Oracle® Retail Integration Cloud Service

Universal Service Mapper User Guide

G17863-01
October 2024

Oracle Retail Integration Cloud Service Universal Service Mapper User Guide,

G17863-01

Contents

Send Us Your Comments

Oracle® Universal Service Mapper User Guide

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document.

Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

> **Note:**
>
> Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the Online Documentation available on the Oracle Technology Network Web site. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: retail-doc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at http://www.oracle.com.

Preface

This document describes the Universal Service Mapper user interface. It provides step-by-step instructions to complete most tasks that can be performed through the user interface.

Audience

This document is for users and administrators of Oracle Retail Universal Service Mapper. This includes merchandisers, buyers, business analysts, and administrative personnel.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Customer Support

To contact Oracle Customer Support, access My Oracle Support at the following URL:

https://support.oracle.com

When contacting Customer Support, please provide the following:

- Product version and program/module name
- Functional and technical description of the problem (include business impact)
- Detailed step-by-step instructions to re-create
- Exact error message received
- Screen shots of each step you take

Improved Process for Oracle Retail Documentation Corrections

To more quickly address critical corrections to Oracle Retail documentation content, Oracle Retail documentation may be republished whenever a critical correction is needed. For critical corrections, the republication of an Oracle Retail document may at times not be attached to a numbered software release; instead, the Oracle Retail document will simply be replaced on the Oracle Technology Network Web site, or, in the case of Data Models, to the applicable My Oracle Support Documentation container where they reside.

This process will prevent delays in making critical corrections available to customers. For the customer, it means that before you begin installation, you must verify that you have the most recent version of the Oracle Retail documentation set. Oracle Retail documentation is available on the Oracle Technology Network at the following URL:

http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html

An updated version of the applicable Oracle Retail document is indicated by Oracle part number, as well as print date (month and year). An updated version uses the same part number, with a higher-numbered suffix. For example, part number E123456-02 is an updated version of a document with part number E123456-01.

If a more recent version of a document is available, that version supersedes all previous versions.

Oracle Retail Documentation on the Oracle Help Center (docs.oracle.com)

Oracle Retail product documentation is also available on the following Web site:

https://docs.oracle.com/en/industries/retail/index.html

(Data Model documents can be obtained through My Oracle Support.)

Conventions

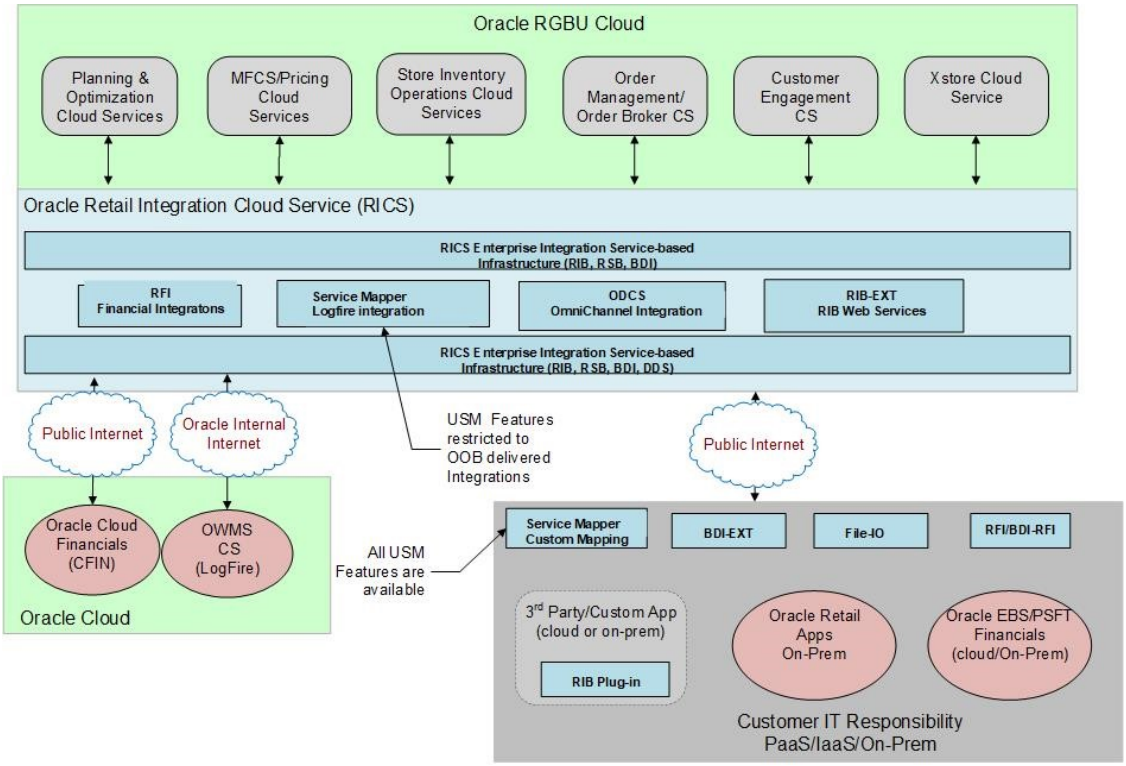
The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Documentation Note

Universal Service Mapper is one of the RTG Tools that is packaged with the RICS SaaS Cloud Service and the Retail Integration Suite for the 22.1.201.0 Release.



RICS USM

The RICS version of USM is deployed with a supported Out-Of-Box Integration, such as the Oracle Warehouse Management Service (LogFire) integration. The features available to customers are restricted to READ-ONLY and to pre-configured integration flows.

Retail Integration Suite's USM

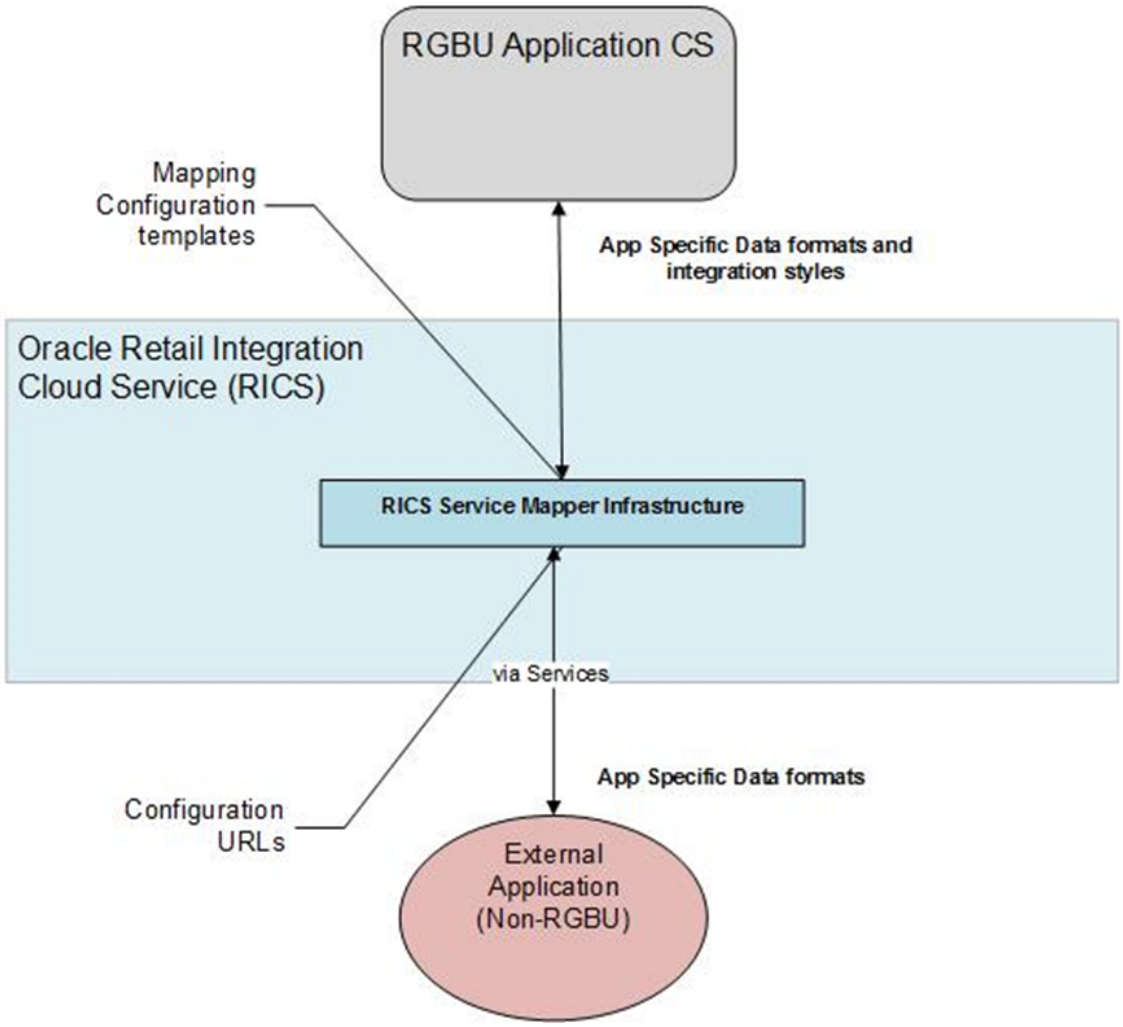
The USM installation into any Customer Responsible Environment (On-Prem/IaaS/PaaS) will be full featured as documented in this Guide.

2
Introduction

The Universal Service Mapper (USM) is an application component of Retail Integration Cloud Service (RICS) that allows the definition, mapping, and configurations needed to support the integration between two heterogeneous applications. Typically, this is an Oracle Retail application found in the Merchandise Foundation Cloud Service and an application external to Oracle Retail, such as Oracle Warehouse Management.

RICS USM supports two of styles of input for an integration: message-based and service-based. Within the RGBU, message-based flows are performed across the Retail Integration Bus. External applications are predominately service-based, so the output of USM is a call is to an external service. Service calls from an external service are transformed to the correct style and format for the internal application.

The functional requirement for the USM is to act as the place to transform the Oracle Retail application data style and the data format into the data format expected by the external application, and then to perform the transformations of the external application's response.



Support Features

The following table lists the USM features supported in various product offerings.

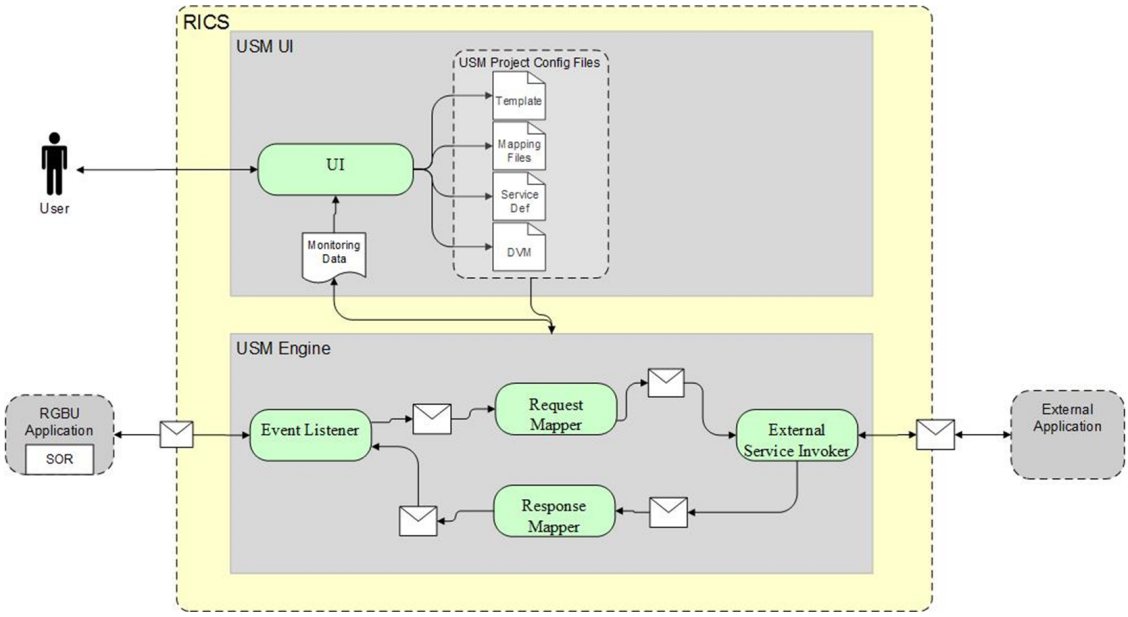
Feature Set	Product Offering		
	On-premises (RIB)	**SaaS (RICS)**	**Hybrid Cloud (USM on PaaS/IaaS integrated with RICS on SaaS)**
Create New/Custom Projects	Self-managed	Oracle Development managed (design-time)	Self-managed
Manage Existing Projects	Self-managed	AMS managed	Self-managed
Manage Configuration	Self-managed	AMS managed	Self-managed
Create new Service Mappers	Self-managed	Oracle Development managed (design-time)	Self-managed
Manage Existing Service Mappers	Self-managed	AMS managed	Self-managed
Create new DVMs	Self-managed	Oracle Development managed (design-time)	Self-managed
Manage Existing DVM's	Self-managed	AMS managed	Self-managed
Import/Export	Self-managed	AMS managed	Self-managed
Monitoring/Traceability	Self-managed	AMS managed	Self-managed
View Logs	Self-managed	AMS managed	Self-managed

3

USM Functional Architecture

Universal Service Mapper (USM) is a platform that allows you to define, map, configure and deploy projects that are required to maintain a seamless integration between two heterogeneous applications.

The application has two components, the User interface and the Engine.



USM User Interface

The user interface gives you the ability to do the following:

- Create and Manage:
  - Projects in USM
  - Service Mapper Files
  - Configuration Files
- View:
  - App statistics
  - Metrics about the message flow
  - System Logs

USM Engine

The USM engine is the logic part of the system. It is where the data is received from the source application, mapped to other data, and the mapped data is sent to the target applications. Data is communicated through service calls.

USM hosts all the necessary web services required by the participating sender and receiver applications. USM has a configuration file that needs up-to-date service URLs for the participating applications.

USM also has the templates that contain the mapping information, the code that does the mapping, and also the configuration files that need to be configured to make the application work.

USM Project

A USM Project has the templates that contain the mapping information, the code that does the mapping, and the configuration files that need to be configured to make the application work.

There is one Project per integration. For example, there would be one Project integrating RMS with Oracle Warehouse Management Cloud Service.

There can be multiple Projects (integrations) hosted by one USM instance. For example, a single USM instance can host the integration between Oracle Warehouse Management and RMS, and an integration between Oracle Customer Management and Oracle ATG Web Commerce.

Oracle Retail creates the initial USM Projects for supported integrations and packages and ships them with the base product.

Modules

Each project in USM has a property named “Modules". The artifacts of this project are identified by the modules associated with the project. Each artifact having a prefix with a project module is associated with the project. Each project can have a minimum of one module and a maximum of 4 modules.

Templates

Template files are the main files holding the actual mapping information used during a mapping. Templates associate different fields in different payloads with one another, mapping fields from one application format to another using the XML format.

There are three different types of templates being used to map data. These files are of the XML data descriptors. The three types are:

- Request Templates
- Response Templates
- Failure Templates

The templates are used to perform data mapping when the participating applications need to communicate with each other.

The Request templates are used when the participating source application sends a message with data that has to be mapped to destination application data format.

The Response templates are the result of the mapping that has been performed on the source application data format.

The Failure templates are also the result of the mapping but, instead of actual mapped data, they contain error codes and specified error messages because of errors caused by missing data or unexpected server events that might have occurred during application runtime.

For greater detail refer to the *USM Implementation Guide* for the template content and use of the templates.

Service Definition Files

The service definition JSON files store the data required for the communication between the participating applications. They contain the host URLs of the source and destination applications along with usernames and passwords, if any, for such applications.

These are of the format JSON, meaning the data is stored in a key-value fashion. The USM application uses the RIB-LGF and LogFire URL set here to communicate with the respective applications.

The USM Implementation will give a greater insight about the fields that can be configured and the usage of the file.

Orchestration Files

These files which contain the actual mapping logic. These are in smo format. These files contain scripts that map data coming from a source application to a data format the destination application can work with. The mapping happens with all the fields mapped using a one-to-one mapping. Fields not required, if any, by any of the applications are simply dropped, and non-present fields present in any of the applications is mapped with a predetermined default value.

> **Note:**
>
> These scripts are strictly read-only and should not be modified.

Domain Value Maps

A Domain Value Map (DVM) is a table containing mappings between related information in participating applications. They enable you to equate lookup codes and other static values across applications. These DVM tables are used in transforming the messages from one system into the expected format of the other system.

Administrators can extend the list of mapped values by adding more maps. The DVM data should be synchronized with what the participating applications use. This synchronization should occur before any initial loads are run or any incremental transactional flows are initiated.

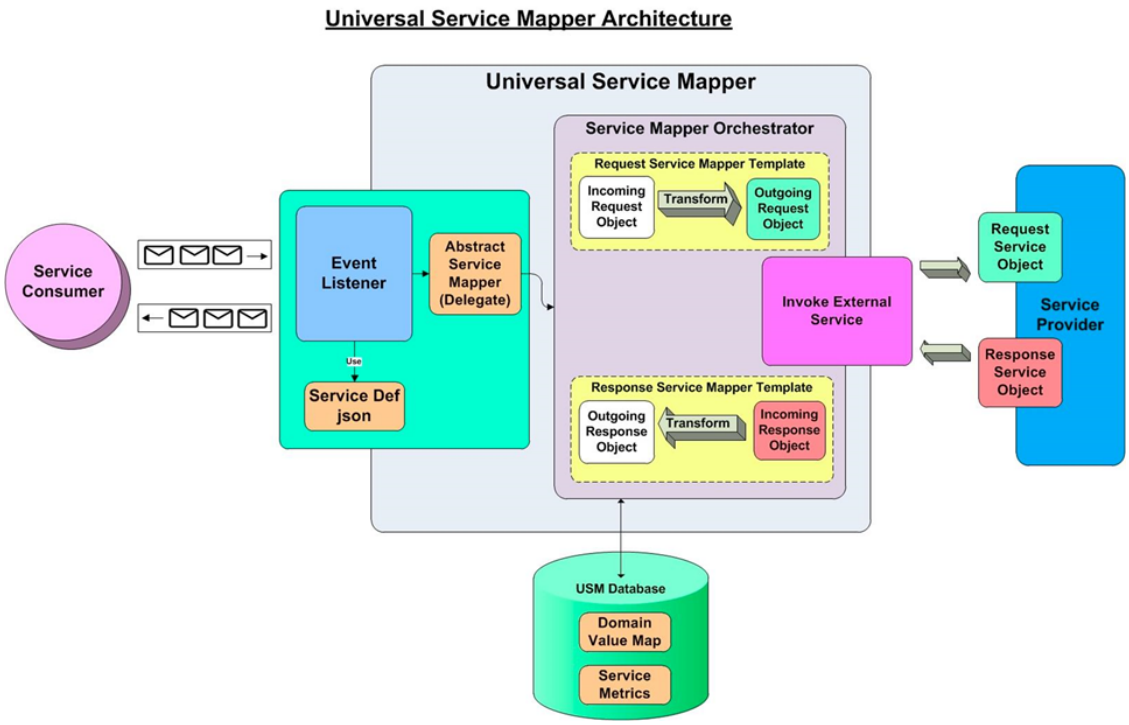
Data that needs to be stored as foundation/seed data and data that does not have many/any modifications, is stored in Static DVMs. These DVMs are created beforehand. Data can be added or removed at any time but, the data is mostly unchanging data.

Data that is to be stored during runtime of the application is stored in Dynamic DVMs. The data is stored and fetched in these DVMs as per request and the data present here can change, as per request, anytime during the runtime of the application.

> **Note:**
>
> Dynamic DVM has an auto-purge and archive feature that can be configured as a part of the **Configurations** tab (for more details, see USM User Interface). The key name for this is `DVM_AUTO_PURGE` and has the default value `TRUE` (upper-case). The purge delay can also be configured using the value for the key `DVM_PURGE_DELAY`. The default value is `550` (in days) and the hard minimum limit is `365` (in days). The purged data is archived to a DVM history table and can be retrieved later, as required.

4

USM Technical Architecture



Universal Service Mapper has 3 major components:

- Event Listener [Abstract Service Mapper, Service Def JSON]
- Service Mapper Orchestration [Orchestrator, Template and DVM]
- External Service Invocation and Service Provider

Event Listener

The event listener is a service hosted by the USM application which is open to receiving data from any application that is connected to it. The application here is either RIB-LGF or WMS Cloud. The applications have the following URL pattern set in their target for USM.

```
http://<host>:<port>
```

When application sends data, the event listener internally calls the abstract service mapper which determines family, message type and the operation(s) from the message received by referring to the Service Def JSON file.

Service Mapper Orchestration

The abstract service mapper now calls the service mapper orchestrator, which decides what data populates the mapper templates. The orchestrator does the field-by-field mapping from

the source application to the destination application. Certain key-value pairs in the DVM maintain context between the applications.

Service Provider and External Services

The Service Mapper Orchestrator calls the services hosted by the service providers after the mapping operations are completed. The service providers here are either RIB-LGF or WMS Cloud, which consume these services through USM. The calls are REST calls. USM holds the information necessary for it to call these services in a JSON file with the prefix `external_env_info` for the respective application. These are stored as key-value pairs in a JSON file.

5

USM User Interface

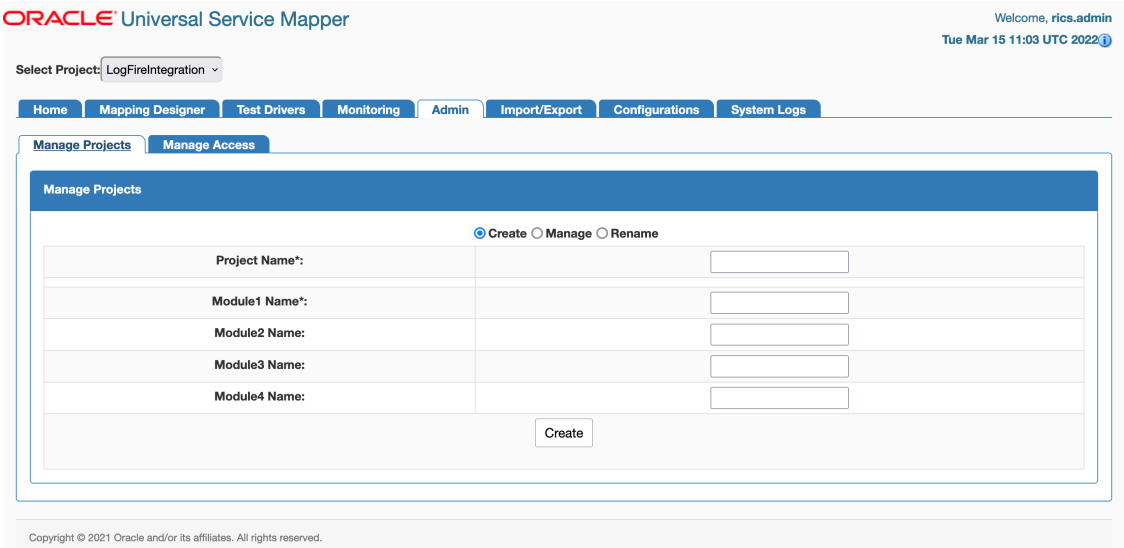
The USM web application allows you to manage and create project and project artifacts for service mapping to enable communication between two different applications.

There are 3 different type of users in USM who will have access to certain tabs based on their role. The Admin Role user is the administrator of the application and has access to all the tabs; the Operator Role user has restricted access to certain functions; and the Monitor Role user can only view the information. The following list shows the tabs with decreasing order of access from top to bottom.

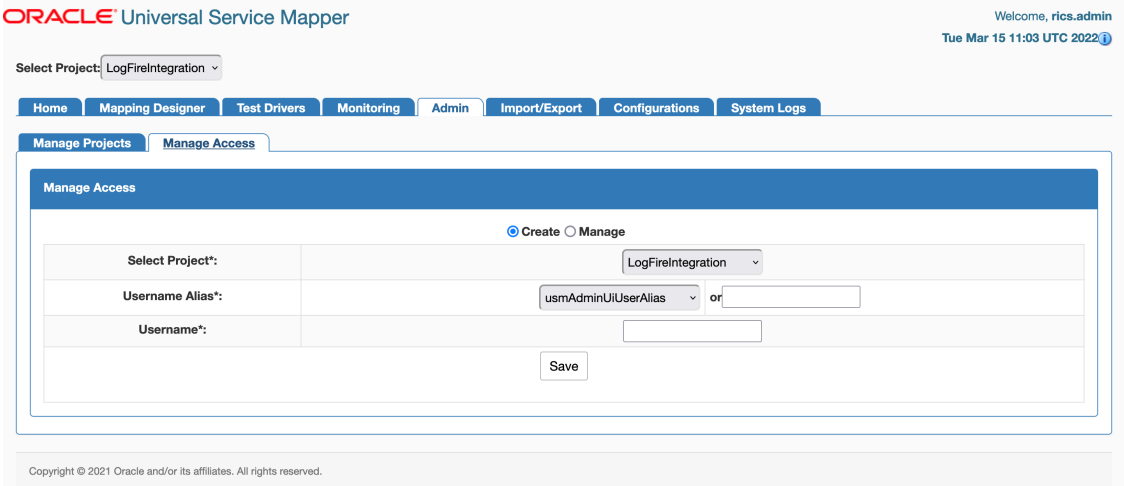
- Admin Role user
  - Admin tab
  - Configurations tab
- Operator Roles user
  - Mapper Designer tab
  - Import/Export tab
- Monitor Role user
  - Home tab
  - Monitoring tab
  - System Logs tab

Admin

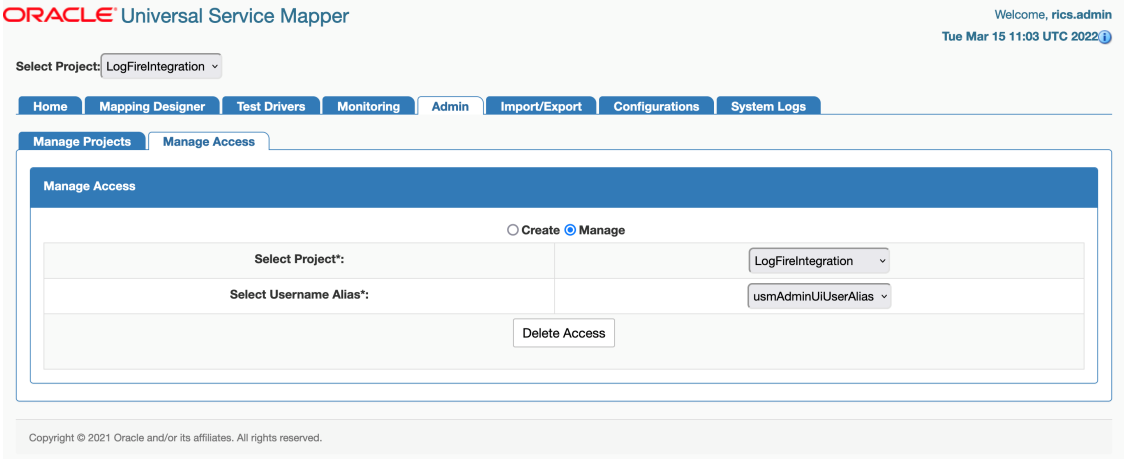
The Admin tab allows Administrators to manage projects and project access. In the projects sub-tab, administrators can create, update, rename, and delete projects.



In the Access sub-tab, Administrators can create and manage access. Using the **Create** option, you can add users to projects by providing usernames and username aliases.

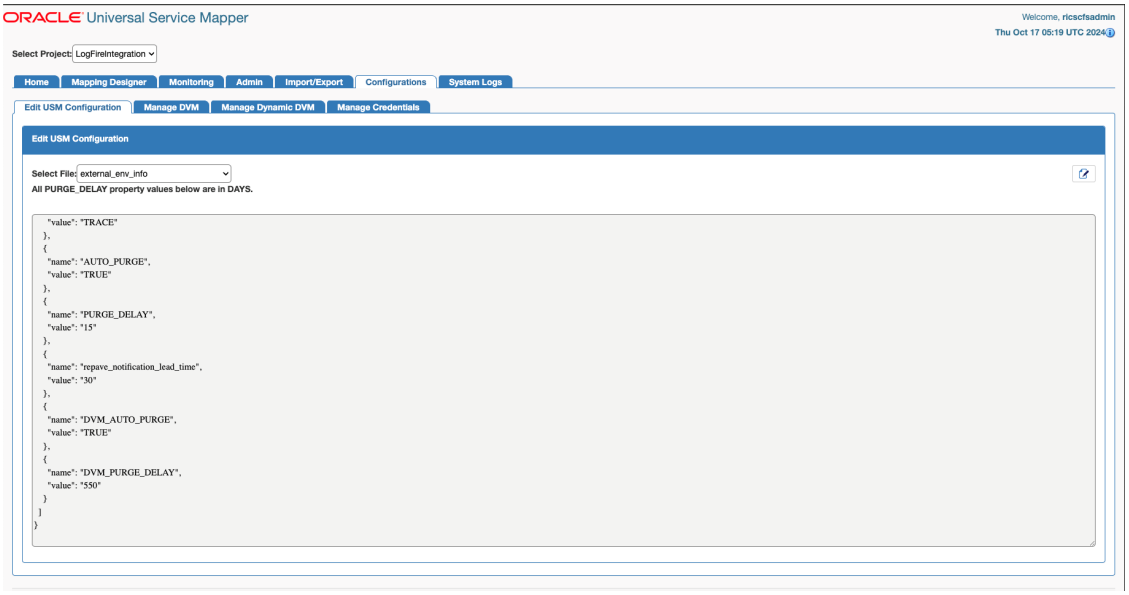


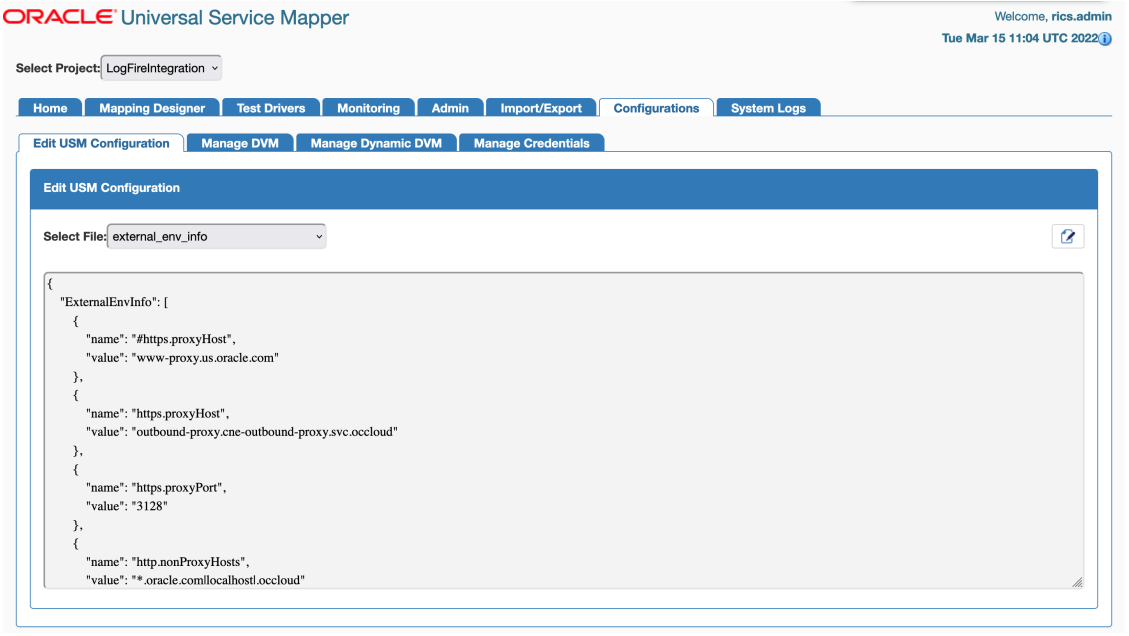
Using the **Manage** option, you can remove user access.



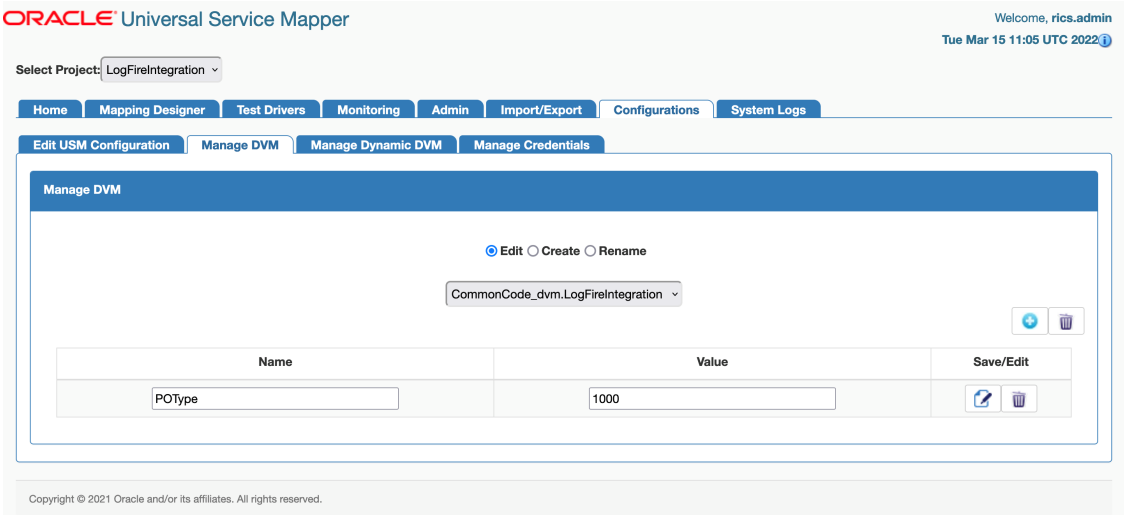
Configuration Tab

Configuration tab allows you to edit configuration files and manage DVM for the selected project. In the **Edit USM Configuration** tab, you can edit the configuration file.

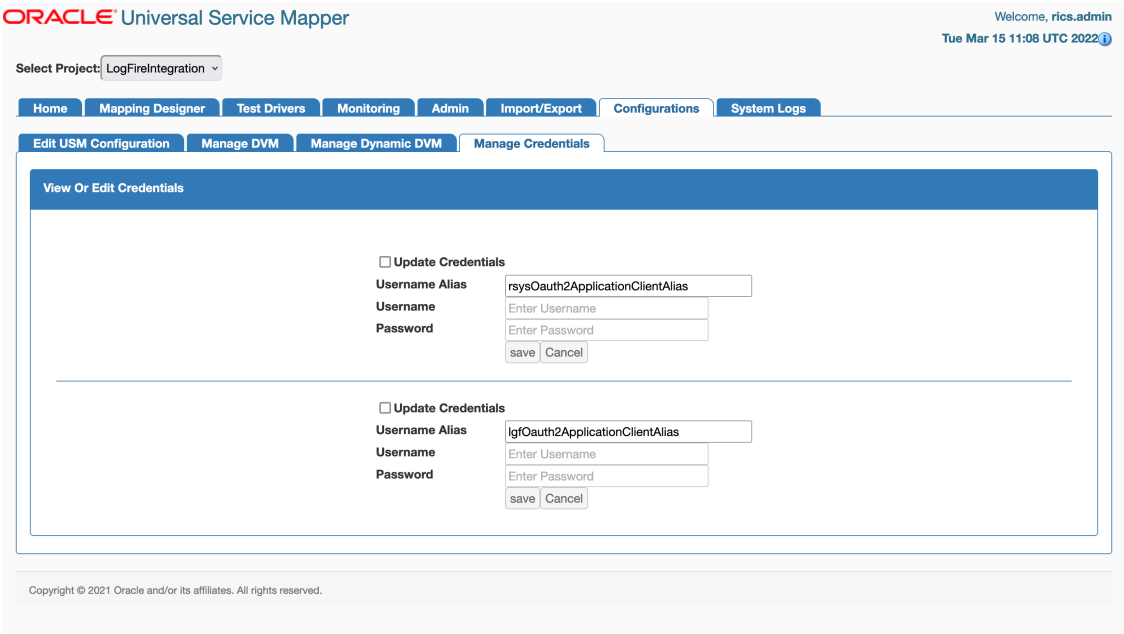




In the **Manage DVM** tab, you can edit DVM data. It also allows you to create, delete and rename DVM.

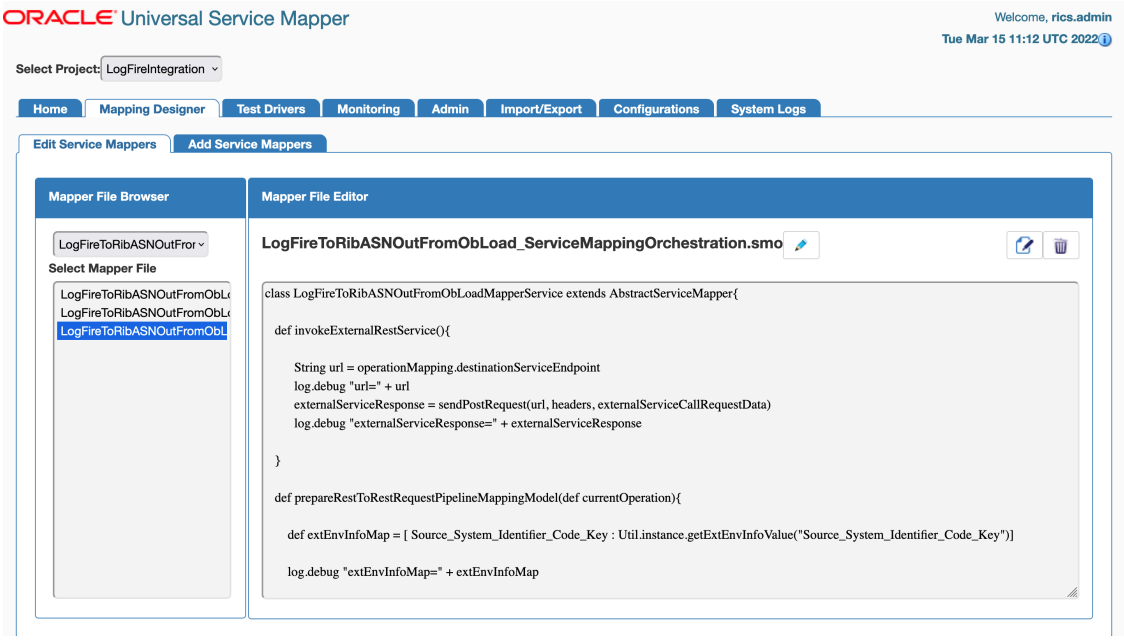


In the **Manage Credentials** tab, you can update credentials.

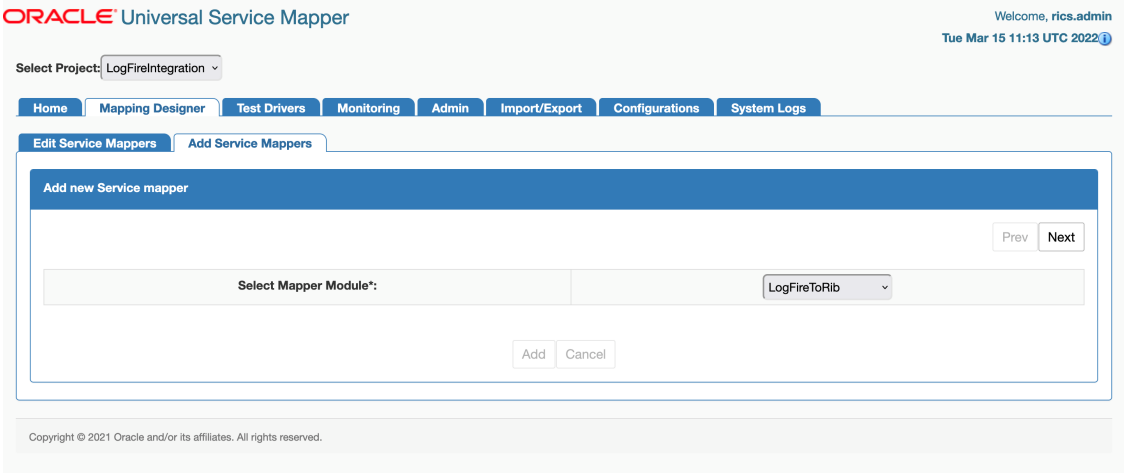


Mapping Designer

This tab allows you to manage and view Service Mappers for the selected project. In the **Edit Service Mappers** sub-tab you can browse existing service mappers, edit service mapper files, rename mappers, and delete mappers.

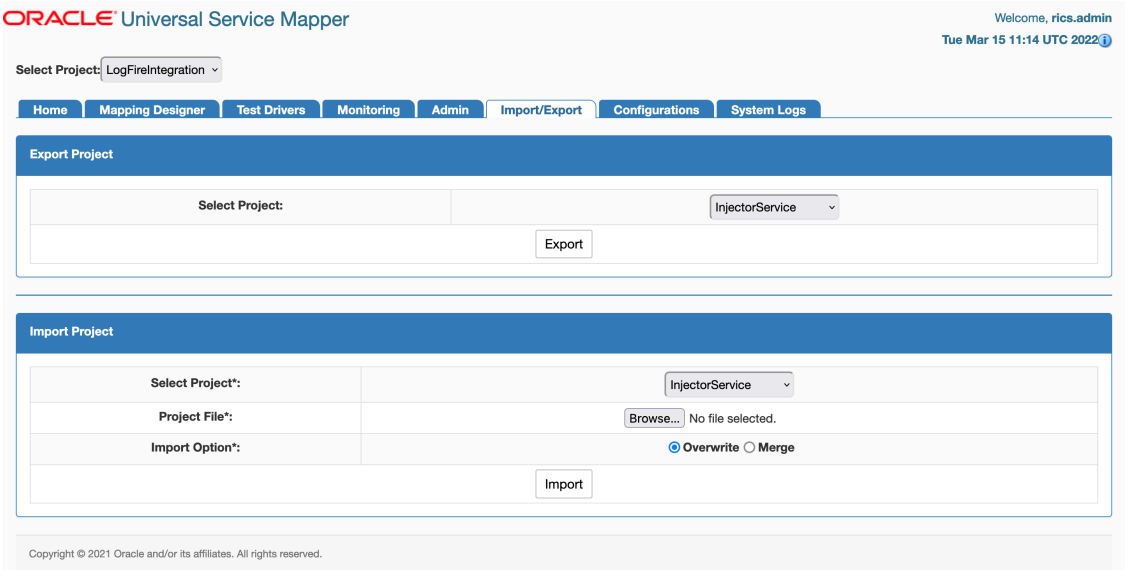


In the **Add Service Mapper** sub-tab, you can create new service mappers.



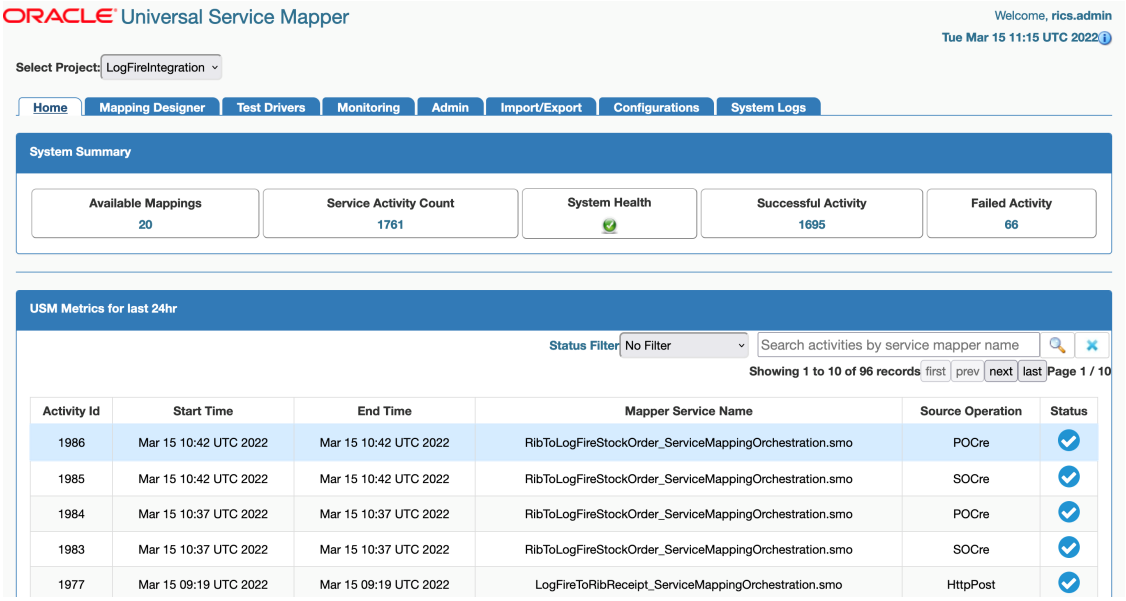
Import/Export Tab

The **Import/Export** tab allows you to import and export project files in .zip format.



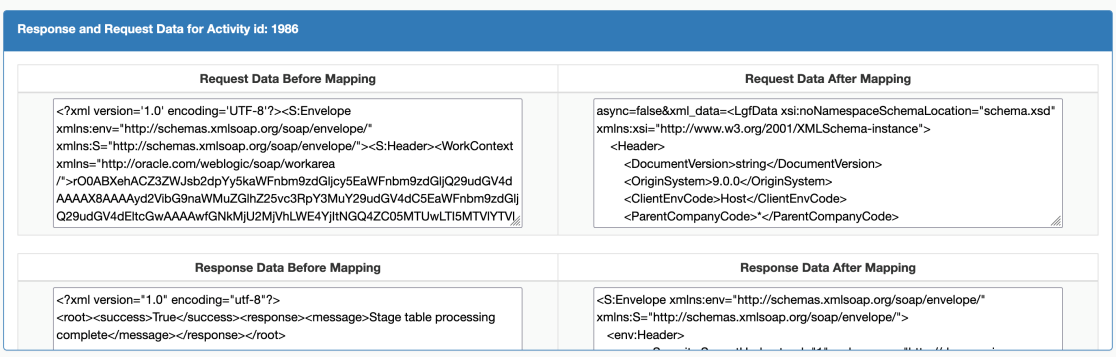
Home

The **Home** tab displays the summary of the service mapper application. The System summary panel displays the available mappings, service activity count, and system health, successful and failed activity.



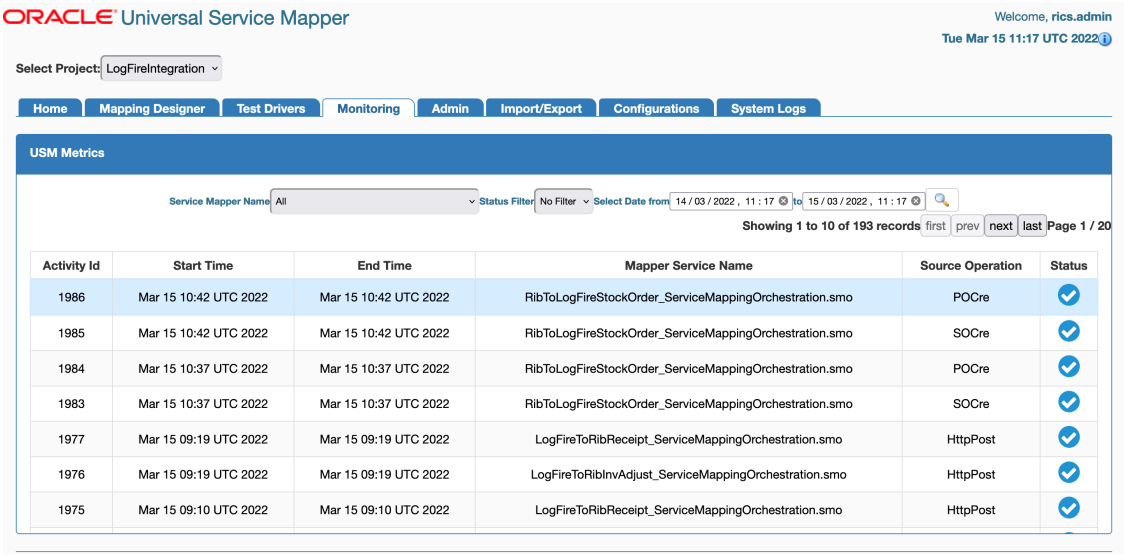
USM metrics for the Today panel show the mappings since midnight. You can search, filter, and select a mapping from the table to view the request and response mapping before and after the mapping.

> **Note:**
>
> USM message logging is optimized to improve performance. Large messages will be truncated to 1 MB and then saved to the database. Those truncated messages show up as “before mapping” and “after mapping” on the GUI.



Monitoring

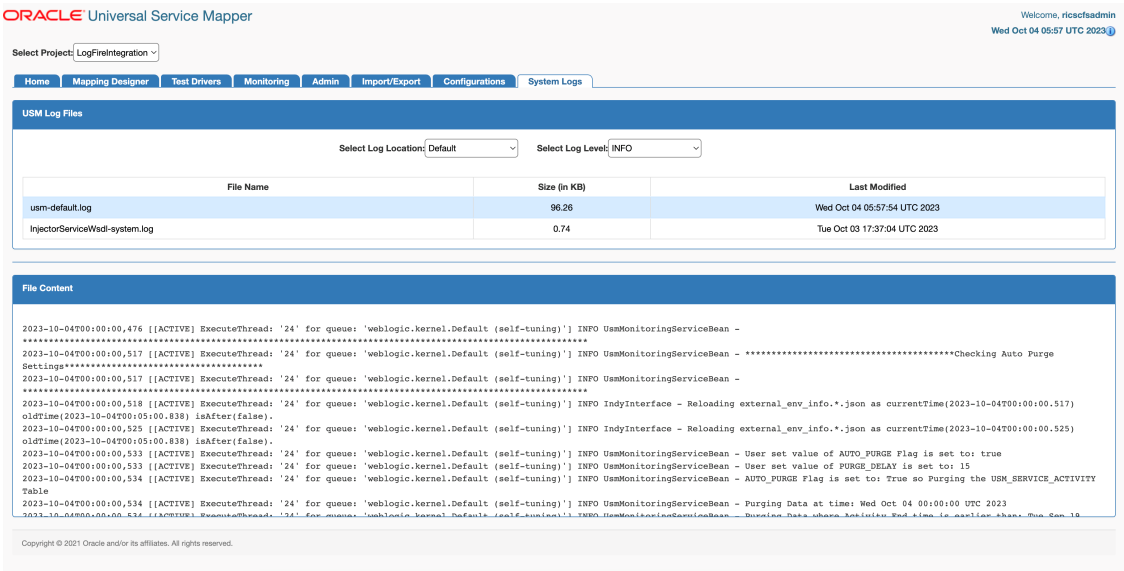
Monitoring tab displays USM metrics in a tabular format. The data on the monitoring tab has filters service mapper name and Date. User can view all the service mappings with the selected filters using the provided pagination buttons. User can also view the request and response data before and after the mapping by clicking the service mapping activity in the table. By default, the monitoring tab displays the service mappings for all the mappers from last 24 hours.



> **Note:**
>
> USM Service mapping logs have an auto-purge feature that can be configured in the **Configurations** tab (for more details, see USM User Interface). The key name for this is `AUTO_PURGE` and has the default value is `TRUE` (upper-case). The purge delay can also be configured using the value for the key `PURGE_DELAY`. The default value for this is `15` (in days) and there is no upper or lower limit for this delay.

System Logs Tab

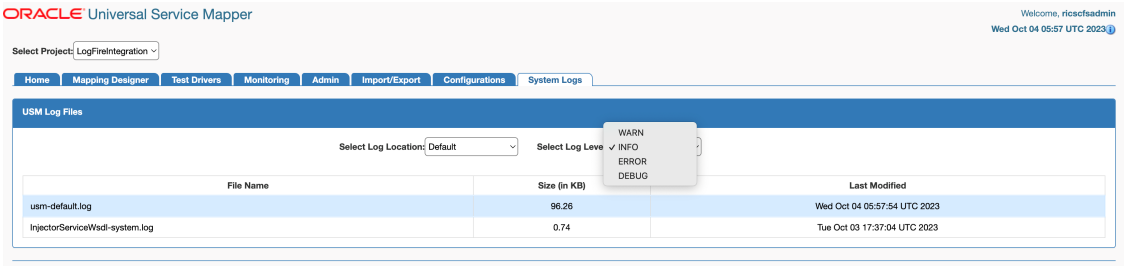
In the System Logs Tab user can browse through universal service mapper logs.



Log level can be configured from the Select Log Level dropdown.

There is an option to configure 4 log levels WARN, INFO, ERROR, DEBUG.

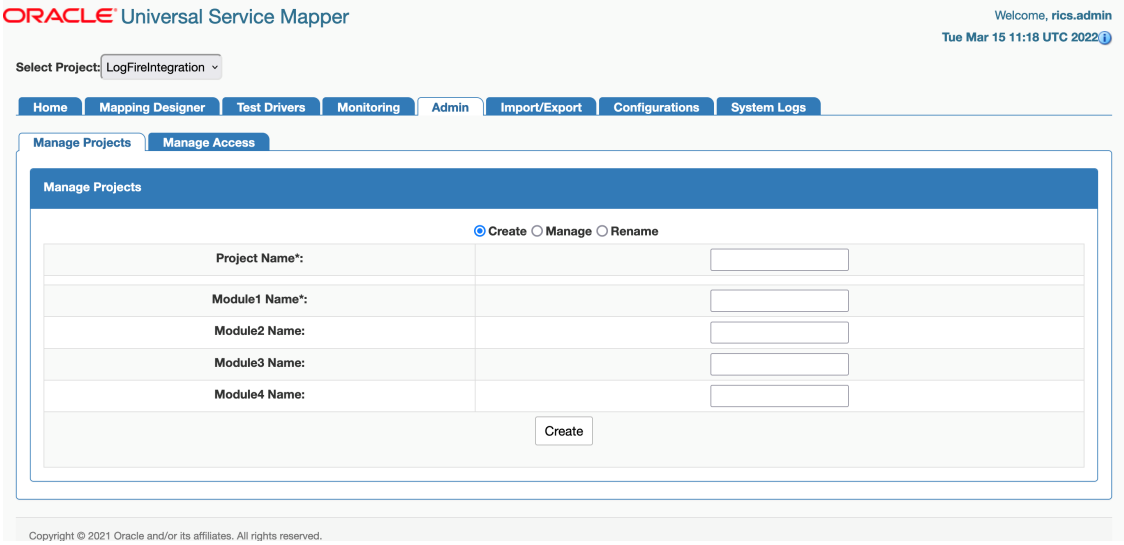
Log level will be automatically saved as soon as it is selected from the dropdown.



Create Project

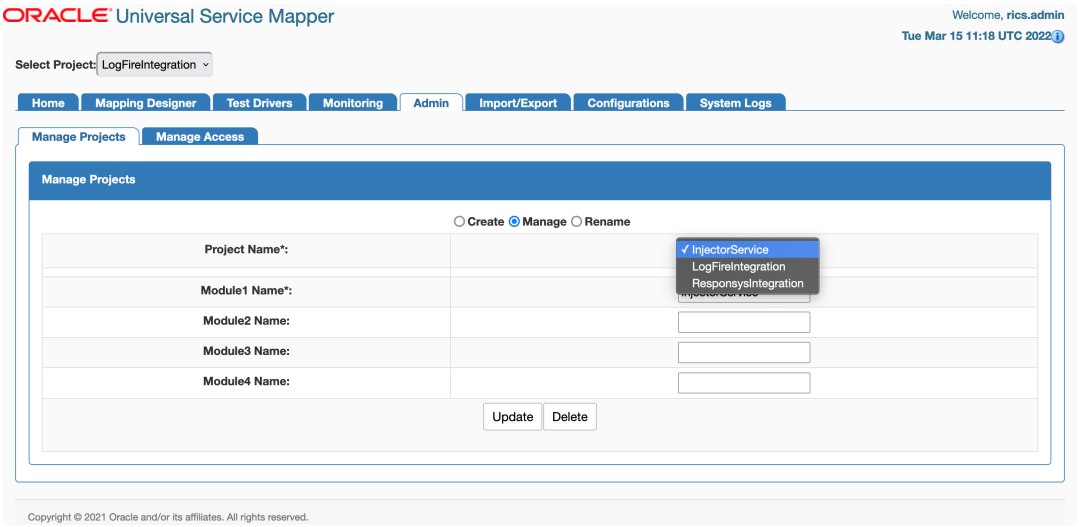
1. Go to the **Admin** tab.
2. Click on the **Manage Projects** sub-tab.
3. Select the **Create** radio button to create a new project.
4. Enter a new project name and a new module name.
5. Click on the **Create** button when done.

   Now the Project is created.

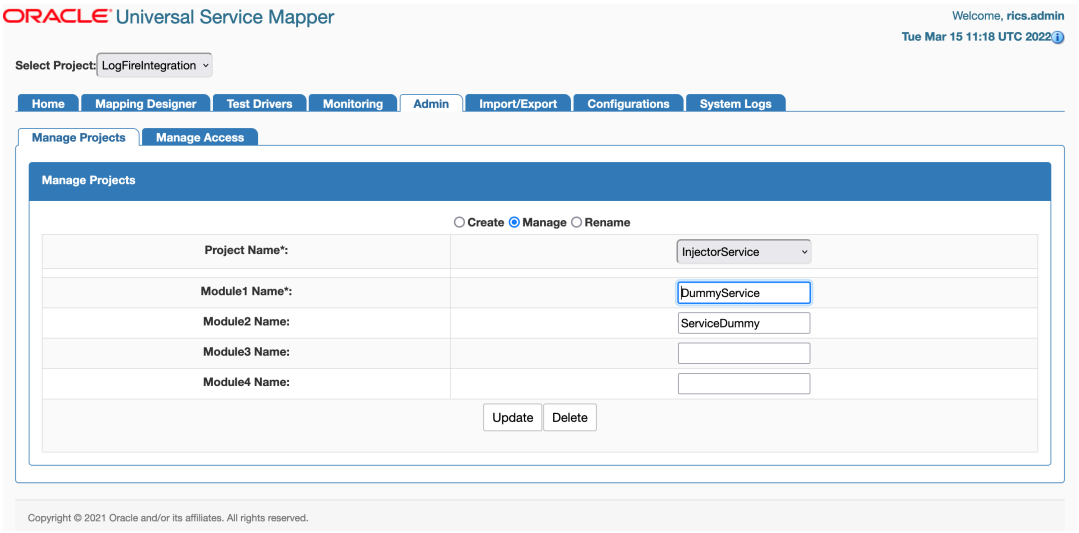


Update Project Modules

1. Go to the **Admin** Tab.
2. In the Admin Tab, click on the **Manage Projects** sub-tab.
3. Click the **Manage** radio button to update the project's modules.
4. Select **Project Name** from the drop down.



5. Now in the text fields, update the project module names, add or remove project modules as necessary.

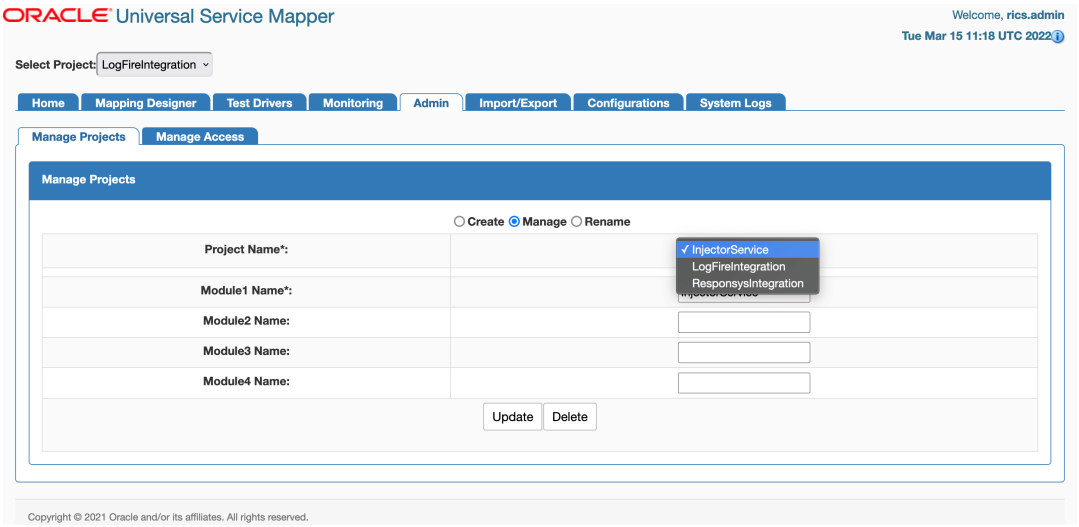


6. Click the **Update** button once done.

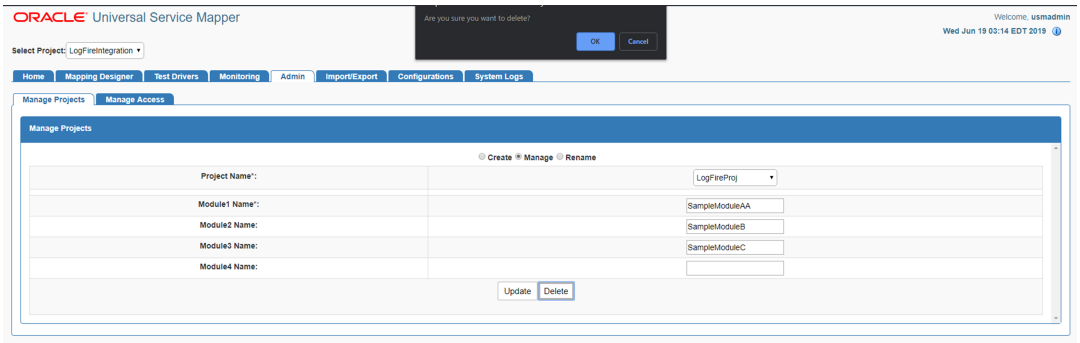
   Now the Project has been updated with new Modules.

Delete Project

1. In the **Admin** Tab, go to the **Project** sub-tab.
2. Click on the **Manage** radio button.
3. Select the **Project Name** from drop down.



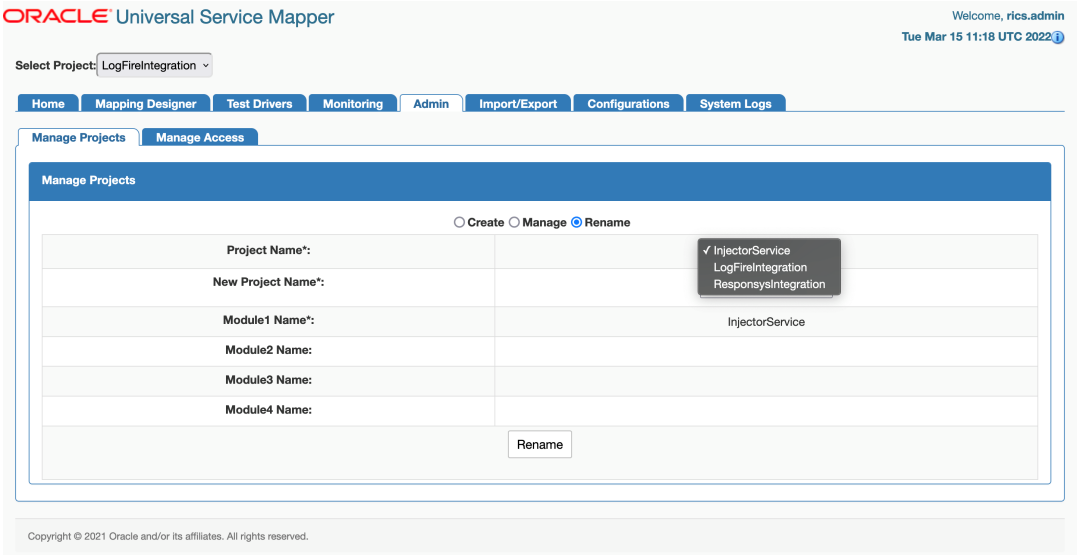
4. Click the **Delete** button.
5. A confirmation dialog appears, click on the **Okay** button.



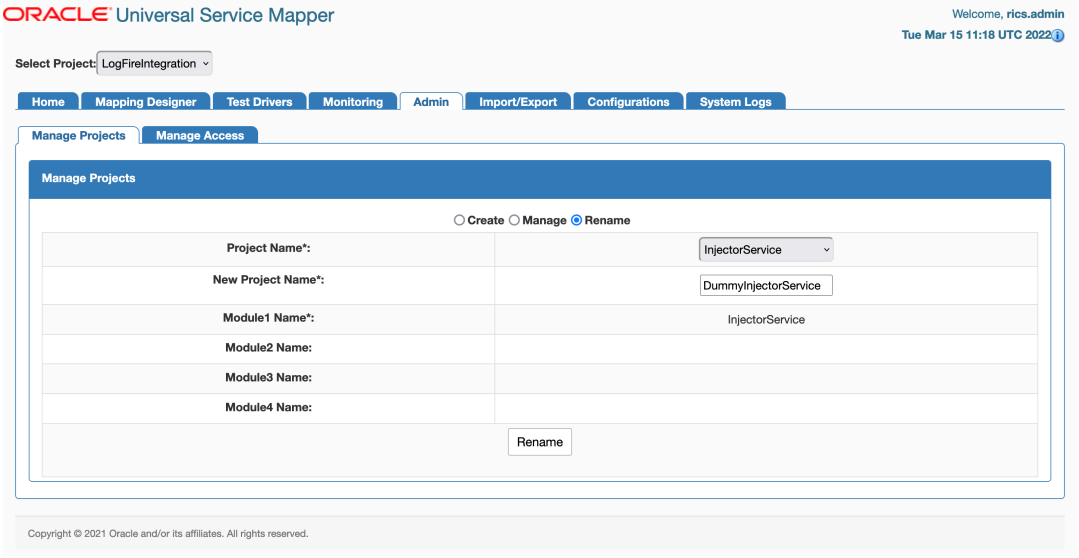
Now the selected project is deleted.

Rename Project

1. In the **Admin** tab, go to the **Project** sub-tab.
2. Click on the **Rename** radio button.
3. Select **Project Name** from the drop down list box.



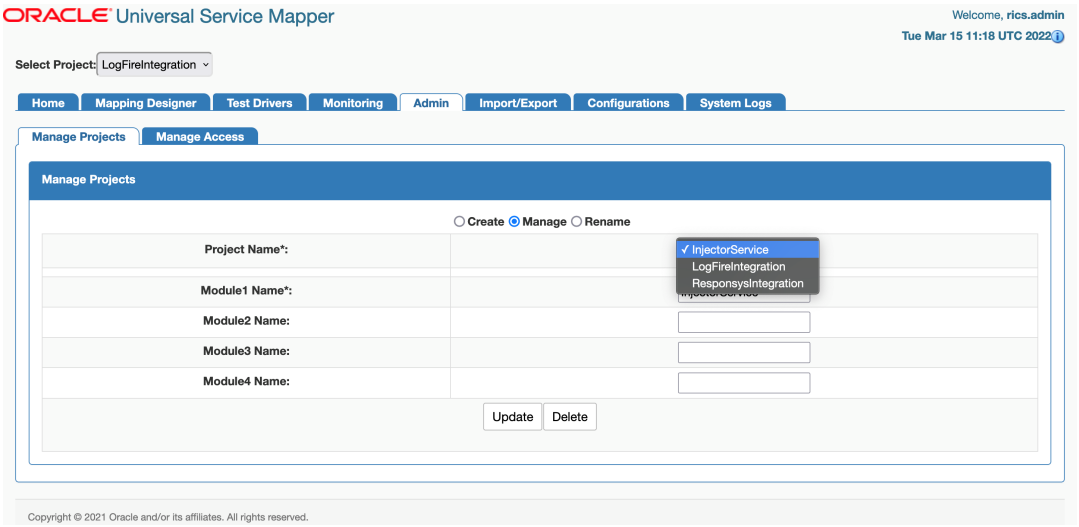
4. Enter the new project name in the **New Project Name** textbox.



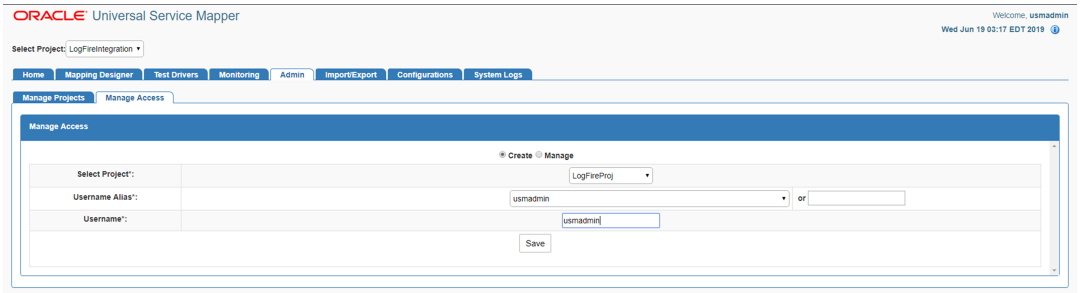
5. Click on **Rename** button to rename the project.

Provide User Access to a Project

1. In the **Admin** tab, go to the **Access** sub-tab.
2. Select the **Project Name** from the drop down list box for which access has to be given.



3. Enter the **Username Alias** and **Username** to which access has to be granted.

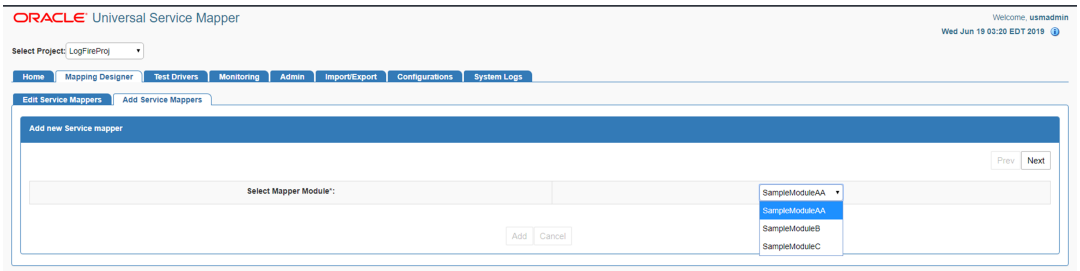


4. Click the **Save** button.

   The user now has access to the project.

Create New Service Mapper

1. Go the **Mapping Designer** tab.
2. Open the **Add Service Mappers** sub-tab.
3. Select the module name from the drop down list box and click on next.

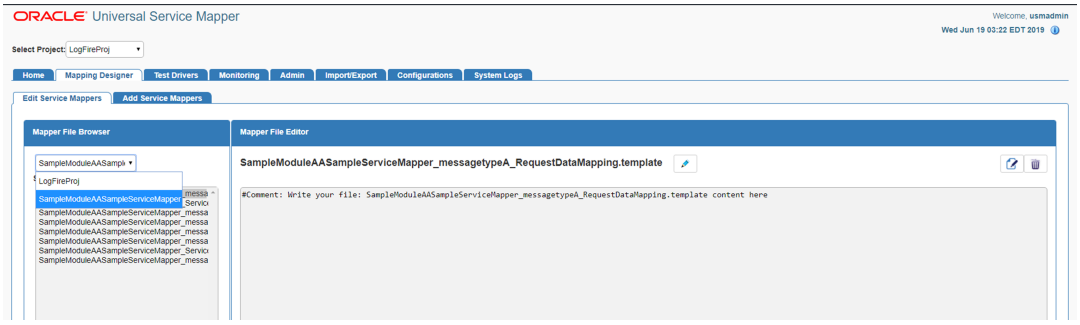


4. Enter the **Service Mapper** name of your choice and click **Next**.
5. Enter the **Message Types** that are to be supported by the service mapper, in a comma separated format.
6. Click on the **Add** button.

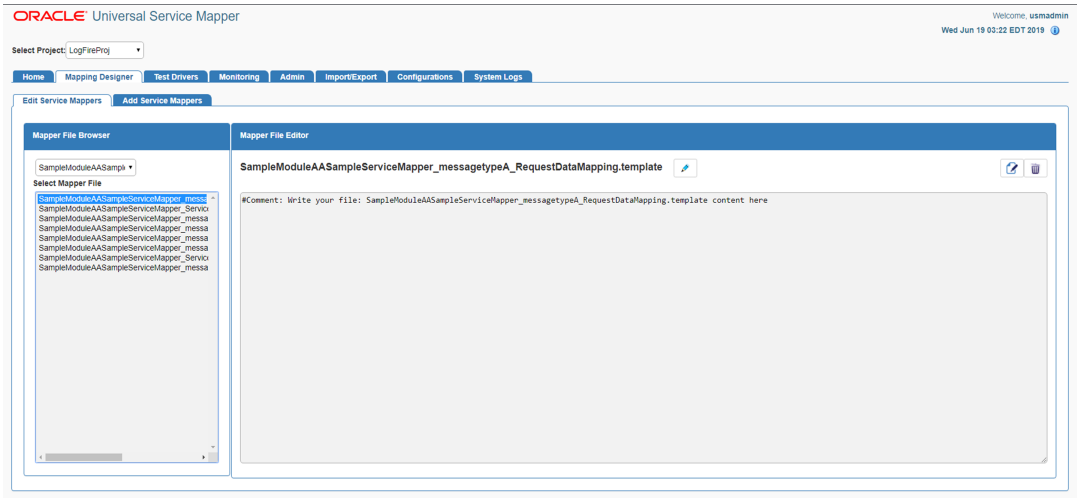
Now the new Service Mapper is created with all the necessary files.

Update Service Mapper Files

1. Go the **Edit Service Mapper** sub-tab in the **Mapping Designer** tab.
2. Select the service mapper prefix from the drop down list box on the left side of the screen.

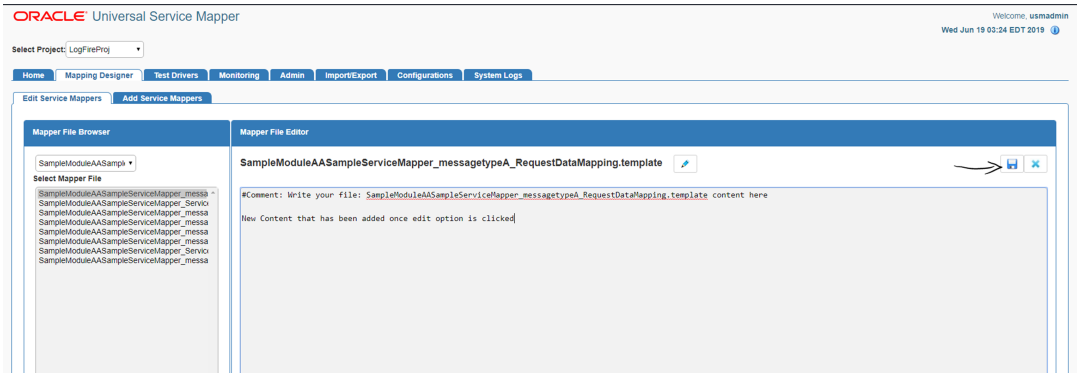


3. Select the mapper file name from the list that appears below it.



4. Once the file loads, click on the Edit icon on the right side of the screen.

   The text field should be enabled for editing.
5. Edit the content as desired.

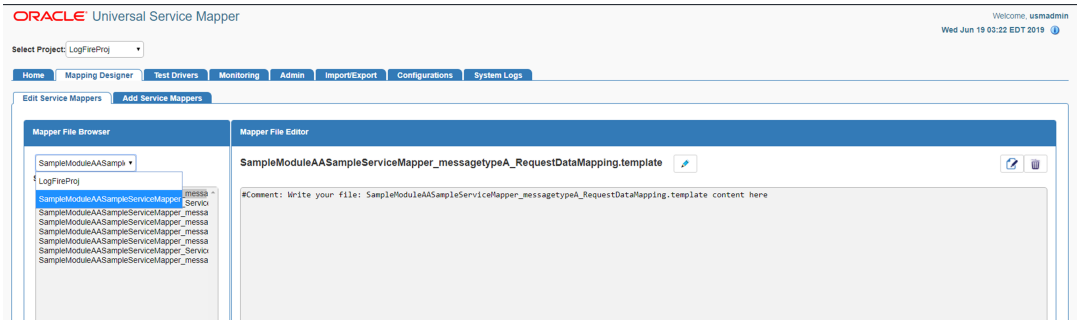


6. Once the editing is done, click the Save icon (it replaced the Edit button).

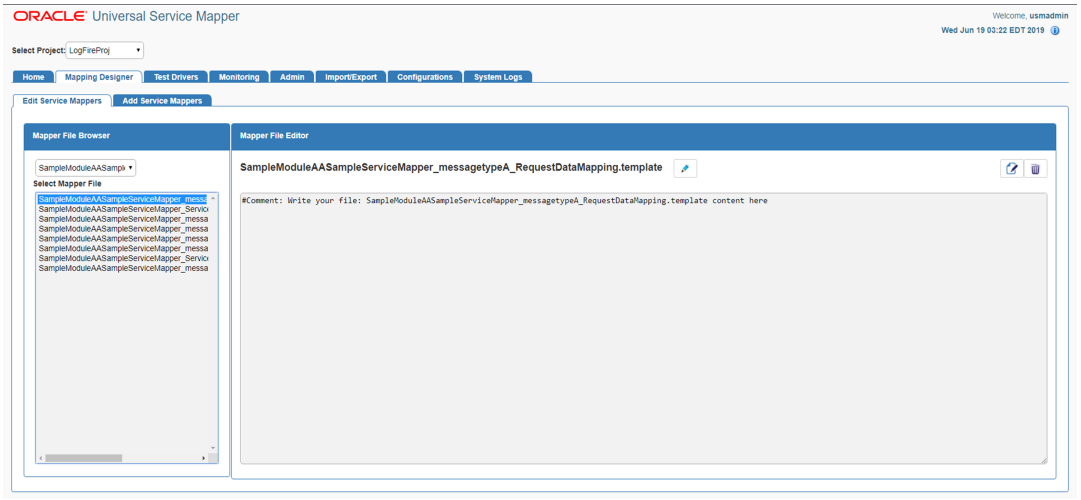
   The updates to the service mapper are saved.

Rename Service Mapper File

1. Go to the **Edit Service Mapper** sub-tab in the **Mapping Designer** tab.
2. Select the service mapper prefix from the drop down list box.



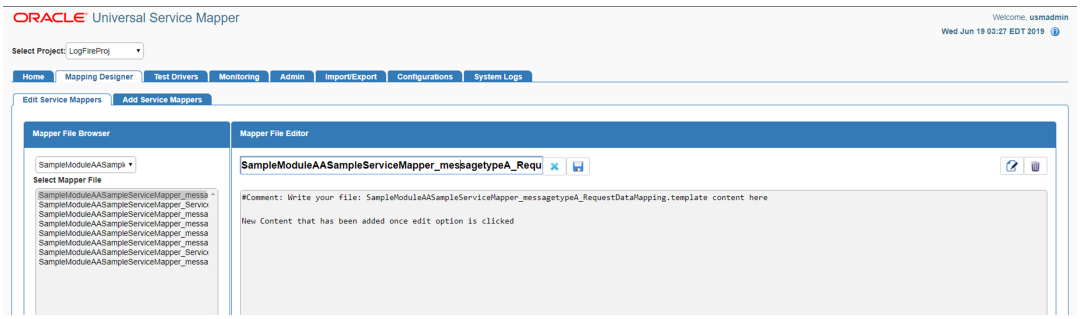
3. Select the mapper file whose name has to be changed.



4. Once the file is loaded, click the pencil icon next to the name of the service mapper on the right pane.

   An Edit box opens.

5. Change the name of the mapper file as required.

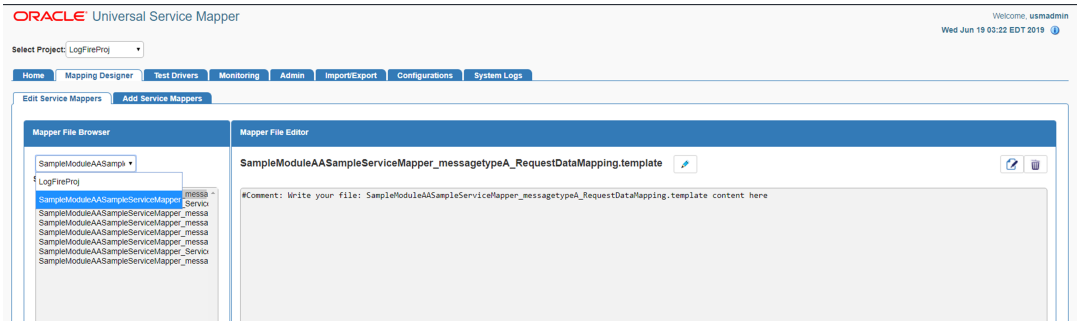


6. Click the Save button (it replaced the Edit button).

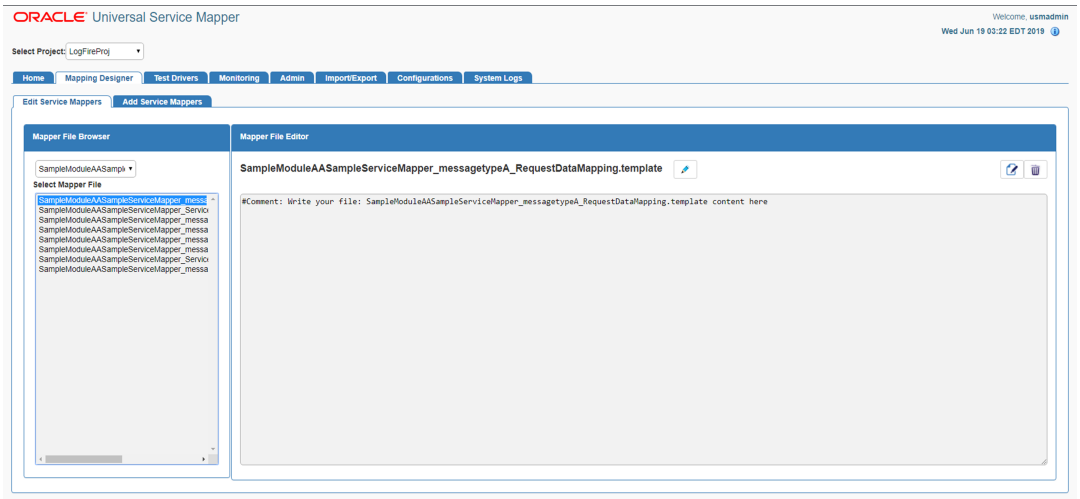
   The mapper file has been renamed.

Delete Service Mapper File

1. Go to the **Edit Service** mapper sub-tab in the **Mapping Designer** tab.
2. Select the mapper prefix from the drop down on the left side of the screen.



3. Select the mapper file to be deleted once the list below loads.



4. Once the selected mapper file loads, click the Delete icon on the far right end of the screen on the right pane.

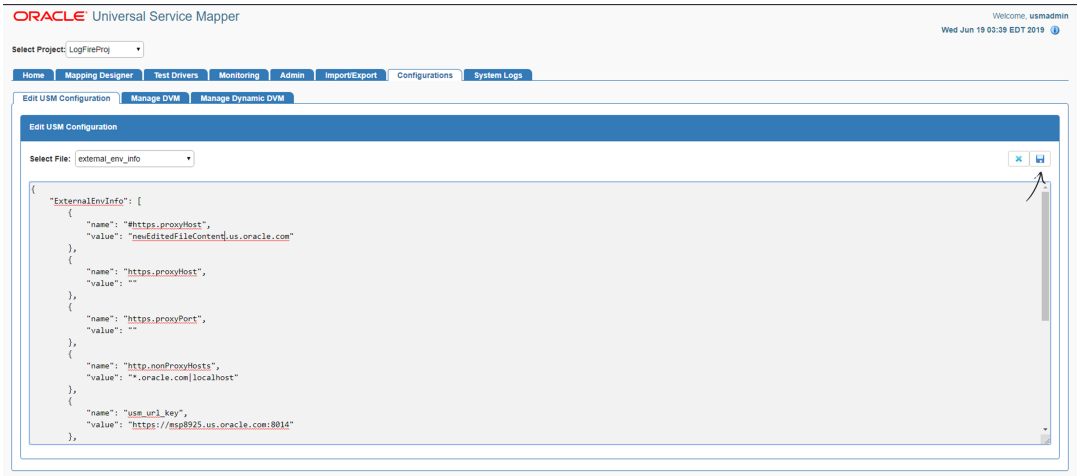
   A confirmation dialog appears.

5. Click **Okay** to continue.

   The mapper file is deleted.

Edit Configuration File

1. Go to the **Edit USM Configuration** sub-tab in the **Configurations** tab.
2. Click the **Edit** button icon on the right side of the screen.
3. Edit the contents of the file as desired.

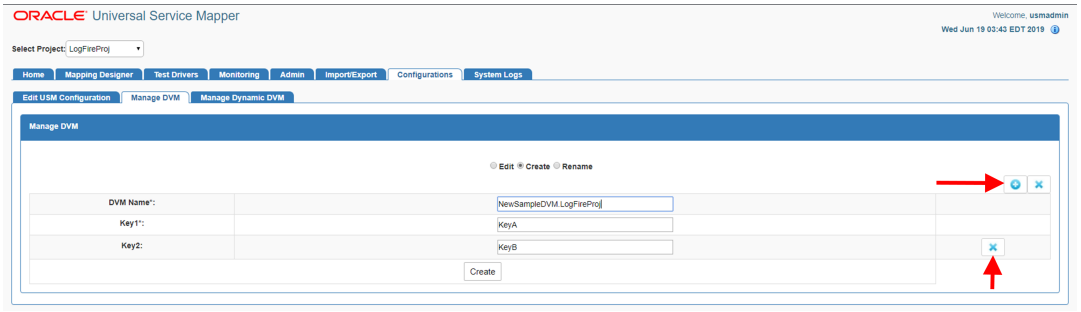


4. Once done, click the **Save** button.

   The Configuration file is now updated.

Create DVM

1. Go to the **Manage DVMs** sub-tab in the **Configurations** tab.
2. Click on the **Create** radio button.
3. Enter the **DVM Name** and key in the text boxes.
4. Click on the Add icon to add more keys or remove unneeded keys from the list by click on the Remove icon next to a key.

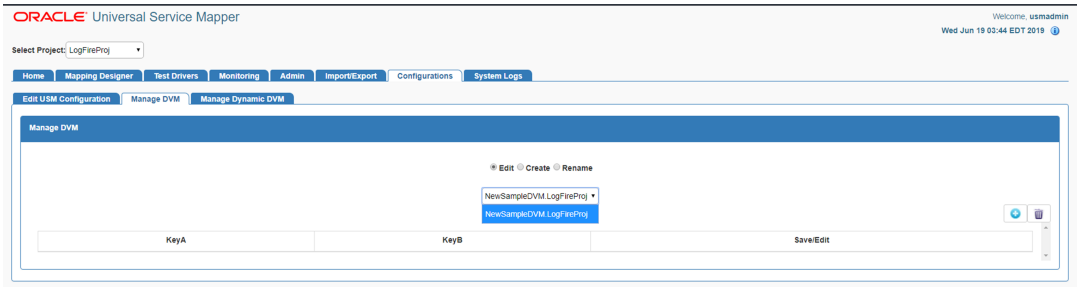


5. Once done, click on **Save** to create the DVM.

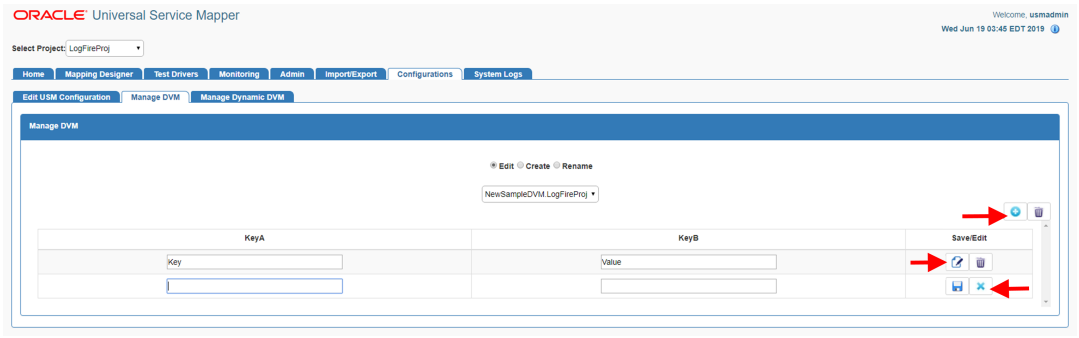
   Now the new DVM is created.

Update DVM

1. Go to the **Manage DVM** sub-tab in the **Configurations** tab.
2. Click the **Edit** radio button.
3. Select the **DVM Name** to be edited from the drop down list box.

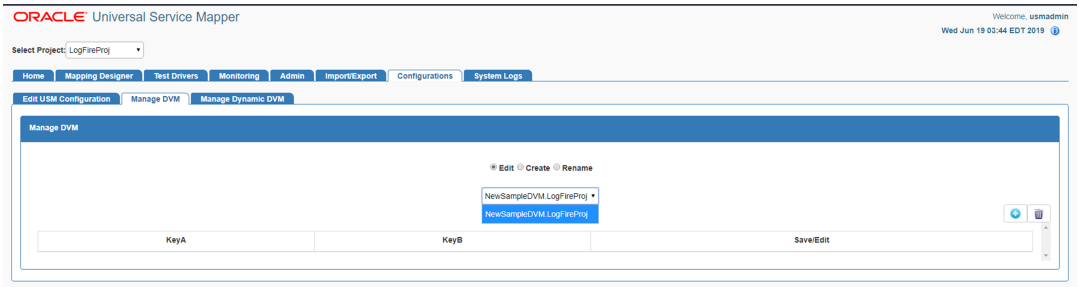


4. Changes are made to the DVM as rows are added, edited, or deleted:
   - Click the Edit icon to edit the DVM row.
   - Click the Delete icon to delete the row.
   - Click the Insert icon on the top right corner of the table view to add more DVM rows.

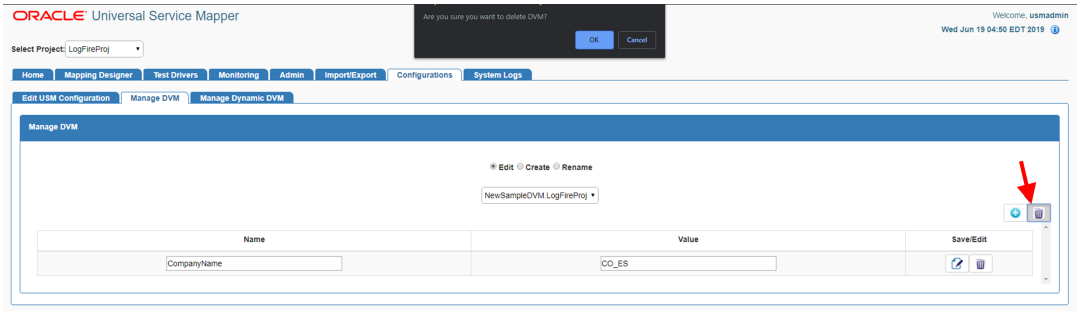


Delete DVM

1. Go to **Manage DVM** sub-tab in the **Configurations** tab.
2. Click the **Edit** radio button.
3. Select the **DVM Name** from the drop down list box.



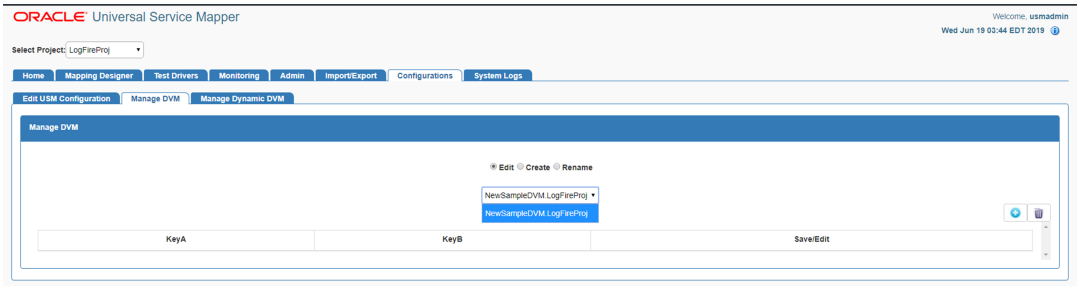
4. Click the Delete button on the top right corner of the table view.
5. A delete confirmation dialog appears, click **OK** to confirm the operation.



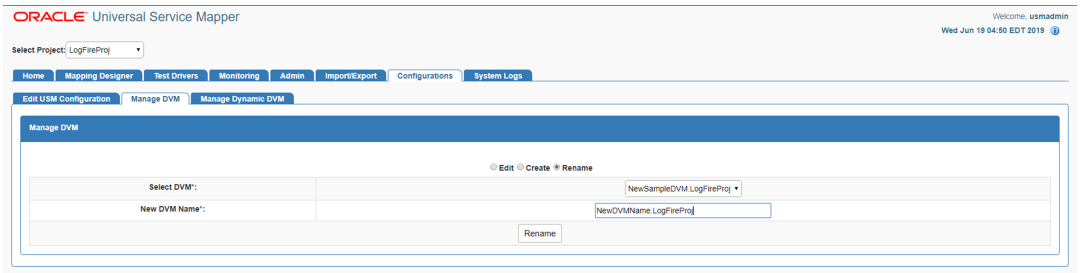
The DVM table is deleted.

Rename DVM

1. Go to the **Manage DVM** sub-tab in the **Configurations** tab.
2. Click the **Rename** radio button.
3. Select the DVM from the drop down list box.



4. Enter the new name for the DVM in the **DVM Name** text box.
5. Once done, click the **Rename** button to rename the DVM.



Now the DVM table has been renamed.

Mandatory Post-Deployment Setup

After deployment, perform the following procedures.

Set the WMS Cloud and RIB-LGF Application Links

Once the USM UI is up, do the following:

1. Log into the application and proceed to the **Configurations** tab.
2. Click the **Edit USM Configurations** sub-tab in the Configurations tab.
3. Select the `external_env_info.json` file from the drop down list box.
4. Change the following field:

```
{"name":"usm_url_key", "value": "[http://<hostname>:<port_number> /]"}
```

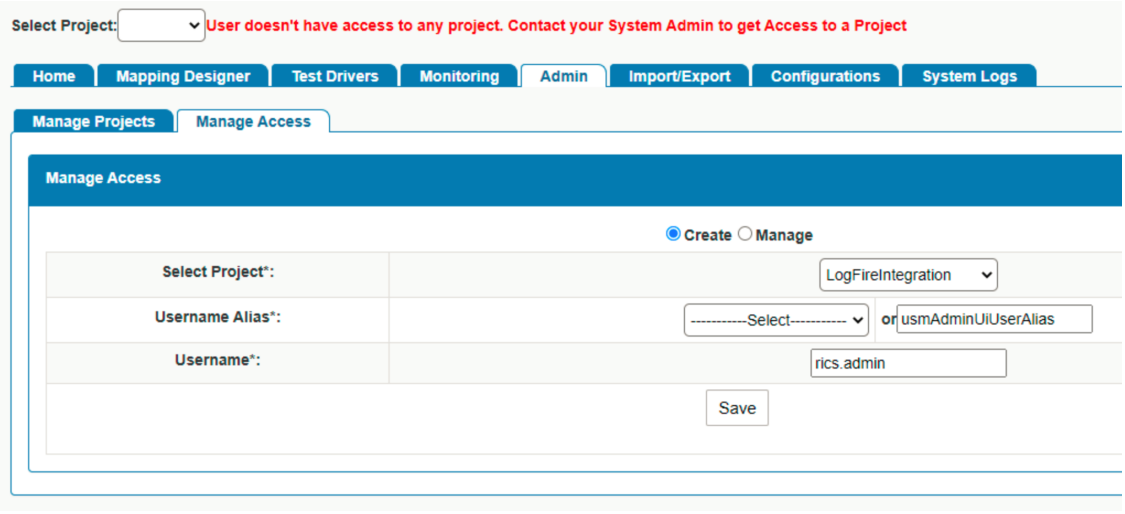
5. Save the file.
6. Next select the `external_env_info.LogFireIntegration.json` file from the drop down list.
7. Change the following fields:

```
{"name": "LogFire_Host_Url_Key", "value": "https://<hostname>:<port_
number>/rgbu_test"}
{"name": "RibLgf_Host_Url_Key", "value": "http://<hostname>:<port_number>/
rib-lgf-services-web/resources/publisher/publish"}
{"name":"rib_lgf_host_UrlSecurityPolicyKey", "value": "PolicyC"}
```

Configure Initial Project

To configure the initial project, perform the following steps:

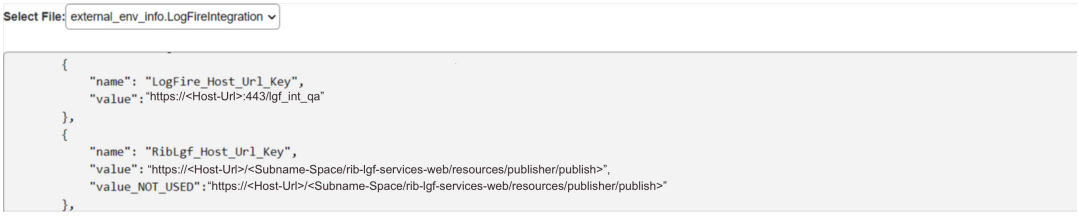
1. Login to USM UI as an admin.
2. Go to the **Admin -> Manage Access** tab and enter the information:
   - **Select Project** - Select `LogFireIntegration`
   - **Username Alias** - Select or enter `usmAdminUiUserAlias`
   - **UserName** - Enter the admin username (for example, `rics.admin`)
3. Click **Save**.



Update External JSON

1. Go to **Configurations -> Edit Usm Configuration**
2. Select **external_env_info.LogFireIntegration** from the dropdown menu, click the **Edit** button, and enter the values:
   - **name**: Enter `LogFire_Host_Url_Key`
   - **value**: Enter the logFire Host URL. For example:

     `https://<Host-Url>:443/lgf_int_qa`

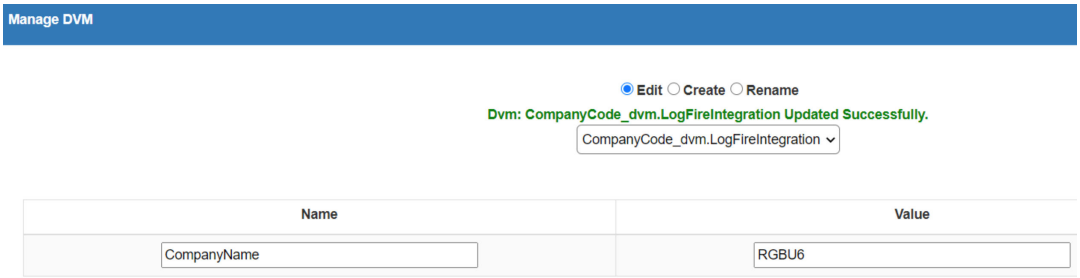


3. Click the **Save** button.

Update DVM

1. Go to the **Configurations -> Manage DVM** tab.
   - Select **CompanyCode_dvm.LogFireIntegration** from the dropdown menu.
   - Click the **Edit** button of the row to edit.
   - Update the value of **CompanyName** for the LogFire application

     For example: `RGBU6`
   - Click the **Save** button.



2. Select **FacilityCode_dvm.LogFireIntegration** from the dropdown menu.
   - Click the (**+**) button to add a new row and enter the **FacilityId**, **FacilityType**, and **FacilityTimeZone** for the LogFire application.

     For example:
     - **FacilityId** - `55`
     - **FacilityType** - `WAREHOUSE`
     - **FacilityTimeZone** - `US/Eastern`

Test the Deployment

After you deploy the server successfully, USM Web Application can be accessed using the following URL:

```
http://<host-server>/<Sub-name-space>/usm/
```

Information on Roles and Groups in USM Application

USM Application has some basic roles and groups which are used to determine the type of user:

Roles

- **AdminRole** - Users with this role have access to all the functions of the USM app. They can also setup the security permissions for other users.
- **OperatorRole** - Users with this role have the ability to read, write and modify content in the service mapper files. However they will not have access to the admin functions and cannot see the admin tab at all.
- **MonitorRole** - Users with this role can only view the data in the service mapping files.

Groups

- **UsmAdminGroup** - Users that belong to this group can perform all operations
- **UsmOperatorGroup** - Users that belong to this group can perform all operations except access the admin tab. The admin tab is not visible unless the user is logged in as an admin user.
- **UsmMonitorGroup** - Users that belong to this group can only view the data.

Functions by Role and Group

The following table lists all the functions which can be performed by the roles and groups mentioned above:

Role Name	**Admin Role**	**Operator Role**	**Monitor Role**
Group Name	**UsmAdminGroup**	**UsmOperatorGroup**	**UsmMonitorGroup**
Admin Tab Functions	Yes	No	No
Project Files Editing and Management	Yes	Yes	No
Service Mapper Files Editing and Management	Yes	Yes	No
Configuration File Editing	Yes	Yes	No

In the above table Editing and Management refers to all functions like create, delete, update, and rename operations.

6

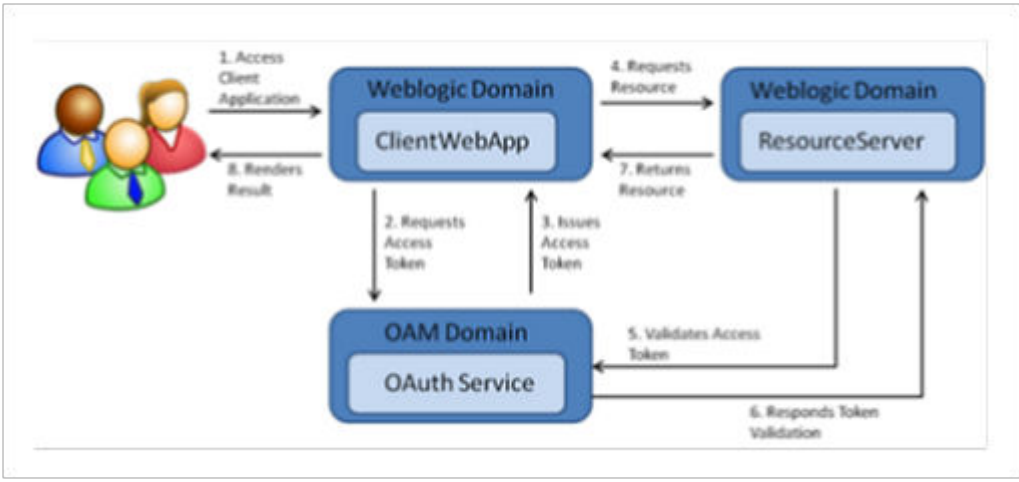
OAuth 2.0

OAuth 2.0 is the industry-standard protocol for authorization. The OAuth 2.0 authorization framework enables a third-party application to obtain limited access to an HTTP service, either on behalf of a resource owner by orchestrating an approval interaction between the resource owner and the HTTP service, or by allowing the third-party application to obtain access on its own behalf.

ORACLE CLOUD INFRASTRUCTURE CONSOLE AND THE IDENTITY AND ACCESS MANAGEMENT (OCI IAM) provides out-of-the-box OAuth Services, which allows a Client Application to access protected resources that belong to an end-user (that is, the Resource Owner).

OAuth 2.0 Architecture Diagram

Figure 6-1 OAuth 2.0 Architecture Diagram



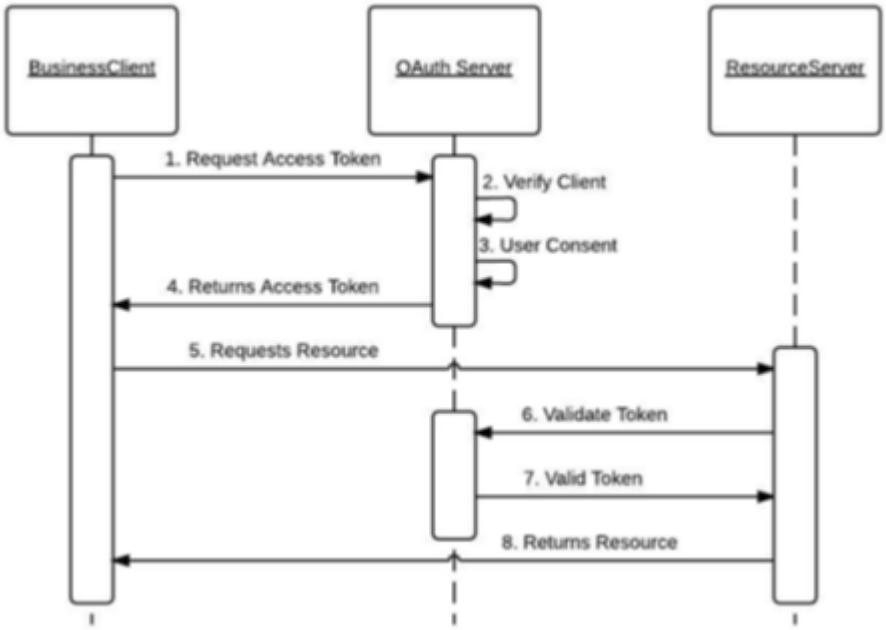
OAuth 2.0 Concepts

Business to Business (2-legged flow):

- It usually represents an application that calls another application or service without end-user intervention.
- A client (Business Client application) will make a call to a service, business service (in OAuth spec, a resource server), and request some business information while passing the access token.
- Because there is no end-user intervention, the client is pre-authorized to have access to the resource.

OAuth 2.0 Use Case Flow

Figure 6-2 OAuth 2.0 Use Case Flow



OAuth 2.0 Terms

- **Resource Server** - The server hosting the protected resource.
- **Resource Owner** - An entity capable of granting access to a protected resource.
- **Client** - An application making protected resource requests on behalf of the resource owner. It can be a server-based, mobile, or a desktop application.
- **Authorization Server** - The server issuing access tokens to the clients after successfully authenticating the resource owner and obtaining authorization.

OAuth2 Service Consumer

A step-by-step guide to retrieve a `clientId` and secret for `grant_type=Password` (Resource Owner Password Credentials) when configuring Logfile/WMS.

1. Create a screen using module `api/oauth2/applications`.
2. Log in to the Oracle WMS cloud using credentials `https://<wms-domain>/<env-name>/`.

   For example: `https://***.wms.ocs.oraclecloud.com/lgf_int_qa/`

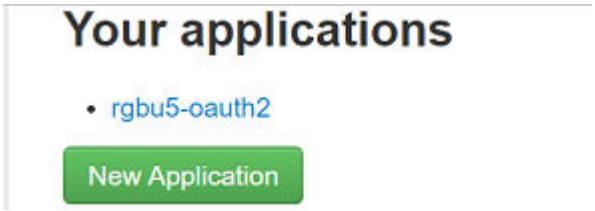
   - **Username:** <username>
   - **Password:** <password>
3. Append `api/oauth2/applications` to the above URI.

   For example:

```
https://<wms-domain>/<env-name>/api/oauth2/applications
```

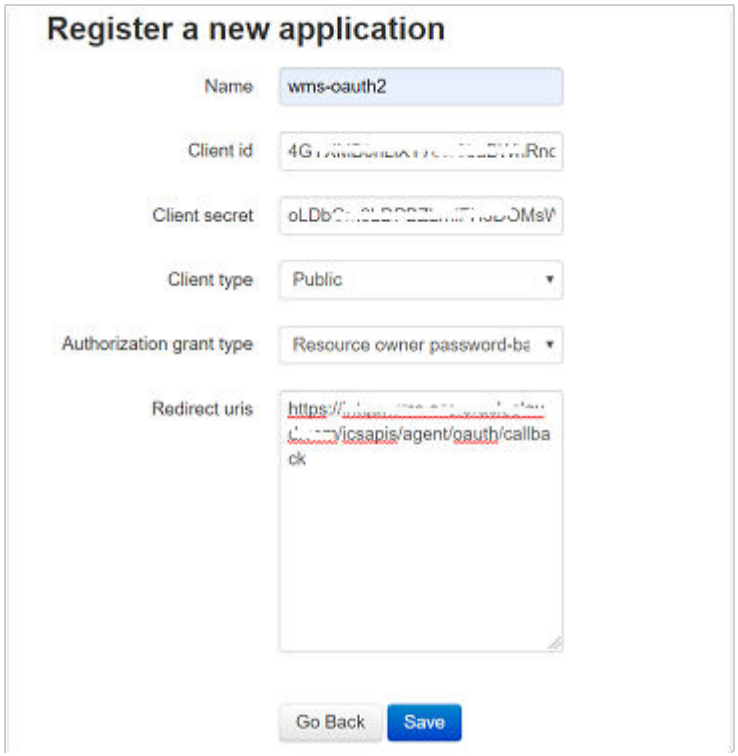
> **Note:**
>
> If you access the URL without first signing you, you will receive a "Forbidden error".

4. Open the URL created in step 3 in a web browser. The **Your applications** screen opens:



5. Click the **New Application** button.

   The **Register a new application** screen opens.



6. Register a new application using this screen.
   - Enter the **Provide Name**, **Client Type**, **Authorization grant type**, and **Redirect uris**.
   - **Client type** can be **public/confidential**.
   - **Client id** and **Secret** are generated.
7. Click the **Save** button.
8. Provide steps for grant type **Resource Owner Password Credentials**.

9. **Redirect uri** is optional for grant type **Resource Owner Password Credentials**, but without a URI, it is not able to register.

   So provide the **Redirect uri** as:

   ```
   <wms-domain>/<env-name>/icsapis/agent/oauth/callback - https://<wms-domain>/<env-name>/icsapis/agent/oauth/callback
   ```

10. Request an access token for grant type **Resource Owner Password Credentials**. **Scope** is optional.

    Enter the following values:

    - **Client id** - <Value generated in the 'Register a new application' screen (step 5-6)>
    - **Client secret** - <Value generated in the 'Register a new application' screen> (step 5-6)>

11. Retrieve the token using the **clientId** and **secret** through a `curl` statement.

    ```
    curl -v -X POST -u  "<ClientId>:<Secret>" -d "grant_type=password&username=<username>&password=<pwd>" <wms-domain>/<env-name>/api/oauth2/token/
    ```

    For example:

    ```
    curl -v -X POST -u  "<ClientId>:<Secret>" -d "grant_type=password&username=rgbu5_adm&password=welcome1#" https://***.wms.ocs.oraclecloud.com/lgf_int_qa/api/oauth2/token/
    ```

    A successful response will be in the following format:

    ```
    {"access_token": "<access-token>", "token_type": "Bearer", "expires_in": 36000, "refresh_token": "<refresh-token>", "scope": "read write"
    ```

12. Test the token by accessing the Logfile URL using the access_token with a curl statement:

    ```
    curl -X POST -i -H 'Authorization: Bearer <access-token>' \
    'https://***.wms.ocs.oraclecloud.com/lgf_int_qa/wms/api/init_stage_interface/' --data "@./ItemLgfDataNoNewLine.xml"
    ```

    A successful response has the following format:

    ```
    <?xml version="1.0" encoding="utf-8"?>
    <root><success>True</success><response><message>Stage table processing com-plete</message></response></root>
    ```

Access Logfire Services Using OAuth2 Consumer

The Logfire services are consumed by using the following security policies:

- Basic Authentication 2.OAuth2.

  By configuring this property in the configuration file, you can switch between "basic" and "oauth2" authentication.

- OAuth2 Consumer Configuration:

Table 6-1 external_env_info.LogFireIntegration.json

Configuration Property	Description
`"name": "Lgf_Oauth2_Authentication","value": "true"`	1. To enable OAuth for logfire, change the value of flag `Lgf_Oauth2_Authentication` to `true`. 2. To enable basic authorization for logfire, change the value of `Lgf_Oauth2_Authentication` to `false`.
`"name": "lgf_oauth2_alias_key", "value": "lgfOauth2ApplicationClientAlias"`	Save the **ClientId** and **Secret** in the credential store using the alias `lgfOauth2ApplicationClientAlias`.
`"name": "LogFire_Host_Url_Key","value": "<Logfire Login URL>`	Logfire URL used for the OAuth token.

After receiving a Logfire **clientId** and **secret** from the above steps:

1. Store these credentials in the credential store for further reference in the USM application to create an OAuth token.

   Once the OAuth token is issued, further API calls are made.

2. Save the **clientId** and **secret** in the credential store with the alias name `lgfOauth2ApplicationClientAlias`, as defined in the JSON.

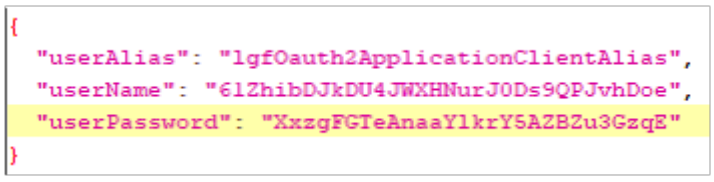
   The USM application uses this alias to make a call to Logfire and retrieve the OAuth token. Once obtained, the OAuth2 token services calls are made.

3. Pass the JSON request to the service. This saves the credentials (**clientId**/**secret** combination).

   JSON request format:

```
{
"userAlias": "<Alias>",
"userName": "<Id>",
"userPassword": "<password>"
}
```

   For example:



USM consumer simplifies access of services protected by OAuth 2.0. The USM consumer executes the following steps:

1. Gets the token from the Logfire server using client ID, client secret, and scope.
2. Adds the "Authorization Bearer <token>" HTTP header.
3. Calls the service.