviceId}/schemas/{schemaId}

Table 6-10 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Fixed value: default.
serviceId	String	Yes	Microservice ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^.*$
schemaId	String	Yes	Schema ID.

Request

None

Response

Response parameters

Table 6-11 describes the parameters.

Table 6-11 Parameter

Parameter	Type	Description
schema	String	Microservice schema content.

Example

Example request

```
GET /v4/default/registry/microservices/e0f0da073f2c91e8979a89ff2d7c69t6/schemas/com.test.TestService
```

Example response

```
{"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

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.7 Modifying a Microservice Schema

Function

This API is used to modify a microservice schema based on the schema ID. By default, the schema of a professional microservice engine cannot be modified, but that of an exclusive microservice engine can.

URI

PUT /v4/{project_id}/registry/microservices/{serviceId}/schemas/{schemaId}

Table 6-12 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Fixed value: default.
serviceId	String	Yes	Microservice ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^.*$
schemaId	String	Yes	Microservice schema ID, which must be unique. The value must be 1 to 160 characters long. Regular expression: ^[a-zA-Z0-9]{1,160}$\|^[a-zA-Z0-9][a-zA-Z0-9_\-.]{0,158}[a-zA-Z0-9]$

Request

Table 6-13 Request body parameters

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

Response

Response parameters

None

Example

Example request

```
{
  "schema": "string",
  "summary": "string"
}
```

Example response

None

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.8 Creating a Dependency Between Services

Function

This API is used to create a dependency between services. For **consumer**, **version** must be a confirmed version and **serviceName** cannot be *. **consumer** must be an existing service while **provider** can be a service that has not been created.

URI

PUT /v4/{project_id}/registry/dependencies

Table 6-14 Path parameters

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

Request

Table 6-15 Request body parameters

Parameter	Type	Mandatory	Description
dependencies	Array of objects	Yes	Set of rules. See **Table 6-16**. The array can contain 1–100 objects.

Table 6-16 MicroServiceDependency parameters

Parameter	Type	Mandatory	Description
consumer	Object	Yes	Consumer microservice. For details, see **DependencyMicroService**.
providers	Array of objects	No	Provider microservice. For details, see **DependencyMicroService**.

Response

Response parameters

None

Example

Example request

```
{
  "dependencies": [
    {
      "consumer": {
        "environment": "string",
        "appId": "string",
        "serviceName": "string",
        "version": "string"
      },
      "providers": [
        {
          "environment": "string",
          "appId": "string",
          "serviceName": "string",
          "version": "string"
        }
      ]
    }
  ]
}
```

Example response

None

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.9 Querying All Providers of a Microservice

Function

This API is used to query all providers of a microservice based on the consumer ID.

URI

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

Table 6-17 Path parameters

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

Request

None

Response

Response parameters

Table 6-18 describes the parameters.

Table 6-18 Parameter

Parameter	Type	Description
providers	Array of objects	Microservice struct. For details, see MicroService.

Example

Example request

```
GET /v4/default/registry/microservices/e5788b23b640232730c7a99a6d806b4f31be453e/providers
```

Example response

```
{
    "providers": [{
        "serviceId": "string",
        "appId": "string",
        "serviceName": "string",
        "version": "string",
```

```
      "description": "string",
      "level": "string",
      "status": "up",
      "timestamp": "string",
      "framework": {
         "name": "string",
         "version": "string"
      }
   }]
}
```

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.10 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 6-19 Path parameters

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

Table 6-20 Query parameters

Parameter	Type	Mandatory	Description
type	String	Yes	Request type. Options: • microservice • schema

Parameter	Type	Mandatory	Description
env	String	No	Microservice environment. Value: development, testing, acceptance, or production. Default value: development.
appId	String	Yes	Application ID, which needs to be transferred when the resource type is **microservice**. The value must be 1 to 160 characters long. Regular expression: ^[a-zA-Z0-9]*$\|^[a-zA-Z0-9][a-zA-Z0-9_\-.]*[a-zA-Z0-9]$
serviceName	String	Yes	Microservice name, which needs to be transferred when the resource type is **microservice**. The value must be 1 to 128 characters long. Regular expression: ^[a-zA-Z0-9]*$\|^[a-zA-Z0-9][a-zA-Z0-9_\-.]*[a-zA-Z0-9]$
version	String	Yes	Microservice version, which needs to be transferred when the resource type is **microservice**. The value must be 1 to 64 characters long. Regular expression: ^[0-9]*$\|^[0-9]+(\.[0-9]+)*$
serviceId	String	Yes	Microservice ID, which needs to be transferred when the resource type is **schema**. The value must be 1 to 64 characters long. Regular expression: ^.*$
schemaId	String	Yes	Schema ID, which needs to be transferred when the resource type is **schema**. The value must be 1 to 160 characters long. Regular expression: ^[a-zA-Z0-9]{1,160}$\|^[a-zA-Z0-9][a-zA-Z0-9_\-.]{0,158}[a-zA-Z0-9]$

Request

None

Response

Response parameters

Table 6-21 describes the parameters.

Table 6-21 Parameters

Parameter	Type	Description
serviceId	String	Unique microservice ID, which is returned when a microservice is queried.
schemaId	String	Unique schema ID, which is returned when a schema is queried.

Example

Example request

```
GET /v4/default/registry/existence
```

Example response

```
{
  "serviceId": "string"
}
```

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.11 Registering a Microservice Instance

Function

This API is used to register a microservice instance after static microservice information is created. Information about the instance must be provided during registration.

The instance ID can be customized. If a customized instance ID is available, use it for registration. If it is not available, the system automatically generates an instance ID.

URI

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

Table 6-22 Path parameters

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

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

Request

Request parameters

Table 6-23 Parameter

Parameter	Type	Mandatory	Description
instance	Object	Yes	Microservice instance struct. For details, see **MicroServiceInstance**.

Response

Response parameters

Table 6-24 describes the parameters.

Table 6-24 Parameter

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

Example

Example request

```
{
  "instance": {
    "instanceId": "string",
    "serviceId": "string",
    "version": "string",
    "hostName": "string",
    "endpoints": [
      "string"
    ],
    "status": "string",
    "healthCheck": {
      "mode": "string",
      "port": 0,
      "interval": 0,
      "times": 0
```

```
    },
    "dataCenterInfo": {
      "name": "string",
      "region": "string",
      "availableZone": "string"
    },
    "timestamp": "string",
    "modTimestamp": "string"
  }
}
```

Example response

```
{
  "instanceId": "string"
}
```

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.12 Querying All Instances of a Microservice Based on the Service ID

Function

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

URI

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

Table 6-25 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Fixed value: default.
serviceId	String	Yes	Microservice ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^.*$

Request

Table 6-26 Request header parameter

Parameter	Type	Mandatory	Description
X-ConsumerId	String	No	Unique ID of a service consumer.

Response

Response parameters

Table 6-27 Parameter

Parameter	Type	Description
instances	Array of objects	Set of microservice instances. For details, see **MicroServiceInstance**.

Example

Example request

```
GET /v4/default/registry/microservices/e0f0da073f2c91e8979a89ff2d7c69t6/instances
```

Example response

```
{
  "instances": [
    {
      "instanceId": "string",
      "serviceId": "string",
      "version": "string",
      "hostName": "string",
      "endpoints": [
        "string"
      ],
      "status": "string",
      "healthCheck": {
        "mode": "string",
        "port": 0,
        "interval": 0,
        "times": 0
      },
      "dataCenterInfo": {
        "name": "string",
        "region": "string",
        "availableZone": "string"
      },
      "timestamp": "string",
      "modTimestamp": "string"
```

```
  }
 ]
}
```

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.13 Deregistering a Microservice Instance

Function

This API is used to deregister an instance based on the instance ID.

URI

DELETE /v4/{project_id}/registry/microservices/{serviceId}/instances/{instanceId}

Table 6-28 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Fixed value: default.
serviceId	String	Yes	Microservice ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^.*$
instanceId	String	Yes	Microservice instance ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^[A-Za-z0-9_.-]*$

Request

None

Response

Response parameters

None

Example

Example request

```
DELETE /v4/default/registry/microservices/a0c572b9ede946712698081a87552a7a87169bcb/instances/
17dd265a8b6f11ebb23100ff14de2ec3
```

Example response

None

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.14 Querying Details About a Microservice Instance

Function

This API is used to query details about an instance based on the service ID and instance ID after the instance is registered.

URI

GET /v4/{project_id}/registry/microservices/{serviceId}/instances/{instanceId}

Table 6-29 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Fixed value: default.
serviceId	String	Yes	Microservice ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^.*$
instanceId	String	Yes	Microservice instance ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^[A-Za-z0-9_.-]*$

Request

Request parameters

None

Response

Response parameters

Table 6-30 describes the parameters.

Table 6-30 Parameter

Parameter	Type	Description
instance	Object	Microservice instance. For details, see **MicroServiceInstance**.

Example

Example request

```
GET /v4/default/registry/microservices/a0c572b9ede946712698081a87552a7a87169bcb/instances/
2370d7938b7111ebb23100ff14de2ec3
```

Example response

```
{
  "instance": {
    "instanceId": "string",
    "serviceId": "string",
    "version": "string",
    "hostName": "string",
    "endpoints": [
      "string"
    ],
    "status": "string",
    "healthCheck": {
      "mode": "string",
      "port": 0,
      "interval": 0,
      "times": 0
    },
    "dataCenterInfo": {
      "name": "string",
      "region": "string",
      "availableZone": "string"
    },
    "timestamp": "string",
    "modTimestamp": "string"
  }
}
```

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.15 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 the instance ID after the instance is registered.

URI

PUT /v4/{project_id}/registry/microservices/{serviceId}/instances/{instanceId}/properties

Table 6-31 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Fixed value: default.
serviceId	String	Yes	Microservice ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^.*$
instanceId	String	Yes	Microservice instance ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^[A-Za-z0-9_.-]*$

Request

Request parameters

Table 6-32 Request body parameters

Parameter	Type	Mandatory	Description
properties	Map<String, String>	No	Extended attribute of the microservice instance. You can customize a key and value.

Response

Response parameters

None

Example

Example request

```
{
   "properties": {
      "attr1": "value1"
   }
}
```

Example response

None

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.16 Changing the Status of a Microservice Instance

Function

This API is used to change the status of a microservice instance based on the instance ID after the instance is registered.

URI

PUT /v4/{project_id}/registry/microservices/{serviceId}/instances/{instanceId}/status

Table 6-33 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Fixed value: default.
serviceId	String	Yes	Microservice ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^.*$
instanceId	String	Yes	Microservice instance ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^[A-Za-z0-9_.-]*$

Table 6-34 Query parameters

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

Request

Request parameters

None

Response

Response parameters

None

Example

Example request

```
PUT /v4/default/registry/microservices/a0c572b9ede946712698081a87552a7a87169bcb/instances/831113358b7811ebb23100ff14de2ec3/status?value=UP
```

Example response

None

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.17 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.

URI

PUT /v4/{project_id}/registry/microservices/{serviceId}/instances/{instanceId}/heartbeat

Table 6-35 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Fixed value: default.
serviceId	String	Yes	Microservice ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^.*$
instanceId	String	Yes	Microservice instance ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^[A-Za-z0-9_.-]*$

Request

None

Response

Response parameters

None

Example

Example request

```
PUT /v4/default/registry/microservices/a0c572b9ede946712698081a87552a7a87169bcb/instances/2370d7938b7111ebb23100ff14de2ec3/heartbeat
```

Example response

None

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.18 Querying a Microservice Instance by Filter Criteria

Function

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

URI

GET /v4/{project_id}/registry/instances

Table 6-36 Path parameters

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

Table 6-37 Query parameters

Parameter	Type	Mandatory	Description
appId	String	Yes	Application ID, which must be unique. The value must be 1 to 160 characters long. Regular expression: ^[a-zA-Z0-9]*$\|^[a-zA-Z0-9][a-zA-Z0-9_\-.]*[a-zA-Z0-9]$
serviceName	String	Yes	Microservice name. The value must be 1 to 128 characters long. Regular expression: ^[a-zA-Z0-9]*$\|^[a-zA-Z0-9][a-zA-Z0-9_\-.]*[a-zA-Z0-9]$
version	String	Yes	Version rule. Options: • 1: exact version matching. Example: 0.0.1. • 2: subsequent version matching. Example: 0.0.1+. • 3: latest version. Example: latest. • 4: version range. Example: 0.1.0–0.2.0.
env	String	No	Value: development, testing, acceptance, or production.

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

Request

Table 6-38 Request header parameter

Parameter	Type	Mandatory	Description
X-ConsumerId	String	No	Unique ID of a service consumer.

Response

Response parameters

Table 6-39 describes the parameters.

Table 6-39 Parameter

Parameter	Type	Description
instances	Array of objects	Set of microservice instances. Each item in the array is an instance. For details, see **MicroServiceInstance**.

Example

Example request

```
GET /v4/default/registry/instances
```

Example response

```
{
  "instances": [
    {
      "instanceId": "string",
      "serviceId": "string",
      "version": "string",
```

```
    "hostName": "string",
    "endpoints": [
      "string"
    ],
    "status": "string",
    "healthCheck": {
      "mode": "string",
      "port": 0,
      "interval": 0,
      "times": 0
    },
    "dataCenterInfo": {
      "name": "string",
      "region": "string",
      "availableZone": "string"
    },
    "timestamp": "string",
    "modTimestamp": "string"
  }
 ]
}
```


Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.19 Querying Configurations

Function

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

URI

GET /v3/{project_id}/configuration/items

Table 6-40 Path parameters

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

Table 6-41 Query parameters

Parameter	Type	Mandatory	Description
dimensionsInfo	String	Yes	Service information, which consists of the service name (mandatory), application (mandatory), and version (optional). The combination format is as follows (note that **@** and **#** must be escaped when being added to the HTTP URL. **@** is escaped to **%40**, and **#** is escaped to **%23**): {serviceName}@{appId}#{version} or {serviceName}@{appId} The value cannot contain spaces or the following special characters: $%^&+/\
keyFilter	String	No	**key** filter criteria. A regular expression is supported. If special characters are contained, escape them.
revision	String	No	Version of a configuration item. 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.

Request

Request parameters

Table 6-42 Request header parameter

Parameter	Type	Mandatory	Description
x-environment	String	No	Environment whose configurations need to be queried. Value: development, testing, acceptance, or production. Other values are regarded as null.

Response

Response parameters

Table 6-43 describes the parameters. (If the input revision value is the same as the current revision value, no response message is returned.)

Table 6-43 Parameters

Parameter	Type	Description
{dimension} (The parameter name and number of parameters are not fixed.)	Map<String, String>	Configuration content.
revision	Table 6-44	revision

Table 6-44 revision

Parameter	Type	Description
version	String	Version.

Example

Example request

```
GET /v3/default/configuration/items?dimensionsInfo=service%40app%231.0.0
```

Example response

```
{
   "service@app": {
      "k1": "v1"
   },
   "service@app#1.0.0": {
      "k2": "v2"
   }
}
```

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.20 Deleting Static Information About a Microservice

Function

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

URI

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

Table 6-45 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Fixed value: default.
serviceId	String	Yes	Microservice ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^.*$

Table 6-46 Query parameters

Parameter	Type	Mandatory	Description
force	Boolean	No	Whether to forcibly delete information about a microservice. Options: • true: Forcibly delete such information. • false: Do not forcibly delete such information. If the value is **true**, all service instances are automatically deregistered and related service dependencies are deleted. If the value is not specified and the microservice has instances, the microservice cannot be deleted.

Request

None

Response

Response parameters

None

Example

Example request

```
DELETE /v4/default/registry/microservices/e0f0da073f2c91e8979a89ff2d7c69t6
```

Example response

None

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.21 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.

URI

DELETE /v4/{project_id}/registry/microservices

Table 6-47 Path parameters

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

Request

Request parameters

Table 6-48 Request body parameters

Parameter	Type	Mandatory	Description
serviceIds	Array of strings	Yes	Microservice ID list.
force	Boolean	No	Whether to forcibly delete static information about microservices. Options: • false: Do not forcibly delete such information. • true: Forcibly delete such information. Default value: false. If the value is **true**, all service instances are automatically deregistered and related service dependencies are deleted. If the value is not specified and a microservice is the dependent service of other microservices or has instances, the microservice cannot be deleted.

Response

Response parameters

Table 6-49 describes the parameters.

Table 6-49 Parameter

Parameter	Type	Description
services	Array of objects	See **DelServicesResponse**.

Example

Example request

```
{
  "serviceIds": [
    "string"
  ],
  "force": boolean
}
```

Example response

```
{
  "services": [
```

```
  {
    "serviceId": "string"
  }
 ]
}
```

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.22 Querying Microservice Instances in Batches

Function

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

URI

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

Table 6-50 Path parameters

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

Table 6-51 Query parameters

Parameter	Type	Mandatory	Description
type	String	Yes	Operation type. Fixed value: query.

Request

Table 6-52 Request header parameter

Parameter	Type	Mandatory	Description
X-ConsumerId	String	No	Unique ID of a service consumer.

Table 6-53 Parameters

Parameter	Type	Mandatory	Description
services	Array of objects	No	Microservice request struct. For details, see **Table 6-54**.
instances	Array of objects	No	Microservice instance request struct. For details, see **Table 6-55**.

Table 6-54 FindService parameters

Parameter	Type	Mandatory	Description
service	Object	Yes	Microservice request struct. For details, see **DependencyMicroService**.
rev	String	No	Version number cached by the client.

Table 6-55 FindInstance parameters

Parameter	Type	Mandatory	Description
instance	Object	Yes	Microservice instance request struct. For details, see **HeartbeatSetElement**.
rev	String	No	Version number cached by the client.

Response

Response parameters

Table 6-56 describes the parameters.

Table 6-56 Parameters

Parameter	Type	Description
services	BatchFindResult	Response structs of microservices that are discovered.
instances	BatchFindResult	Response structs of microservice instances that are discovered.

Table 6-57 BatchFindResult parameters

Parameter	Type	Description
failed	Array of objects	Response structs of microservices that cannot be discovered. For details, see **Table 6-58**.
notModified	Array of integers	Index array of instances that remain unchanged. The value of each item in the array corresponds to an index of a microservice in the request struct.
updated	Array of objects	Response structs of instances that are modified. For details, see **Table 6-59**.

Table 6-58 FindFailedResult parameters

Parameter	Type	Description
indexes	Array of integers	Index array of microservices that cannot be discovered. The value of each item in the array corresponds to an index of a microservice in the request struct.
error	Object	Error information struct. For details, see **Error**.

Table 6-59 FindResult parameters

Parameter	Type	Description
index	Integer	Index array of instances that remain unchanged. The value of each item in the array corresponds to an index of a microservice in the request struct.
rev	String	Version number cached by the client.
instances	Array of objects	Set of microservice instances. For details, see **MicroServiceInstance**.

Example

Example request

```
curl -X POST "/v4/default/registry/instances/action?type=query" -H "accept: application/json" -H "Content-Type: application/json" -d '
{
  "services": [
    {
      "service": {
        "environment": "string",
        "appId": "string",
        "serviceName": "string",
        "version": "string"
      },
      "rev": "string"
    }
  ],
  "instances": [
    {
      "instance": {
        "serviceId": "string",
        "instanceId": "string"
      },
      "rev": "string"
    }
  ]
}
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Server: SERVICECENTER/0.0.1
Vary: Accept-Encoding
Vary: Origin
X-Response-Status: 200
Date: Fri, 21 Dec 2018 02:36:48 GMT
Content-Length: 1952

{
  "services": {
    "failed": [
      {
        "indexes": [
          0
        ],
        "error": {
          "errorCode": "string",
          "errorMessage": "string",
          "detail": "string"
        }
      }
    ],
    "notModified": [
      0
    ],
    "updated": [
      {
        "index": 0,
        "rev": "string",
        "instances": [
          {
            "instanceId": "string",
            "serviceId": "string",
            "version": "string",
            "hostName": "string",
            "endpoints": [
              "string"
            ],
            "status": "string",
            "healthCheck": {
              "mode": "string",
              "port": 0,
              "interval": 0,
              "times": 0
            },
            "dataCenterInfo": {
              "name": "string",
              "region": "string",
              "availableZone": "string"
            },
            "timestamp": "string",
            "modTimestamp": "string"
          }
        ]
      }
    ]
  },
  "instances": {
    "failed": [
      {
        "indexes": [
          0
        ],
        "error": {
          "errorCode": "string",
          "errorMessage": "string",
          "detail": "string"
        }
```

```
    }
  ],
  "notModified": [
    0
  ],
  "updated": [
    {
      "index": 0,
      "rev": "string",
      "instances": [
        {
          "instanceId": "string",
          "serviceId": "string",
          "version": "string",
          "hostName": "string",
          "endpoints": [
            "string"
          ],
          "status": "string",
          "healthCheck": {
            "mode": "string",
            "port": 0,
            "interval": 0,
            "times": 0
          },
          "dataCenterInfo": {
            "name": "string",
            "region": "string",
            "availableZone": "string"
          },
          "timestamp": "string",
          "modTimestamp": "string"
        }
      ]
    }
  ]
 }
}
```


Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

6.23 Querying All Schema Information About a Microservice

Function

This API is used to query all schema information (including **schemaId** and **summary**) about a microservice.

URI

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

Table 6-60 Path parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Fixed value: default.
serviceId	String	Yes	Microservice ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^.*$

Table 6-61 Query parameters

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

Request

Request parameters

None

Response

Response parameters

Table 6-62 describes the parameters.

Table 6-62 Parameter

Parameter	Type	Description
schemas	Array of objects	Struct list of all microservice schemas. For details, see **Schema**.

Example

Example request

None

Example response

```
{
   "schemas": [
      {
         "schemaId": "xxxx",
         "schema": "xxxx", // A value is returned only when the value of withSchema in the URL is 1. By default, no value is returned.
         "summary": "abcd"
      }
   ]
}
```

Status Code

See **Status Codes**.

Error Code

See **CSE Error Codes**.

7 Data Structure

7.1 Parameters

7.1.1 HealthCheck

Table 7-1 Parameters

Parameter	Type	Mandatory	Description
mode	String	Yes	Check mode. Value: push or pull.
port	Integer	No	Port. The value contains a maximum of 65,536 characters. Regular expression: ^[0-9]*$
interval	Integer	Yes	Heartbeat interval. Unit: s. If the value is less than 5s, the registration is performed at an interval of 5s. The value contains a maximum of 4,294,967,296 characters. Regular expression: ^[0-9]+$
times	Integer	Yes	Retry times. The value contains a maximum of 4,294,967,296 characters. Regular expression: ^[0-9]+$

7.1.2 MicroServiceInstance

Table 7-2 Parameters

Parameter	Type	Mandatory	Description
instanceId	String	No	Instance ID, which must be unique. The instance ID is generated by the service center.
serviceId	String	No	Microservice ID, which must be unique. The microservice ID in the URL will be used.
version	String	No	Microservice version.
hostName	String	Yes	PC name, which can be a domain name or IP address.
endpoints	Array of strings	Yes	Example: rest:127.0.0.1:8080.
status	String	Yes	Instance status. Options: • UP • DOWN • STARTING • OUTOFSERVICE
properties	Map<String, String>	No	Extended attribute. You can customize a key and value.
healthCheck	**HealthCheck**	No	Health check mode. Value: push or pull. **interval** indicates the pull interval (unit: s), and **times** indicates the maximum request attempts. When in the pull mode, you need to define **port** to ensure that the registration center actively connects to the port. The default value of **healthCheck** is **push**, that of **interval** is **30**, and that of **times** is **3**.
timestamp	String	No	Time when an instance is created, which is automatically generated.
modTimestamp	String	No	Update time.
dataCenterInfo	**DataCenterInfo**	No	Time zone.

7.1.3 HeartbeatSetElement

Table 7-3 Parameters

Parameter	Type	Mandatory	Description
serviceId	String	Yes	Microservice ID. The value must be 1 to 64 characters long. Regular expression: ^.*$
instanceId	String	Yes	Microservice instance ID. The value must be 1 to 64 characters long. Regular expression: ^[A-Za-z0-9_.-]*$

7.1.4 InstanceHbRst

Table 7-4 Parameters

Parameter	Type	Mandatory	Description
serviceId	String	No	Microservice ID.
instanceId	String	No	Microservice instance ID.
errMessage	String	No	Error information. If the operation is successful, the value is empty. If the operation fails, the error information is returned.

7.1.5 DelServicesResponse

Table 7-5 Parameter

Parameter	Type	Mandatory	Description
serviceId	String	No	Microservice ID.

7.1.6 MetricData

Table 7-6 Parameters

Parameter	Type	Mandatory	Description
name	String	Yes	Service name.
appId	String	Yes	Application ID.
serviceId	String	No	Service ID.
version	String	Yes	Service version.
instanceId	String	No	Instance ID.
instance	String	Yes	Instance name.
thread	String	No	Number of threads.
cpu	String	No	CPU usage.
memory	String	No	Memory.
interfaces	Table 7-7	No	API data details.

NOTE

Parameters **serviceId** and **instanceId** must be specified or left blank at the same time.

Table 7-7 items parameters

Parameter	Type	Mandatory	Description
items	FunctionMetricInfo	No	API-level aggregation metric data.

7.1.7 FunctionMetricInfo

Table 7-8 Parameters

Parameter	Type	Mandatory	Description
AggregateMetricInfo	Object	No	Aggregation metric data.
l995	Integer	No	99.5% requests have a value smaller than the value of this parameter.

Parameter	Type	Mandatory	Description
l99	Integer	No	99% requests have a value smaller than the value of this parameter.
l90	Integer	No	90% requests have a value smaller than the value of this parameter.
l75	Integer	No	75% requests have a value smaller than the value of this parameter.
l50	Integer	No	50% requests have a value smaller than the value of this parameter.
l25	Integer	No	25% requests have a value smaller than the value of this parameter.
l5	Integer	No	5% requests have a value smaller than the value of this parameter.

7.1.8 InstanceMetricInfo

Table 7-9 Parameters

Parameter	Type	Mandatory	Description
AggregateMetricInfo	Object	No	Aggregation metric data.
thread	Integer	No	Number of threads.
cpu	number	No	CPU usage.
memory	Object	No	Memory information.

7.2 Common Request Parameters

7.2.1 MicroService

Table 7-10 Parameters

Parameter	Type	Mandatory	Description
serviceId	String	No	Microservice ID, which must be unique. The value must be 1 to 64 characters long. Regular expression: ^.*$
environment	String	No	Service stage. Value: development, testing, acceptance, or production. Schemas can be added or modified only when this parameter is set to **development**, **testing**, or **acceptance**. Default value: development.
appId	String	Yes	Application ID, which must be unique. The value must be 1 to 160 characters long. Regular expression: ^[a-zA-Z0-9]*$\|^[a-zA-Z0-9][a-zA-Z0-9_\-.]*[a-zA-Z0-9]$
serviceName	String	Yes	Microservice name, which must be unique in an application. The value must be 1 to 128 characters long. Regular expression: ^[a-zA-Z0-9]*$\|^[a-zA-Z0-9][a-zA-Z0-9_\-.]*[a-zA-Z0-9]$
version	String	Yes	Microservice version. The value must be 1 to 64 characters long. Regular expression: ^[0-9]*$\|^[0-9]+(\.[0-9]+)*$
description	String	No	Microservice description. The value contains a maximum of 256 characters.
level	String	No	Microservice level. Options: • FRONT • MIDDLE • BACK
timestamp	String	No	Microservice registration time.

Parameter	Type	Mandatory	Description
schemas	String	No	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	No	Microservice status. Value: UP or DOWN. Default value: UP.
paths	ServicePath	No	Service path.
framework	Framework	No	Microservice development framework.
registerBy	String	No	Microservice registration mode. Options: • SDK • PLATFORM • SIDECAR • UNKNOWN
modTimestamp	String	No	Latest modification time (UTC).
properties	Map<String, String>	No	Extended attribute. You can customize a key and value.
alias	String	No	Microservice alias. The value contains 1 to 128 characters. Regular expression: ^[a-zA-Z0-9_\-.:]*$

Table 7-11 ServicePath parameters

Parameter	Type	Mandatory	Description
Path	String	No	Route address. The value must be 1 to 160 bytes long. Only digits, letters, and special characters (.,?'\/+&%$#=~_-@{}) are allowed.
Property	**Properties**	No	Extended attribute.

Table 7-12 Framework parameters

Parameter	Type	Mandatory	Description
name	String	No	Microservice development framework. Default value: UNKNOWN.
version	String	No	Version of the microservice development framework. The value must be 1 to 64 bytes long.

7.2.2 Properties

Table 7-13 Parameters

Parameter	Type	Mandatory	Description
< * >	String	No	Extended attribute. You can customize a key and value. The value must be at least 1 byte long.

7.2.3 DependencyMicroService

Table 7-14 Parameters

Parameter	Type	Mandatory	Description
appId	String	Yes	Application ID, which must be unique. The value must be 1 to 160 characters long. Regular expression: ^[a-zA-Z0-9]*$\|^[a-zA-Z0-9][a-zA-Z0-9_\-.]*[a-zA-Z0-9]$

Parameter	Type	Mandatory	Description
serviceName	String	Yes	Microservice name. For a provider microservice, the value can be *****, indicating all services of a tenant. If the value is *****, both **appId** and **version** can be omitted. For a consumer microservice, the value cannot be *****. The value must be 1 to 128 characters long. Regular expression: ^[a-zA-Z0-9]*$\|^[a-zA-Z0-9][a-zA-Z0-9_\-.]*[a-zA-Z0-9]$
version	String	Yes	Microservice version. For a provider microservice, the value can be the version+, fixed version, or latest version. For example, **1.0.1+** indicates version 1.0.1 or later. For a consumer microservice, the value can only be a fixed version. The value must be 1 to 64 characters long. Regular expression: ^[0-9]*$\|^[0-9]+(\.[0-9]+)*$
environment	String	No	Microservice environment. Options: • development • testing • acceptance • production

7.2.4 Rule

Table 7-15 Parameters

Parameter	Type	Mandatory	Description
ruleId	String	No	Blacklist or whitelist ID.
ruleType	String	Yes	List type. Value: WHITE (whitelist) or BLACK (blacklist).

Parameter	Type	Mandatory	Description
attribute	String	Yes	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	String	Yes	Matching rule. The value is a regular expression containing 1 to 64 characters.
timestamp	String	No	Rule creation time. This parameter is used only when you obtain a rule.
description	String	No	Description of a rule. The value contains a maximum of 256 characters.
modTimestamp	String	No	Latest modification time (UTC).

7.2.5 AddOrUpdateRule

Table 7-16 Parameters

Parameter	Type	Mandatory	Description
ruleType	String	No	List type. Value: WHITE (whitelist) or BLACK (blacklist).
attribute	String	No	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	String	No	Matching rule. The value is a regular expression containing 1 to 64 characters.
description	String	No	Description of a rule. The value contains a maximum of 256 characters.

7.2.6 DataCenterInfo

Table 7-17 Parameters

Parameter	Type	Mandatory	Description
name	String	Yes	Data center name. The value must be 1 to 128 bytes long. Only digits, letters, underscores (_), hyphens (-), and periods (.) are allowed.
region	String	Yes	Region where the data center is located. The value must be 1 to 128 bytes long. Only digits, letters, underscores (_), hyphens (-), and periods (.) are allowed.
availableZone	String	Yes	AZ where the data center is located. The value must be 1 to 128 bytes long. Only digits, letters, underscores (_), hyphens (-), and periods (.) are allowed.

7.2.7 Schema

Table 7-18 Parameters

Parameter	Type	Mandatory	Description
schemaId	String	No	Microservice schema ID. The value must be 1 to 160 bytes long. Only digits, letters, underscores (_), hyphens (-), and periods (.) are allowed.
schema	String	No	Microservice schema content. The value must be 1 to 2048 bytes long.
summary	String	No	Microservice schema summary. The value contains a maximum of 128 bytes. Only digits and letters are allowed.

7.2.8 environment_setting

Table 7-19 environment_setting parameters

Parameter	Type	Description
cluster_id	String	Kubernetes cluster ID.
language	String	Language type and version. At least one language type and version must be provided, for example, Java 8.
node_label_selector	String	Label for filtering nodes.

7.2.9 phases

Table 7-20 phases parameters

Parameter	Type	Description
build	Table 7-21	Build syntax struct.

Table 7-21 build parameters

Parameter	Type	Description
commands	Table 7-22	Build command struct.

Table 7-22 commands parameters

Parameter	Type	Description
commands	String	Software compilation command.

7.3 Common Response Parameters

7.3.1 WatchMicroServiceKey

Table 7-23 Parameters

Parameter	Type	Mandatory	Description
appId	String	No	Application ID, which must be unique. The value must be 1 to 160 characters long. Regular expression: ^[a-zA-Z0-9]*$\|^[a-zA-Z0-9][a-zA-Z0-9_\-.]*[a-zA-Z0-9]$
serviceName	String	No	Microservice name. For a provider microservice, the value can be *****, indicating all services of a tenant. If the value is *****, both **appId** and **version** can be omitted. For a consumer microservice, the value cannot be *****.
version	String	No	Microservice version. For a provider microservice, the value can be the version+, fixed version, or latest version. For example, **1.0.1+** indicates version 1.0.1 or later. For a consumer microservice, the value can only be a fixed version.

7.3.2 SuccessdResponse

Table 7-24 Parameters

Parameter	Type	Mandatory	Description
Result	String	Yes	Returned result.

7.3.3 ServiceInfo

Table 7-25 Parameters

Parameter	Type	Mandatory	Description
serviceId	String	Yes	Microservice ID, which must be unique. The value is a UUID. Regular expression: ^.*$
appId	String	Yes	Application ID, which must be unique. The value must be 1 to 160 characters long. Regular expression: ^[a-zA-Z0-9]*$\|^[a-zA-Z0-9][a-zA-Z0-9_\-.]*[a-zA-Z0-9]$
serviceName	String	Yes	Microservice name, which must be unique in an application. The value must be 1 to 128 characters long. Regular expression: ^[a-zA-Z0-9]*$\|^[a-zA-Z0-9][a-zA-Z0-9_\-.]*[a-zA-Z0-9]$
version	String	Yes	Microservice version. The value must be 1 to 64 characters long. Regular expression: ^[0-9]*$\| ^[0-9]+(\.[0-9]+)*$'
level	String	Yes	Microservice level. Options: • FRONT • MIDDLE • BACK
status	String	Yes	Service status. • UP • DOWN
timestamp	String	Yes	Timestamp.

7.3.4 AggregateMetricInfo

Table 7-26 Parameters

Parameter	Type	Mandatory	Description
time	Integer	No	Report time.
name	String	No	API name.
qps	Number	No	Current Query Per Second (QPS).
latency	Number	No	Average latency.
rate	Integer	No	Success rate.
total	Integer	No	Total number of requests.
isCircuitBreakerOpen	Boolean	No	Whether the circuit breaker is enabled.
failure	Integer	No	Total number of failed requests.
shortCircuited	Integer	No	Total number of short circuits.
semaphoreRejected	Integer	No	Total number of rejected semaphores.
threadPoolRejected	Integer	No	Total number of rejected threads.
countTimeout	Integer	No	Total number of timeout requests.

7.3.5 ServiceDetail

Table 7-27 Parameters

Parameter	Type	Mandatory	Description
microService	**MicroService**	No	Service information.
instances	**MicroServiceInstance**	No	Instance information.
schemaInfos	**Schema**	No	Microservice API details.
rules	**Rule**	No	Blacklist and whitelist.
providers	**MicroService**	No	All provider information.
consumers	**MicroService**	No	All consumer information.

Parameter	Type	Mandatory	Description
tags	Tags	No	All tag information about a microservice.
microServiceVersions	Array. Each item in the array is a string.	No	All microservice versions.

Table 7-28 tags parameters

Parameter	Type	Mandatory	Description
tags	Properties	No	Request struct of an extended attribute of a microservice.

7.3.6 TenantProject

Table 7-29 TenantProject parameters

Parameter	Located In	Type	Description
tenant	body	String	Tenant name.
projectId	body	String	ID.

7.3.7 Error

Table 7-30 Parameters

Parameter	Type	Mandatory	Description
errorCode	String	Yes	Error code.
errorMessage	String	Yes	Error message.
detail	String	No	Location details.

8 Permissions Policies and Supported Actions

8.1 Introduction

This chapter describes fine-grained permissions management for your ServiceStage. If your account does not need individual Identity and Access Management (IAM) users, then you may skip over this chapter.

By default, new IAM users do not have permissions assigned. You need to add a user to one or more groups, and attach permissions policies or roles to these groups. Users inherit permissions from the groups to which they are added and can perform specified operations on cloud services based on the permissions.

You can grant users permissions by using roles and policies. Roles are a type of coarse-grained authorization mechanism that defines permissions related to user responsibilities. Policies define API-based permissions for operations on specific resources under certain conditions, allowing for more fine-grained, secure access control of cloud resources.

NOTE

Policy-based authorization is recommended if you want to allow or deny the access to an API.

Your account has all the permissions required to call all APIs, but IAM users under your account must be assigned the required permissions. The permissions required for calling an API are determined by the actions supported by the API. Only users who have been granted permissions allowing the actions can call the API successfully.

Supported Actions

There are two kinds of policies: system-defined policies and custom policies. If the permissions preset in the system do not meet your requirements, administrators can create custom policies and apply these policies to user groups for refined access control. Operations supported by policies are specific to APIs. The following lists common concepts related to policies:

- Permissions: Defined by actions in a custom policy.
- Actions: Added to a custom policy to control permissions for specific operations.
- APIs: REST APIs that can be called in a custom policy.

Table 8-1 ServiceStage actions

Permissions	Actions	APIs
Creating an Application	servicestage:app:create	POST /v2/{project_id}/cas/applications
Deleting an Application Based on the Application ID	servicestage:app:delete	DELETE /v2/{project_id}/cas/applications/{application_id}
Modifying Application Information	servicestage:app:modify	PUT /v2/{project_id}/cas/applications/{application_id}
Obtaining All Applications	servicestage:app:list	GET /v2/{project_id}/cas/applications
Obtaining Application Details Based on the Application ID	servicestage:app:get	GET /v2/{project_id}/cas/applications/{application_id}
Querying Static Information About a Microservice	cse:registry:get	GET /v4/{project_id}/registry/microservices/{serviceId}
Querying Static Information About All Microservices	cse:registry:get	GET /v4/{project_id}/registry/microservices
Creating Static Information for a Microservice	cse:registry:modify	POST /v4/{project_id}/registry/microservices
Modifying Static Information About a Microservice	cse:registry:modify	PUT /v4/{project_id}/registry/microservices/{serviceId}/properties
Querying a Microservice Schema	cse:registry:get	GET /v4/{project_id}/registry/microservices/{serviceId}/schemas/{schemaId}
Modifying a Microservice Schema	cse:registry:modify	PUT /v4/{project_id}/registry/microservices/{serviceId}/schemas/{schemaId}
Creating a Dependency Between Services	cse:registry:modify	PUT /v4/{project_id}/registry/dependencies
Querying All Providers of a Microservice	cse:registry:get	GET /v4/{project_id}/registry/microservices/{serviceId}/providers

Permissions	Actions	APIs
Querying the Unique Service or Schema ID of a Microservice	cse:registry:get	GET /v4/{project_id}/registry/existence
Registering a Microservice Instance	cse:registry:modify	POST /v4/{project_id}/registry/microservices/{serviceId}/instances
Querying a Microservice Instance Based on Service ID	cse:registry:get	GET /v4/{project_id}/registry/microservices/{serviceId}/instances
Deregistering a Microservice Instance	cse:registry:modify	DELETE /v4/{project_id}/registry/microservices/{serviceId}/instances/{instanceId}
Querying Details About a Microservice Instance	cse:registry:get	GET /v4/{project_id}/registry/microservices/{serviceId}/instances/{instanceId}
Modifying the Extended Information About a Microservice Instance	cse:registry:modify	PUT /v4/{project_id}/registry/microservices/{serviceId}/instances/{instanceId}/properties
Changing the Status of a Microservice Instance	cse:registry:modify	PUT /v4/{project_id}/registry/microservices/{serviceId}/instances/{instanceId}/status
Sending Heartbeat Information	cse:registry:modify	PUT /v4/{project_id}/registry/microservices/{serviceId}/instances/{instanceId}/heartbeat
Querying a Microservice Instance by Filter Criteria	cse:registry:get	GET /v4/{project_id}/registry/instances
Querying Configurations	cse:config:get	GET /v3/{project_id}/configuration/items
Deleting Static Information About a Microservice	cse:registry:modify	DELETE /v4/{project_id}/registry/microservices/{serviceId}
Deleting Static Information About Microservices in Batches	cse:registry:modify	DELETE /v4/{project_id}/registry/microservices
Querying Microservice Instances in Batches	cse:registry:modify	POST /v4/{project_id}/registry/instances/action

Permissions	Actions	APIs
Querying All Schema Information About a Microservice	cse:registry:get	GET /v4/{project_id}/registry/microservices/{serviceId}/schemas

9 Appendix

9.1 Status Codes

Table 9-1 describes the status codes.

Table 9-1 Status codes

Status Code	Message	Description
200	-	OK
400	BadRequest	Invalid request. The client should not repeat the request without modifications.
401	Unauthorized	The authentication information is incorrect or invalid.
404	NotFound	The requested resource cannot be found. The client should not repeat the request without modifications.
422	UnprocessableEntity	The request was well-formed but was unable to be followed due to semantic errors.
500	InternalServerError	The server is able to receive the request but it could not understand the request.

9.2 ServiceStage Error Codes

If an error occurs after a request is sent to the system, a response containing an error code will be returned. The following lists the error codes.

Error Codes of Application Management APIs

Table 9-2 Error codes

Status Code	Error Code	Error Message	Description	Measure
400	SVCSTG.00100400	Invalid parameter.	Bad Request	Enter a valid value and try again.
	SVCSTG.00100402	The component has been deployed. Uninstall it first.	Bad Request	Uninstall the component and try again.
	SVCSTG.00100414	The job ID cannot be empty.	Bad Request	Specify a job ID.
	SVCSTG.00100418	The component instance operation has not been defined.	Bad Request	Specify a defined operation.
	SVCSTG.00100420	The runtime is not supported.	Bad Request	Specify a supported runtime.
	SVCSTG.00100422	Basic resources cannot be empty.	Bad Request	Set basic resources.
	SVCSTG.00100425	The artifact parameters cannot be empty.	Bad Request	Enter correct artifact parameters.
	SVCSTG.00100449	The environment name has been used.	Bad Request	Provide a new environment name.
404	SVCSTG.00100403	The application cannot be found.	Not Found	Specify an existing application.

Status Code	Error Code	Error Message	Description	Measure
	SVCSTG.00100404	The component cannot be found.	Not Found	Specify an existing component.
	SVCSTG.00100408	The request cannot be found.	Not Found	Specify an existing request.
	SVCSTG.00100411	The environment cannot be found.	Not Found	Specify an existing environment.
	SVCSTG.00100412	The instance cannot be found.	Not Found	Specify an existing instance.
	SVCSTG.00100424	The ELB instance cannot be found.	Not Found	Specify an existing ELB instance.
	SVCSTG.00100429	The job cannot be found.	Not Found	Specify an existing job.
	SVCSTG.00100433	The application release task cannot be found.	Not Found	Specify an existing release task.
	SVCSTG.00100436	The snapshot cannot be found.	Not Found	Specify an existing snapshot.
	SVCSTG.00100438	The template image cannot be found.	Not Found	Specify an existing template image.
	SVCSTG.00100441	The instance version has been used in the same environment.	Not Found	Provide a new version number.
405	SVCSTG.00100421	The account is suspended.	Method Not Allowed	Ensure that the account is normal.

Status Code	Error Code	Error Message	Description	Measure
	SVCSTG.00100427	Permission denied.	Method Not Allowed	Perform authorization again.
408	SVCSTG.00100428	Request timeout.	Request Timeout	The system is busy. Try again later.
409	SVCSTG.00100401	The application is not empty.	Conflict	Delete all components of the application and then delete the application.
	SVCSTG.00100413	The instance name already exists.	Conflict	Use another instance name.
	SVCSTG.00100419	You are not allowed to operate in the current state.	Conflict	The current operation has not been finished. Please try again later.
	SVCSTG.00100423	The component instance cannot be deleted.	Conflict	The current operation has not been finished. Please try again later.
	SVCSTG.00100426	The environment is already in use.	Conflict	Delete the instances deployed in the environment and then delete the environment.
	SVCSTG.00100432	The component is being released.	Conflict	Wait until the current task is complete and try again.

Status Code	Error Code	Error Message	Description	Measure
	SVCSTG.00100435	The release task cannot be deleted.	Conflict	The current operation has not been finished. Please try again later.
	SVCSTG.00100439	The ELB instance has been used in a component.	Conflict	Use another ELB instance.
500	SVCSTG.00100500	Internal service error.	Internal Server Error	The system is busy. Try again later.
	SVCSTG.00100501	Remote service error.	Internal Server Error	The system is busy. Try again later.

Error Codes of Git Repository Access APIs

Table 9-3 Error codes

Status Code	Error Code	Error Message	Description	Measure
400	SVCSTG.REPO.0400	Invalid parameter.	Bad Request	Enter a valid value and try again.
	SVCSTG.REPO.0401	Repository not supported.	Bad Request	Select another repository.
	SVCSTG.REPO.0402	Invalid OAuth code or state.	Bad Request	Perform OAuth 2.0 authorization again to obtain a correct authentication code and random string.
	SVCSTG.REPO.0403	The resource already exists.	Bad Request	Check whether the resource already exists.
	SVCSTG.REPO.0450	No repository authorization.	Bad Request	Check whether the authorization name in the request header is correct.
	SVCSTG.REPO.0451	Invalid repository authorization.	Bad Request	Perform authorization again.

Status Code	Error Code	Error Message	Description	Measure
	SVCSTG.REPO.0451	Invalid clone URL.	Bad Request	Enter a correct clone URL. Example: https://user-name@github.com/user-name/demo.git.
500	SVCSTG.REPO.0500	Internal service error.	Internal Server Error	Network error. Try again later.
	SVCSTG.REPO.0501	Remote repository error.	Internal Server Error	Check whether the request parameter is correct and the remote Git repository file has been modified.

9.3 CSE Error Codes

If an error occurs after a request is sent to the system, a response containing an error code will be returned. **Table 9-4** describes the error codes.

Table 9-4 Error codes

Category	Status Code	Error Code	Description	Measure
Configuration management	400	SVCSTG.00300607	Invalid dimension.	Change the value of dimensionsInfo as prompted.

Category	Status Code	Error Code	Description	Measure
		SVCSTG.00400641	Configuration item format incorrect	The format must meet the following requirements: • The value of a configuration item can contain 1 to 2048 characters, including digits, letters, and special characters (.-_:[]). • The value can contain digits, letters, and special characters in the parentheses ('-!@#$ %^&*()_\{}|:"<>?`-[], ./=). For the professional CSE, the value can contain 1 to 2048 characters. For the exclusive CSE, the value can contain 1 to 131072 characters.
		SVCSTG.00400642	Configured quota exceeding the upper limit	• Exclusive CSE: Delete some configurations. • Professional CSE: Delete some configurations or apply for exclusive CSE.
	401	SVCSTG.00300401	Unauthorized	Enter a correct token.
	500	SVCSTG.00300605	Internal service error.	Internal error. Contact technical support.
		SVCSTG.00300500	Internal service error.	Internal error. Contact technical support.
		SVCSTG.00300501	Internal service error.	Internal error. Contact technical support.
Microservice	400	400001	Invalid parameter.	Change parameters as prompted.
		400010	The service already exists.	Modify the service ID or microservice description in the body of the request for creating a microservice.
		400012	The service does not exist.	Enter a valid service ID.

Category	Status Code	Error Code	Description	Measure
		400013	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	The schema ID does not exist.	Enter a valid schema ID.
		400015	The schema cannot be modified.	The schema has been registered and cannot be modified.
		400016	The schema does not exist.	Register the schema first.
		400017	The instance does not exist.	Enter a valid instance ID.
		400023	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.
		400026	The microservice version does not exist.	Enter a correct version number or range.
		400100	Insufficient quota.	The quotas of resources such as microservices, instances, or schemas are insufficient. Delete some resources and create again.
	401	401002	Unauthorized	Enter a correct token.
	500	500003	Internal error.	Internal error. Contact technical support.

Category	Status Code	Error Code	Description	Measure
		500011	Backend error.	Internal error. Contact technical support.

9.4 Obtaining the Project ID

Obtaining the Project ID from the Console

A project ID is required for some URLs when an API is called. To obtain a project ID, perform the following operations:

Step 1 Sign up and log in to the management console.

Step 2 Hover the mouse pointer over the username and choose **My Credentials** from the drop-down list.

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

----End

9.5 Obtaining the Account ID

An account ID is required for some URLs when an API is called. To obtain an account ID, perform the following operations:

Step 1 Sign up and log in to the management console.

Step 2 Hover the mouse pointer over the username and choose **My Credentials** from the drop-down list.

View the account ID.

----End

9.6 Obtaining the Connection Address of Service Center

When calling an API of an exclusive microservice engine, you need to use the connection address of service center of the exclusive microservice engine to access the API. To obtain the service registry and discovery address, perform the following steps:

Step 1 Log in to ServiceStage and choose **Infrastructure** > **Cloud Service Engines**.

Step 2 Locate the target exclusive microservice engine, and view or click to copy the connection address.

----End