webMethods API-Portal Administrator's Guide

Version 9.10

April 2016

Document ID: YAP-AG-910-20160415

Table of Contents

About this Guide

This guide describes how you can use webMethods API-Portal and other webMethods components to effectively manage APIs for services that you want to expose to consumers, whether inside your organization or outside to partners and third parties. In addition to describing the API management components and workflow, the guide explains how to configure API-Portal for use with CentraSite and webMethods Mediator, how to manage API-Portal and its users, and how to manage APIs published to API-Portal.

To use this guide effectively, you should have an understanding of the APIs that you want to expose to the developer community and the access privileges you want to impose on those APIs.

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
Narrowfont	Identifies storage locations for services on webMethods Integration Server, using the convention *folder.subfolder:service* .
UPPERCASE	Identifies keyboard keys. Keys you must press simultaneously are joined with a plus sign (+).
Italic	Identifies variables for which you must supply values specific to your own situation or environment. Identifies new terms the first time they occur in the text.
`Monospace font`	Identifies text you must type or messages displayed by the system.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
|	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the | symbol.

Convention	Description
[ ]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information

Software AG Documentation Website

You can find documentation on the Software AG Documentation website at http://documentation.softwareag.com. The site requires Empower credentials. If you do not have Empower credentials, you must use the TECHcommunity website.

Software AG Empower Product Support Website

You can find product information on the Software AG Empower Product Support website at https://empower.softwareag.com.

To submit feature/enhancement requests, get information about product availability, and download products, go to Products.

To get information about fixes and to read early warnings, technical papers, and knowledge base articles, go to the Knowledge Center.

Software AG TECHcommunity

You can find documentation and other technical information on the Software AG TECHcommunity website at http://techcommunity.softwareag.com. You can:

- Access product documentation, if you have TECHcommunity credentials. If you do not, you will need to register and specify "Documentation" as an area of interest.
- Access articles, code samples, demos, and tutorials.
- Use the online discussion forums, moderated by Software AG professionals, to ask questions, discuss best practices, and learn how other customers are using Software AG technology.
- Link to external websites that discuss open standards and web technology.

1 Overview

Why Do Organizations Expose APIs?

Organizations often lack the resources to support mobile Bring Your Own Device (BYOD), supply chain, or eCommerce initiatives. By opening a set of APIs to external developers, organizations can reduce costs, expand the reach of their products or services, and create new channels of revenue in the following ways:

- Mobile application developers can create mashups and apps that satisfy a particular user niche and are optimized for specific mobile device types and platforms.
- Enterprise application developers can leverage APIs to simplify integration with suppliers and B2B partners.
- The involvement of external developers fosters innovation and collaboration throughout the development community. In return, the resulting developed applications offer the organization additional potential revenue as those applications reach new markets or customers in new ways.

Why Do APIs Need to Be Managed?

The APIs that an organization chooses to expose contain core assets the organization would naturally want to protect. As with the services they support, these APIs have a life cycle, need to be managed and governed, and require mediation and security at run time.

From an API provider's perspective, an API management tool is needed that enables the provider to do the following:

- Maintain an inventory of APIs and their associated resources.
- Publish, secure, and retire APIs according to defined service level agreements.
- Onboard API developers and give those developers the ability to publish APIs on behalf of the organization.
- Onboard API consumers who will use the published APIs in their own applications.
- Provide tiered access to APIs, for example according to authorization level.
- Track key performance indicators (KPIs) to help monitor and interpret API use.

From an API consumer’s perspective, an API management tool should provide the ability to:

- Browse a catalog of APIs and obtain details and code samples for a specific API.
- Sign up and request and manage access tokens to download an API and its associated resources and documentation.
- Test the functionality of an API.

- Collaborate with other API consumers by way of forums or integration with social media.

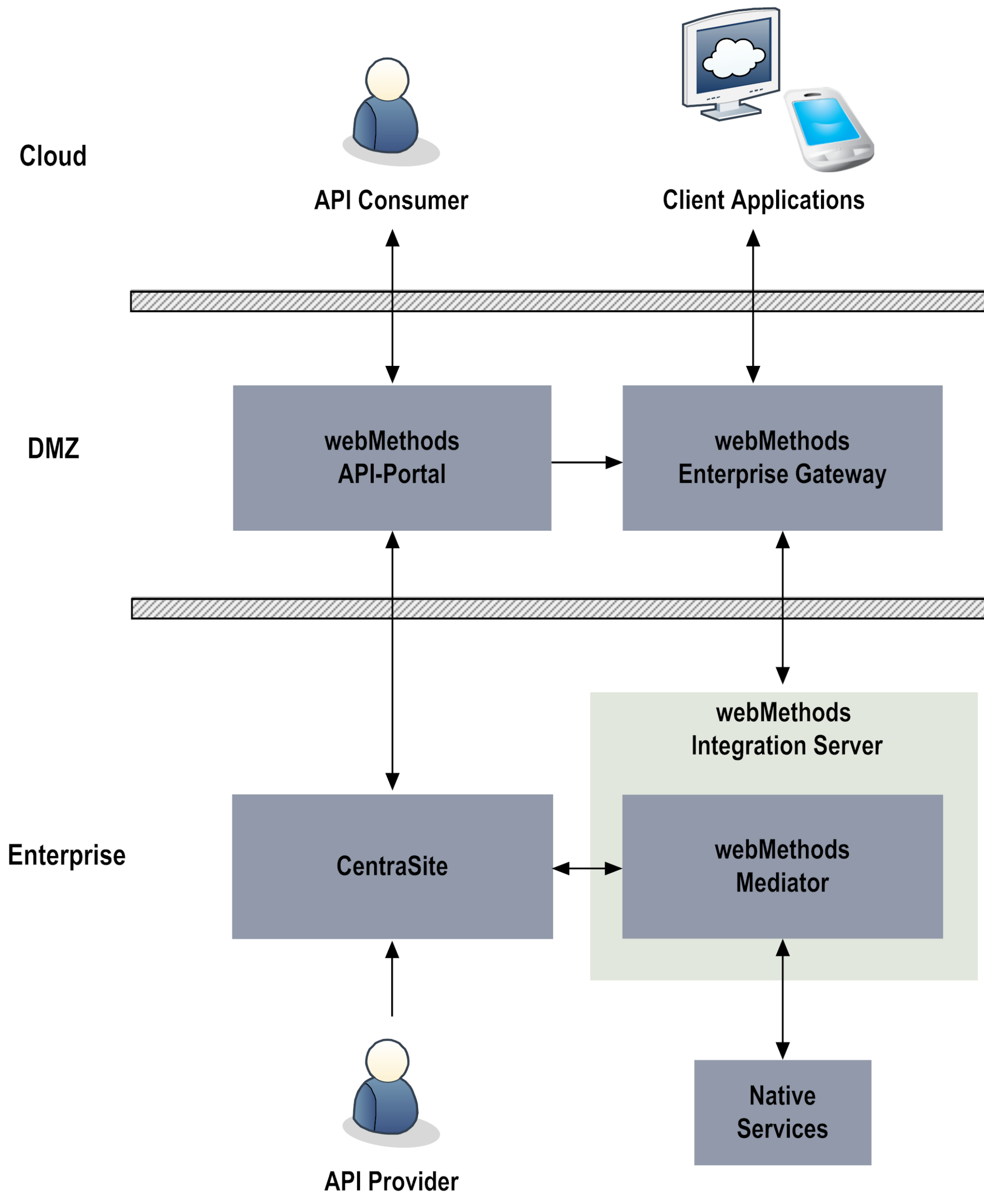
What Is webMethods API-Portal?

webMethods API-Portal is a web-based, self-service portal that enables an organization to securely expose APIs to external developers, partners, and other consumers for use in building their own apps on their desired platforms. API-Portal provides the following features:

- **Branding and customization.** API-Portal administrators can customize their portal's logo, colors, and fonts to match their organization's corporate identity. Administrators can further customize their portal by modifying pages, incorporating widgets, and changing the appearance and organization of APIs in the gallery for easier discovery. For example, APIs in a large catalog can be grouped by business domain, free versus paid, or public versus B2B partner. APIs can also be flagged based on maturity level (for example, beta versus production or release).
- **Support for SOAP and REST APIs.** API-Portal supports traditional SOAP-based APIs as well as REST-based APIs. This support enables organizations to leverage their current investments in SOAP-based APIs while they adopt REST for new APIs.
- **Quick, secure providing of access tokens.** Approval workflows in CentraSite simplify the provisioning of API keys and OAuth2 credentials. These workflows enable the API provider to individually approve access token requests that developers submit from API-Portal.
- **Easy discovery and testing of APIs.** Full text search capabilities help developers quickly find APIs of interest. API descriptions and additional documentation, usage examples, and information about policies enforced at the API level provide more details to help developers decide whether to adopt a particular API. From there, developers can use the provided code samples and expected error and return codes to try out APIs they are interested in, directly from within API-Portal, to see first-hand how the API works.
- **Quick, secure onboarding of new users.** Easy to configure approval workflows in API-Portal graphical user interface to define how the user onboarding should take place, with or without confirmations.
- **Community support.** API-Portal provides a collaborative community environment where API consumers can rate APIs and contribute to open discussions with other developers.
- **Built-in usage analytics.** API-Portal provides information about where visitors are coming from, how many visitors become registered users, how many provisioned access tokens are actually used, what pages gather the most interest, and which APIs are more popular than others. This information is available by way of dashboards to API providers who have an API administrator role in API-Portal. With this information, providers can understand how their APIs are being used, which in turn

can help identify ways to improve their users' portal web experience and increase API adoption.

The following diagram illustrates a typical scenario of products that make up the webMethods API management product suite.



In this scenario, webMethods API management suite products include the following:

- **webMethods API-Portal**. In API-Portal, API consumers browse the catalog of APIs that a provider has published. When the consumer finds an API of interest, the consumer can sign up and request an access token to download the API for further investigation and testing.

API providers who have an API administrator role in API-Portal can also view dashboards containing details about API run-time usage.

Provided with each API-Portal installation is a sample portal called SAGTours. The SAGTours sample provides an end-to-end scenario using CentraSite, webMethods Mediator, and API-Portal to demonstrate how the fictitious company SAGTours has customized the content as well as the look and feel of an out-of-the-box API-Portal. For details about installing the SAGTours demo and using it as a basis for customizing your own portal, see *webMethods API-Portal Online Help*.

- **CentraSite**. CentraSite provides a registry and repository for APIs and offers complete design-time governance of those APIs. API providers add APIs to CentraSite by defining the APIs and their associated resources as objects. When APIs are ready to be made available to consumers, API providers publish the APIs from CentraSite to API-Portal.

  CentraSite administrators do the following API management tasks:

  - Register instances of API-Portal.
  - Manage API provider and API consumer user accounts.
  - Manage the API catalog.
  - Deploy virtualized APIs to webMethods Mediator.
  - Configure policies to be enforced at run time.
  - Manage API and OAuth2 keys and API access tokens.

  When API providers add API services to CentraSite as assets, the providers can attach supporting documents to the API assets. Examples of such documents include input files containing WSDL or schema definitions, programming guides, sample code, legal notices and terms of use, and associated contracts and plans. When the APIs are published from CentraSite to API-Portal, these supporting documents are published to API-Portal as well.

- **webMethods Mediator**. Mediator provides complete run-time governance of APIs published to API-Portal. Mediator acts as an intermediary between service consumers and service providers. Mediator also enforces access token and operational policies such as security policies for run-time requests between consumers and native services. Using Mediator, API providers can do the following:

  - Enforce security, traffic management, monitoring, and SLA management policies.
  - Transform requests and responses into expected formats as necessary.
  - Perform routing and load balancing of requests.
  - Collect run-time metrics on API consumption and policy evaluation.

- **webMethods Integration Server**. Integration Server hosts Mediator and initiates connections to webMethods Enterprise Gateway. Integration Server also orchestrates the services and provides the connection to back-end systems.
- **webMethods Enterprise Gateway**. Enterprise Gateway protects the APIs on Mediator and other webMethods products installed behind the firewall from malicious attacks initiated by external client applications. Administrators can secure traffic between API consumer requests and the execution of services on Mediator by doing the following:
  - Filter requests coming from particular IP addresses and blacklist specified IP addresses.
  - Detect and filter requests coming from particular mobile devices.
  - Avoid additional inbound firewall holes through the use of reverse invoke.

2 Configuring API-Portal

Before You Configure API-Portal

The webMethods API-Portal solution consists of CentraSite, webMethods Mediator, and webMethods API-Portal. At a minimum, CentraSite is required to publish APIs to the API-Portal. Ensure you have a CentraSite instance installed before configuring API-Portal. webMethods Mediator is optional.

If you are using Enterprise Gateway to secure traffic between API consumer requests and the execution of services on Mediator, ensure that CentraSite, Integration Server, and Mediator are installed behind the firewall. In this scenario, the Enterprise Gateway Server is installed in the DMZ for external access. Alternatively, for internal users only, you can use Integration Server with Mediator deployed in a secure network behind the firewall.

For more information about installing these products, see *Installing Software AG Products*.

Security Considerations

Use the following information to ensure your API-Portal installation is protected.

Securing Client Requests

webMethods API-Portal supports both HTTP and HTTPS, allowing it to listen on an HTTP port for non-secure client requests and an HTTPS port for secure requests.

Unlike HTTP, HTTPS provides for secure data transmission. HTTPS does this through encryption and certificates. Without HTTPS, unauthorized users might be able to capture or modify data, use IP spoofing to attack servers, access unauthorized services, or capture passwords.

By default, the API-Portal load balancer component is set to allow both unencrypted HTTP and encrypted HTTPS/SSL access. Software AG recommends using HTTPS to ensure a secure connection, and disabling the HTTP port.

For instructions on how to disable the HTTP port, see "Disabling a Port" on page 21.

Preventing Use of the HTTP OPTIONS Method

The OPTIONS request method, while part of the HTTP standard, has the potential for allowing incoming requests to obtain information about API-Portal server capabilities or to get information about resources, even though the request does not specify a resource action or retrieve a resource.

By default, the API-Portal load balancer component is set to allow HTTP OPTIONS method requests. Software AG recommends deactivating the OPTIONS method in the load balancer, preventing it from responding to the requests.

To deactivate the OPTIONS method

1. Stop the loadbalancer component from the API-Portal Cloud Controller (ACC).
2. In a text editor, open the httpd-custom.conf and the http-custom-ssl.conf files from the following directory: *Software AG_directory* \API_Portal\server\bin\work \work_loadbalancer_m\httpd\conf\extra.
3. Add the following lines to the files:

```
<Location/>
   <Limit OPTIONS>
      Deny from all
   </Limit>
</Location>
```

4. Start the loadbalancer component from the ACC.

Implementing Secure Password Policies

If the API-Portal default password policy does not comply with your security requirements, you can change the password policy settings.

To change the password policies

1. Start a web browser and go to the API-Portal User Management Console (UMC) application at: http://*host* :*port* /umc
2. Click **Configuration**, and then select **Password policy** from the drop-down list to display all the related properties.
3. The current password policy settings are shown. Change the properties as needed. To see a description of each property, hover your cursor over the property name.

For additional information about setting passwords and using the Configuration page, see the *webMethods API-Portal Online Help*.

Sending Email Notifications

Both API-Portal and CentraSite can send email messages to notify administrators and users about important events and to convey status information.

API-Portal can send user management related email messages to notify users about:

- Status of access token requests, renewals, and expiration
- Critical events
- User registration status, including approval workflow notifications

API-Portal can also reply to user requests for forgotten passwords. Additionally, during access token requests, if CentraSite is not reachable, API-Portal will send an email alert to all users with the API-Portal Administrator role.

CentraSite can send email notifications about:

- Access tokens expiring
- Policy-related actions
- Service deployment

For CentraSite to issue email messages, an administrator must first configure CentraSite's email server settings. CentraSite also provides predefined email templates for use with API key or OAuth token generation, renewal, and expiration. By default, these templates are configured in the centrasite.xml file. But, if you do not want to use the predefined email templates, you can create your own templates and configure the centrasite.xml file as necessary. For instructions to configure CentraSite email server settings, see the *CentraSite Administrator's Guide*. For more information about the predefined email templates, see the section on managing API keys and OAuth 2.0 tokens in *Working with the CentraSite Business UI*.

Configuring the SMTP Mail Server Connection for API-Portal

To enable API-Portal to send email notifications, you need to register your SMTP server and set the sender's email address.

This can be achieved in one of the following ways:

- "Configuring the SMTP Mail Server Connection for API-Portal using the User Management Component" on page 18
- "Configuring the SMTP Mail Server Connection for API-Portal using ACC" on page 19

Configuring the SMTP Mail Server Connection for API-Portal using the User Management Component

You can customize your system configuration to meet your requirements at runtime without having to restart the system. You carry out this part of the configuration in the User Management Component. You must have the Technical configuration function privilege.

To register the SMTP mail server and set the sender's email address using the user management component

1. Log in into the **User Management Component**.

   http://*host* :*port* /umc

2. Click **Configuration**.
3. Select **SMTP** from the drop-down list to display all the related parameter.
4. Type the SMTP mail server address, including the domain. For example:

   `API-Portal@MyCompany.com`

5. Type the port number; the port number should be 25.
6. Define the sender's email address.

   You configured the SMTP Mail Server Connection using the User Management Component.

Configuring the SMTP Mail Server Connection for API-Portal using ACC

To enable API-Portal to send email notifications, you need to register your SMTP server and set the sender’s email address.

To register the SMTP mail server and set the sender’s email address using ACC

1. Set the SMTP mail server.
   a. Start the ACC.
   b. At the command line, run the `register` external service, where *smtp_server* is the SMTP mail server address, including the domain:

      ```
      register external service smtp host=smtp_server port=25
      ```

   c. Verify the setting by entering the following command, and examining the output to see the email server is listed:

      ```
      list external services
      ```

2. Set the sender’s email address.
   a. Start a web browser and go to the API-Portal UMC application:

      http://*host* :*port* /umc

   b. Click **Configuration**, and select **SMTP** from the drop-down list to display all the related parameters.
   c. Double click the com.aris.umc.notification.sender parameter and type your mail server address. For example:

      ```
      API-Portal@MyCompany.com
      ```

3. In the ACC, restart API-Portal.
   a. Type `stopall` to stop API-Portal.
   b. Type `list` to verify the status of the API-Portal components and ensure that all are in the STOPPED state before proceeding.
   c. After all components have stopped, type `startall` to start API-Portal.

Usage Reports

Usage reports allow API providers to monitor the usage of their APIs. Usage reports in API-Portal show the number of requests made by all the applications for an API for a specified period of time.

Usage reports are sent directly to consumers through email, so you have to make sure that you only provide reports that make sense for the consumers and that are considered safe in terms of security and regulatory legislation. The reports are executed with a higher privileged user role.

Before you can schedule a usage report in API-Portal, you have to share the report template from CentraSite to the instance of API-Portal you require. For more information about report templates in CentraSite, see the *Working with the CentraSite Business UI*.

API consumers can define the frequency of the report they require. API consumers receive the scheduled report in the PDF format through email. For more information about scheduling API usage reports, see *webMethods API-Portal Online Help*.

Creating a Usage Report

To create a usage report

1. Click the ☰ in the right top corner of the API-Portal window and click Reports.
2. Click Generate.
3. Provide a name and a description of the report.
4. Select an active token from the **Report to schedule** drop-down list.
5. Specify the frequency to execute the report.
6. Click **Schedule**.

Configuring Ports

API-Portal listens for requests on ports that you specify. Each port is associated with a specific type of protocol: HTTP, HTTPS, or email.

By default, the API-Portal load balancer component is set to allow both unencrypted HTTP and encrypted HTTPS/SSL access. API-Portal has the following pre-configured ports:

Port Type	Default Port Number	Description
HTTP	18101	Unsecured/unencrypted port
HTTPS	18102	Secure/encrypted port
Email	25	SMTP port

About Enabling/Disabling a Port

API-Portal accepts port connection requests as soon as it receives them. If you want to temporarily prevent API-Portal from accepting requests on one of its ports, you can disable that port. This action blocks incoming requests from reaching the API-Portal server. When a port is disabled, clients receive an error message when they issue requests to it. Later, you can enable the port. If you stop and restart API-Portal, the port remains disabled until you enable it. Disabling a port is a convenient way to eliminate developer access to an API-Portal once it goes into production.

Disabling a Port

Disabling a port allows you to stop the port from accepting connections or dispatching more requests. For example, you may need to temporarily disable ports for testing, or disable the HTTP port and use only the HTTPS port for secure client requests. To disable a port, use the ACC `reconfigure` command and set the port to 0.

To disable a port

1. Start the ACC.
2. Stop the load balancer component.
3. At the ACC command prompt type:

   ```
   reconfigure loadbalancer_m  +HTTPD.port=0
   ```

4. Start the load balancer component.
5. To verify that you have deactivated the port, try logging in. The welcome screen is not visible, if you have deactivated the port.

About Enabling/Disabling a Port

API-Portal accepts port connection requests as soon as it receives them. If you want to temporarily prevent API-Portal from accepting requests on one of its ports, you can disable that port. This action blocks incoming requests from reaching the API-Portal server. When a port is disabled, clients receive an error message when they issue requests to it. Later, you can enable the port. If you stop and restart API-Portal, the port

remains disabled until you enable it. Disabling a port is a convenient way to eliminate developer access to an API-Portal once it goes into production.

About Enabling/Disabling a Port

API-Portal accepts port connection requests as soon as it receives them. If you want to temporarily prevent API-Portal from accepting requests on one of its ports, you can disable that port. This action blocks incoming requests from reaching the API-Portal server. When a port is disabled, clients receive an error message when they issue requests to it. Later, you can enable the port. If you stop and restart API-Portal, the port remains disabled until you enable it. Disabling a port is a convenient way to eliminate developer access to an API-Portal once it goes into production.

Enabling a Port

When you are ready to have API-Portal begin accepting requests on a port you previously disabled, you must enable it. To enable a port, use the ACC `reconfigure` command and set the port number.

To enable a port

1. Start the ACC.
2. Stop the load balancer component.
3. At the ACC command prompt type:

   ```
   reconfigure loadbalancer_m  +HTTPD.port=port_number
   ```
4. Start the load balancer component.

Testing for HTTPS Requests

To test whether your server is listening to HTTPS requests on the port you specified, bring up your browser and type `https://localhost:`*`port`*. If the port is working properly, you will see the logon screen for API-Portal. If API-Portal does not display, check to see if a service running on the machine is listening to the same port.

Considerations for Machines with Multiple Network Interfaces

By default, the load balancer is set at installation time with the server host name and port details. These details are persisted in all load balancer configurations. If you are configuring multiple machines (for example, cloud instances) and have cloned one to many, you need to reconfigure the load balancer for each machine. This ensures each machine has its own identity, and prevents problems at startup.

Reconfiguring the Load Balancer when you configure multiple API-Portal machines

If you are configuring multiple API-Portal machines, you need to check the load balancer components on each one and ensure that the `HTTPD.servername` parameter is set to the correct IP address of server.

To reconfigure the load balancer component

1. Start the ACC.
2. At the command line, type `show instance loadbalancer_m` to see the current configuration settings of the load balancer component. For example:

```
ACC+ localhost>show instance loadbalancer_m
ID: loadbalancer_m state:STARTED type:com.aris.runnables.httpd.httpd-run-prod-98.0.0
classifier=runnable type=zip
Configuration parameters:
        HTTPD.LimitRequestFieldSize=32768
        HTTPD.access.root=granted
        HTTPD.keepalive=on
        HTTPD.modjk.max_packet_size=32768
        HTTPD.modjk.stickySessions.abs=true
        HTTPD.modjk.stickySessions.ads=true
        HTTPD.modjk.stickySessions.ecp=true
        HTTPD.modjk.stickySessions.processboard=true
        HTTPD.modjk.stickySessions.umc=true
        HTTPD.port=18101
        HTTPD.servername=192.0.2.11
        HTTPD.ssl.port=18102
        appcontext.abs=abs
        appcontext.cop=/
        appcontext.ecp=collaboration
        plugin.ping.search.for.processes=false
        zookeeper.application.instance.host= portal01.my.org
        zookeeper.connect.string=localhost:18043
```

3. Examine the `HTTPD.servername` parameter. If it needs to point to another server IP, you can change it by using the `reconfigure` command. To do so:

   a. Stop the load balancer component:

   ```
   stop loadbalancer_m
   ```

   b. Change the value of the `HTTPD.servername` parameter and specify the new IP address:

   ```
   reconfigure loadbalancer_m +HTTPD.servername=new_IP
   ```

   c. Start the load balancer component:

   ```
   start loadbalancer_m
   ```

Reconfiguring the Loadbalancer in case of a DMZ/Reverse Proxy Setup

For on-premise deployments, the recommendation is to place API-Portal behind a reverse proxy in the DMZ. The reverse proxy is required for tunneling of requests for the API-Portal. The reverse proxy can either be an arbitrary reverse proxy like an Apache with mod_proxy or mod_rewrite modules, a third party security appliance, or a wM Enterprise Gateway (for the pure API traffic).

The API-Portal load balancer by default redirects any clients to its fully-qualified host name. For example, if the server's host name is "myportal.mycompany.com" and a user who is in the domain "mycompany.com" can access the server with its non-qualified name "myportal". The user (or rich client and so on being used) will be redirected to "myportal.mycompany.com".

When API-Portal is placed behind a reverse proxy in the DMZ the redirect may not work properly with certain network configurations where there are internal and external IPs and no hostname. This can be solved by reconfiguring the Loadbalancer settings.

You have to reconfigure the following parameter settings:

- `HTTPD.servername` parameter to manually tell the load balancer the proper external hostname of the reverse proxy (or IP address) to be used to redirect.
- `HTTPD.RewriteEngine` parameter to disable the automatic redirection.

The loadbalancer registers itself as a service in zookeeper. This registration includes registration of a port, a hostname, and a scheme. This information can then be used by other API-Portal applications to generate URLs that are passed to users (for example, in email notifications). In the reverse proxy case, the hostname used for registration should be the external (reverse proxy) address. This has to be configured by reconfiguring the `zookeeper.application.instance.host` parameter.

Reconfiguring the Loadbalancer Settings

1. Start the ACC
2. At the command line, type `show instance loadbalancer_m` to see the current configuration settings of the load balancer component.

   ```
   ACC+ localhost>show instance loadbalancer_m
   ```

3. Stop the load balancer component.

   ```
   stop loadbalancer_m
   ```

4. Reconfigure the following loadbalancer settings for redirect as required:
   - `HTTPD.servername` parameter to redirect to the proxy server IP. You can change it by using the reconfigure command as follows:

```
reconfigure loadbalancer_m
+HTTPD.servername=<reverse_proxy_Address>
```

- Disable the automatic redirection using the following command:

  ```
  reconfigure loadbalancer_m +HTTPD.RewriteEngine=off
  ```

5. Change the value of the `zookeeper.application.instance.host` parameter and specify the proxy server IP address:

   ```
   reconfigure loadbalancer_m zookeeper.application.instance.host
   ="<reverseproxy_address>"
   ```

6. Start the load balancer component:

   ```
   start loadbalancer_m
   ```

Configuring CentraSite, Mediator, and API-Portal

To add and manage API-Portal instances in CentraSite, you must have the privileges of a CentraSite Administrator role or at least the API-Portal Administrator role.

- If you have the privileges of a CentraSite Administrator role, you can manage any API-Portal within any organization.
- If you have the privileges of an API-Portal Administrator role for an organization, you can manage all the API-Portal instances in that particular organization.

  The API-Portal administrator is responsible for installing, configuring, and maintaining API-Portal. The administrator is also responsible for ensuring the server is secure, available to clients, and running at peak performance. Usually, one person is appointed as the API-Portal administrator, although most sites identify at least one other person to act as a backup.

For more information about roles, see "Overview of Managing Users" on page 64 and the *CentraSite Administrator's Guide*.

After the components listed in "Before You Configure API-Portal " on page 16 are installed, additional configuration tasks must be performed to set up the environment in preparation for publishing APIs to API-Portal. The following table lists these high-level configuration steps and where to go for more information:

Step	**Where to find the procedures**
Import the API-Portal license file.	" API-Portal License File" on page 30
Configure the CentraSite infrastructure, including LDAP, email server, and log purging.	*CentraSite Administrator's Guide*

Step	Where to find the procedures
Set up an organization structure for API provider and consumer organizations.	*Working with the CentraSite Business UI*
(Optional) Configure Mediator in a clustered Integration Server environment.	Section on configuring communication with CentraSite in *Administering webMethods Mediator*
Configure the communication link between Mediator and Enterprise Gateway.	Section on configuring Enterprise Gateway in *webMethods Integration Server Administrator's Guide*
Configure the communication link between CentraSite and Mediator by defining Mediator targets in CentraSite.	*CentraSite Administrator's Guide*
Create a technical user in CentraSite. Specify this user when registering an API-Portal instance in CentraSite. It is considered a best practice not to tie critical actions (such as publishing data or APIs) to a real user whose credentials can expire. **Note:** Software AG recommends specifying the same user credentials as the technical user created in Mediator and API-Portal.	*Working with the CentraSite Business UI*
Create a technical user for Mediator and add the user to the Administration group so that it may be used by CentraSite when publishing APIs to Mediator. **Note:** Software AG recommends specifying the same user credentials as the technical user created in CentraSite and API-Portal.	Section on adding user accounts and adding users to a group in *webMethods Integration Server Administrator's Guide*

Step	Where to find the procedures
Create a technical user in API-Portal and assign the user to the API Provider role. Specify this user when registering an API-Portal instance in CentraSite. It is considered a best practice not to tie critical actions (such as publishing data or APIs) to a real user whose credentials can expire. **Note:** Software AG recommends specifying the same user credentials as the technical user created in CentraSite and Mediator.	*webMethods API-Portal Online Help*
Create API Provider users (developers who can publish APIs to API-Portal) and assign the users to the API-Portal Administrator role.	Sections on adding a user, assigning a user to a group, and assigning a role to a user in *CentraSite Administrator's Guide*
Register API-Portal instances with CentraSite.	*Working with the CentraSite Business UI*
Configure settings for proxy server, analytics, and geolocation.	"API-Portal Configuration Parameters" on page 28
Set up and customize email templates to be used when informing users about access token request, renewal, and expiration.	Section on managing API keys and OAuth 2.0 tokens in *Working with the CentraSite Business UI*
Create taxonomies in CentraSite that you can use to categorize APIs in the API-Portal Gallery page.	*CentraSite Administrator's Guide*
Customize the API-Portal user registration and the e-mail-templates for user registration.	"Registering Users in API-Portal " on page 32
Define the advanced customizations of API-Portal.	"Advanced Configuration of API-Portal " on page 38

Step	Where to find the procedures
Customize the user registration with social media for API-Portal.	"User Registration in API-Portal with Social Login" on page 43
Customize the API-Portal branding and set up API-Portal templates.	*webMethods API-Portal Online Help*

API-Portal Configuration Parameters

This section contains a description of the parameters you can specify in the API-Portal configuration file (apiportal.properties). This configuration file contains parameters specific to API-Portal data analysis. The file is located in the *Software AG_directory*`\API-Portal\server\bin\work\work_apiportalbundle_m\base\webapps\abs\WEB-INF\classes\com\aris\modeling\server\webapp\apiportal\configuration` directory.

To set parameters in the `apiportal.properties` file, stop API-Portal, and then edit the file directly with a text editor. After you save the changes, restart (start) API-Portal.

The server uses default values for many of the parameters. If a parameter has a default, it is listed with the description of the parameter. If the service request must be routed through a proxy server, these properties specify the proxy server alias for the proxy server through which API-Portal routes the HTTP/S request.

apiportal.analytics.

apiportal.analytics.enabled
Specifies whether API-Portal collects analytics data. When the `apiportal.analytics.enabled` parameter is set to `true`, API-Portal collects analytics data. When the parameter is set to `false`, API-Portal does not collect analytics data. The default is `true`.

apiportal.analytics.enabled.category
When API-Portal is set to collect analytics data, use the `apiportal.analytics.enabled.category` property to identify which categories of data will be collected. API-Portal analytics collects three different categories of data: PageViews, UserAudit, and AccessTokenAudit. If you specify multiple categories, separate the names with commas (,).

The default is `PageViews,UserAudit,AccessTokenAudit`

apiportal.proxy.

apiportal.proxy.url
Specifies the proxy server URL used to communicate with the services through the Internet.

Use the format http://*host* :*port*

where *host* is the host name or IP address of the proxy server and *port* is the port number of the proxy server.

apiportal.proxy.username
Optional. If your proxy server requires authentication, use this parameter to specify the username the proxy server uses to authenticate incoming requests.

apiportal.proxy.password
Optional. If your proxy server requires authentication, use this parameter to specify the password the proxy server uses to authenticate incoming requests.

apiportal.proxy.noProxyHosts

apiportal.proxy.noProxyHosts
Specifies the servers that API-Portal connects to directly and not through a proxy server.

Use the format *host_1* |*host_2* |*host_n*

where *host* is the host name or IP address of the server. You can specify a list of servers, each separated by a |. You can also specify a wildcard character (*) for matching.

For example:

```
apiportal.proxy.noProxyHosts=localhost|127.0.0.1|*.int.abc.org
```

apiportal.geo.

apiportal.geo.service.url
When resolving the location of the user who is accessing API-Portal based on the IP address of the client, API-Portal uses the external service, telize.com, to resolve the location based on the IP address of the client.

The value for this property must be: `http://www.telize.com/geoip/`

If you are using a proxy server, ensure that you also set the properties apiportal.proxy.url, apiportal.proxy.username, and apiportal.proxy.password to ensure API-Portal can communicate the external service.

Preparing to Publish APIs to API-Portal

Before API providers can publish APIs to API-Portal, additional steps need to be performed. The following table lists these high-level steps and where to find more information:

Step	Where to find the procedures
(Optional) Configure design-time and change-time policies using the predefined policies Publish to API-Portal and UnPublish from API-Portal.	*Working with the CentraSite Business UI*

Step	Where to find the procedures
This step is needed only if your organization requires design-time governance.	
Set up approval and onboarding workflows in CentraSite.	Section on working with design/ change-time policies in *Working with the CentraSite Business UI*
Create the SOAP or REST API and attach any supporting documents to this asset.	*Working with the CentraSite Business UI*
Create a new virtual alias for the API.	Section on creating a virtual API in *Run-Time Governance with CentraSite*
Configure the run-time policies to enforce for the API you want to expose.	*Run-Time Governance with CentraSite*
Configure consumption settings for the API.	Section on configuring the API consumption settings in access key management policies in *Working with the CentraSite Business UI*
Deploy the API to Mediator.	Section on publishing an API to Mediator in *Run-Time Governance with CentraSite*
Publish the API to API-Portal.	Section on publishing an API to API-Portal in *Run-Time Governance with CentraSite*

API-Portal License File

API-Portal cannot be viewed or used without a license. If the license was not provided during installation, you must import it before you can use API-Portal. There are two ways to import the license and then get API-Portal up and running: from the API-Portal Cloud Controller (ACC) console or from the API-Portal User Management Console (UMC).

For instructions see:

- "Importing the API-Portal License File using the ACC" on page 31
- "Importing the API-Portal License File using the UMC" on page 31

Importing the API-Portal License File using the ACC

To import the API-Portal license file from the ACC

1. Start API-Portal if it is not already running.
2. Start the API-Portal Cloud Controller (ACC).
3. Ensure all runnables are started. To do so, issue the `list` command.
4. Type the following command:

   ```
   invoke delegate on apiportalbundle_m app=umc
   command=enhancement_importLicense tenant.name=<tenantname>
   tenant.user.name=<adminusername> tenant.user.pwd=<adminpassword>
   enhancement.path=c:\licensefile.zip
   ```

5. Type `stopall` to stop API-Portal.
6. Type `startall` to restart API-Portal.

For instructions on how to perform the steps above, see:

- "Starting API-Portal (Windows)" on page 56
- "Starting API-Portal (Linux/UNIX)" on page 57
- "Stopping API-Portal (Windows)" on page 58
- "Stopping API-Portal (Linux/UNIX)" on page 58

Importing the API-Portal License File using the UMC

To import the API-Portal license file from the UMC

1. Start API-Portal if it is not already running.
2. Start a web browser and go to the API-Portal UMC application at: http://*host* :*port* / umc
3. Log in as system user (default password `manager`).
4. In the UMC, click **Licenses** and then import the license: (http://*host* :*port* /umc/? tenant=default#licenses)
5. Stop API-Portal.
6. Start API-Portal.

For instructions on how to start and stop API-Portal, see:

- "Starting API-Portal (Windows)" on page 56
- "Starting API-Portal (Linux/UNIX)" on page 57
- "Stopping API-Portal (Windows)" on page 58

- "Stopping API-Portal (Linux/UNIX)" on page 58

Registering Users in API-Portal

When visitors to your API-Portal decide they want to use the APIs there, they need to have full access as validated users to log in to the portal. To log in users can either:

- Register with API-Portal and provide an email address and create a password. Upon approval, API-Portal creates an account for the user in the UMC, and the user can log in with the email address and password provided at registration. Users can manage their own account details and change the password from the Profiles link in API-Portal.
- Use their credentials for a current social account (Google, Facebook). Upon approval, API-Portal notifies the user. At the first login, API-Portal creates an account for the user in the UMC. Users must use the social account to change account details and their password. If you want to allow requesters to use their social account, you need to configure it.

Depending on your security requirements, you can choose the level of approval needed when registration requests and social logins arrive:

- **Approval workflows.** Ensure security while simplifying operations during user registration by using approval workflows within API-Portal for security credentials provisioning. With approval workflows, API-Portal can notify an administrator (or a group of approvers) about user registration requests and allow the approvers to approve or reject the requests. Upon approval, API-Portal notifies the requester by email. If the user registered with an email address and password, API-Portal creates the user account at approval. If the user registered with a social account, API-Portal creates the user account when the user first logs in after approval.

  Note: Approval workflows in API-Portal are separate from the approval workflows that are used with run-time policies in CentraSite.

- **Email confirmation.** An alternative to implementing approval workflows, email confirmation provides a simple way to register new users. Upon receiving a user registration request, API-Portal sends an email to the requester at the email address provided at registration. The requester simply clicks the link in the email to activate the user account and the user credentials. If the user logs in with a social account, API-Portal creates the user account when the user first logs in after clicking the link in the email.
- **Automatic registration.** If it's not essential to review each user registration request, you can use automatic registration, where API-Portal automatically processes all user registration requests or social log in requests upon receipt. With automatic registration, API-Portal creates the user account and notifies the requester by email that their account is activated and ready to use. The requester needs to log in to the portal using the email address or social account credentials that were provided at registration.

By default, API-Portal stores all user registration approvals and email notifications. Depending on the volume of user registration activity for your portal, you may want to periodically purge the approval and email notification entries. For more information, see "Advanced Configuration of API-Portal " on page 38.

Registering for webMethods API-Portal

You can sign up for a new account or log in using an existing social media account. For information on logging in using an existing social media account, refer to *webMethods API-Portal Administrator's Guide*. In the current version, login from Google+ and Facebook is supported.

To sign up for a new webMethods API-Portal account

1. Open webMethods API-Portal.
2. Click **Register**. The **Register** dialog opens.
3. Type the required data in the respective fields.
4. Click **Register**.

The registration request is sent. As soon as the request is processed, you will receive an e-mail containing a link. Click the link to activate your account. You are now able to log in to the webMethods API-Portal with your user name and password.

Configuring Approval Workflows for User Registration

Before you configure the registration process, you need to:

- **Customize email notification templates.** Use the default text provided or customize the text as needed. You can use pre-defined tokens as placeholders for specific information. For more information, see "Customizing Email Templates" on page 37.
- **Configure the SMTP mail server.** If you have not already done so, configure the SMTP server to enable API-Portal to send email notifications. For instructions, see "Configuring the SMTP Mail Server Connection for API-Portal using ACC" on page 19
- **Configure social accounts.** If you want to allow requesters to use a social account to access the portal, you need to configure that access. For instructions, see "User Registration in API-Portal with Social Login" on page 43.

When a requester clicks **Register** on the API-Portal landing page or clicks **Log in** and types the social login credentials, a registration request appears in the approver's Pending Approvals page. If a requester submits another request while the first request is still pending, API-Portal automatically rejects the second request. By default, users with the API Administrator role can approve or reject user registration requests. To have additional users who can participate in the approval process, add the users to the approver group. API-Portal provides a pre-defined approver group, **API User Registration Approvers**.

To use approval workflows you need to:

1. **Assign users as approvers by adding them to the approver group.** Determine who in your organization will review user registration requests, and approve or reject them. In the UMC, add users, who are to review and approve or reject registration requests, to the approver group.
2. **Define the workflow approval process.** In API-Portal, specify that you want to require approval for all registration requests, and select at which points in the approval process API-Portal has to send email notifications.

   API-Portal will automatically approve registration in the following situations:

   - The **API User Registration Approvers** group is not configured. For example, if the **API User Registration Approvers** group has been renamed or deleted.
   - There are no users in the **API User Registration Approvers** group.

Assigning Users to the Approver Group

API-Portal provides a pre-defined approver group, **API User Registration Approvers**. To specify which users receive user registration requests to approve or reject, add the users in the approver group.

Important: Do not change the name of the API user registration approver group. The name must be **API User Registration Approvers**.

To add or remove users as approvers

1. Start a web browser and go to the API-Portal UMC application at: `http://host:port/umc`
2. Click **User Management**, and then click **User groups**.
3. Click the **API User Registration Approvers** user group name, and then click **Associated Users**.
4. Click **Edit Assignment**. Add or remove users as approvers as follows:
   - **To add users:** Select the users you want to add from the **Available items** box, and then click **Add**. The selected users are transferred to the **Assigned items** box, and can now receive and approve user registration requests.
   - **To remove users:** Select the users you want to remove from the **Assigned items** box, and then click **Remove**. The selected users are moved to the **Available items** box, and can no longer receive registration requests.
   - **To add or remove all users at once:** Click **Add All** or **Remove All**.
5. Click **Save**.

Configuring the Approval Workflow

To configure your workflow approval, you need to specify at which approval step an email notification has to be sent, the subject and the content in the email.

To configure the approval workflow

1. In API-Portal, click **Administration > User Registration**.
2. Click **Approval required**.
3. Select the notifications that you want to send for each workflow step that you want to use. For each step, the **Subject** and **Content** fields contain the content that will be used for all notifications sent from API-Portal. Use the default content or change the content, as required. For more information, see "Email Notifications Templates and Tokens" on page 37.
4. Click **Apply**.

Working with Pending Approvals

Users who are assigned the API Administrator role and users included in the approver group use the Pending Approvals page in API-Portal to view and approve or reject registration requests.

For additional information about the Pending Approvals page, see the *webMethods API-Portal Online Help*.

To approve or reject pending approvals:

1. In API-Portal, select **Pending Approvals** from your user menu.
2. On the Pending Approvals page, review all the pending requests and do one of the following:
   - **Approve a request:** Select the request and click **Approve**. In the confirmation dialog, click **OK**.
   - **Reject a request:** Select the request and click **Reject**. In the confirmation dialog, provide a reason for the rejection, and then click **Reject**. The reason text will be included in the email notification sent to the requester.
   - **Approve or reject multiple requests:** Select all requests and click **Approve** or **Reject**.

Configuring Email Confirmation for User Registration

Upon receiving a registration request, API-Portal sends an email notification to the requester at the email address provided during the registration process or to the social account email address. The email contains an activation link that the requester clicks to access the portal. Email confirmation is selected by default.

To configure email confirmation:

1. In API-Portal, select **Administration > User Registration**.
2. Click **E-mail confirmation required**.
3. The **Subject** and **Content** fields contain the content that will be used for all notifications sent from API-Portal. Use the default content or change the content, as required. For more information, see "Email Notifications Templates and Tokens" on page 37.
4. Click **Apply**.

Configuring Automatic User Registration

With automatic registration, API-Portal automatically processes and approves all registration requests and social logins upon receipt. API-Portal notifies the user by email that their account is activated and ready to use. The user needs to log in to the portal using the email address or social account credentials they registered with.

To automatically process user registrations :

1. In API-Portal, select **Administration > Configuration > User Registration > Automatic registration**.
2. The **Subject** and **Content** fields contain the content that will be used for all notifications sent from API-Portal. Use the default content or change the content, as required. For more information, see "Email Notifications Templates and Tokens" on page 37.
3. Click **Apply**.

View and Manage Users

After a user is approved, a user account is created for the user in the UMC. User account details are available from the Profiles link after logging in to API-Portal.

- If the user registered with an email address and created a password, API-Portal creates the user account when the registration request is approved.
- If the user chooses to log in using a social account (Facebook, Google), API-Portal creates the user account after the user logs in for the first time as a registered user. Users must use their social account to change account details.

  Note: Social account passwords cannot be changed through the UMC.

From the UMC, an administrator can view and change the user details, such as the email address and phone number, groups the user is assigned to, and privileges assigned to the user. You can also delete a user and its associated account. For more information about user management, see the *webMethods API-Portal Online Help*.

Email Notifications Templates and Tokens

API-Portal provides default email templates that you can customize as needed. The templates are used in the Define user registration page in API-Portal. The templates also use tokens that you can use as placeholder information.

Customizing Email Templates

For each type of registration process on the Define registration process page, there are email templates provided that API-Portal uses to send email notifications to requesters about the status of their requests. You can change the default subject and content for any of the email templates.

To change the content of email templates

1. In API-Portal, click **Administration > User Registration**.
2. Click to select the type of registration process.
3. Edit the **Subject** and **Content** fields as required.
4. Click **Apply**.

Email Tokens

You can use any of the following tokens within your email templates. The tokens are valid in both the email **Subject** and **Content** fields. Before sending the email notification, API-Portal replaces the token with the corresponding value.

Token	Will be replaced with...
`@approver.name`	The name of the person who approved the request.
`@requester.name`	The name of the user who submitted the registration request.
`@comment`	When used in an approval workflow, this token will be replaced with the reason the approver entered when rejecting the request.
`@api-portal.url`	A link to the URL of the API-Portal that the user is registered to use.
`@activation.token`	The access token the user needs to access API-Portal. This token is valid only in email confirmation.

Advanced Configuration of API-Portal

Some user registration parameters are configurable by a REST service. To configure user registration parameters using a REST service, you must have API Administrator privileges. Any REST client tool can be used to invoke the mentioned APIs.

REST Resource	Parameters	Description
Active user approver group	Endpoint: `http://`*`host:port`*`/abs/apirepository/configurations/com.aris.umc.apiportal.useronboarding.approval.approvergroup/` Method: POST Request Body: *New approver group name*	Create an Active new User Registration Approvers group.: Sample Request: Endpoint: `http://`*`your serverg`*`/abs/apirepository/configurations/com.aris.umc.apiportal.useronboarding.approval.approvergroup/` Method: POST Request Body: Approvers
Onboarding request token	Endpoint: `http://`*`host:port`*`/abs/apirepository/configurations/com.aris.umc.apiportal.useronboarding.validateemail.token.ttl/` Method: POST Request Body: *New expiration period in minutes*	REST Service to change the expiration period of the link generated during email confirmation workflow. The default value is 30 minutes. Sample Request: Endpoint: `http://`*`your server`*`/abs/apirepository/configurations/com.aris.umc.apiportal.useronboarding.validateemail.token.ttl/` Method: POST Request Body: 60
Data purging	Endpoint: `http://`*`host:port`*`/abs/apirepository/configur`	REST Service to modify the default purging behavior for the approval objects. The default value is **true**. If the value is set

REST Resource	Parameters	Description
	`ations/ com.aris.umc.apiportal.useronboarding.approval.purge/` Method: POST Request Body: `true|false` Endpoint: `true| http://host:port/abs/apirepository/purge/ false` Method: DELETE Sample Request Payload: {"purgeSettings": [{"objectType":"User", "until":"2014-11-29T00:00:00"}, {"objectType": "Approval", "until":"2014-11-29T00:00:00"}]} Expected Response code is 204	to false, the API Administrator must perform manual purging. Metadata for pending approvals will be deleted after the requester is approved or rejected. By this time, the token, which we see in the URL, will also be deleted. API-Portal cannot differentiate between an invalid token (any token entered by the requester) and a valid, but expired, token (which is deleted by the first request). Sample Request: Endpoint: `http://`*`your server`*`/abs/apirepository/configurations/ com.aris.umc.apiportal.useronboarding.approval.purge/` Method: POST Request Body: false

The expected HTTP response code for all REST resources listed in the table is 202.

Configuring API-Portal with External Databases

API-Portal by default uses Postgres database. API-Portal can be configured with the following external databases:

- Microsoft SQL® Server 2012
- Microsoft SQL® Server 2014
- Oracle Database 11g
- Oracle Database 12c

Configure API-Portal with Microsoft® SQL Server 2012 or Microsoft® SQL Server 2014

To customize the API-Portal with Microsoft® SQL Server 2012 or Microsoft® SQL Server 2014 you need the following components:

- An operating Microsoft® SQL Server 2012 or Microsoft® SQL Server 2014 database.
- The Microsoft® JDBC Driver sqljdbc4.jar. You can download this driver from the Microsoft Web Site to a directory of your choice.
- SQL scripts and all additional files. These scripts can be downloaded from the ARIS Download Center.

The SQL scripts creates a database and necessary database objects required by the API-Portal components.

1. Go to the directory *Software AG Installation Directory* \API_Portal\Add-ons \DatabaseScripts\Design&ConnectServer\mssql.
2. Configure `envset.bat`
   a. Log on remotely onto the system where the database is installed and copy the database scripts to that system.

      In the Design&ConnectServer directory you can find the following files:

      - `create_schema_for_tenant.bat`
      - `envset.bat`
      - `fix_collation_for_tenant.bat`
      - `inst.bat`

   b. Edit the following lines in the envset.bat file:

      SET MSSQL_SAG_MSSQL_SERVER_NAME=*DB Server Name*

      SET MSSQL_SAG_MSSQL_LOGIN_NAME=*DB User Name*

      SET MSSQL_SAG_DATABASE_NAME=*Name of the database*

      SET MSSQL_SAG_FILEGROUP_FILE_DIR=*Location to store the file groups*
3. Run the database scripts. Before running the database scripts ensure that the Microsoft SQL Server client (sqlcmd) is available in the command prompt. Execute the `inst.bat` file. If you need additional tenants please use the `create_schema_for_tenant.bat` file.
4. Add the JDBC drivers to API-Portal classpath.
   a. Start API-Portal Cloud Controller.

   b. Stop all runnables except zoo_m.

   c. In ACC, type `enhance apiportalbundle_m with commonsClasspath local file "`*`sqljdbc4.jar location`*`"`

5. Register the external service database.

   a. In ACC, type `register external service db url="jdbc:sqlserver://`*`servername`*`:`*`port`*`;DatabaseName=`*`databasename`*`" driverClassName="com.microsoft.sqlserver.jdbc.SQLServerDriver" username="`*`username`*`" password="`*`password`*`" maxIdle=15 maxActive=300 maxWait=10000 removeAbandoned=false removeAbandonedTimeout=600 logAbandoned=true defaultAutoCommit=false rollbackOnReturn=true host=`*`servername`*` jmxEnabled=true`

      An external service identifier will be returned once the above command is executed, for example it returns the service id as db0000000000.

   b. Execute the following command from ACC console to assign the service to the tenant default: `assign tenant default to service db0000000000 com.aris.cip.db.schema=CIP_DEFAULTassign tenant master to service db0000000000 com.aris.cip.db.schema=CIP_MASTER`

6. Start all runnables.

You configured API-Portal with a Microsoft® SQL Server 2012 or Microsoft® SQL Server 2014 database.

Configure API-Portal with an Oracle Database

To customize the API-Portal with Oracle 11g or Oracle 12c database you need the following components:

- An operating Oracle database.
- The ojdbc6.jar driver. You can download this driver from the Oracle Web Site to a directory of your choice.
- SQL scripts and all additional files. These scripts can be downloaded from the ARIS Download Center.

The SQL scripts creates a database and necessary database objects required by the API-Portal components.

1. Go to the directory *Software AG Installation Directory* \API_Portal\Add-ons \DatabaseScripts\Design&ConnectServer\oracle.
2. Configure `envset.bat`

   a. Log on remotely onto the system where the database is installed and copy the database scripts to that system.

      In the Design&ConnectServer directory you can find the following files:

      - `cip_create_app_user.bat`

- `cip_create_schema_for_tenant.bat`
- `cip_update_schema_for_tenant.bat`
- `envset.bat`

b. Edit the following lines in the envset.bat file:

   SET CIP_ORA_BIN_PATH=*Path where sqlplus.exe can be found*

   SET TARGET_HOST=*DB Server Name*

   SET TARGET_PORT=*Port*

   SET TARGET_SERVICE_NAME= *Name of the database*

   SET CIP_INSTALL_USER=*User Name*

   SET CIP_INSTALL_PWD=*Password*

3. Run the database scripts.

   a. Run the script `cip_create_app_user.bat`. The application user is the oracle user connecting ARIS and the Oracle database.

   b. Type `cip_create_schema_for_tenant.bat aris_master` (mandatory)

   c. Type `cip_create_schema_for_tenant.bat aris_default` (mandatory). Two schemes are mandatory. One for the master tenant and one for the default tenant.

4. Add the JDBC drivers to API-Portal classpath.

   a. Start API-Portal Cloud Controller.

   b. Stop all runnables except zoo_m.

   c. In ACC type `enhance apiportalbundle_m with commonsClasspath local file "`*`ojdbc6.jar location`*`"`

5. Register the external service database.

   a. In ACC, type
   ```
   register external service db
   url="jdbc:oracle:thin:@servername:port:databasename"
   driverClassName=oracle.jdbc.OracleDriver jmxEnabled=true
   maxActive=100 maxIdle=15 logAbandoned=true rollbackOnReturn=true
   maxWait=10000 removeAbandoned=false defaultAutoCommit=false
   username=User Name password=Paswordhost=servername
   ```

   An external service identifier will be returned once the above command is executed, for example it returns the service id as db0000000000.

   b. Execute the following command from ACC console to assign the service to the tenant default:
   ```
   assign tenant default to service db0000000000
   com.aris.cip.db.schema=default schemaassign tenant master to
   service db0000000000 com.aris.cip.db.schema=master schema
   ```

6. Start all runnables.

You configured API-Portal with an Oracle 11 g or Oracle 12c database.

User Registration in API-Portal with Social Login

By default, API-Portal asks new users to register by providing a valid email address and a password. Upon approval, the user logs in to the portal using the email address and password. But you can also enable users to access the portal through a social login. Giving users access with their existing Google or Facebook account means they do not have to register or remember another set of credentials—they simply log in to the portal using their social account. API-Portal authenticates a user by accessing their social account.

Social login is a form of single sign-on using existing login information from a social network to sign in to a third-party application. Before an application can access private data of a social media user, it must obtain an access token that grants access to the OAuth provider API.

Social login works with all API-Portal registration approval processes. After being approved or clicking on an email confirmation link, users can access the portal. Users who are rejected or who do not have a valid email confirmation are denied access.

When you allow social login to API-Portal:

- At the user's first login, API-Portal stores the user's social login information, if authorized by the user.
- Users who access the portal with their social account will not be able to change their user profile information or password from the API-Portal Profiles link. All user profile fields, with the exception of the **Language** field, are read only, and there is no password change link. Instead users need to go to their social account and make changes there.
- Users can delete their API-Portal account from the API-Portal Profiles link.
- Dashboards in API-Portal can capture and track which social app users access the portal with.

After access with a social account is configured, valid users will see a login dialog where they can sign in to API-Portal with their social credentials if they are not already logged in to their social account.

What is OAuth?

OAuth is a standard for authorization that enables client applications to securely access resources on behalf of a resource owner. OAuth specifies processes that allow resource owners to authorize third-parties to access their resources without having to share credentials. OAuth allows an authorization server to issue access tokens to third party clients with the approval of a resource owner or end user. The client can then use the access token to access protected resources offered by the server. OAuth is most commonly used to allow users to log in to a web site using their Google, Facebook, Twitter, or any other social media account, without worrying about their credentials being compromised.

There are several ways to request an access token from the provider. The process used by API-Portal is described below.

1. The user clicks the **Sign in with *social_network*** link on the API-Portal login screen.
2. The application creates an authorization URL for the requested provider and redirects the user to that URL.
3. If the user is already logged in to the social network, he is redirected back to the API-Portal landing page where he is already logged in based on the approval process defined.
4. If the user is not already logged in, he is offered the possibility to log in at the OAuth provider. After logging in, the user is prompted to grant the permissions requested by API-Portal. This process is called *user consent*. If the user gives consent, the OAuth provider redirects the user back to API-Portal including a temporary code. If the user does not give consent, the OAuth provider returns an error.
5. After API-Portal obtains an access token, it uses the permitted API to determine the identity of the user, and creates a user account in the UMC, and finally logs in the user.

Configuring Google Login

To enable users to log in to the portal through Google, you must first create a Google app and then configure API-Portal to access Google account information to authenticate users.

Note: If you do not already have a Google account, you will need to create one. For complete information about Google sign-in, see the documentation available on developers.google.com.

To use Google for user registration and log in

1. Log in to developers.google.com and create an app for API-Portal registration.
   a. Go to the Google Developer Documentation.
   b. Log in using your Google account credentials.
   c. Click **Create Project**. Provide the project name and project id, and then click **Create**.
   d. Click on the link for your project. You will be taken to your project. In the left side navigation, go to **APIs & auth > Credentials**, and then click **Create new Client ID**.
   e. Select the application type **Web application**. The warning says that a product name needs to be set. Click **Configure consent screen** and set the product name.
   f. At the Create Client ID screen, the application requires that the domain be public. Provide the URL and the relevant port of your website in the **Authorized Javascript**

Origins field, and provide the redirect URL and the relevant port in the **Authorized Redirect URLs** field. The URLs must be HTTPS. Click **Create Client ID**.

g. In the left side navigation, go to **APIs & auth > Credentials**. You will find the client id and client secret on the right.

h. Make note of the client key and the client secret from the Google app, because you will need it in the next step.

i. In the left side navigation, go to **APIs & auth > APIs**. Search for Google+ API and enable it. You will then find the Google+ API in the **Enabled APIs** section.

2. Log in to UMC as Administrator, and configure the OAuth settings for the Google app as described in "OAuth Properties for Social Login" on page 46.
3. If you have not already done so, choose the level of approval needed, as described in "Registering Users in API-Portal " on page 32.

Configuring Facebook Login

To enable users to log in to the portal through Facebook, you must first create a Facebook app and then configure API-Portal to access Facebook account information to authenticate users.

Note: If you do not already have a Facebook ID, you will need to create one. For complete information about Facebook login, see the documentation available on developers.facebook.com.

To use Facebook for user registration and log in

1. Log in to developers.facebook.com and create an app for API-Portal registration.
   a. Go to: http://developers.facebook.com
   b. Log in using your Facebook account credentials.
   c. Go to **Apps > Add a New App**.
   d. On the Add a New App page, click **Website**.
   e. At the next screen, enter a name for your application and click **Create New Facebook App ID**.
   f. Choose the category **Apps For Pages**.
   g. Scroll down. In the **Tell us about your website** section, provide the URL of your website. The website URL should be an HTTPS URL and include the corresponding port.
   h. Click **Skip Quick Start**. This will create your Facebook app and display the Dashboard for your app.

i. Go to the Settings page by clicking the **Settings** link in the left side navigation. Provide a valid contact email here. This is required to make your application public to all users.

j. Go to the Status & Review page by clicking the **Status & Review** link in the left side navigation.

k. Make your application public by toggling the button from **No** to **Yes**. If the application is not made public, only the developer and the test users of the application will be allowed to log in using the app.

l. Make note of the client key (**App ID**) and the **App Secret**, because you will need it in the next step.

2. Log in to UMC as Administrator, and configure the OAuth settings for the Facebook app, as described in "OAuth Properties for Social Login" on page 46.
3. If you have not already done so, choose the level of approval needed, as described in "Registering Users in API-Portal " on page 32.

OAuth Properties for Social Login

If you are using social logins to provide access to your API-Portal, you need to configure OAuth settings in UMC to authorize the portal to work with the social apps. Log in to UMC as Administrator, and then click the Configuration tab. Select **OAuth** from the drop-down box to show only the OAuth properties. Set the properties as described below.

com.aris.umc.oauth

com.aris.umc.oauth.active

Indicates whether OAuth is used for authenticating portal access from social logins. Set the value to `true` to enable users to log in with their social account. Set the value to `false` to disable social login, and restrict access to valid users with an account in the UMC. The default is `false`.

com.aris.umc.oauth.api.keys

A comma-separated list of API keys obtained from each social app provider used for login. The order of the values specified in this property should match the order of the values specified in the com.aris.umc.oauth.providers property.

com.aris.umc.oauth.api.secrets

A comma-separated list of API secrets obtained from each social app provider used for login. The order of the values specified in this property should match the order of the values specified in the com.aris.umc.oauth.providers property.

com.aris.umc.oauth.debug

Specifies whether debug level output is provided for OAuth operations. Set the value to `true` to enable detailed debug output. Set the value to `false` to disable debug output, and not provide detailed information. The default is `false`.

com.aris.umc.oauth.providers
A comma-separated list of OAuth providers for each social app used for login. Both values are optional. The list of providers specified here defines how many login possibilities are displayed. If, e.g. only Google is configured, then the login page will show **Login with Google** only.

com.aris.umc.oauth.tenant
Specifies the default tenant used for social authentication. This value is read-only.

Sample Configuration for Google

In this example, we want to see full debug information for Google logins on the “default” tenant:

```
com.aris.umc.oauth.active=true
com.aris.umc.oauth.debug=true
com.aris.umc.oauth.providers=facebook,google
com.aris.umc.oauth.api.keys=facebook_key,google_key
com.aris.umc.oauth.api.secrets=facebook_secret_ID,google_secret_ID
com.aris.umc.oauth.tenant= default
```

Removing Social Login

If you no longer want to allow social login from any social account, you need to disable that access in the UMC. Doing so means that users can only register for an account with a valid email address and password.

To remove social registration and login access

1. Log in to UMC as Administrator, and disable OAuth as follows:
   a. Go to the API-Portal UMC application at: http://*host*:*port*/umc, and log in with your administrator credentials.
   b. Click the **Configuration** tab.
   c. Set the `com.aris.umc.oauth.active` property to `false`.
2. Log out from the UMC.

Connecting to Specific Servers

If you need to connect to specific servers instead of going through a proxy server, set the `apiportal.proxy.noProxyHosts` parameter to list those servers.

You can find this parameter in the apiportal.properties file. The file is located in the *Software AG_directory*\API-Portal\server\bin\work\work_apiportalbundle_m\base\webapps\abs\WEB-INF\classes\com\aris\modeling\server\webapp\apiportal\configuration directory.

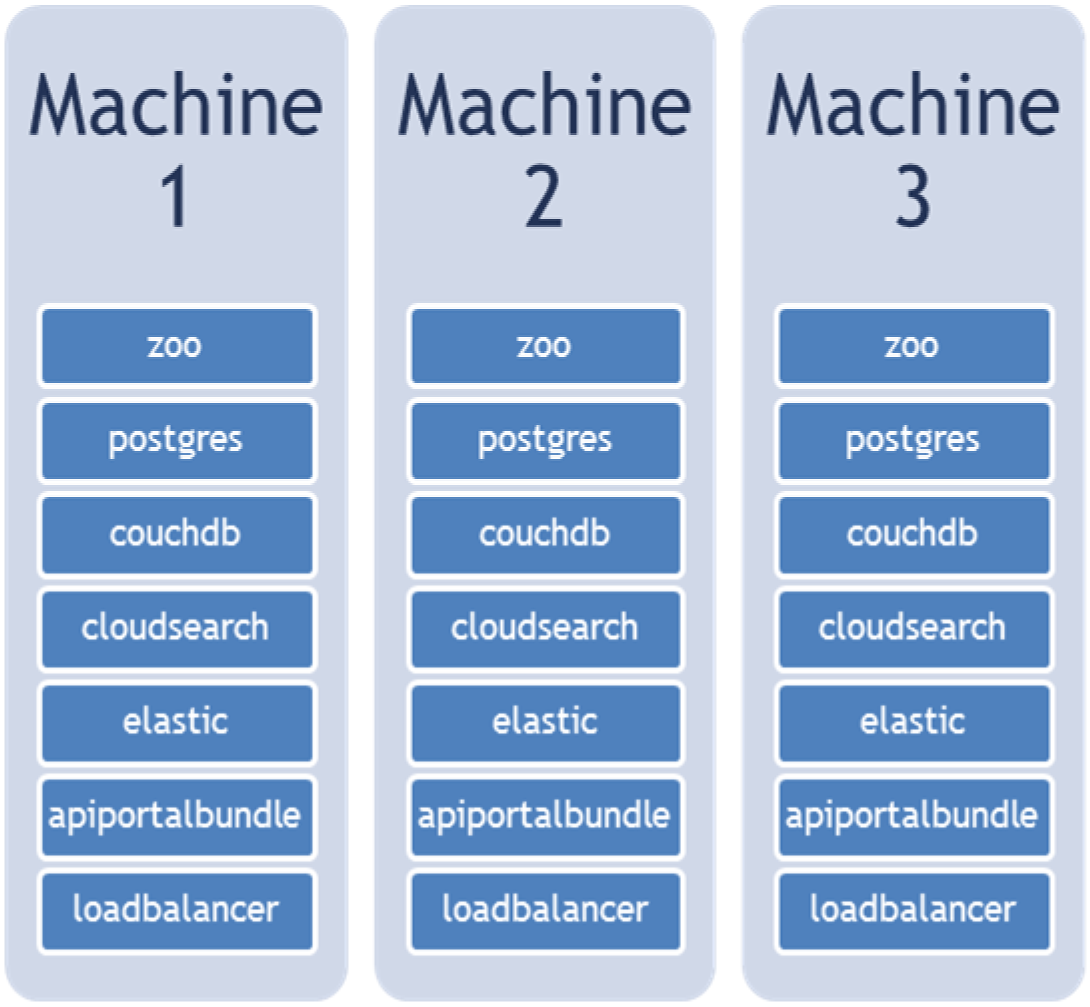
High Availability setup in API-Portal

The High Availability(HA) set up in API-Portal ensures continuous operation in cases of a fail-over scenario where up to one runnable of each kind or a whole machine becomes unavailable.

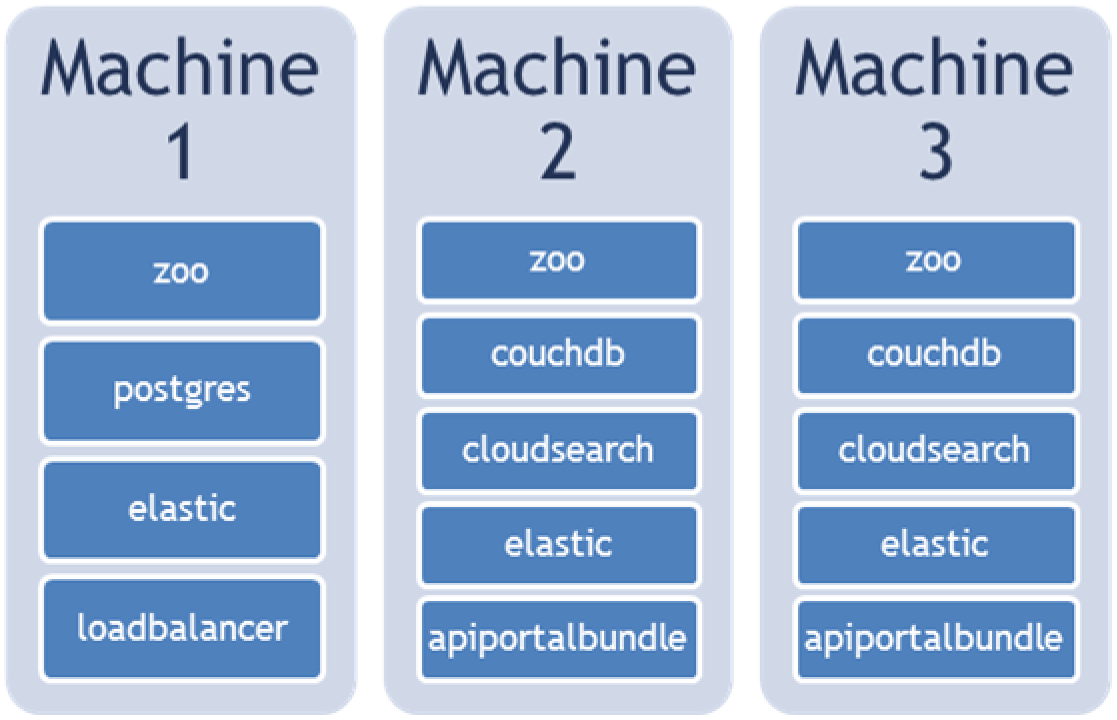
Prerequisites:

- A minimum of three machines with API-Portal set up on each of them.
- Switch-off firewalls or open appropriate ports on each machine so that they can access all components on the other machines.

On installing API-Portal 9.10, the runnables are installed and are visible in the ARIS Cloud Controller (ACC). The figure below depicts a general 3-machine API-Portal setup with the runnables installed.



The figure below depicts a 3-machine HA set up for API-Portal and the distribution of the runnables there after.



Machine 2 and 3 are configured the same way. Either when the full machine goes down or the runnables zoo and elastic stop working, with the runnables on machine 2 and 3 running properly, the system is still in the operation mode.

Setting up API-Portal HA setup

This section gives a detailed procedure for setting up the HA setup for API-Portal and test whether it is working properly.

Prerequsites:

Install API-Portal 9.10 but do not start any runnables.

1. Create a 3-node environment.
    a. On machine1, create a nodelist.pt file in the folder C:\SoftwareAG\API_Portal \server which contains the following lines:

        add node n1 machine1 @18004 Clous g3h31m

        add node n2 machine2 @18004 Clous g3h31m

        add node n3 machine3 @18004 Clous g3h31m

        set current node n1
    b. Replace machine1, machine2 and machine3 with the names or IP addresses of your machines.
    c. Run the following command:

```
C:\SoftwareAG\API_Portal\server>acc\acc.bat -n
C:\SoftwareAG\API_Portal\server\nodelist.pt -c
C:\SoftwareAG\API_Portal\server\generated.apptypes.cfg
```

This creates an ensemble between the instances in the cluster.

d. To view the 3-node cluster in ACC, run the command:

```
ACC+ n1>list nodes
```

the 3-node cluster has all nodes listed listening on port 18002 using REST services as follows:

n1 : machine1 (18004) OK

n2 : machine2 (18004) OK

n3 : machine3 (18004) OK

2. Cleanup the unnecessary runnables by running the following commands, in ACC, to deconfigure the runnables for all the 3 nodes:

```
ACC+ n1>on n1 deconfigure zoo_m
ACC+ n1>on n1 deconfigure couchdb_m
ACC+ n1>on n1 deconfigure cloudsearch_m
ACC+ n1>on n1 deconfigure apiportalbundle_m
ACC+ n1>on n2 deconfigure zoo_m
ACC+ n1>on n2 deconfigure postgres_m
ACC+ n1>on n2 deconfigure loadbalancer_m
ACC+ n1>on n3 deconfigure zoo_m
ACC+ n1>on n3 deconfigure postgres_m
ACC+ n1>on n3 deconfigure loadbalancer_m
```

3. Create a zookeeper cluster in ACC by running the following commands:

```
ACC+ n1>on n1 add zk
ACC+ n1>on n2 add zk
ACC+ n1>on n3 add zk
ACC+ n1>commit zk changes
```

You can view the following configuration using the ACC command after you start the zoo runnables:

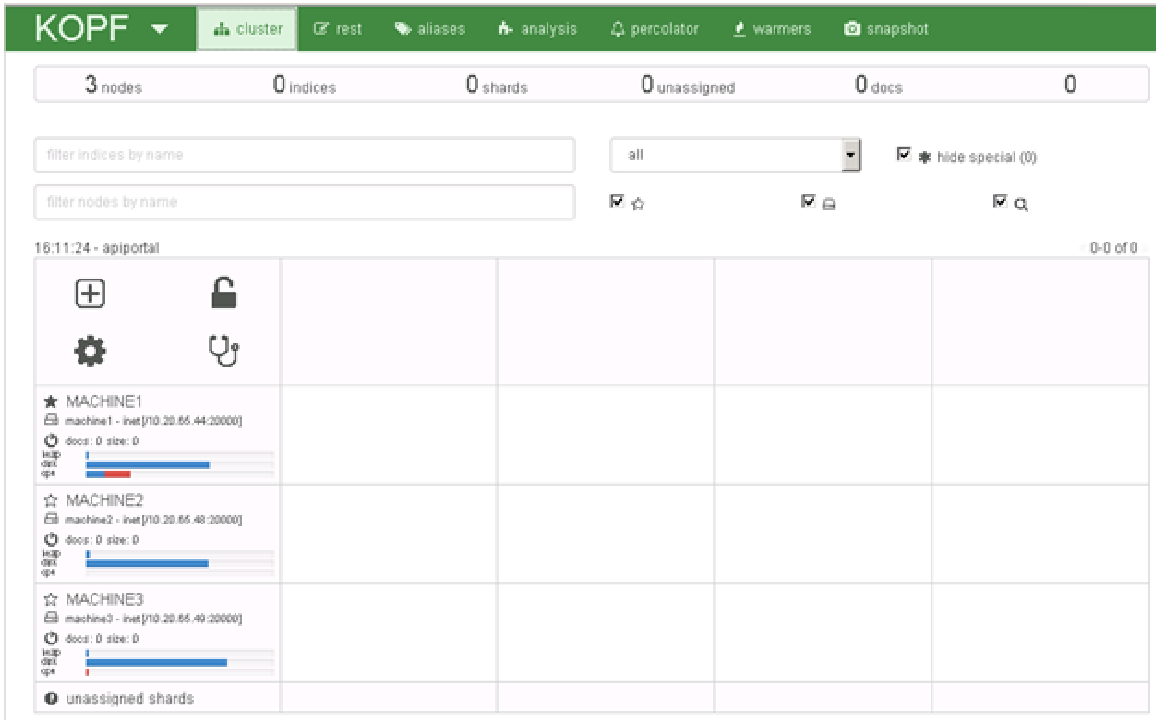
```
ACC+ n1>list zk instances
3 Zookeeper instances:
Node InstID MyID State     Cl Port Port A Port B Type
n1   zoo0   1    STARTED   2181    2888   3888   Master
n2   zoo0   2    STARTED   2181    2888   3888   Master
n3   zoo0   3    STARTED   2181    2888   3888   Master
```

4. Reconfigure the three elasticsearch runnables to form a cluster through ACC by running the following commands:

```
ACC+ n1>on n1 reconfigure elastic_m +ELASTICSEARCH.cluster.name = apiportal
+ELASTICSEARCH.discovery.zen.minimum_master_nodes=2 -zookeeper.connect.string
+ELASTICSEARCH.node.local = false +ELASTICSEARCH.index.number_of_replicas=1
-ELASTICSEARCH.sonian.elasticsearch.zookeeper.client.host
ACC+ n1>on n2 reconfigure elastic_m +ELASTICSEARCH.cluster.name = apiportal
+ELASTICSEARCH.discovery.zen.minimum_master_nodes=2 -zookeeper.connect.string
+ELASTICSEARCH.node.local = false +ELASTICSEARCH.index.number_of_replicas=1
-ELASTICSEARCH.sonian.elasticsearch.zookeeper.client.host
ACC+ n1>on n3 reconfigure elastic_m +ELASTICSEARCH.cluster.name = apiportal
+ELASTICSEARCH.discovery.zen.minimum_master_nodes=2 -zookeeper.connect.string
+ELASTICSEARCH.node.local = false +ELASTICSEARCH.index.number_of_replicas=1
```

```
-ELASTICSEARCH.sonian.elasticsearch.zookeeper.client.host
```

To see whether everything is working fine view the cluster at http://<*machine name*>/_plugin/kopf. You can see that the elasticsearch cluster consists of three nodes, that the cluster name is apiportal and that machine1 is the master node, indicated by a solid star as shown.



5. To reconfigure the PostgreSQL database on n1 so that it knows about all zookeeper cluster members and accepts connections from all locations:

   ```
   reconfigure postgres_m -zookeeper.connect.string +postgresql.listen_addresses = "'*'"
   ```

6. Define two cloudsearch instances, onthe nodes n2 and n3, where each one belongs to a different data center:

   ```
   reconfigure cloudsearch_m -zookeeper.connect.string
   +zookeeper.application.instance.datacenter = n2
   reconfigure cloudsearch_m -zookeeper.connect.string
   +zookeeper.application.instance.datacenter = n3
   ```

7. Reconfigure the couchdb runnable on the nodes n2 and n3 as follows:

   ```
   reconfigure couchdb_m -zookeeper.connect.string
   ```

   You can check if the couchdb runnable works by using the URLs http://<*machine name*>/_utils/fauxton/ or http://<*machine name*>/_utils/fauxton/

8. Reconfigure the apiportalbundle runnable on the nodes n2 and n3 as follows:

   ```
   reconfigure apiportalbundle_m -zookeeper.connect.string
   ```

9. Reconfigure the loadbalancer on n1 to point to all three zookeeper cluster members as follows:

   ```
   reconfigure loadbalancer_m -zookeeper.connect.string
   ```

10. Change the startup order of the runnables by running the following commands:

```
ACC+ n1>on n1 set runnable.order = "zoo0 < (elastic_m, postgres_m) < loadbalancer_m"
ACC+ n1>on n2 set runnable.order = "zoo0 < (elastic_m, couchdb_m) < cloudsearch_m
< apiportalbundle_m"
ACC+ n1>on n3 set runnable.order = "zoo0 < (elastic_m, couchdb_m) < cloudsearch_m
< apiportalbundle_m"
```

11. Start all runnables using the startup script.

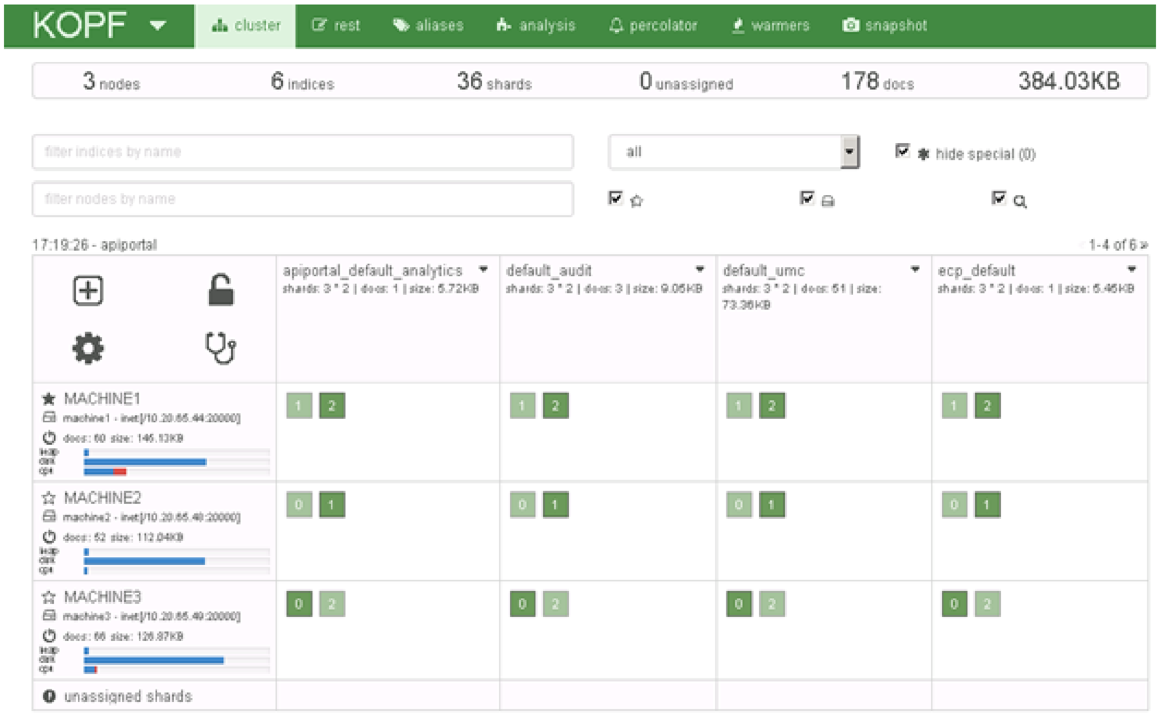
```
#
# start Zookeeper Ensemble
#
on n1 start zoo0
on n2 start zoo0
on n3 start zoo0
on n1 wait for STARTED of zoo0
on n2 wait for STARTED of zoo0
on n3 wait for STARTED of zoo0
#
# start ElasticSearch Cluster
#
on n1 start elastic_m
on n2 start elastic_m
on n3 start elastic_m
on n1 wait for STARTED of elastic_m
on n2 wait for STARTED of elastic_m
on n3 wait for STARTED of elastic_m
#
# start CouchDB
#
on n2 start couchdb_m
on n3 start couchdb_m
on n2 wait for STARTED of couchdb_m
on n3 wait for STARTED of couchdb_m
#
# start PostgreSQL Database
#
on n1 start postgres_m
on n1 wait for STARTED of postgres_m
#
# start CloudSearch
#
on n2 start cloudsearch_m
on n3 start cloudsearch_m
on n2 wait for STARTED of cloudsearch_m
on n3 wait for STARTED of cloudsearch_m
#
# start API-Portal Bundle
#
on n2 start apiportalbundle_m
on n3 start apiportalbundle_m
on n2 wait for STARTED of apiportalbundle_m
on n3 wait for STARTED of apiportalbundle_m
#
# finally, start loadbalancer
#
on n1 start loadbalancer_m
on n1 wait for STARTED of loadbalancer_m
```

12. Ensure the HA setup is successfully running.

In the default installation, the User Management Component (UMC) can be accessed at http://<*machine name*> /umc, the ARIS Document Storage (ADS) at http://<*machine name*> /ads, the collaboration component at http://<*machine name*> /collaboration and the API-Portal at http:/<*machine name*> .

Once you put some documents in ADS, the couchdb monitoring tool will show additional entries.

additional indices will be shown in elasticsearch. You can see that each index is split in 3 parts, called shards, which are replicated once and distributed over the three available nodes as shown. If one node goes offline, the system can still up the complete index making it a fail-over system.



3 nodes | 6 indices | 36 shards | 0 unassigned | 178 docs | 384.03KB

17:19:26 - apiportal

1-4 of 6 »

	apiportal_default_analytics shards: 3 * 2 | docs: 1 | size: 5.72KB	default_audit shards: 3 * 2 | docs: 3 | size: 9.05KB	default_umc shards: 3 * 2 | docs: 51 | size: 73.36KB	ecp_default shards: 3 * 2 | docs: 1 | size: 5.46KB
MACHINE1 machine1 - inet[/10.20.65.44:20000] docs: 60 size: 146.13KB heap disk cpu	1 2	1 2	1 2	1 2
MACHINE2 machine2 - inet[/10.20.65.40:20000] docs: 52 size: 112.04KB heap disk cpu	0 1	0 1	0 1	0 1
MACHINE3 machine3 - inet[/10.20.65.49:20000] docs: 66 size: 126.87KB heap disk cpu	0 2	0 2	0 2	0 2
unassigned shards				

Managing Tenants

API-Portal allows you to manage tenants for webMethods API-Portal installation type. You can create or delete different tenants. You should be a user in the master tenant and have both Tenant administrator and User administrator privileges to create or delete a tenant.

Creating tenants

You should be a user in the master tenant and have both Tenant administrator and User administrator privileges to create a tenant.

1. Create a tenant by logging into API-Portal Cloud Controller (ACC) and executing the following ACC commands.

```
acc > set acc cfg create.tenant.app.types=UMC,ECP
acc > set acc config create.tenant.operations=createTenant
acc > create tenant <<tenant.name>>
```

Note: <<tenant.name>> is the name of the new tenant being created. If the master tenant user password has been modified, you can supply the credentials as part of create tenant command as shown below.

```
acc > create tenant <<tenant.name>>
              master.tenant.user.name=<<master.tenant.user.name>>
              master.tenant.user.pwd=<<master.tenant.user.pwd>>
              tenant.user.pwd=<<tenant.user.pwd>>
              tenant.superuser.pwd=<<tenant.superuser.pwd>>
```

The tenant.user.pwd and tenant.superuser.pwd are the passwords that you want to set for the system and superuser accounts of the new tenant.

2. Import the license into the new tenant by executing the following command.

```
acc > invoke enhancement_importLicense on apiportalbundle_m tenant.name=
<<tenant.name>> local file enhancement.path="<<location to license file>>"
```

3. Prepare the new tenant for API-Portal installation by executing the following command.

```
acc> invoke prepareTenant on apiportalbundle_m tenant.name=<<tenant.name>>
isDemo=false
```

Note: If the tenant user name and passwords are changed, the default values can be overridden using parameters `tenant.user.name`/`tenant.user.pwd` as explained in step 1. The isDemo flag has to be set to true if you want to prepare a demo tenant similar to sagtours.

Note: If you face any problem while preparing tenant for API-Portal, you can lookup the logs located at <*SAG_InstallDir*> \API_Portal\server\bin\work \work_apiportalbundle_m \ apiPortalTenantProvisioning.log

Deleting Tenants

You should be a user in the master tenant and have both Tenant administrator and User administrator privileges to delete a tenant.

1. To delete a tenant execute the following command.

```
acc > set acc cfg delete.tenant.app.types=ECP,ABS,ADS,UMC
acc > delete tenant <<tenant.name>>
```

Note: If the tenant user name and passwords are changed, you can supply the credentials as part of delete tenant command as follows:

```
acc > set acc cfg delete.tenant.app.types=ECP,ABS,ADS,UMC
acc > delete tenant <<tenant.name>>
       master.tenant.user.name=<<master.tenant.user.name>>
       master.tenant.user.pwd=<<master.tenant.user.pwd>>
```

3 Managing API-Portal

Overview of Managing API-Portal

Managing API-Portal consists of starting and stopping API-Portal and the API-Portal Cloud Controller (ACC), verifying the status of all API-Portal components, and opening the API-Portal user interface in a browser to ensure that the portal looks and functions as intended.

What Happens When You Start API-Portal?

When API-Portal is installed from the Software AG Installer, the installer installs all the required API-Portal components (also known as runnables).

On Windows, the API-Portal Cloud Controller is installed as a Windows service and is set to start automatically when the machine on which it is installed initializes. You do not have to manually restart API-Portal following a machine restart.

On Linux systems, you need to manually start API-Portal.

When you start API-Portal for the first time after installation, the following items are created:

- A default tenant (which includes loading the initial tenant database)
- License file imports/assignments
- Default roles for API-Portal Administrator, API Provider, and API Consumer

Starting API-Portal (Windows)

All API-Portal components must be started before the API-Portal user interface can be opened in a browser. If any of the components are not started, your browser displays an error. For example:

```
403 Forbidden You don't have permission to access / on this server.
```

To start API-Portal on Windows

1. Start API-Portal automatically or manually by doing one of the following:

   - **Automatic:** Start API-Portal automatically using a Windows shortcut, as follows:

     Start > All Programs > Software AG > Start Servers > Start API-Portal n.n

   - **Manual:** Start API-Portal manually from the API-Portal Cloud Controller (ACC), as follows:

     Start > All Programs > Software AG > Administration > API-Portal Cloud Controller n.n

     In the command window, type `startall` to start all the components.

2. Verify the status of all API-Portal components. For details, see "Verifying the Status of API-Portal Components" on page 60.

Note: You can configure API-Portal in a way that all services start automatically. You can set the autostart mode with ACC>set autostart.mode=all to automatically start all the portal services once the Cloud Controller starts.

Starting API-Portal (Linux/UNIX)

All API-Portal components must be started before the API-Portal user interface can be opened in a browser. If any of the components are not started, your browser displays an error. For example:

```
403 Forbidden You don't have permission to access / on this server.
```

To start API-Portal on Linux/UNIX

1. Start the API-Portal Cloud Controller (ACC) by running the *Software AG_directory*/ `API_Portal/server/acc/acc.sh` script, specifying the following:

   - Machine on which the cloud agent is running (this will always be localhost) with the `-h` command line switch.

     Note: If the cloud agent is not running, you can start it by running the CloudAgentApp.sh script. Navigate to the <*Software AG directory*> / `API_Portal/server/bin` directory and run the command `./ CloudAgentApp.sh start`.

   - Username (default: Clous) of the cloud agent user with the `-u` command line switch.
   - Password (default: g3h31m) of the cloud agent user with the `-pwd` command line switch.

     You can also omit the password. If you do so, the ACC will prompt you for it.
   - Port information for the cloud agent with the `-p` command line switch.

   For example, to start the ACC installed in the directory Software_AG on localhost and using the default username and password, use the command:

   ```
   Software_AG/API_Portal/server/acc/acc.sh -h localhost -u Clous -pwd g3h31m
   -p 18003
   ```

2. At the ACC command prompt, type `startall` to start all components..
3. Verify the status of all API-Portal components. For details, see "Verifying the Status of API-Portal Components" on page 60.

Note: You can configure API-Portal in a way that all services start automatically. You can set the autostart mode with ACC>set autostart.mode=all to automatically start all the portal services once the Cloud Controller starts.

Stopping API-Portal (Windows)

To stop API-Portal on Windows

1. Stop API-Portal manually or automatically by doing one of the following:
   - **Automatic:** Stop API-Portal automatically using a Windows shortcut, as follows:

     Start > All Programs > Software AG > Start Servers > Stop API-Portal n.n
   - **Manual:** Stop API-Portal manually from the API-Portal Cloud Controller (ACC), as follows:

     Start > All Programs > Software AG > Administration > API-Portal Cloud Controller n.n

     In the command window, type `stopall`.
2. Verify the status of all API-Portal components. For details, see "Verifying the Status of API-Portal Components" on page 60.

Stopping API-Portal (Linux/UNIX)

To stop API-Portal on Linux/UNIX

1. Start the API-Portal Cloud Controller (ACC) by running the *Software AG_directory* / API_Portal/server/acc/acc.sh script, specifying the following:
   - Machine on which the cloud agent is running (this will always be localhost) with the `-h` command line switch.

     Note: If the cloud agent is not running, you can start it by running the CloudAgentApp.sh script. Navigate to the <*Software AG directory*>/API_Portal/server/bin directory and run the command `./CloudAgentApp.sh start`.
   - Username (default: Clous) of the cloud agent user with the `-u` command line switch
   - Password (default: g3h31m) of the cloud agent user with the `-pwd` command line switch

     You can also omit the password. If you do so, the ACC will prompt you for it.
   - Port information for the cloud agent with the `-p` command line switch.

   For example, to start the ACC installed in the directory Software_AG on localhost and using the default username and password, use the command:

```
Software_AG/API_Portal/server/acc/acc.sh -h localhost -u Clous -pwd g3h31m
-p 18003
```

2. At the ACC command prompt, type `stopall`.
3. Verify the status of all API-Portal components. For details, see "Verifying the Status of API-Portal Components" on page 60.

Opening the API-Portal User Interface in a Browser

To open the API-Portal user interface, open your browser and point it to the port on the host machine where the API-Portal instance is running. By default, API-Portal runs on port 18101.

All API-Portal components must be started to open the API-Portal user interface. If any of the components are not started, the browser displays an error. For example:

```
403 Forbidden You don't have permission to access / on this server.
```

To open the API-Portal user interface

1. Start your browser, and then point it to the host and port where API-Portal is running. For example:
   - If API-Portal is running on the default port on the same machine where you are running the API-Portal components, you would enter:

     ```
     http://localhost:18101
     ```
   - If the API-Portal components are running on port 4040 on a machine called QUICKSILVER, you would enter:

     ```
     http://QUICKSILVER:4040
     ```
2. When API-Portal loads in the browser, log in with your user name and password.

   If you are logging in to perform administration tasks, log in with your API-Portal Administrator credentials. The default values are:
   - User Name: system
   - Password: manager

API-Portal Components

API-Portal includes the following components (runnables). All must be in the STARTED state for the API-Portal user interface to come up in the browser.

Instance ID (runnable)	Description
zoo_m	Service registry
postgres_m	PostgreSQL database

Instance ID (runnable)	Description
couchdb_m	Document store persistence
cloudsearch_m	Intelligent API search capabilities for API-Portal
elastic_m	Search storage engine with search capabilities
apiportalbundle_m	API-Portal business logic
loadbalancer_m	Load balancer to distribute the request load across servers

For more information about the ACC, see "Verifying the Status of API-Portal Components" on page 60.

Verifying the Status of API-Portal Components

Use the ACC to manually manage API-Portal components. To monitor the status of all API-Portal components (runnables), use the `list` command.

In the following example, the response from the `list` command shows the component apiportalbundle_m with a status of STARTED.

```
ACC+ localhost>list

On node localhost 12 runnables are installed.

zoo_m          : STARTED (com.aris.runnables.zookeeper-run-prod-1.0.0-RC22-
Trunk-SNAPSHOT)

postgres_m     : STARTED (com.aris.runnables.PostgreSQL-run-prod-1.0.0-RC22-
Trunk-windows64-SNAPSHOT)

couchdb_m      : STARTED (com.aris.runnables.couchdb-run-prod-1.1.1-RC22-Trunk-
windows-SNAPSHOT)

cloudsearch_m  : STARTED (com.aris.cip.y-cloudsearch-run-prod-3.0.0-RC32-Trunk-
SNAPSHOT)

elastic_m      : STARTED (com.aris.runnables.elasticsearch-run-prod-1.0.0-RC22-
Trunk-SNAPSHOT)

apiportalbundle_m : STARTED (com.aris.apiportalbundle.y-apiportalbundle-run
-prod-12.0.0-RC29-Trunk-SNAPSHOT)

loadbalancer_m : STARTED (com.aris.runnables.httpd.httpd-run-prod-1.0.0-RC22-
Trunk-windows64-SNAPSHOT)
```

For more details about the API-Portal components and their status in the ACC, see " API-Portal Components" on page 59 and "Understanding API-Portal Component Status in ACC" on page 61.

Understanding API-Portal Component Status in ACC

When using the `list` command in the ACC, the command response lists components by their component name (runnable instance ID), followed by the state of the component. Possible states are as follows:

State	Meaning
UNKNOWN	The component state is not yet known. This state is normally only shown directly after the ACC service is (re-)started.
STOPPED	The component is currently not running.
STARTING	The component is starting, but this process is not yet complete.
STARTED	The component is running.
STOPPING	The component is stopping, but this process is not yet complete.
DOWN	The component has crashed. The agent will attempt to automatically restart the component momentarily.
FAILED	The component has crashed. The agent has given up trying to restart the component (or automatic restarting has been disabled).

If a component does not start properly, look at the logs of the component. The logs are available at *Software AG_directory*\API_Portal\server\bin\work \work_*component_name* \base\logs. If you need additional help to determine why components are not starting, contact Software AG Global Support.

Starting and Stopping API-Portal Components

API-Portal components can be started and stopped independently, but most components will not work on their own.

Start the ACC and enter these commands at the prompt:

Command	Description and Notes
`startall`	Starts all API-Portal components in the correct order.

Command	Description and Notes
	To monitor the progress, use the `list` command. If successful, the system responds: Successfully started all not yet running runnables on node localhost
`start` *`instanceId`*	Starts the specified API-Portal component. For example, the command to start the apiportalbundle_m component is: `start apiportalbundle_m` If successful, the system responds: Successfully started runnable apiportalbundle_m on node localhost If issues arise, ACC returns additional information. For example: Could not start runnable apiportalbundle_m on node localhost: Ant stop exited with 1
`stopall`	Stops all API-Portal components. To monitor the progress, use the `list` command. If successful, the system responds: `Successfully stopped all running runnables on node localhost.`

`stop` *`instanceId`*	Stops the specified API-Portal component.
	
	For example, the command to stop the apiportalbundle_m component is:
	
	`stop apiportalbundle_m`
	
	If successful, the system responds:
	
	`Successfully stopped runnable apiportalbundle_m on node localhost`
	
	If issues arise, ACC returns additional information. For example:
	
	`Could not stop runnable apiportalbundle_m on node localhost: Ant stop exited with 1`

4 Managing Users

Overview of Managing Users

API-Portal users are managed through CentraSite and API-Portal User Management Console (UMC). In CentraSite, user management centers more around allowing and controlling access to who can do certain things regarding assets and objects. In API-Portal, the focus is on who can publish the APIs to API-Portal and access API-Portal itself.

Users, Groups, Roles, and Permissions in CentraSite

Users identify individuals that are known to CentraSite. The roles and permissions that are assigned to users specify which operations they can perform and which registry objects they can access.

A *group* describes a set of CentraSite users. The group always belongs to exactly one organization, but it can contain users from different organizations. Groups are visible to all users.

In CentraSite, access control is enabled through a system of permissions and roles. The roles to which you belong and the permissions they include dictate what portions of the CentraSite user interface you see, what objects you can work with, and what operations you can perform.

- A *permission* enables a user to perform a specified operation on a specified object. Permissions also enable users to work with specified parts of the CentraSite user interface.
- A *role* is a collection of permissions that can be assigned to an individual user or a group.

For more information about users, groups, roles, and permissions, see *CentraSite Administrator's Guide*.

Predefined Roles in CentraSite

Roles in CentraSite can be assigned to users or groups defined in an organization. Users or groups who have roles receive all of the permissions associated with the roles. CentraSite provides some predefined roles for you to use. You can also create custom roles as needed.

For complete information about the predefined roles and creating custom roles in CentraSite, see *CentraSite Administrator's Guide*.

Predefined CentraSite Role	Description
CentraSite Administrator (CSA)	System-level role. This role includes all available permissions. Users in this role have all the permissions to manage the complete system (that is, users with “superuser” permissions). Only a CentraSite Administrator can change the permission for a system role. Additionally, the CSA can create and delete a system role. The CentraSite Administrator role cannot be modified or deleted.
Organization Administrator	Organization-level role. Users in this role can manage objects within a particular organization. This includes all child organizations of that organization. The Organization Administrator role cannot be modified or deleted.
Asset Provider	Organization-level role. Users (providers of the API) in this role can create and view assets within his or her organization. This role also has the ability to register users to consume assets. By default, all CentraSite users in an organization belong to this role.
Asset Consumer	Organization-level role. Users (consumers of the API) in this role can only see the set of APIs that providers have given them permission to view. Additionally, they can invoke (call) the APIs exposed to them. By default, all CentraSite users belong to this role, giving them permission to view the assets for their organization.
API Runtime Provider	Organization-level role. Users in this role can manage the API lifecycle, including API creation and documentation, API virtualization (publishing to Mediator), policy

Predefined CentraSite Role	Description
	enforcement, and API analytics. These users can also create and register API-Portal instances in CentraSite.
API Publisher	Organization-level role. Users in this role can publish run-time policies within an organization. This role also has the ability to publish APIs to gateways, such as to Mediator.
API-Portal Administrator	Organization-level role. Users in this role can manage API-Portal instances within an organization. This role is required to publish APIs to API-Portal.

For more information, see the section on getting started with API Management in *Working with the CentraSite Business UI*.

Other Users in CentraSite

The CentraSite user account to which you belong is either an API provider or an API consumer:

- **API providers** (owners of the API) who belong to the API Runtime Provider role in CentraSite are allowed to create and manage (view, edit, delete) all APIs within their organization. Additionally, these users can virtualize and publish APIs to the run-time layer.
- **API consumers** (users of the API) who belong to the Asset Consumer role in CentraSite can only work with the set of APIs that API providers have given them permission to view. Additionally, API consumers can invoke (call) the APIs exposed to them.

If your API management lifecycle includes approval workflows, some CentraSite users can be designated as an approver. Approvers whose user account includes a valid email address can receive an email message informing them that a request is awaiting their approval. Approvers can be in a designated approver group, and manage various aspects of the approval lifecycle, including review and approval or rejection of API access token requests.

In addition to these users, technical users exist to facilitate communication between systems and applications to ensure that credentials stay the same. A technical user is not associated with a specific user. Rather, a technical user represents a set of credentials and authorizations that is authenticated against an internal list of users, and not with an external set of authentications (for example, Active Directory or LDAP).

Predefined Roles in API-Portal

API-Portal provides predefined roles that you can assign to users and groups defined in an organization. You can also create custom roles as needed. Users or groups who have roles receive all permissions associated with the roles.

The following is a partial list of the roles and function privileges in API-Portal that apply to API users and administration. For a complete list of all function privileges that can be assigned, see *webMethods API-Portal Online Help*. For complete information about the predefined roles and creating custom roles in API-Portal, see the API-Portal User Management help, available from http://*API-Portal_host* :*port* /umc/help/en/handling/index.htm.

Predefined API-Portal Role	Description
API Administrator	Users with this role can start and stop API-Portal, manage API-Portal users, customize the API-Portal user interface to reflect the organization's own branding and look and feel, and switch configuration sets to customize views in API-Portal. API Administrators can create and remove private communities and can also manage all communities. API Administrators can add and remove users from a community and define community administrators or revoke the community administrator role from a user.
Guest	A guest user is an anonymous user who can browse and test the APIs available in API-Portal. When a guest user decides to use an API, the user must register and request an access token. **Important:** Do not change the credentials for the Guest user. Username/password credentials must remain guest/guest. If you change the guest user credentials in a user's properties page in the User Management Console, the API-Portal and/or its page components do not render correctly.

Other Users and Roles in API-Portal

The API-Portal user account to which you belong is either an API provider or an API consumer:

- **API provider.** Users in this group are allowed to publish APIs to API-Portal. These users are registered in CentraSite and APIs are published to API-Portal.
- **API consumer.** Users in this group are allowed to browse the portal (anonymously or as a registered user), register as an API-Portal user, request API access tokens, and test (evaluate) available APIs. API consumers can leave a community.

  After receiving an onboard approval email with an access token, registered developers can build their own applications, including the token in the application.

  If API consumers do not belong to any private community, they can only see public APIs.
- **Community administrator.** A community administrator is an administrator who can create, update, and delete a private community. Users in this group can add, update or remove users from a private community. A community administrator can define community administrators and revoke the community administrator role from a user.

In addition to these roles, technical users exist to facilitate communication between systems and applications to ensure that credentials stay the same. A technical user is not associated with a specific user. Rather, a technical user represents a set of credentials and authorizations that is authenticated against an internal list of users, and not with an external set of authentications (for example, Active Directory or LDAP). API-Portal administrators create technical users in API-Portal, and CentraSite administrators specify the technical user credentials when they register an API-Portal instance in CentraSite.

Note: As a best practice, Software AG recommends using a technical user in CentraSite and API-Portal to publish APIs from CentraSite to API-Portal.

5 Managing Communities

Overview of Communities

Communities in API-Portal define the exposure of APIs to consumers. A member of a private community can consume the APIs assigned to the private community. The APIs assigned to the public community are exposed to all users including unregistered users. A community can have one or more administrators that manage the members of a community.

API Administrators have the privileges to assign community administrators, add users to, and remove users from a community. API Providers assign APIs to specific communities. API Consumers have access to APIs depending on whether they belong to a specific community or not.

There are two types of communities - Private Communities and Public Communities.

Private Community

A *private community* is a group of users in API-Portal that users can join through an invitation. Community membership grants API consumers access to private APIs. When you assign an API to a community, it becomes visible to the community members. Communities can have one or more community administrators. APIs that are associated to a private community are only visible to users that belong to this community. APIs can be hidden from a community. APIs can be grouped in a collection of APIs. These API Groups can be assigned to specific communities. Each private community has one or more community administrators. Community administrators can add users to or remove users from the community.

To add existing users, the administrator needs the full user ID.

New users are invited to join API-Portal and the community of the inviter. New users invited to a private community are not added to the public community.

If users do not belong to any other community including public communities, they are removed from API-Portal. Access tokens belonging to the community are revoked from the removed user.

Public Community

The *public community* is a community any registered user can join. Each API-Portal instance features only one public community. By default, all APIs are assigned to the Public Community and they are visible to all users. This is the default community of a user joining without specifying a private community. Members of the public community can see the public APIs. Guests do not belong to the public community. APIs published to the API-Portal without a community are visible to the public community. Public APIs are visible to all communities including the public community.

For details on creating communities, adding members to or removing members from a community, adding APIs to or removing APIs from a community and defining a community administrator, see *webMethods API-Portal Online Help*

6 Managing API Assets

Planning for API Management

API-Portal functions as the key component of an effective API management solution, along with an instance of CentraSite. CentraSite uses API-Portal to securely publish APIs to external developers and partners and provides design-time governance capabilities to the APIs, whereas API-Portal allows developers to self-register, learn about these APIs, and use the APIs in their applications.

To prepare to manage the APIs that you plan to make available in API-Portal, consider the following questions:

- How many API-Portal instances you will need?
- Which organizations will API-Portal use?
- Which users in the organization will be using API-Portal to consume its published APIs?
- Which taxonomies and categories are needed to organize the APIs?

About API-Portal Assets

API metadata is stored in CentraSite for each registered API-Portal. For each API-Portal, there is an API-Portal object registered. The API-Portal object has the type of API-Portal, which is a virtual type of type Gateway. An API-Portal is associated with an Organization. Multiple API-Portal instances can share the same Organization.

When an API is published to API-Portal, CentraSite creates a relationship between the API and the API-Portal object in the repository. An API can be published to multiple API-Portal instances.

When an API is unpublished (removed) from API-Portal, its metadata is deleted from the repository, and the API is no longer available from API-Portal. The API remains in CentraSite.

When an access token for an API is requested, CentraSite creates an AccessToken object in the repository and deploys it to the Mediator for run-time policy enforcement. This object is associated with the API-Portal object through a requested relationship. For the requesting user, a User object is created within the Organization associated with API-Portal. The User object is created as soon as a user requests an access token for the first time. The User object is deleted when no more AccessTokens exist.

API-Portal Profile in CentraSite

The Service asset type contains a profile named **API-Portal Information** that includes attributes that are of use when CentraSite is integrated with API-Portal.

The **API-Portal Information** profile includes the following attributes:

- **API maturity status.** Defines the maturity of your API based on a customizable set of terms, allowing you to indicate the maturity status for the API. For example, Beta, Experimental, Test, or Production.
- **API grouping.** Groups the APIs by a definable business terminology to indicate the API usage. For example, CRM; Financial, Banking, and Insurance; Sales and Ordering.
- **API subscription terms.** Specifies the category of the key assigned to the client to access the API based on subscription plans. For example, Donationware, Flat Fee, Pay per use.
- **API icon.** Specifies the icon shown in API-Portal to represent the API.
- **Supported access token types.** Specifies the client authentication: API Key or OAuth2.
- **API Details on API-Portal(s).** List of URLs generated for API's details on the actual API-Portal to which the API is published to.

CentraSite provides a number of standard taxonomy categories that you can use to indicate the maturity status, grouping, and subscription terms for an API or you can create your own custom categories. For information about taxonomies and adding a category, see *CentraSite Administrator's Guide*.

API-Portal Information profile will be enabled for the Service asset types (Service, REST Service, and XML Service) and its variants (Virtual Service, Virtual REST Service, and Virtual XML Service) only when API-Portal Gateway is created.

Publishing and Unpublishing APIs to and from API-Portal

To publish APIs from CentraSite to API-Portal, you must have the privileges of a CentraSite Administrator or an API-Portal Administrator.

- If you have the privileges of a CentraSite Administrator, you can publish APIs that belong to any organization.
- If you have the privileges of an API-Portal Administrator for an organization, you can publish APIs that belong to that organization.

For more information about roles and permissions, see *CentraSite Administrator's Guide*.

API-Portal uses the following built-in actions for design/change-time policies to facilitate publishing and unpublishing APIs to and from API-Portal:

- **Publish to API-Portal.** This action bundles API metadata in the CentraSite repository, and then publishes those bundles to API-Portal. You can assign APIs to private communities when you publish them in CentraSite. For more information about assigning APIs while publishing them, see *Working with the CentraSite Business UI*
- **Unpublish from API-Portal.** This action removes the specified API metadata bundle from the API-Portal.

For more information about configuring the design/change-time policies that API-Portal uses, see *Working with the CentraSite Business UI*. For procedures for publishing and unpublishing APIs to and from API-Portal, see *Run-Time Governance with CentraSite*.

Handling Requests for API Access Tokens

When an API developer signs up and requests an API access token, API-Portal sends the API developer's sign-up or key request to CentraSite.

CentraSite generates a key, sends the key through an email to the requesting API developer, and deploys the key to Mediator.

After receiving the email, the API developer includes the key in the application so that, at run time, when the application communicates with the virtual service at the Mediator target endpoint, Mediator will call the native service. Mediator acts as a "key enforcer," validating the key contained in the application's header.

In addition to key validation, Mediator also collects metrics about the application and sends that data to CentraSite for later analysis and displaying the information on dashboards.

API-Portal Extension Points

API-Portal is configurable to work as a standalone component and can be integrated into any third-party API Management systems, which may not necessarily have Mediator or CentraSite present. You can configure API-Portal to work with different key management systems for access token retrieval functions for the APIs deployed in the third-party environments.

Providing API-Portal extension points when API-Portal is deployed in a third-party environment is done in the following stages:

- Managing third-party key management providers
- Managing Access Tokens

Managing Third-party Key Management Providers

API-Portal can depend on CentraSite or other third-party providers for Key management. Depending on the deployment model the configuration parameter **Is CentraSite based API Management Profile active?** has to be configured accordingly.

The default value of the **Is CentraSite based API Management Profile active?** parameter is **true**, which means that API-Portal relies on CentraSite for key management. This value has to be changed to **false** for key management to be handled by the third-party vendor. When this parameter is set to false, API-Portal sends key requests to a group of people for access token management. The API Administrator can assign the users to the UMC group **API Consumption Approvers** and emails will be sent out to all the users in this group for access token requests.

You can modify the property **Is CentraSite based API Management Profile active?** through the configuration page on the API-Portal user interface.

Modifying the property Is CentraSite based API Management Profile active? through the API-Portal User interface

For information on various configurable parameters see API-Portal Configurations page in the *webMethods API-Portal Online Help* .

1. Login to API-Portal as an Administrator.
2. Click **Administration** in the menu options displayed in the right top corner of the API-Portal window.
3. Select **Configuration > Configuration Settings**.

   The API-Portal Configurations page is displayed
4. Modify the **Is CentraSite based API Management Profile active?** parameter value to **false**.

Managing Access Tokens

If the **Is CentraSite based API Management Profile active?** is set to **false**, API-Portal, by default, displays the **Get Access Tokens** link in the API details page. The values entered in the Get Access Token dialog box will be sent in an email to the users in the **API Consumption Approvers** group.

The Get Access Token dialog box is fully customizable so that all the required information can be obtained from the users requesting the keys. For more details, see API-Portal Customization guide.

Once the key is approved and processed, the third party key management system should use the REST Service with endpoint /abs/apirepository/accesstokens/external with the values UserId, Properties (Key Value) to populate the access token details in API-Portal.

EndPoint: http://*<host>* :*<port>* /abs/apirepository/accesstokens/external

Method: POST

Content Type: application/json

Authorization: Basic Auth (Username/password of users in API Consumption Approvers group)

Input Schema (Payload)

```
{
  "user":" Id of the user who requested the token. Can be obtained from the
   email sent from API-Portal",
  "propertyList": [
    {
      "key": "applicationName",    //Mandatory Input
      "value": "App name"
    },
    {
      "key": "Access Token Type",    //Mandatory Input
      "value": "Value 1"
    },
    {
      "key": "Key 2",
      "value": "Value 2"
    }
    …
  ]
}
```

Sample Payload

```
{
"user":"4ee9c60c-c32e-3615-83c4-5dc6822c45de",
  "propertyList": [
    {
      "key": "applicationName",
      "value": "Weather App"
    },
    {
      "key": "Access Token Type",
      "value": "API Key"
    },
    {
      "key": "x-Api-Key",
      "value": "F46805559A46CBB236B34EAB761A8163 "
    },
    {
      "key": "How To Use",
      "value": "use this key in query parameter"
    },
   {
      "key": "Expiration Date",
      "value": "22-10-2016"
    }
  ]
}
```

Sample Response Payload

`672a9c60c-c3se-3115-13c4-5dc6822c45fd` (tokenId - to be stored by third-party system for future reference)

The **Access Tokens** page now displays the list of access tokens and each of these tokens will have an option to Revoke and Renew.

Revoking a token sends an email request with token id and user details to the configured provider. Upon successful deletion of the token, the provider must invoke "/abs/apirepository/accesstokens/external/tokenId". This deletes the token from API-Portal.

EndPoint: http://*<host>* :*<port>* /abs/apirepository/accesstokens/external/tokenId

Method: DELETE

Authorization: Basic Auth (Username/password of users in API Consumption Approvers group)

Renewing a token sends an email request with token id and user details to the configured provider. Upon successful renewal of the token, the provider must invoke PUT on "/abs/apirepository/accesstokens/external/tokenId". This renews the token in API-Portal.

EndPoint: http://*<host>* :*<port>* /abs/apirepository/accesstokens/external/tokenId

Method: PUT

Authorization: Basic Auth (Username/password of users in API Consumption Approvers group)

Input Schema (Payload)

```
{
  "user":" Id of the user who requested the token. Can be obtained from the
   email sent from API-Portal",
  "propertyList": [
    {
      "key": "applicationName",    //Mandatory Input
      "value": "App name"
    },
    {
      "key": "Access Token Type",    //Mandatory Input
      "value": "Value 1"
    },
    {
      "key": "Key 2",
      "value": "Value 2"
    }
    …
  ]
}
```

Sample Payload

```
{
"user":"4ee9c60c-c32e-3615-83c4-5dc6822c45de",
  "propertyList": [
    {
      "key": "applicationName",
      "value": "Weather App"
    },
    {
      "key": "Access Token Type",
      "value": "API Key"
    },
    {
      "key": "x-Api-Key",
      "value": "F46805559A46CBB236B34EAB761A8163 "
    },
    {
      "key": "How To Use",
      "value": ""use this key in query parameter"
    }
           {
  "key": "Expiration Date",
 "value": "22-10-2017"
```

```
                }
   ]
}
```

Note: In case of third party key management, the API Tryout will not enforce any access token restrictions except for BASIC Auth.

Note: For information on using the Get Access Token dialog and Access Tokens page details, see *webMethods API-Portal Online Help*.

7 Managing Data in API-Portal

Configuring Audit Logs

API-Portal provides a level of auditing that allows an API-Portal Administrator to see the state of the API-Portal and get information about events that occur.

Various events that can be audited are:

- Login attempts
- User Deletion
- API lifecycle
- Access token lifecycle
- Communications from API-Portal to CentraSite
- CentraSite instance lifecycle
- Purge logs

In addition to monitoring events through various audit logs the API-Portal administrator can configure the following as required on a tenant basis:

- Type of auditing events
- Purging of different types of logs

Different event types and their respective list properties that have to be configured to enable or disable the auditing of these logs is listed in the following table.

Event Type	List Property Value
Approvals	com.aris.umc.apiportal.useronboarding.approval.purge
Request Access Token	com.aris.apiportal.eventType.requestaccesstoken.log.enabled
Renew Access Token	com.aris.apiportal.eventType.renewaccesstoken.log.enabled
Revoke Access Token	com.aris.apiportal.eventType.revokeaccesstoken.log.enabled
Publish API	com.aris.apiportal.eventType.publish.log.enabled
Republish API	com.aris.apiportal.eventType.republish.log.enabled
Unpublish API	com.aris.apiportal.eventType.unpublish.log.enabled
Purge Log	com.aris.apiportal.eventType.purge.log.enabled

Event Type	List Property Value
User Deletion	com.aris.apiportal.eventType.csuserdeletion.log.enabled

You can configure the Audit logs to enable or disable the required types of auditing events such as login events, API lifecycle events, access token lifecycle, user deletion, and purge logs. These are enabled by default.

You can configure audit logs by invoking the REST service.

Endpoint: http://*<hostname>* :*<port>* /abs/apirepository/configurations/*<PropertyName>* /

Supported HTTP methods:

- GET - Display the current audit setting (enabled/disabled) for the given property.
- POST - Modify the audit setting for the given property. Message body to be set to false (to disable logging).

For example, to disable the audit logging for the publish property type:

Endpoint: http://*<hostname>* :*<port>* /abs//apirepository/configurations/com.aris.apiportal.eventType.publish.log.enabled/

HTTP Method: POST

Message body: false

For example, to enable the audit logging for the publish property type:

Endpoint:http://*<hostname>* :*<port>* /abs//apirepository/configurations/com.aris.apiportal.eventType.publish.log.enabled/

HTTP Method: POST

Message body: true

For example, to get the status of the properties:

Endpoint URL: http://*<hostname>* :*<port>* /abs/apirepository/configurations/com.aris.apiportal.eventType.publish.log.enabled/

HTTP Method: GET

Response:

```
{
propName: "com.aris.apiportal.eventType.publish.log.enabled"
propValue: "true"
}
```

Purging Logs

The process of systematically deleting unwanted data from the database is called purging. Each time you purge a log unit the corresponding purging log entry is created and can be viewed in API Audit log page in the API-Portal User Interface. For example, when you purge a certain set of analytics logs, the corresponding purging log entry is created and is displayed in the API Audit log page in the API-Portal User Interface. You can purge logs by setting the required date or duration per tenant in the API-Portal. There is only a single purge log entry that provides information about the type of logs that have been purged along with the date or duration.

You can purge logs in one of the following ways:

- "Purging logs by invoking a REST service" on page 82
- "Purging logs through the user interface" on page 82

The purge service supports the following types of log entries:

- Audit Logs - API and User audit logs
- Analytics - Page Views and other analytics data
- Approval - Objects of approved users
- Stale Users - Users who have not been activated through the email confirmation workflow
- API Analytics - Run-time analytics data for APIs

Purging logs by invoking a REST service

You can purge logs by invoking a REST service.

1. Endpoint: http://*<hostname>* :*<port>* /abs/apirepository/purge/
2. Method: Delete
3. Content-Type: application/json
4. Request Payload

```
{"purgeSettings":[{"objectType":"Audit Logs", "until":"2014-11-29T00:00:00"},
 {"objectType":"Analytics", "until":"2014-11-29T00:00:00"},{"objectType":
"Approval", "olderThan":"1m"}, {"objectType":"Stale Users", "olderThan":"1m"}]}
```

Purging logs through the user interface

As an Administrator, you can purge the following types - Approvals, Stale User data, Audit Events, Analytics, and API Analytics logs. You can apply filters using Date or Duration.

To purge logs from the API-Portal user interface:

1. Login to API-Portal as an Administrator.
2. Click **Administration** in the menu options displayed in the right top corner of the API-Portal window.

3. Click **Manage Logs**.
4. Click **Purge Logs**.
5. On the **Purge Logs** page, do one of the following:
   - Select **Until** and select a date until which you want the data to be purged. The rest of the data is retained.
   - Select **Duration** and type the duration value for which you want the data to be retained. The rest of the data is purged.
6. Select the type of data that needs to be purged.
7. Click **Purge**. A confirmation message is displayed.
8. Click **Yes**.

Back up and Restore Tenant Data

In this section, an administrator can back up or restore tenant data.

API-Portal is multi-tenant capable and each tenant administrator can manage their data. Generally, the tenant data comprises of

- APIs, Access Keys, and other related data
- Users, user groups, licenses, configurations, and so on
- API documents and files
- Collaboration data

Prerequisites for Backup and Restore in a Distributed Environment

The following points are to be considered if the API-Portal server is installed in a clustered high availability setup. If the API-Portal is a single node installation, then you can skip this section.

In a typical multi-node setup, the different parts of the application (runnables) are installed on the different nodes (machines). It is important to verify that all the nodes are up and accessible. All the nodes must be registered to the parent node and the API-Portal Cloud Controller (ACC) needs to know about your distributed environment. There are a couple of ways to register the different nodes to ACC.

Using the ACC REST Service

1. Post a http request to http://loadbalancer_host:loadbalancer_port/acc/rest/nodes/ with payload:

```
{
 "nodename": "<<hostname>>",
"hostname": "<<hostname>>",
"port": "<<acc_portno>>",
"username": "Clous",
"password": "g3h31m"
}
```

The user must be a member of infrastructure administrator tenant - the master tenant - with Tenant administrator functional privilege.

Your system is now ready for the backups or restores.

Using a Configuration File

1. Stop the component apiportalbundle_m through ACC using the `Stop apiportalbundle_m` command.
2. Edit the file in the path <*Installation directory*> \API-Portal\server\bin\work\apiportalbundle_m\base\webapps\abs\WEB-INF\config\spring\ext\apiportal-servlet.xml.
3. Add the new entry for the distributed ACC server host and port in the map.

```
<bean id="addNodeAction"
  class="com.aris.modeling.server.webapp.apiportal.startup.impl.AddNodeAction">
    <property name="nodeList">
        <map>
            <entry key="127.0.0.1" value="9001" />
            <entry key="localhost" value="18002" />
        </map>
    </property>
</bean>
```

4. Restart the component apiportalbundle_m via ACC using the `Start apiportalbundle_m` command.

Your system is now ready for the backups or restores.

Back up Tenant Data

The backup or restore operation requires that the logged in user is a member of the API-Administrator user group. Non API-Administrators will not be able to do backup or restore data.

To backup tenant data

1. Log in to API-Portal.
2. Click **Administration** from the menu options in the right top corner of the API-Portal window.
3. Click **Manage Data**.
4. Click **Backup and Restore**.
5. Select **Backup**.
6. Select the relevant options. The available options are **All**, **API-Portal objects**, and **Collaboration data**. You must select at least one option.

7. Click **Backup**.
8. Confirm your password.
9. Type the name of the backup file. The file extension will be **.acb**.

The following data are backed up:

- The **API-Portal Objects** option is used to backup API related metadata. API related documents, registered users, user groups, licenses and configuration properties.
- The **Collaboration data** option is used to backup API related collaboration data.

Restore Tenant Data

The backup or restore operation requires that the logged in user is member of the API-Administrator user group. Non API-Administrators will not be able to do backup or restore data.

Note: Restoring will overwrite the existing content in your API-Portal.

To restore a tenant proceed as follows. Depending on which is contained in your backup file, all data, API-Portal objects or Collaboration data are restored.

To restore a tenant

1. Log in to the API-Portal.
2. Click **Administration** from the menu options in the right top corner of the API-Portal window.
3. Click **Manage Data**.
4. Select **Restore**.
5. Click **Upload** and select the relevant backup file to be uploaded.
6. Click **Restore**.

A success message is displayed once the restore process is completed.

Troubleshooting Backup and Restore Failures

In case the back up or restore process fails, check whether all components are running. To do so, use API-Portal Cloud Controller (ACC).

To troubleshoot backup and restore failures

1. Check whether all components are running using ACC.
    a. To start ACC under a Windows operating system click **Start > All Programs > webMethods API-Portal > Administration > Start webMethods API-Portal Cloud Controller**.

To start ACC under a Linux operating system, run the **acc.sh** file. ACC is available if you have copied and installed the files **aris-acc_1.0.0-SNAPSHOT_amd64.deb** or **aris-cloud-agent-1.0.0-1.x86_64.rpm** depending on the Linux operating system.

b. Type **help** or **help <command>** to get information about the usage of the commands.

c. Type **list** to monitor the status of all components (runnables). This example shows ACC of a API-Portal Connect Server installation for a medium number of users.

All Components with their states are listed.

```
ACC+ arisserver>list
7 installed runnables.
apiportalbundle_m :STARTED com.aris.runnables.apiportal_m
                   -run-prod-1.0.0-RC17-Trunk-SNAPSHOT)
zoo_m           : STARTED (com.aris.runnables.zookeeper-
                  run-prod-1.0.0-RC17-Trunk-SNAPSHOT)
postgres_m      : STARTED (com.aris.runnables.PostgreSQL-
                  run-prod-1.0.0-RC17-Trunk-windows-SNAPSHOT)
couchdb_m       : STARTED (com.aris.runnables.couchdb-
                  run-prod-1.1.1-RC17-Trunk-windows-SNAPSHOT)
cloudsearch_m   : STARTED (com.aris.cip.y-cloudsearch-
                  run-prod-3.0.0-RC26-Trunk-SNAPSHOT)
elastic_m       : STARTED (com.aris.runnables.elasticsearch-
                  run-prod-1.0.0-RC17-Trunk-SNAPSHOT)
loadbalancer_m : STARTED (com.aris.runnables.httpd.httpd-
                  run-prod-1.0.0-RC17-Trunk-windows-SNAPSHOT)
ACC+ arisserver>
```

The status of all components represented by their instance IDs are listed. Possible states are:

- **UNKNOWN**

  The component state is not yet known. This state is shown directly after the agent was started.

- **STOPPED**

  The component is currently not running.

- **STARTING**

  The component is starting, but this process is not complete yet.

- **STARTED**

  The component is running.

- **STOPPING**

  The component is stopping, but this process is not complete yet.

- **DOWN**

  This component has and crashed. The agent will attempt to automatically restart the component momentarily.

- **FAILED**

  Component has crashed. The agent has given up trying to restart the component.

2. Perform the following preliminary task in a distributed environment using the ACC REST service

   Post a http request to http://loadbalancer_host:loadbalancer_port/acc/rest/nodes/ with payload:

   ```
   {
   "nodename": "<<hostname>>",
   "hostname": "<<hostname>>",
   "port": "<<acc_portno>>",
   "username": "Clous",
   "password": "g3h31m"
   ```

   Your system is now ready to start the backup or restore operation.

3. Start the backup or restore operation using a configuration file.

   a. Stop the component work_apiportalbundle_m through ACC using the **Stop work_apiportalbundle_m** command.

   b. Edit the file in the path <Installation directory>\API-Portal\server\bin\work\work_apiportalbundle_m\base\webapps\abs\WEB-INF\config\spring\ext\apiportal-servlet.xml.

   c. Add the new entry for the distributed ACC server host and port in the **bean** section.

   ```
   <bean id="addNodeAction"
   class="com.aris.modeling.server.webapp.apiportal.startup.impl.AddNodeAction">
   <property name="nodeList">
   <map>
   <entry key="127.0.0.1" value="9001" />
   <entry key="localhost" value="18002" />
   </map>
   </property>
   </bean>
   ```

   d. Restart the component work_apiportalbundle_m through ACC using the **Start work_apiportalbundle_m** command.

   Your system is now ready for the backup or restore operation.