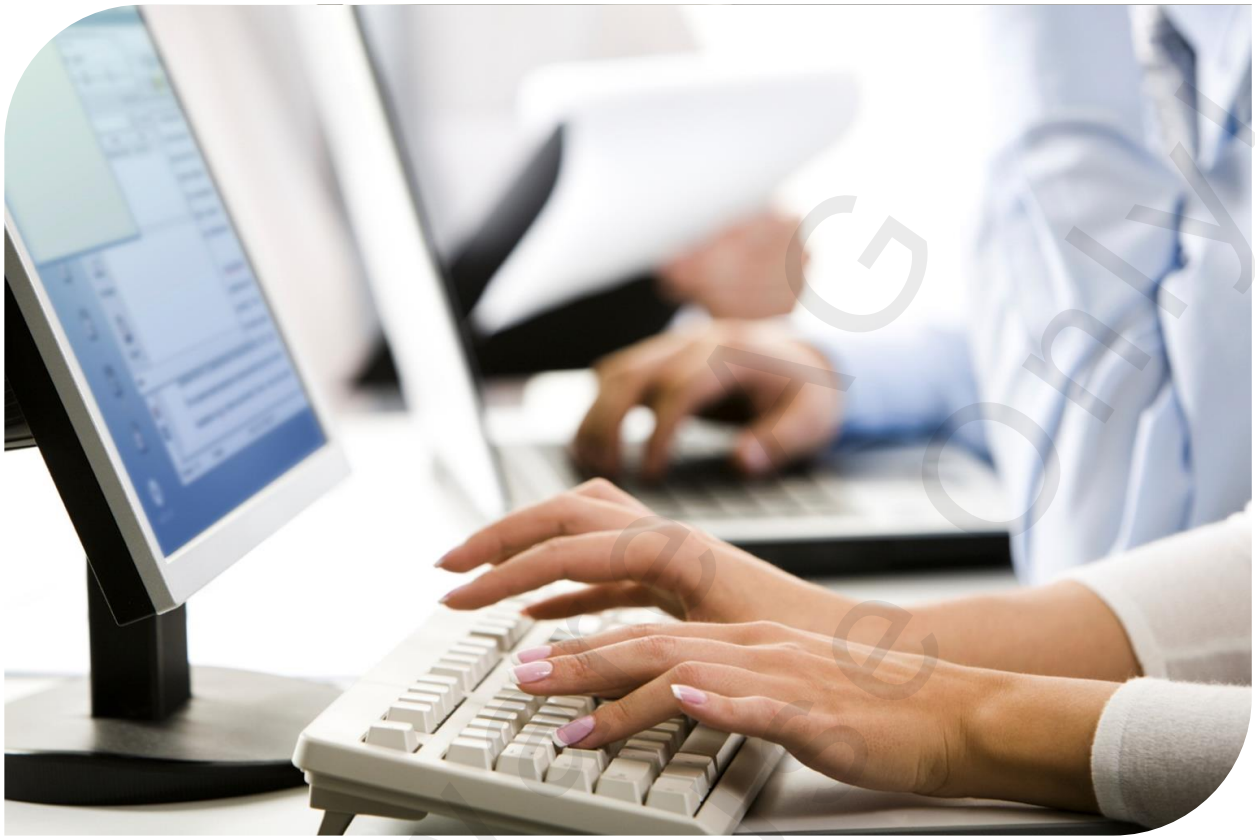


# EXERCISE GUIDE

# API MANAGEMENT WITH webMETHODS PLATFORM

456-71E

# TABLE OF CONTENTS

Introduction: Current Situation of the SAGTours Company ..... 5
Exercise 1: How to set up API Gateway ..... 7
Exercise 2: Creating a REST API using a Swagger File ..... 15
Exercise 3: Identify Business Objects and configure REST APIs ..... 19
Exercise 4: Creating Rest API from Scratch ..... 21
Exercise 5: Adding a Logging Policy ..... 37
Exercise 6: Enforcing Security Policy ..... 43
Exercise 7: Using a WSS Security Token ..... 47
Exercise 8: Creating a Consumer Application ..... 55
Exercise 9: Enforcing a Consumer Relationship using API Key ..... 61
Exercise 10: Managing API Threat Protection ..... 73
Exercise 11: Manage Fine Granular Exposure of APIs ..... 87
Exercise 12: Monitoring the Virtual API ..... 95
Exercise 13: Using a Routing Policy Action ..... 99
Exercise 14: Updating Content Using XSLT Stylesheet ..... 105
Exercise 15: Setting up Integration with API Portal Objectives ..... 109
Exercise 16: Publishing API to API-Portal ..... 113
Exercise 17: Publishing SAGTours APIs to API-Portal ..... 117
Exercise 18: Using API Portal as Consumer ..... 127
Exercise 19: Send Events to API Portal ..... 137
Exercise 20: Managing API Packages and Plans ..... 143
Exercise 21: Try out API secured with OAuth2 token ..... 153
Exercise 22: Editing APIs in API Portal ..... 157



This page intentionally left blank

# Introduction:
# Current Situation of the SAGTours Company

## Overview

SAG Tours is a leading provider of yachting holidays offering worldwide yachting trips. It offers pre-arranged tours that can be booked by individuals (Sailing Cruises) as well as yacht chartering (Charter Cruises). SAG Tours primarily operates through its retail agency business.

## SAG Tours Future Goals

SAG Tours has recognized the opportunity that APIs allow to generate additional revenue.

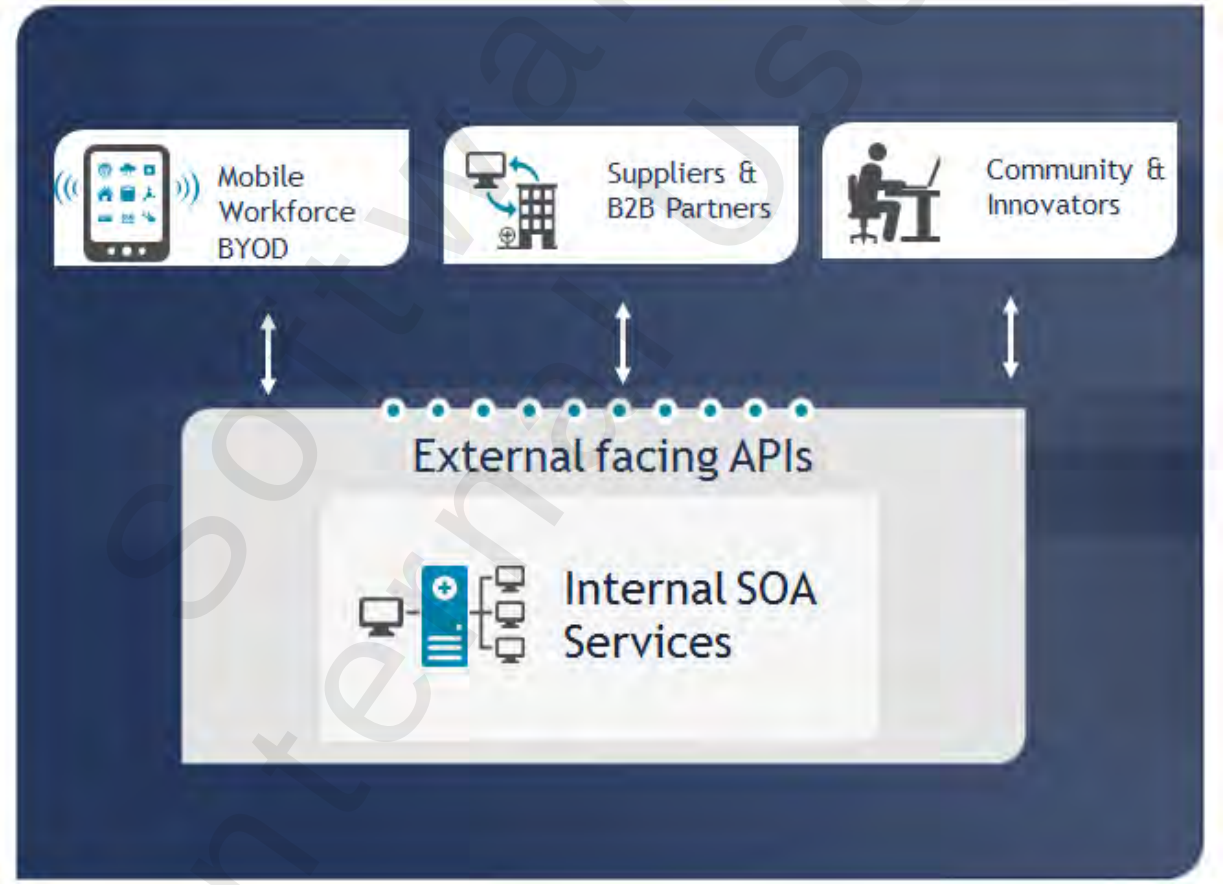
- Support Mobile initiatives and use APIs to serve data to mobile devices
- Leverage APIs for supply-chain and ecommerce initiatives to simplify B2B integration and expand channel outreach
- Encourage external innovation through the community and create incremental revenue

Therefore SAGTours has decided to start an open API initiative along with its traditional sales channels. SAGTours expects that these parties will create innovative new travel applications and mobile applications with the content coming from SAGTours provided through its APIs. Through APIs they want to allow third parties to access booking capabilities that SAGTours uses in its current channels.

To support their API initiative, SAGTours plan to host an API developer portal, to expose their APIs to consumers.

The service/APIs of the booking application should be registered/defined within CentraSite, which is the SAGTours registry and repository. For Runtime Governance SAGTours will use webMethods API Gateway for aspects like logging, security, consumer based monitoring, …..

The API developer portal will be built using webMethods API-Portal. All APIs SAGTours wants to expose to consumer will be published to the API Portal, so that internal and external consumers can search for APIs using API Portal.



# Exercise 1:
# How to set up API Gateway

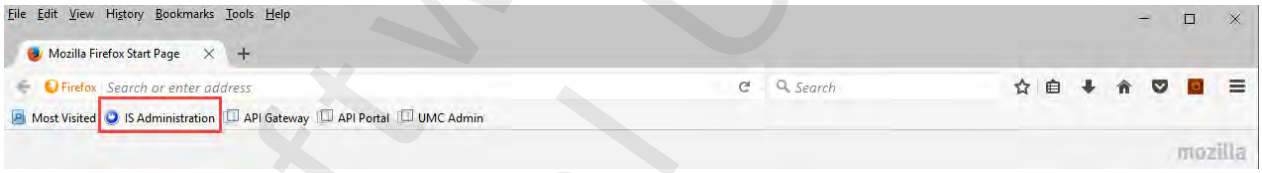
## Objectives

In this exercise, you will set up the environment for API Gateway and the SAGTours scenario.

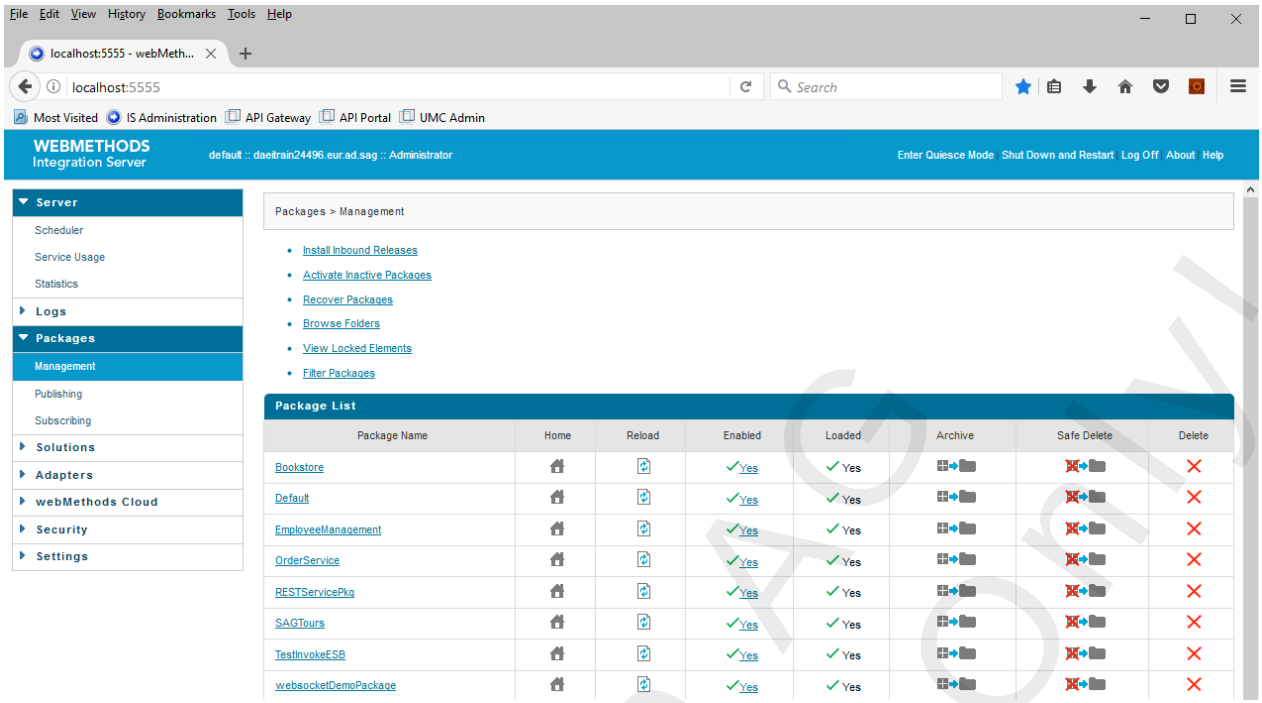
## Steps

1. Open **Windows Services UI** to double check that the following services, needed for API Gateway, are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1 (default)**
      i. **Make sure to start Software AG Integration Server and not the one which is named Internal Integration Server**
   b. **Software AG Event Data Store 10.1 – will be started with Integration Server**
      ii. **It is important to verify that the Event Data Store is started, as API Gateway requires that**
   c. **Software AG Runtime 10.1**
      i. **This service is important so that the SAGTours services are available on port 53307**
2. Login to **Integration Server Administration** by opening **Firefox** and clicking the **IS Administration** link (just below the URL field). Login as Administrator (**Administrator / manage**).

   **Note:** The Integration Server takes a few minutes to start, therefore you may have to refresh the Administration page until you are asked to login.




3. Select **Packages -> Management** and enable the following packages:
   a. **Bookstore**
   b. **SAGTours**
   c. **TestInvokeESB**
   d. **MediatorECRWebservices**




4. Login to **API Gateway** by opening **Firefox** and clicking the **API Gateway** link (just below the URL field). Login as Administrator (**Administrator / manage**).

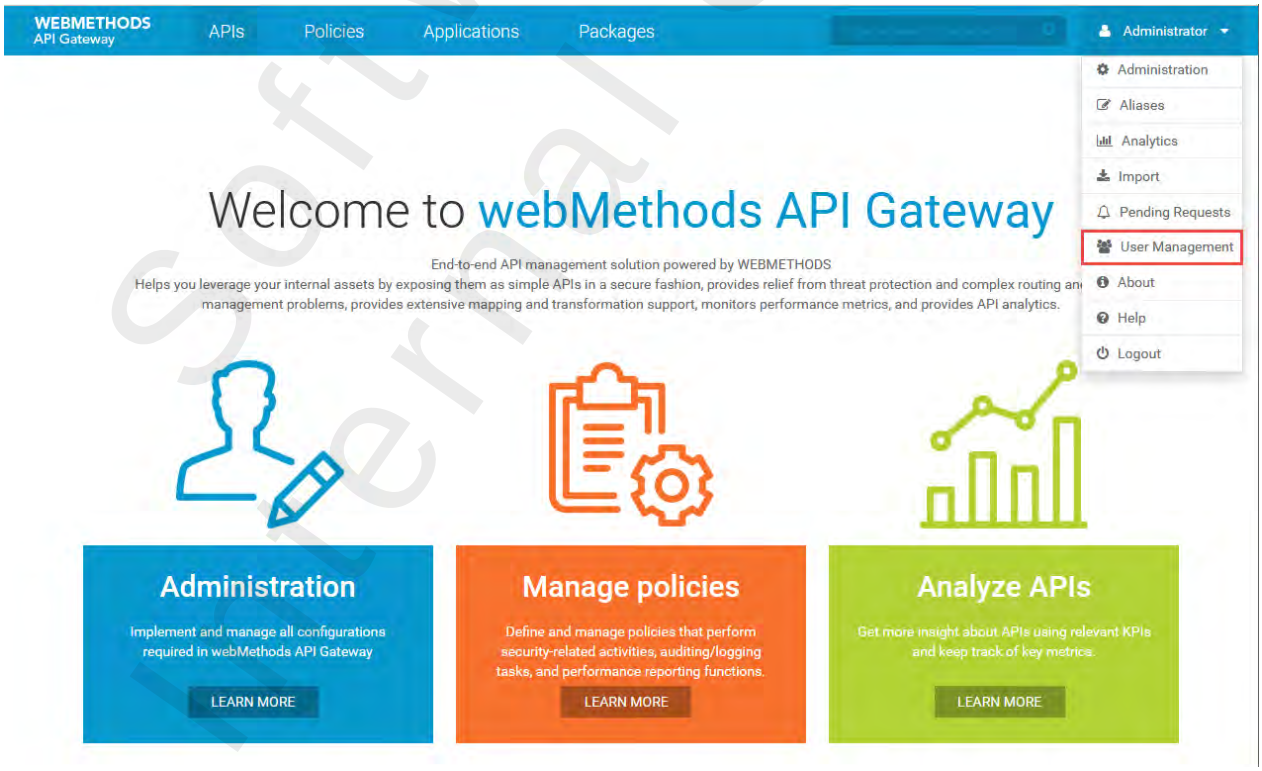
**5.** We want to define different users who have different responsibilities in API Gateway like

   a. Create APIs in API Gateway

   b. Publish APIs to API Portal

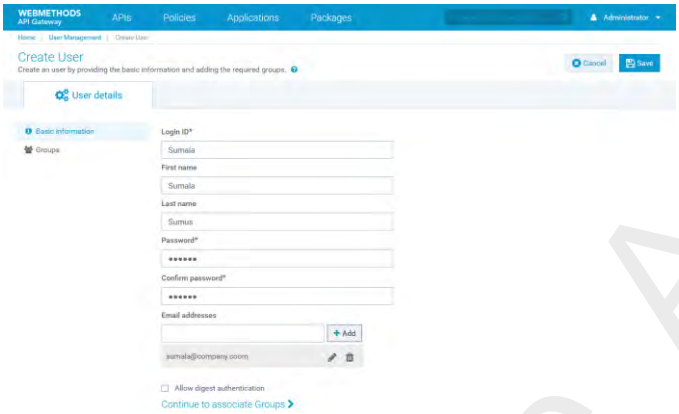
   c. Approve Consumer

   Navigate to **Administration > User Management**.

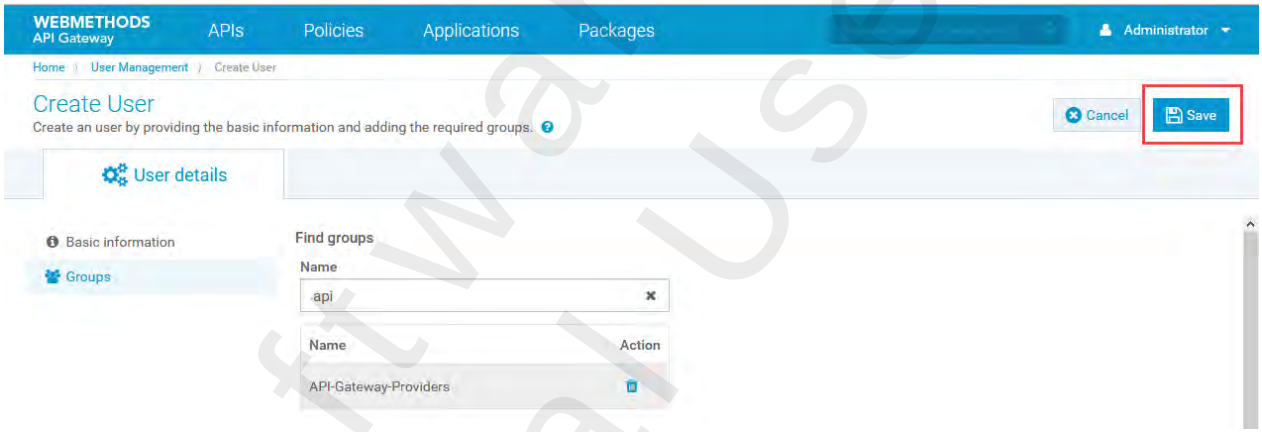



**6.** Within the **Users** tab click **Add User**. Provide the following properties:

a. **Login ID:** Sumala
b. **First Name**: Sumala
c. **Last Name**: Sumus
d. **Password:** manage
e. **Email Address:** sumala@company.com
   i. Click **+Add**




f. Click **Continue to associate Groups >**
   i. Add to group: **API-Gateway-Providers**




Click **Save.**

7. Create a new user with the following properties:
   a. **Login ID:** Andy
   b. **First Name**: Andy
   c. **Last Name**: Roth
   d. **Password:** manage
   e. **Email Address:** andy@company.com
      i. Click **+Add**
   f. Click **Continue to associate Groups >**
      i. Add to group: **API-Gateway-Administrators**

   Click **Save.**

8. Create a new user who is responsible with the following properties:
   a. **Login ID:** Peter
   b. **First Name**: Peter

c. **Last Name**: Green

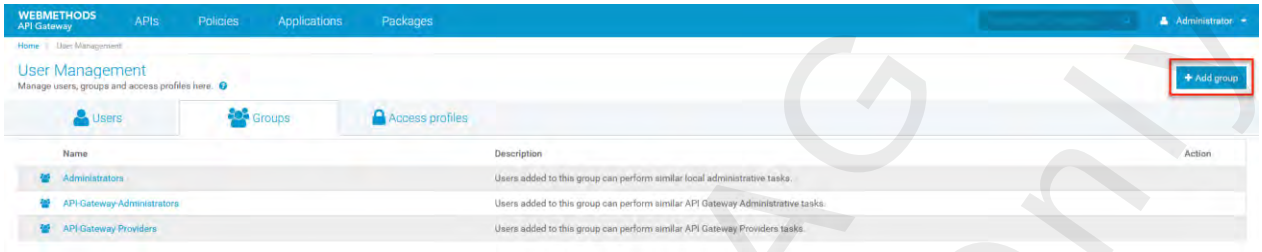
d. **Password:** manage

e. **Email Address:** peter@company.com

i. Click **+Add**

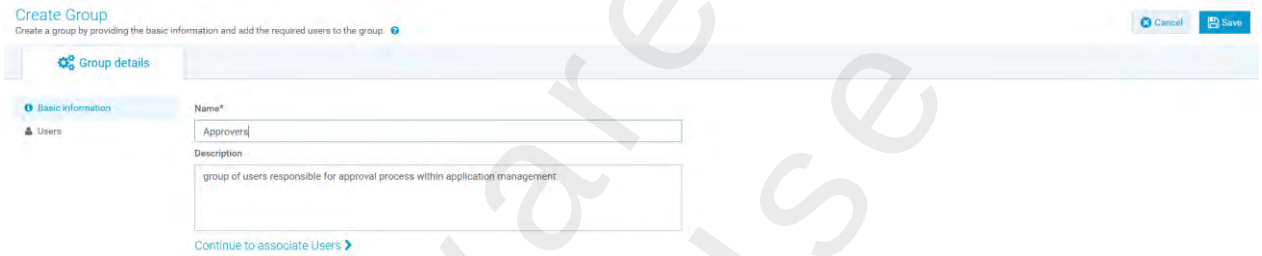
Click **Save.** Without adding Peter to a group.

9. Later on in the exercises we need a group of users who are approvers of specific processes. Navigate to **User Management > Groups.** Click **Add group.**

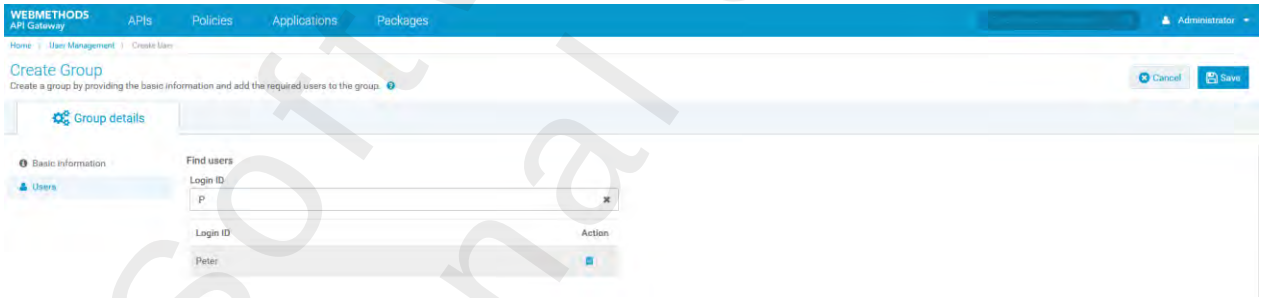



a. **Name:** Approvers

b. **Description:** group of users responsible for approval process within application management

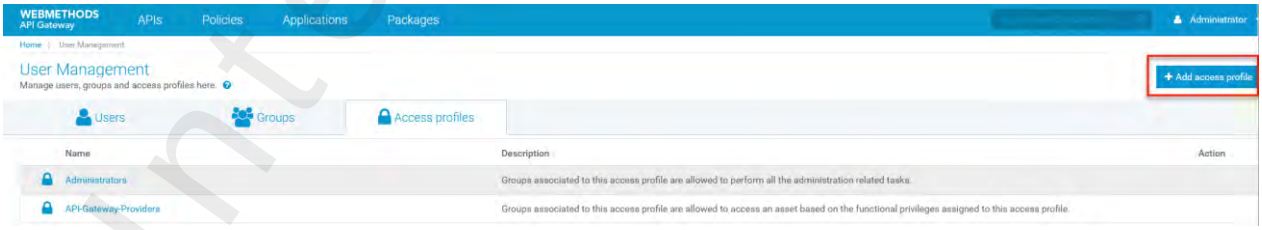



Click **Continue to associate Users**. Provide character P for **Login ID**. From the list of users select **Peter**. Click **+**.




Click **Save**.

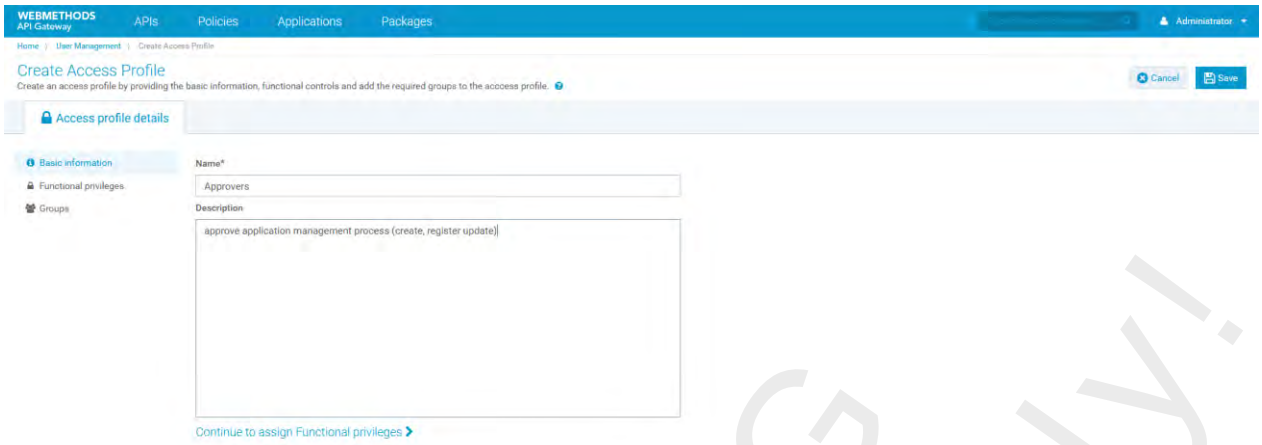
10. Now we need to set up an appropriate access profile for the approver group. Navigate to **User Management > Access profiles**. Click **Add access profile.**




Provide the following properties:

a. **Basic Information:** Approvers

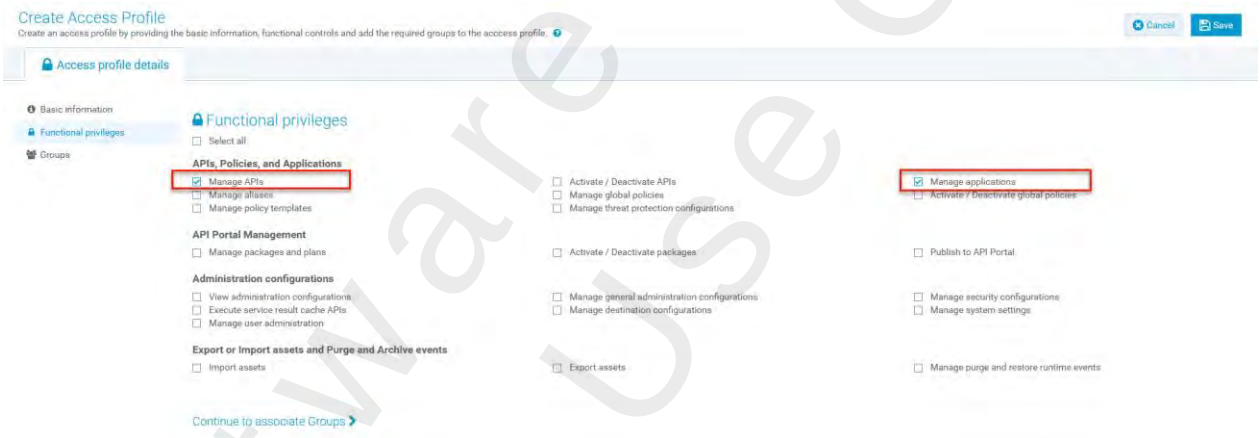
b. **Description:** approve Application management process (create, register, update)



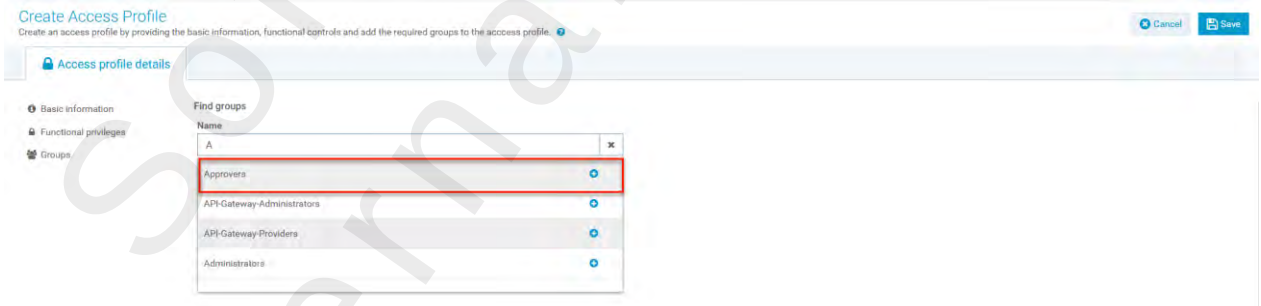

Click **Continue to assign Functional privileges**. Select the following **Functional privileges > APIs, Policies, and Applications**

c. **Manage APIs**

d. **Manage Applications**

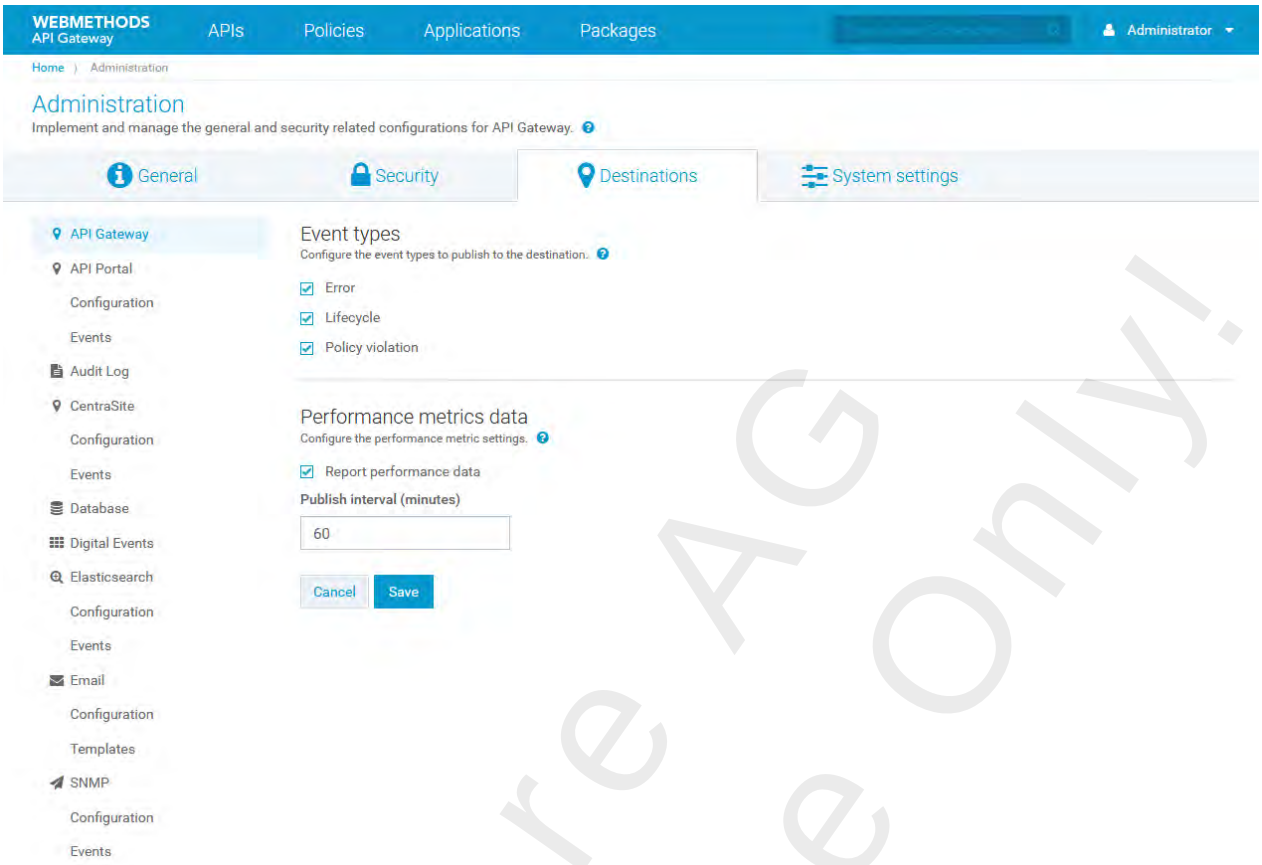



Click **Continue to associate Groups.** Provide character A for **Name**. From the list of groups select **Approvers**. Click **+**.




Click **Save.**

11. Navigate to **Administration > Destinations.**




12. The destination **API Gateway** is preselected. Edit the configuration and set the following values: Some of the settings are already done, and some need to be adapted.

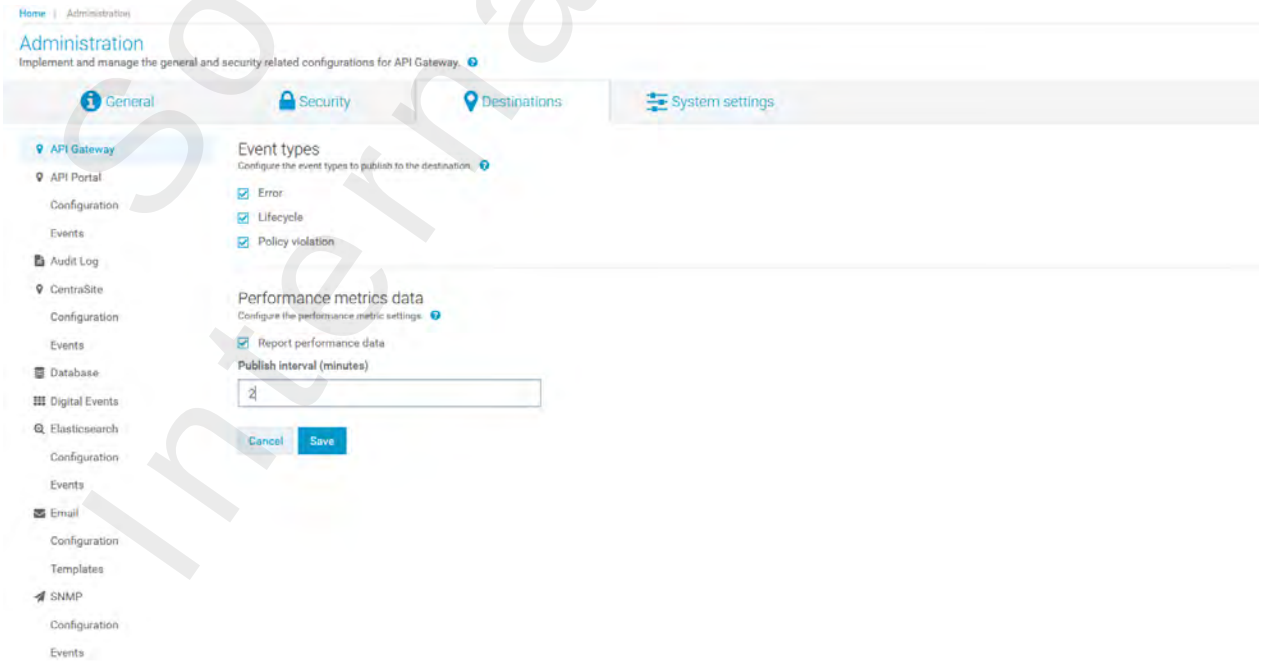
    a. **Error**: - select -

    b. **Lifecycle**: - select –

    c. **Policy violation**: - select –

    d. **Report Performance data**: - select –

    e. **Publish interval (minutes)**: 2



Click **Save.**

This page intentionally left blank

# Exercise 2:
# Creating a REST API using a Swagger File

## Objectives

In this exercise, you will add a  REST Service to API Gateway. You will run the service and look for the performance metrics information in API Gateway

To create and manage services in API-Gateway, users must belong to the following group in Integration Server:

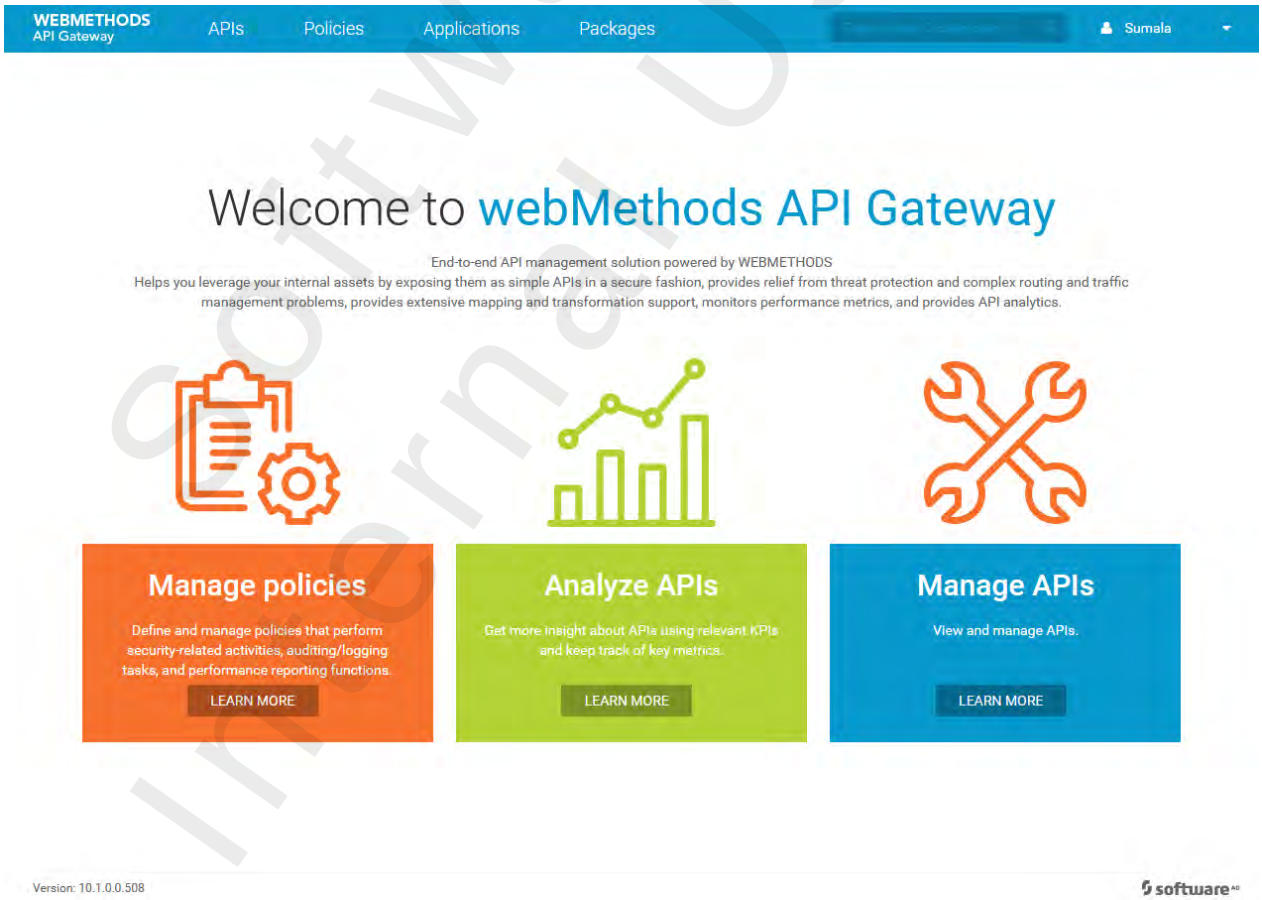
- API-Gateway-Provider

We will keep our exercise as simple as possible**. Sumala** will be the service provider.

## Steps

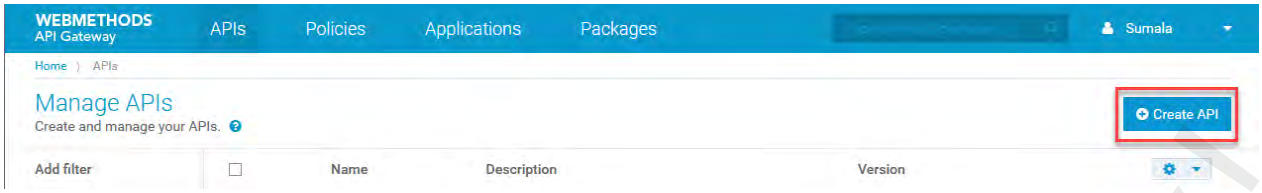
1. Open **Windows Services UI** to double check that the following services, needed for API Gateway are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Event  Data Store 10.1 – will be started with Integration Server**
2. Start **Mozilla Firefox** and **l**ogon to **API Gateway** as user **Sumala/manage**.
3. Create a new asset of type **REST** with name **Bookstore** using the import functionality of API Gateway.

   Click **Manage API** within the Home Page



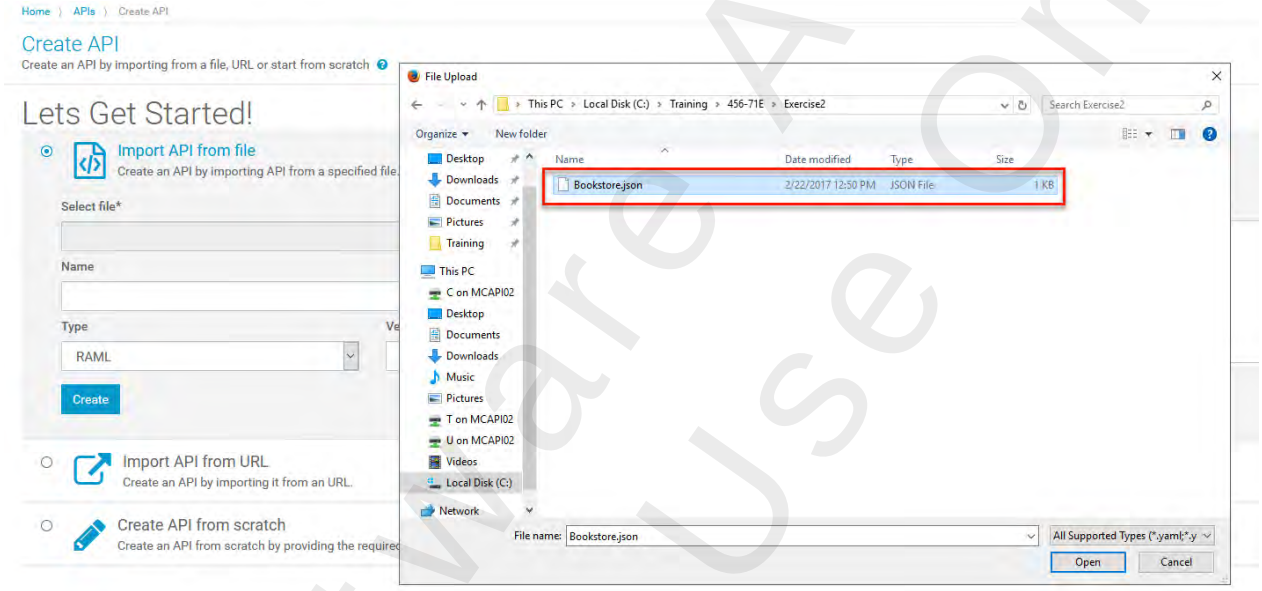

   or navigate to **APIs** in the header section.


4. Click Create API.




A shortcut to this would be using the **Create API** on the Home page.

5. We would like to import a REST API from file. Therefor we leave the default selection. Within the **Import API from File** section click **Browse**. Navigate to **C:/Training/456-71E/Exercise2** and select **Bookstore.json**.



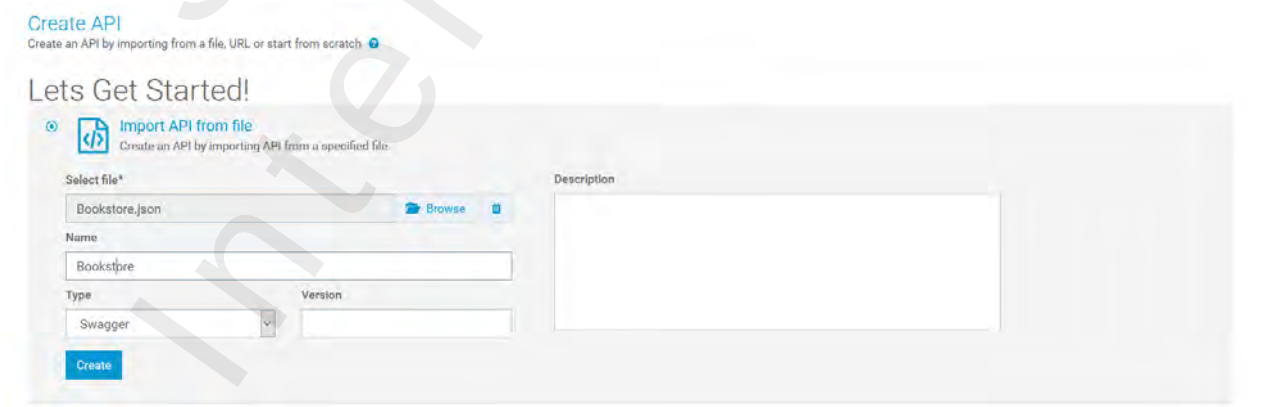

Click **Open**.

Provide the following properties:

**Name**: Bookstore

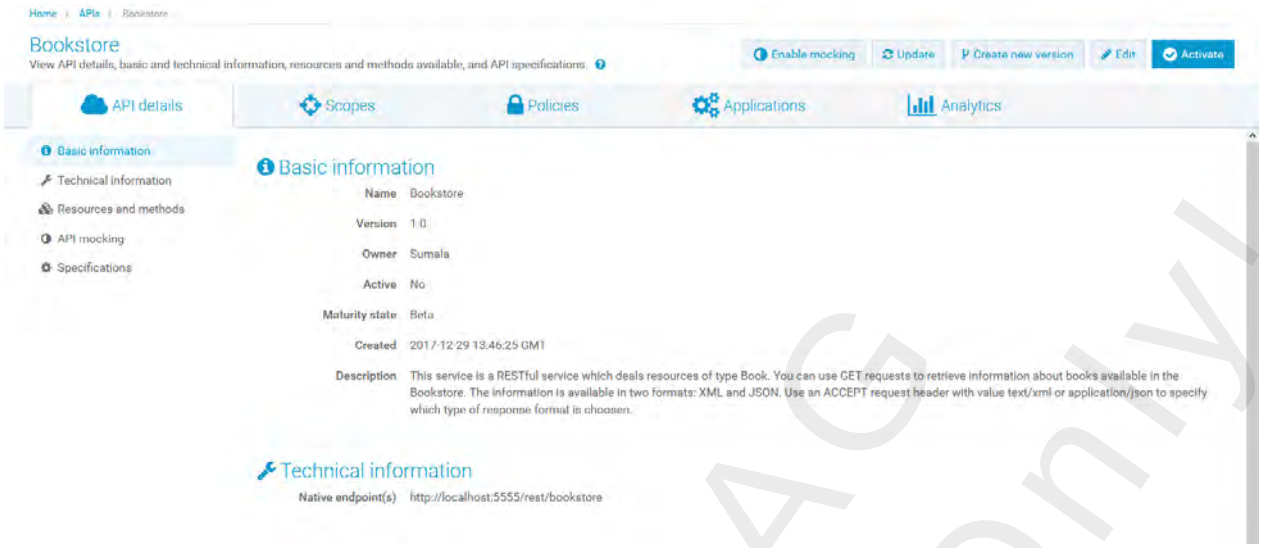
**Type:** Swagger

**Version**: 1.0



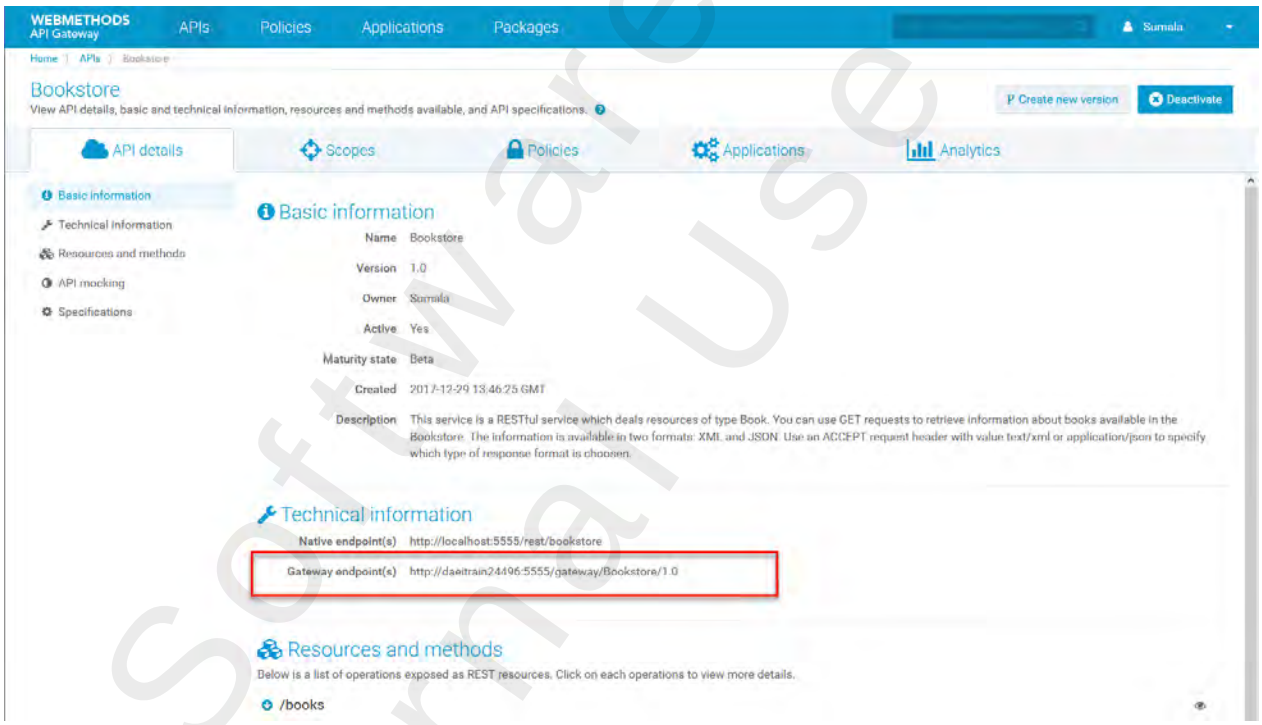

Click **Create.**

6. Review the properties of the generated API.




7. Click **Activate**. Click **Yes**.

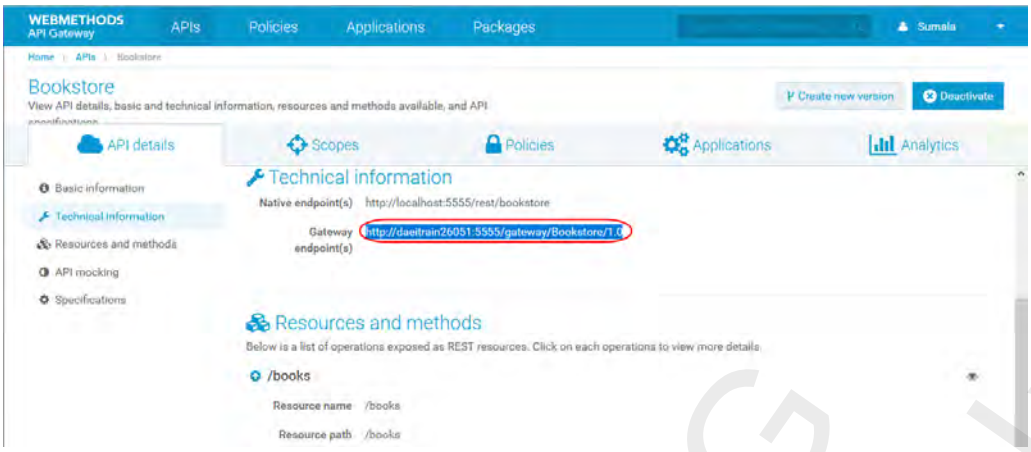
8. Review the API details. Scroll down to **Technical Information** to get the gateway endpoint .



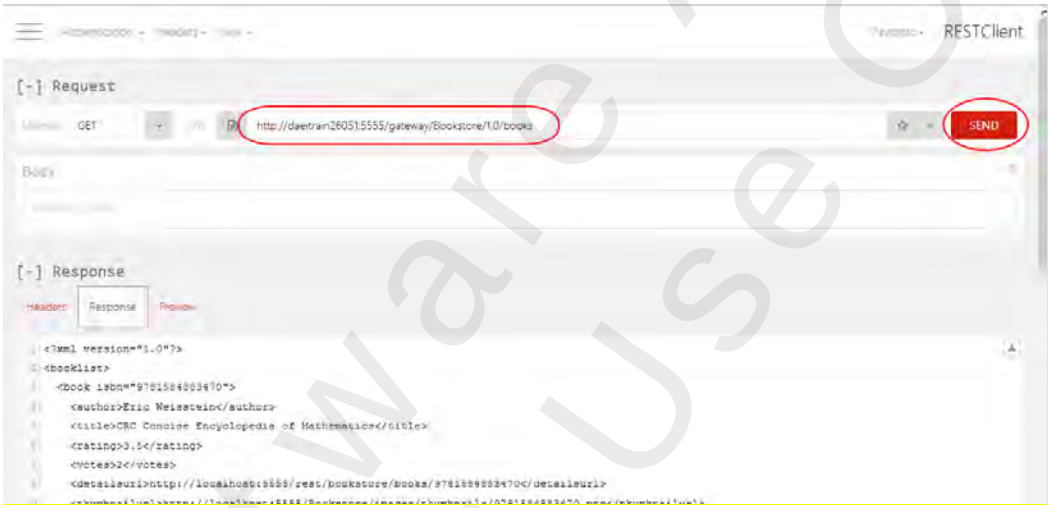

9. Open **Resources and methods** to review the registered Resources. Select the resource /books to view the resource information and the assigned methods.

10. Explore consuming this API:

    a. Copy the Gateway URL as shown, then click on the REST Test Client icon;




b. Paste the URL into the REST client and append “/books” to the URL; the click “SEND”.



# Exercise 3:
# Identify Business Objects and configure REST APIs

## Objectives

In this exercise, you will discuss and think about which Business Objects we have in our current application and how we would define the REST APIs for our consumers

## Steps

1. Design the APIs which **SAGTours** would create in order to expose their Business to outside of the organization
   a. Which business services exist and which are useful to expose as APIs?
   b. How to configure REST APIs using best practices?
      i. Which resources do we have, Path Definition
      ii. Which HTTP methods would we like to support
      iii. Which Parameters would we like to provide
         1. Path parameters
         2. Header Parameter
         3. Query parameter
   c. Think about using best practices like
      i. Partial Responses
      ii. Limit Results
      iii. Pagination
      iv. Server Side Sorting
      v. Design for Versions
   d. Think about roles and permission
      i. API Administrator
      ii. API Developer
      iii. API Provider
      iv. API Consumer
2. This is a Sample Result which we will use in our workshop to expose the SAGTours application. We will add these entities to API Gateway and expose to Consumers:
   a. Following resources
      i. Search just using HTTP Methods: GET
         1. /cruises
         2. /cruises?
         3. /cruises/{cruiseID}
      ii. Booking
      iii. SignUp
   b. Following Groups, Users, Roles
      i. Andy as API as owner of Runtime Environment



This page intentionally left blank

# Exercise 4:
# Creating Rest API from Scratch

## Objectives

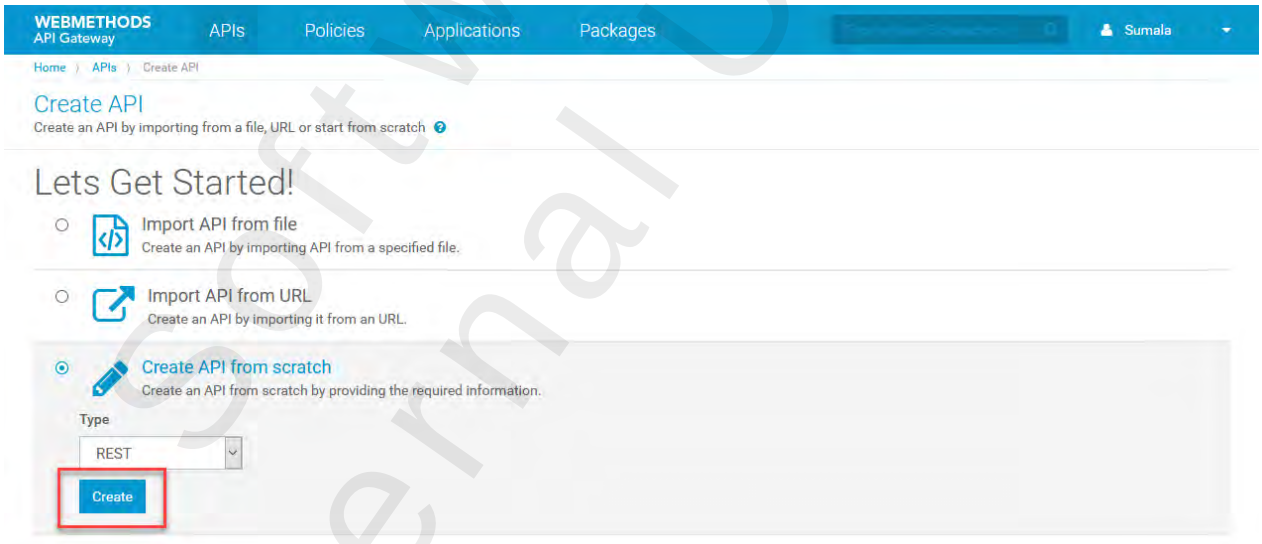
Create a REST API from Scratch.

## Steps

1. Open **Windows Services UI** to double check that the following services, needed for API Gateway, are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Event  Data Store 10.1 – will be started with Integration Server**
2. Logon to **API Gateway** as user **Sumala/manage**.
3. Create a new asset of type **REST** from Scratch. The service enables users to search for tours offered by **SAGTours**. The Search service provides the following search options:
   - Show all available cruises
   - Show cruises based on filter criteria
   - Show a specific cruise based on its cruise ID.

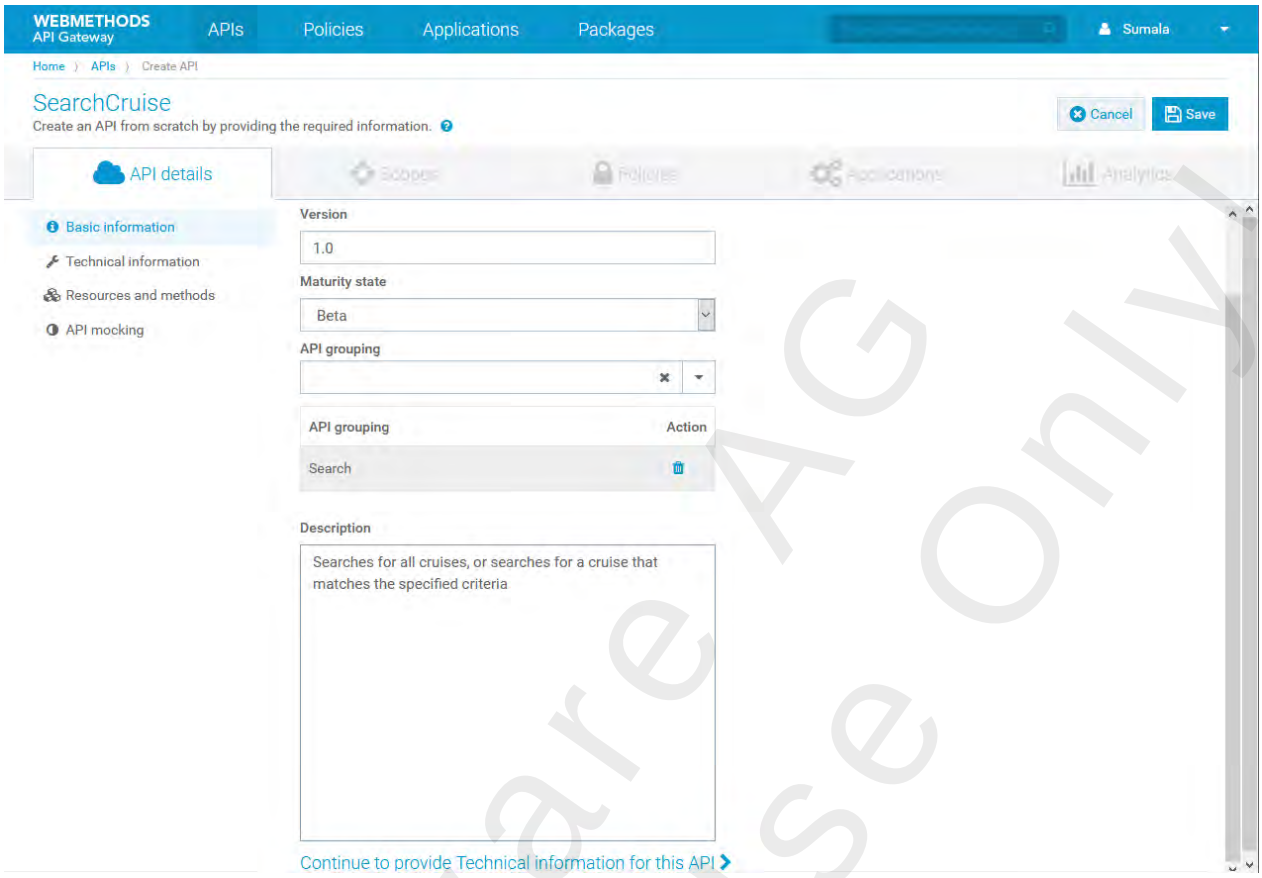
   If an error occurs, the API returns an HTTP status code and, when appropriate, a plain text message in the message body providing the reason for the error.

   Navigate to **Manage APIs,** click **Create API. S**elect **Create REST API from scratch** and click **Create.**




4. Add properties of new REST service – you can copy and paste content out of file **CruisesAPIResourceCruise.txt** in folder **C:\Training\456-71E\Exercise4**
   c. Basic Information:
      i. **Name**: SearchCruise
      ii. **Version**: 1.0
      iii. **Maturity state**: Beta
      iv. **API grouping**: Search

v. **Description**: Searches for all cruises, or searches for a cruise that matches the specified criteria.



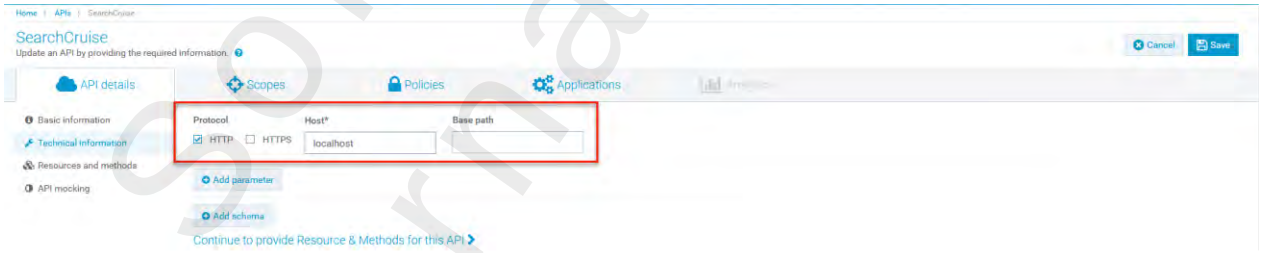

Click **Continue to provide Technical information for this API>**.

d. **Technical information:**

i. **Protocol**: HTTP

ii. **Host:** localhost

iii. **Base path:**




Click **Continue to provide Resource & Methods for this API>**.

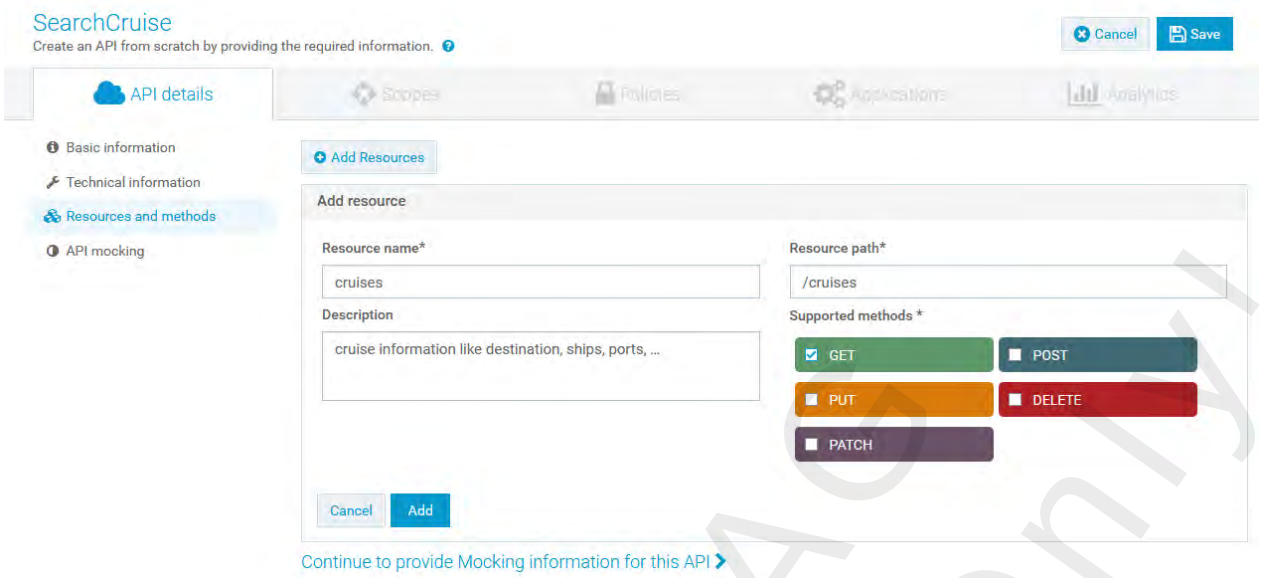
e. **Resources and Methods**:

i. **Resource name:** cruises

ii. **Resource path:** /cruises

iii. **Description:** cruise information like destination, ships, ports, …

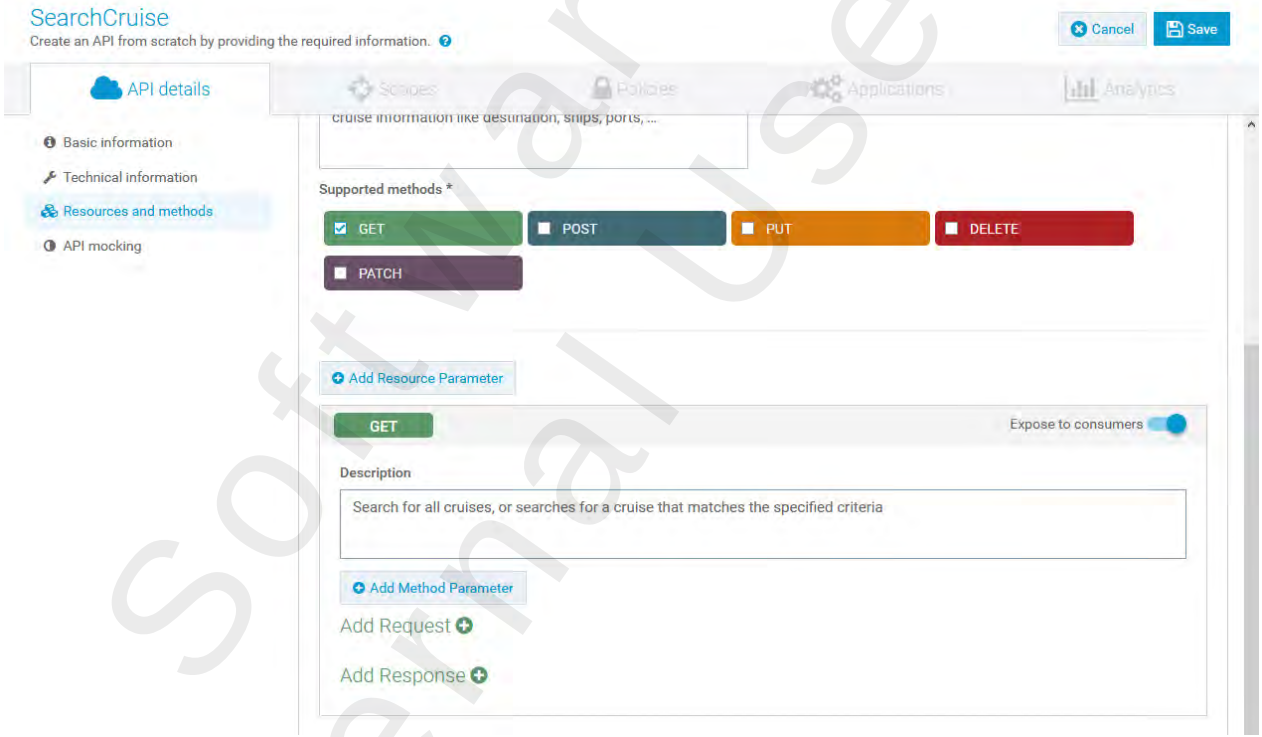
iv. **Supported Methods:** GET




Click **Add**.

f. Scroll down to **GET** methods and provide the following properties.

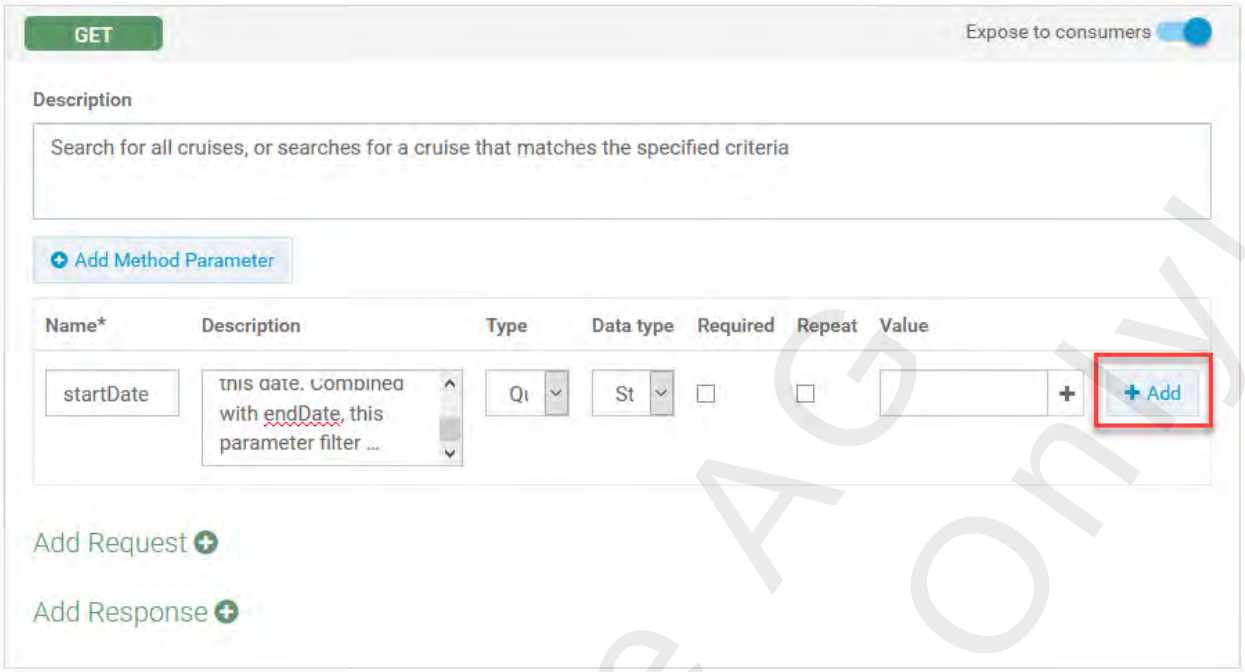
i. **Description:** Search for all cruises, or searches for a cruise that matches the specified criteria




g. Click **Add Method Parameter** section and provide the following properties

1. **Name**: startDate
2. **Description**: Cruise start date, in the format YYYYMMDD. Selects cruises that start on this date. Combined with endDate, this parameter filter …
3. **Type**: Query-string
4. **Data Type**: String
5. **Required:** - leave empty –
6. **Array:** - leave empty –

7. **Value**: - leave empty –

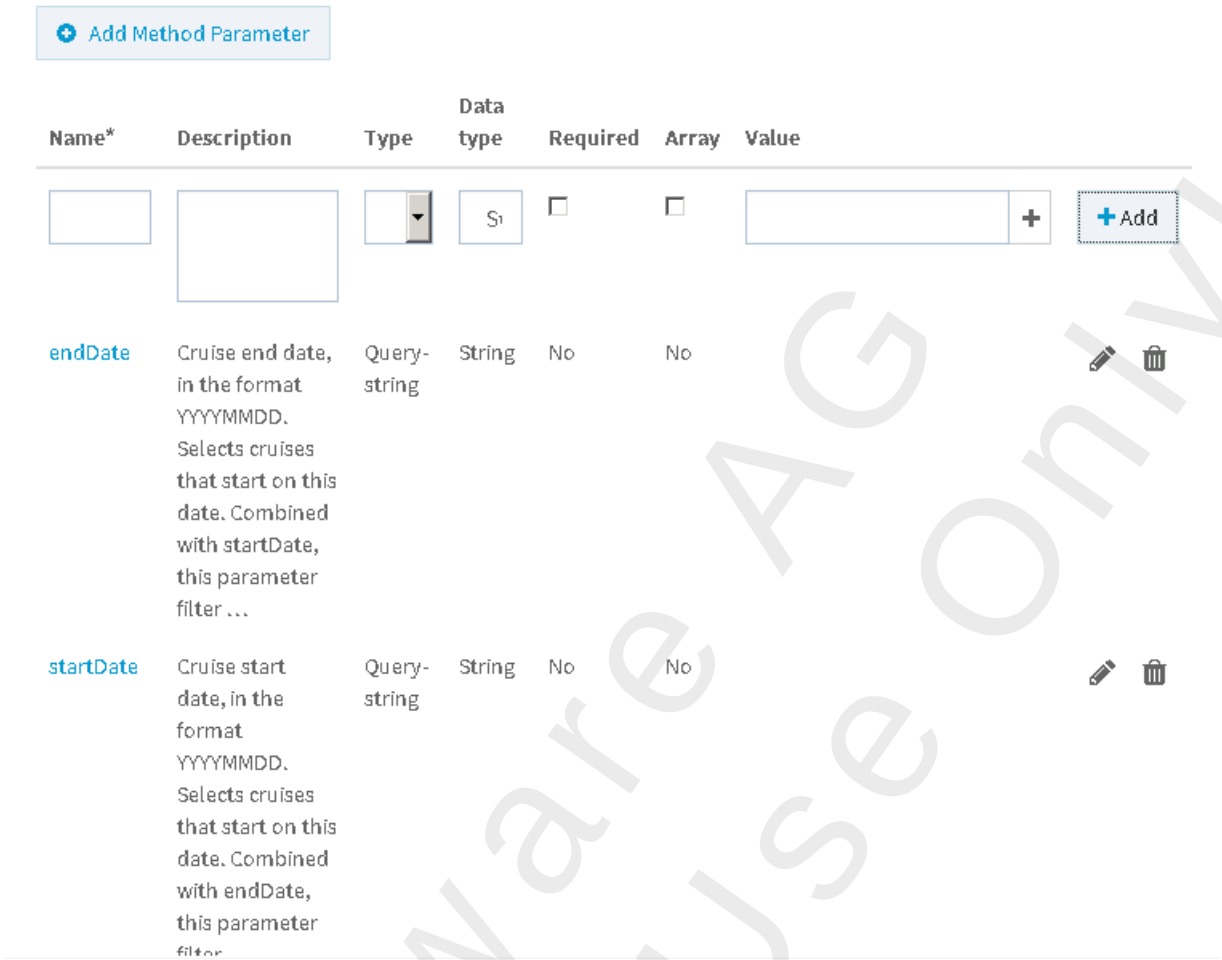



Click **+Add**.

h. Click **Add Method Parameter** section and provide the following properties

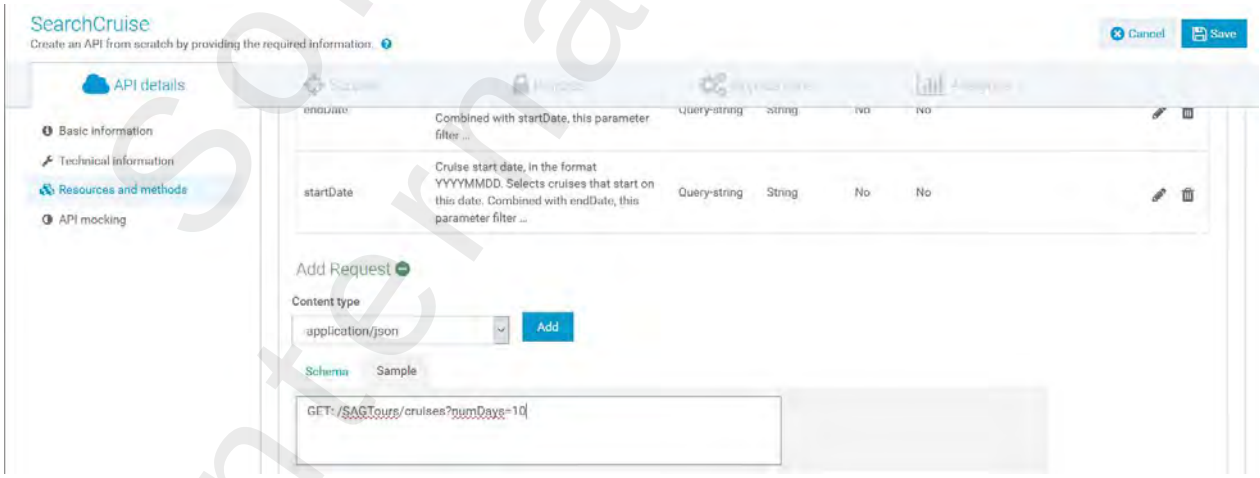
1. **Name**: endDate
2. **Description**: Cruise end date, in the format YYYYMMDD. Selects cruises that start on this date. Combined with startDate, this parameter filter …
3. **Type**: Query-String
4. **Data Type**: String
5. **Required:** - leave empty –
6. **Array:** - leave empty –
7. **Values**: - leave empty –

Click **+ Add**.




i. Click **Add Request** link to define the supported content type of the API. Select **application/json** from the list and click **Add**.

   i. Switch to sample and copy and paste the sample request from the txt-file **CruisesAPIResourceCruise.txt** available in **C:\Training\456-71E\Exercise4**

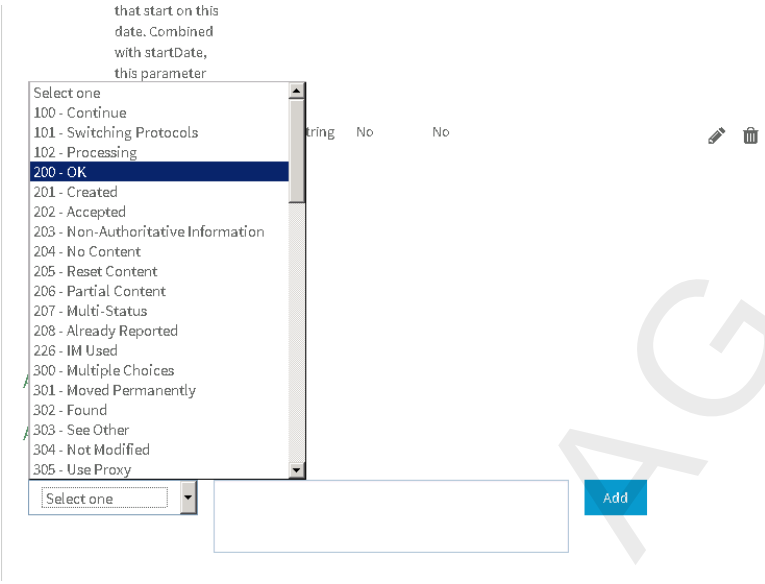



   Click **Add**.

j. Click **Add Response** link to add HTTP Status Code. Use the + icon.

   1. **Status Code**: 200 OK

2. **Description:** Search (with filter) was successful. GET method returned a list of all cruises that match the specified criteria, in JSON format.



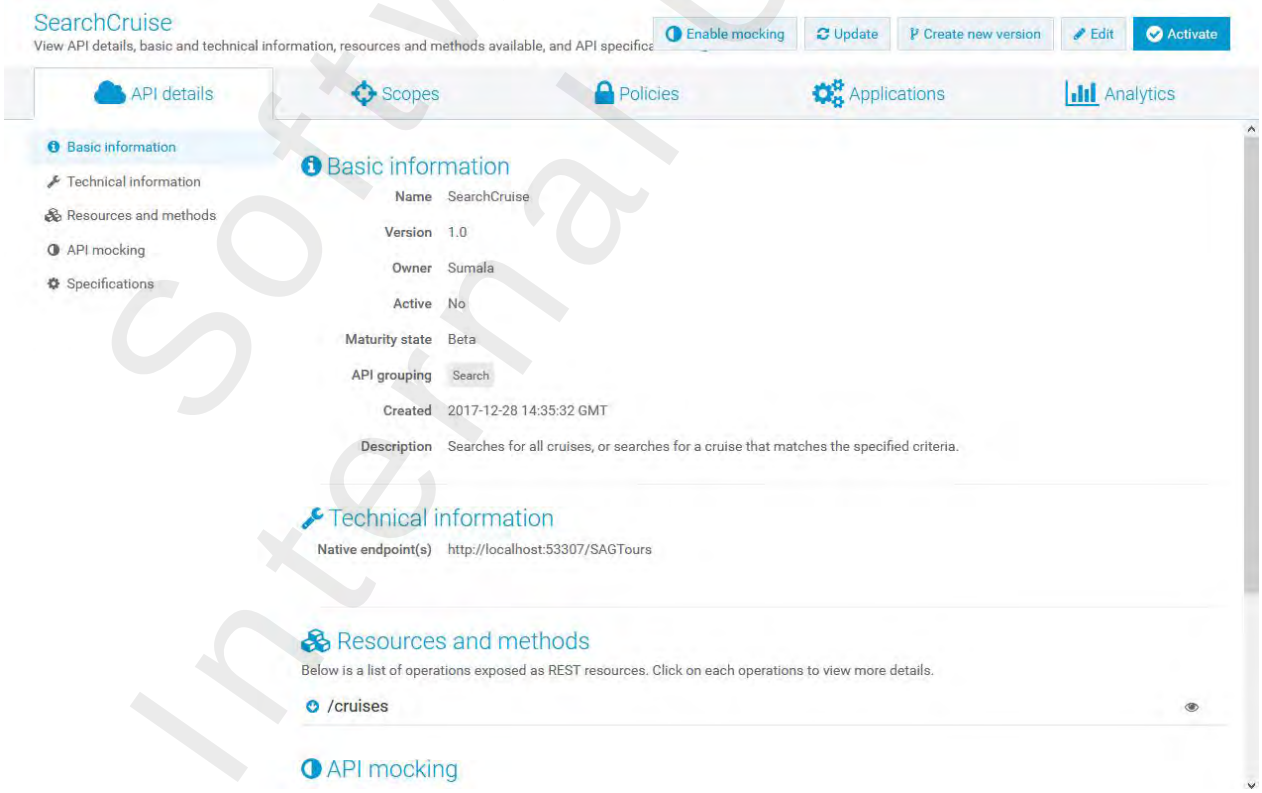

Click **Add**.

k. Add another Response Code:

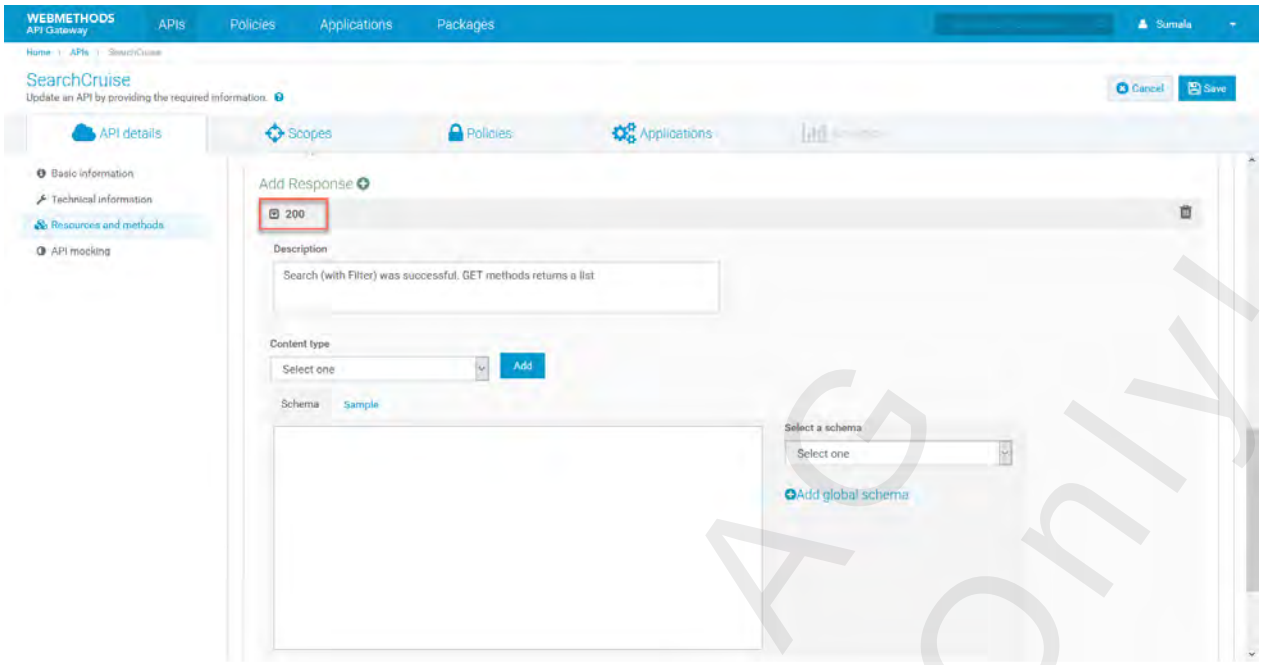
1. **Status Code**: 400 Bad Request
2. **Description:** Query failed because the filter parameters are not valid. Check the resulting message for detail

Click **ADD**.

11. Click **Save**



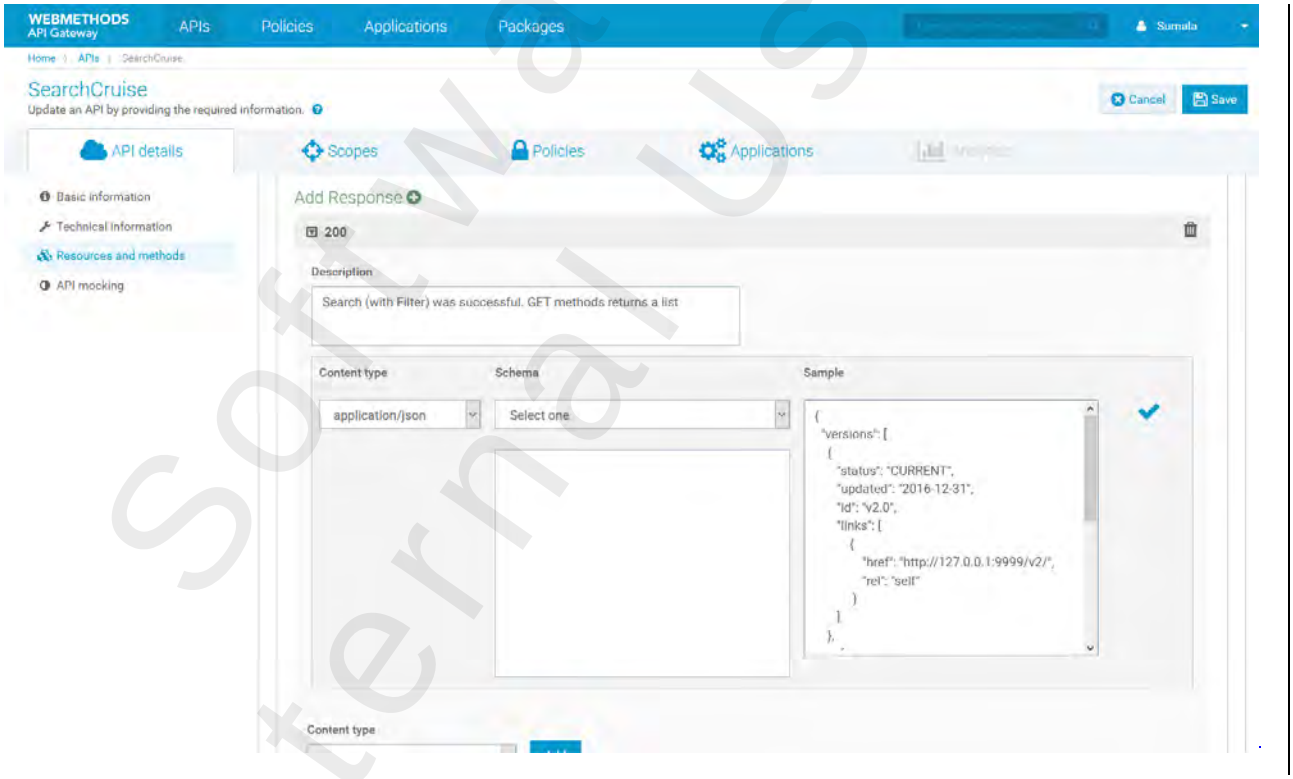

12. Switch over to Edit mode and navigate to **Resources and methods**. Select the response code 200 to provide a sample response document.

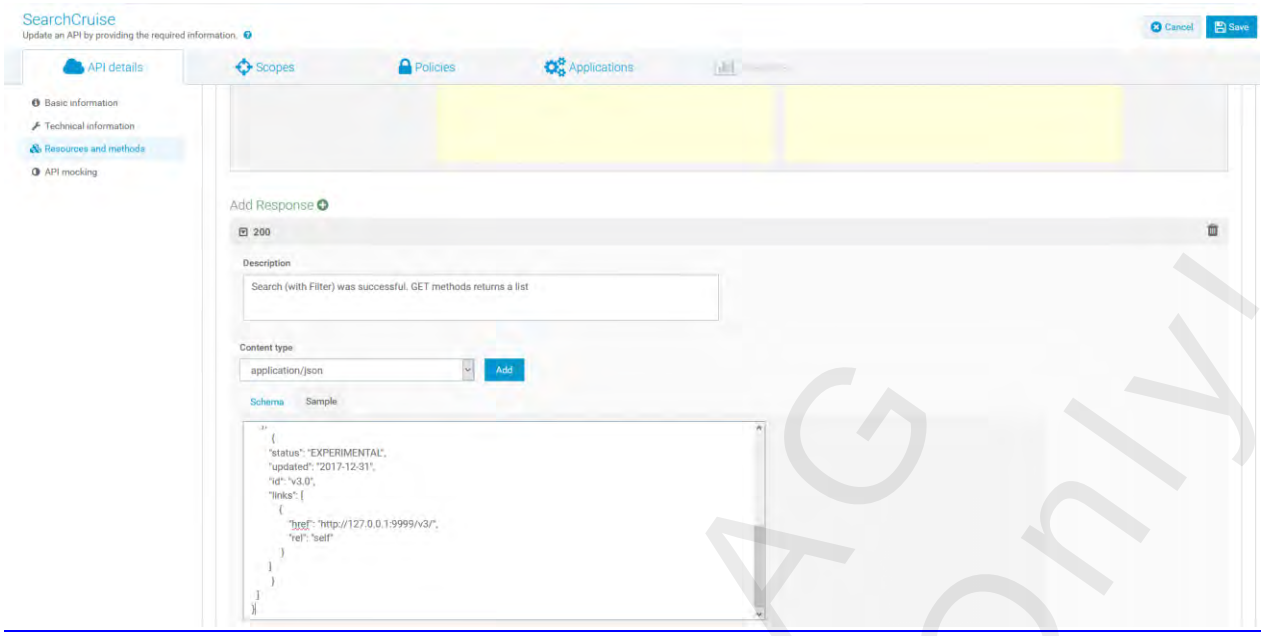



a. Provide the following properties:

   i. Content Type: application/json

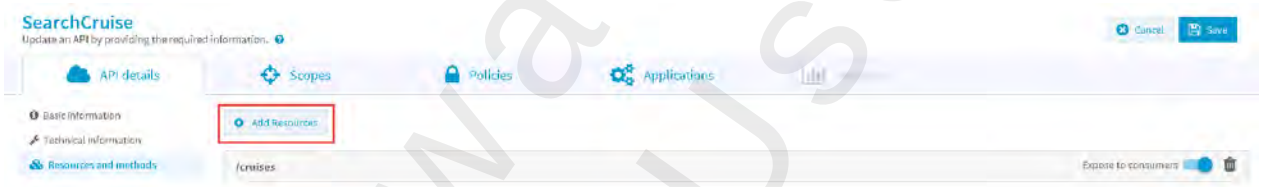
   ii. Sample: - copy and paste 200 response code sample from txt file mentioned above.






Click **Add.**

13. Create another Resource using the cruiseID as Path parameter. You can copy and paste content out of file **CruisesAPIResourceCruiseWithID.txt** in folder **C:\Training\456-71E\Exercise4.** Click **Edit** to change to edit mode. Select **Resources and methods** from the **API details** menu on the left hand side. Click **+ Add Resources**.




Provide the following details

a. **Add Resource**

   i. **Resource name:** cruisesWithCruiseID

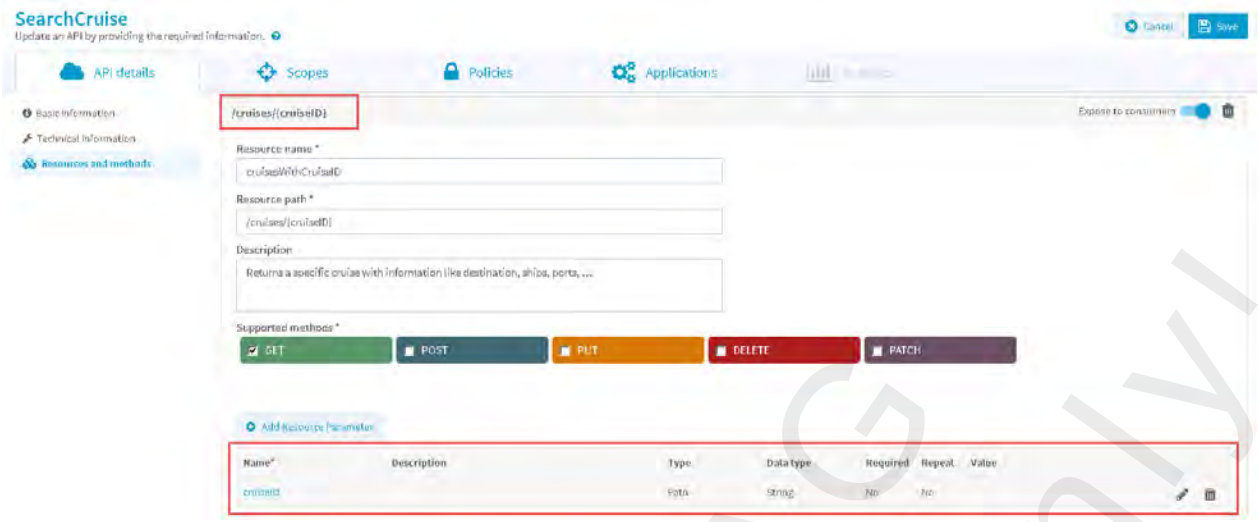
   ii. **Resource path:** /cruises/{cruiseID}

   iii. **Description:** Returns a specific cruise with information like destination, ships, ports, …

   iv. **Supported Methods:** GET

   Click **Add**.

b. Open the new created resource. The Path parameter is already created.




c. Scroll down to **GET** methods and provide the following properties.

   i. **Description:** Returns a specific cruise based on the specified cruise ID

d. Click **Add Request** link to provide Content type application/json. Copy and paste the sample request out of the text file mentioned above. Click **Add**.

e. Click **Add Response** link to add HTTP Status Code. Use the + icon.

   1. **Status Code**: 200 OK
   2. **Description:** Search with cruiseID was successful. GET methods returns a specific cruise..

   Click **ADD**.

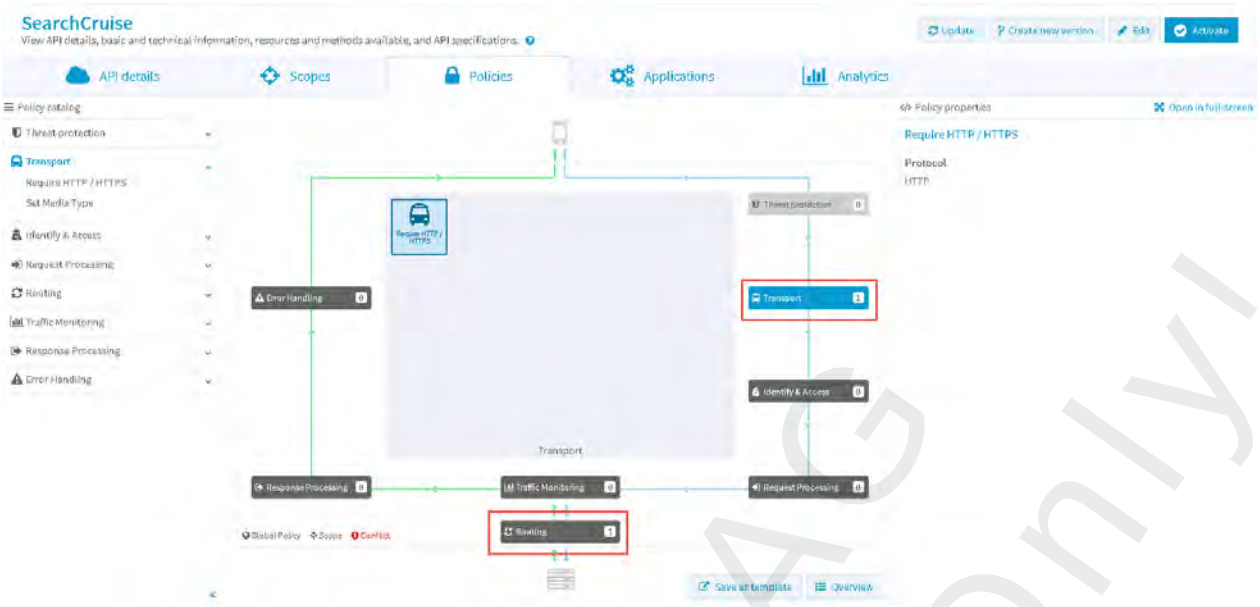
f. Add another Response Code:

   1. **Status Code**: 400 Bad Request
   2. **Description:** Cruise ID is not valid: <cruiseID>

   Click **ADD.**

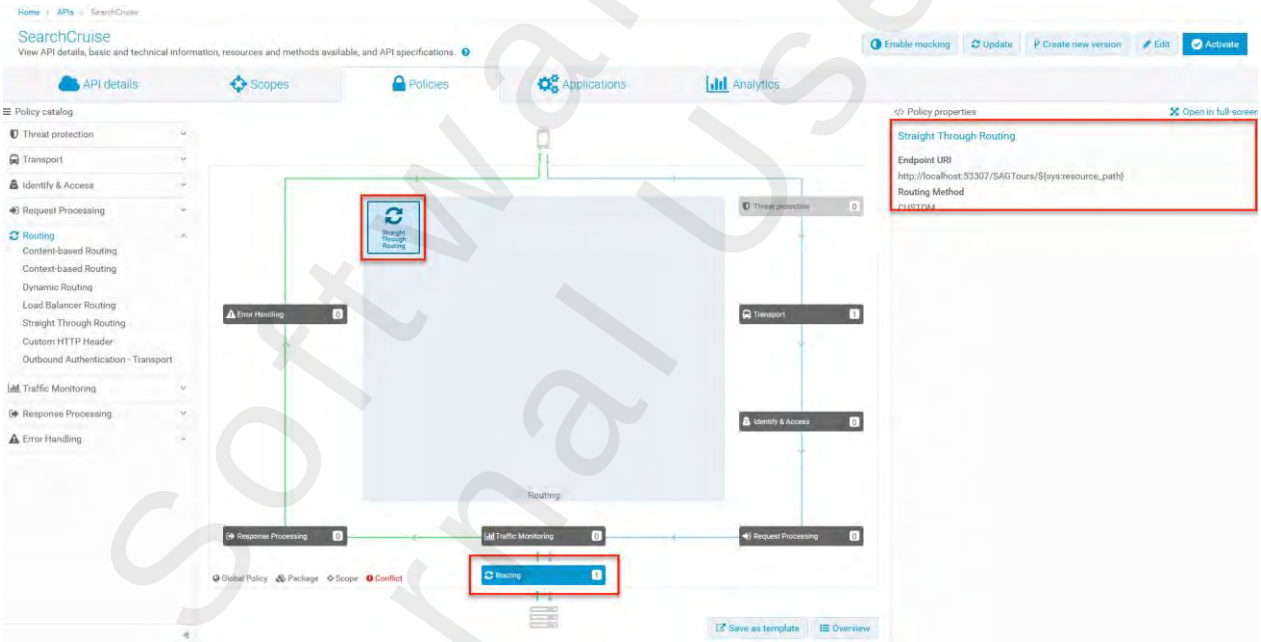

Click **Save.**

14. Review the default Policy Definitions. Select the **Policies** tab. The only policies in place are inside the section **Transport** and inside the section **Routing**.

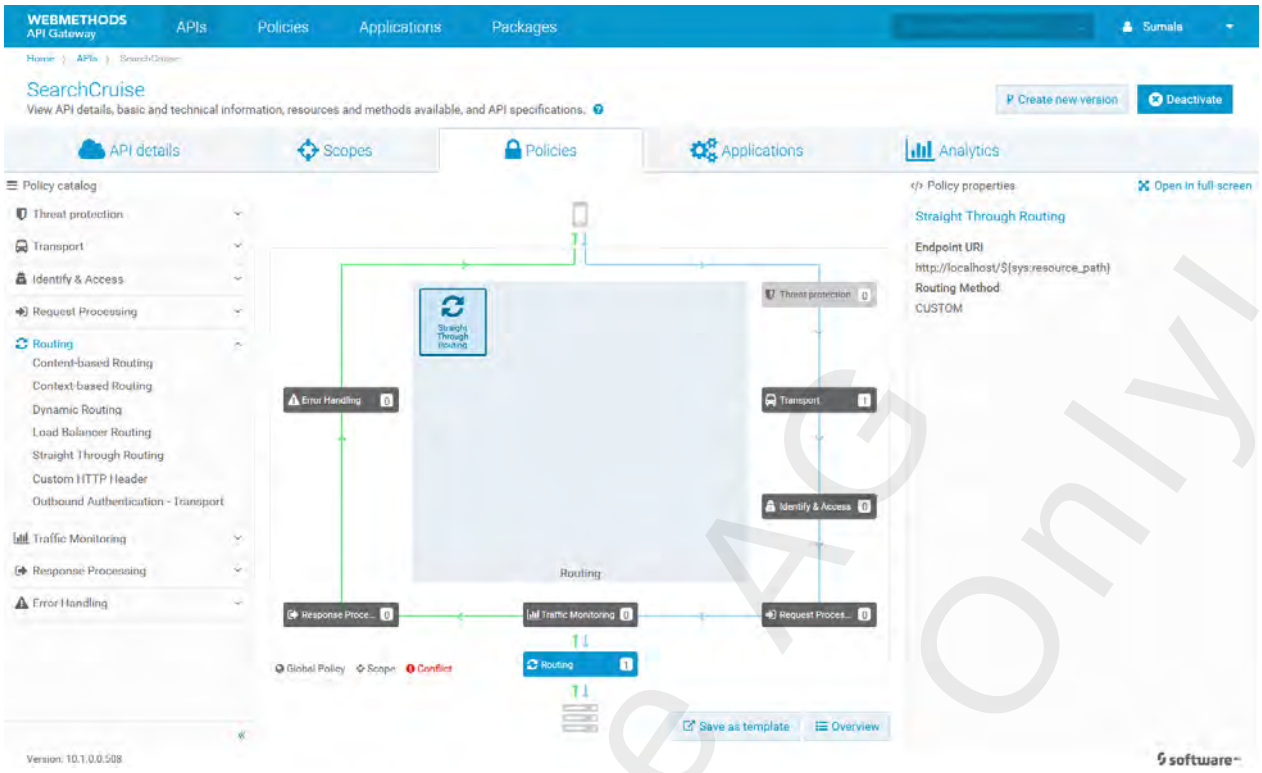



The Transport section is highlighted. The **Require HTTP/HTTPs** policy action is selected by default and configured, so that Protocol is set to **HTTP**.

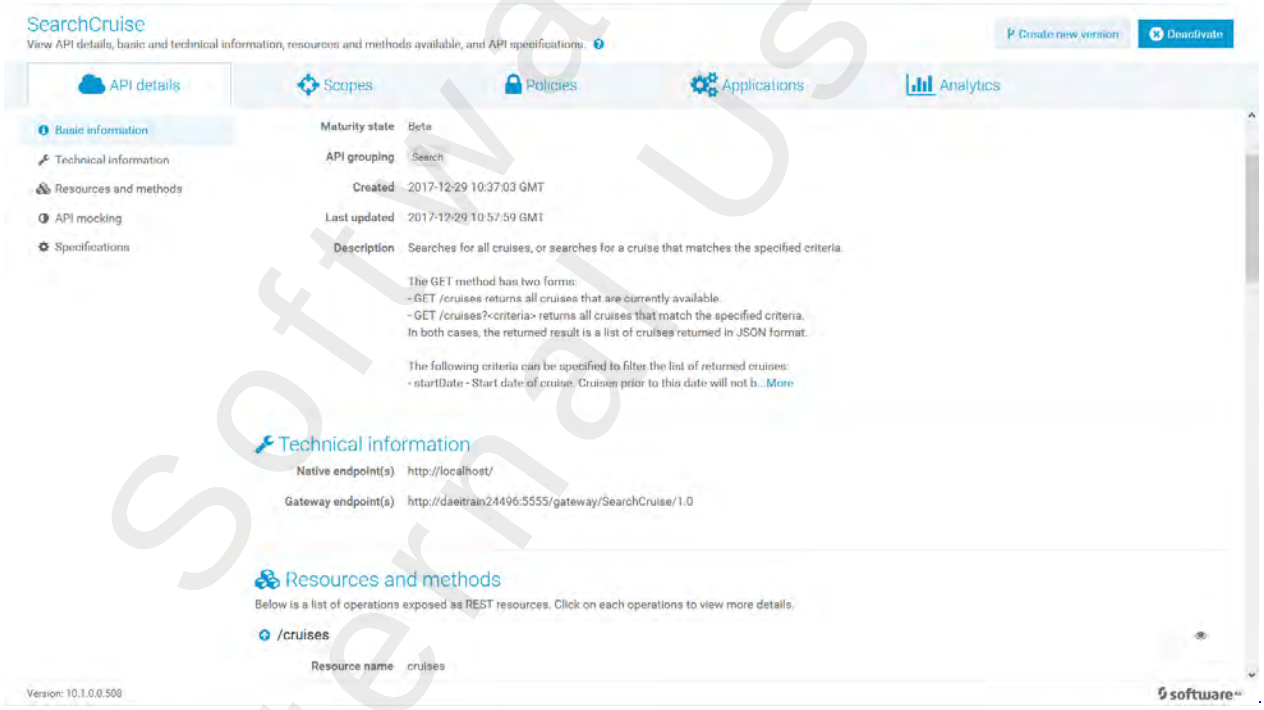
15. Review the Transport policy **Require HTTP / HTTPS**. Review the Routing policy by clicking the **Routing** box in the section Configure Policies. The **Straight Through Routing** policy action is selected by default and pre-configured. Click the policy action.

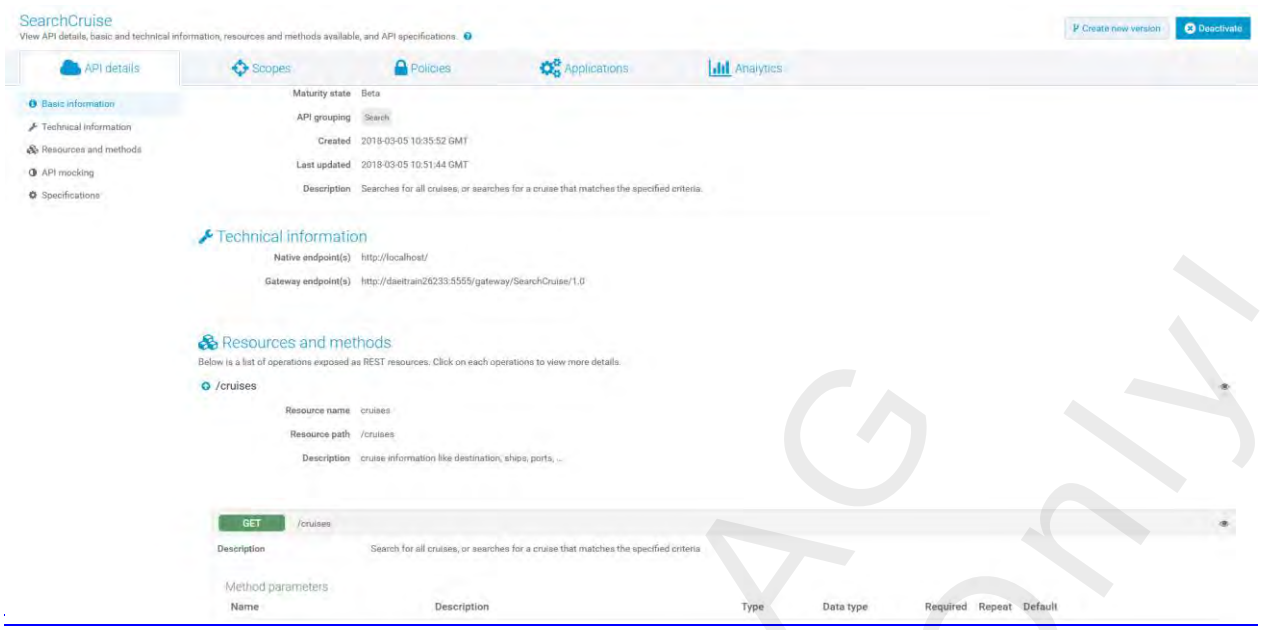



**16.** Click **Activate.** Click **Yes.** The service will now be virtualized and created in **API Gateway.** Select the **API details** tab.




17. Navigate to **API Details** and open the profile **Technical information**. You will find a reference to the native REST endpoint and the URL to access the Gateway endpoint(s)






18. Open **Mozilla Firefox REST client** and configure a GET request against our newly created virtual service.

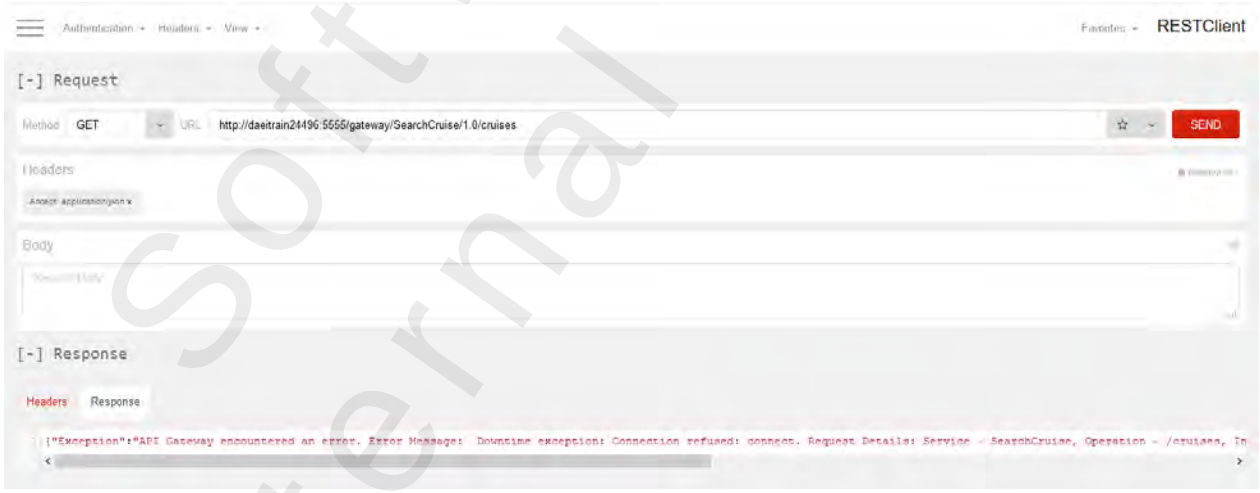
    a. **Methods:** GET

    b. **URL:** Copy and paste the Access URL out of the **Technical information** profile and append the **/cruises** resource from the **Resources and methods** profile

    c. **Custom Headers**:

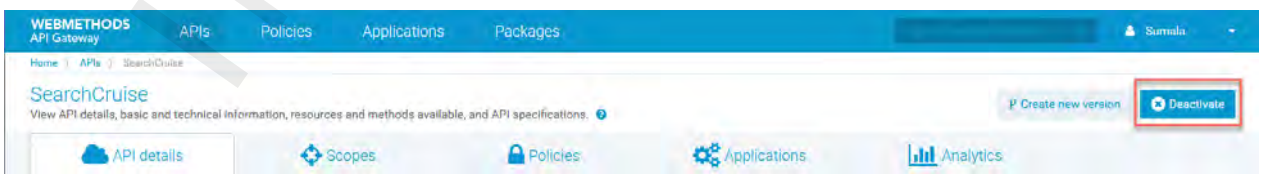
        i. **Accept**: application/json

    Run the request pressing the **Send** button.

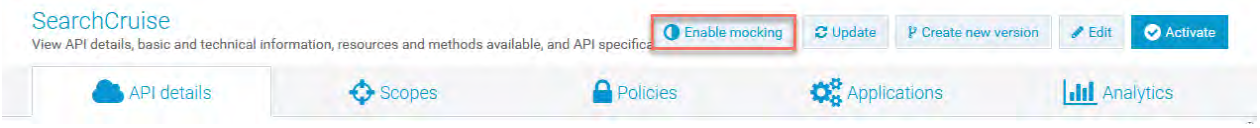



19. The error response for the request shows a downtime exception because we haven't provided a real endpoint for the native service. Therefor we want to use the Mocking feature of API Gateway.

20. Within **API Gateway** select the API and Click **Deactivate**. Click **Yes.**



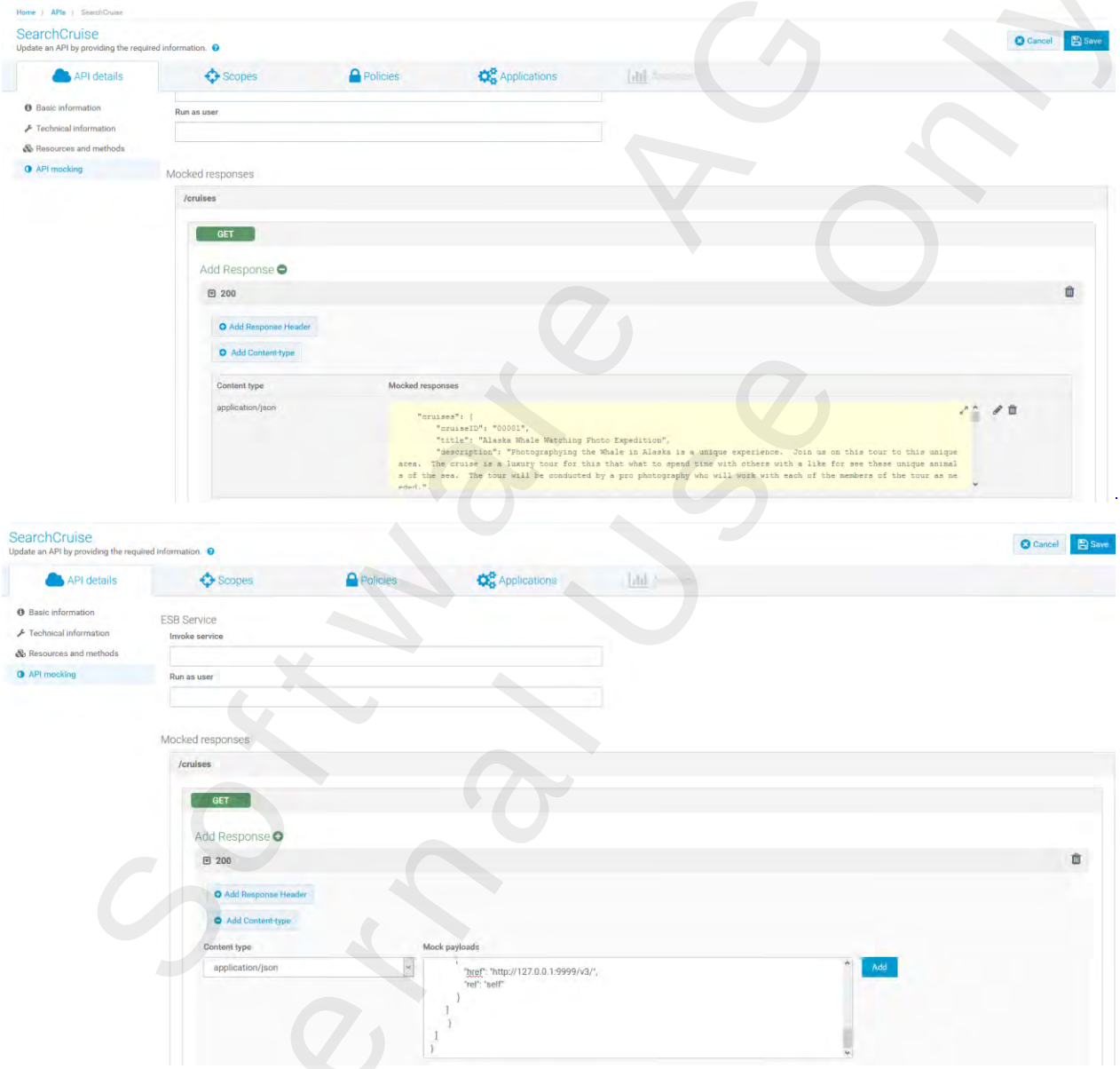
21. Click the **Enable Mocking** button at the top of the **SearchCruise** page.




22. Navigate to **API Details > API Mocking** information for this API and switch over to **Edit** mode.

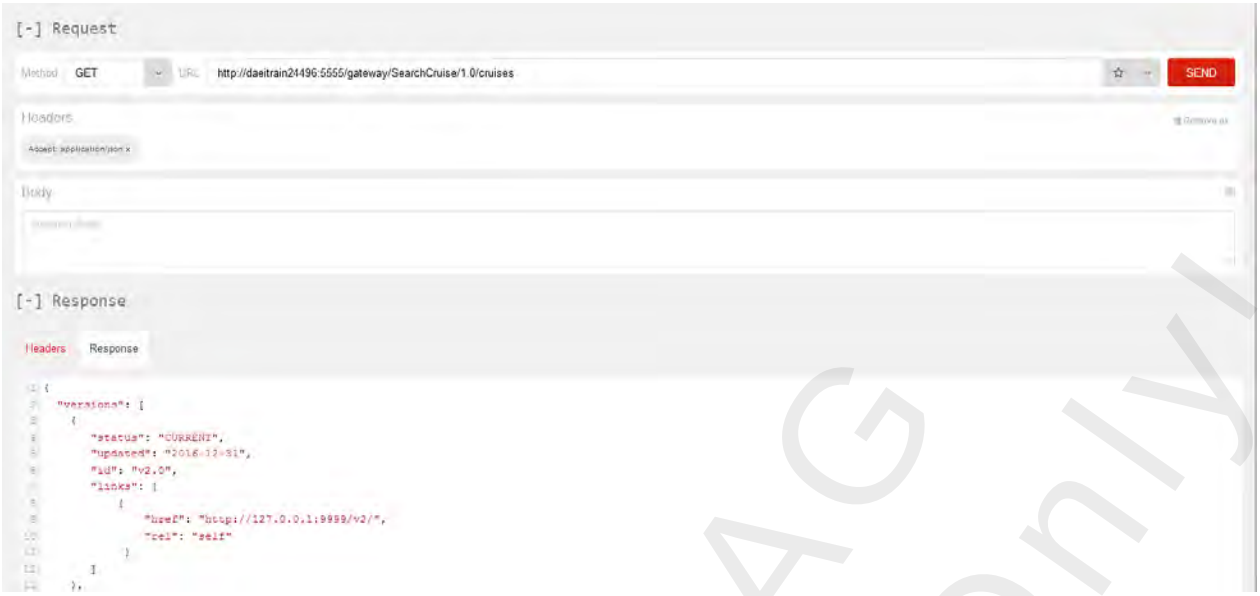
    a. Open the resource within the section **Mocked responses**

    b. Review the Mock payloads for the GET methods and the response **200**.




    c. Click **Save**. Click **Activate**.

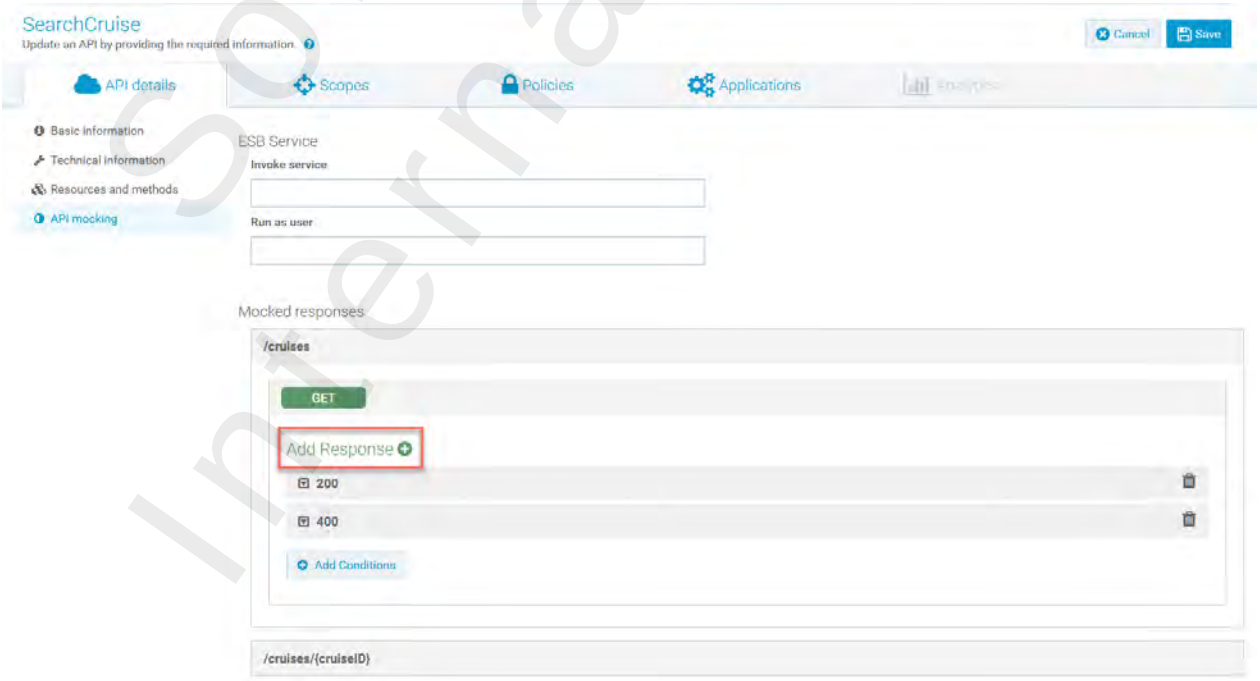
23. Move back to **REST API** and rerun the request. The Mocking response is shown.




24. Change the request and append the cruiseID 000001 to the URL

    a. ResponseCode shows 200, but no response body. We haven't provided a response payload.


25. Within **API Gateway** deactivate the API and switch to edit mode. Navigate to **API Mocking**. Within the mocked responses for **/cruises** and the **GET** method click **Add Response**.

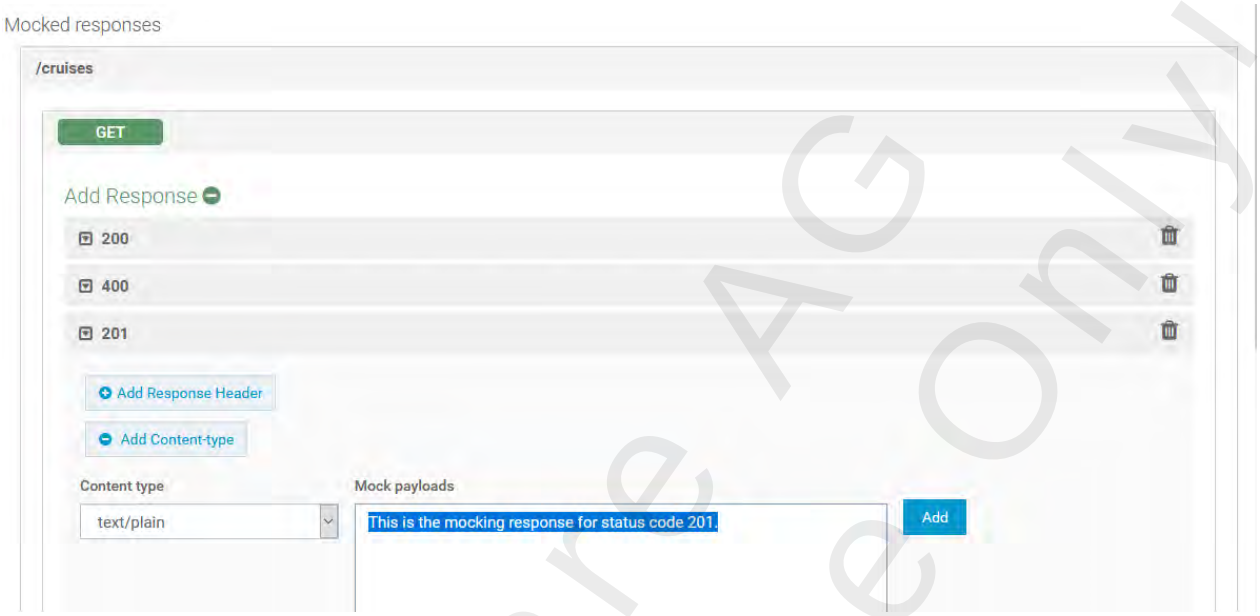


a. Select **Status code 201 – Created** and click **Add**.

b. Select the status code **201**. This will open the section to provide the Mock Payload. Click **Add Content-type**. Provide the following properties:

   i. **Content-type:** text/plain

   ii. **Mock payloads:** This is the mocking response for status code 201.




Click **Add.**

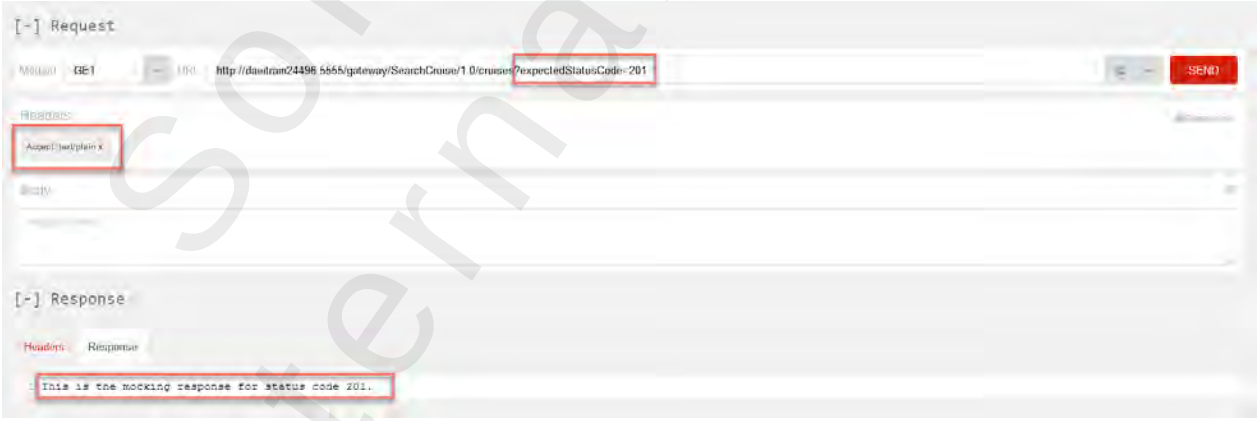
26. Save your changes and activate the API.

27. Move back to **REST Client.** Provide the following changes.

    a. Within the URL definition replace the appendix **/cruiseID** with **?expectedStatusCode=201**

    b. Change Customer Header **Accept** from **application/json** to **text/plain**.

28. Rerun the request. The mocked response is shown.

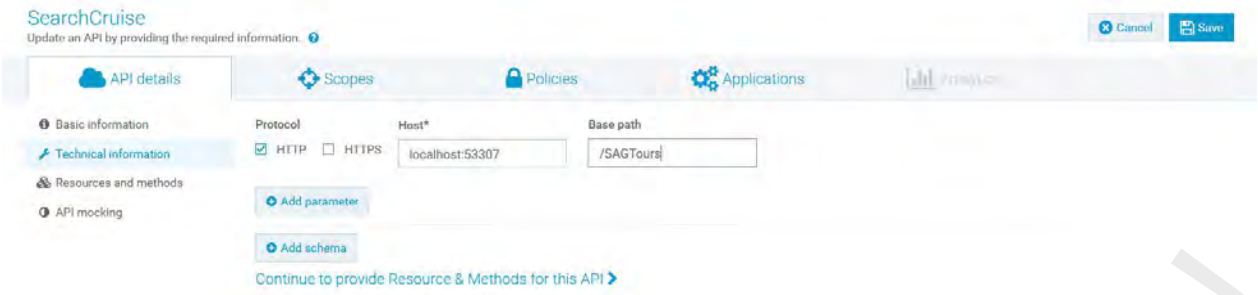



29. Now we want to point to the implementation of the native API and disable the mocking of the API. Within **API Gateway** deactivate the API and switch to edit mode. Disable mocking by clicking the **Disable mocking** button.

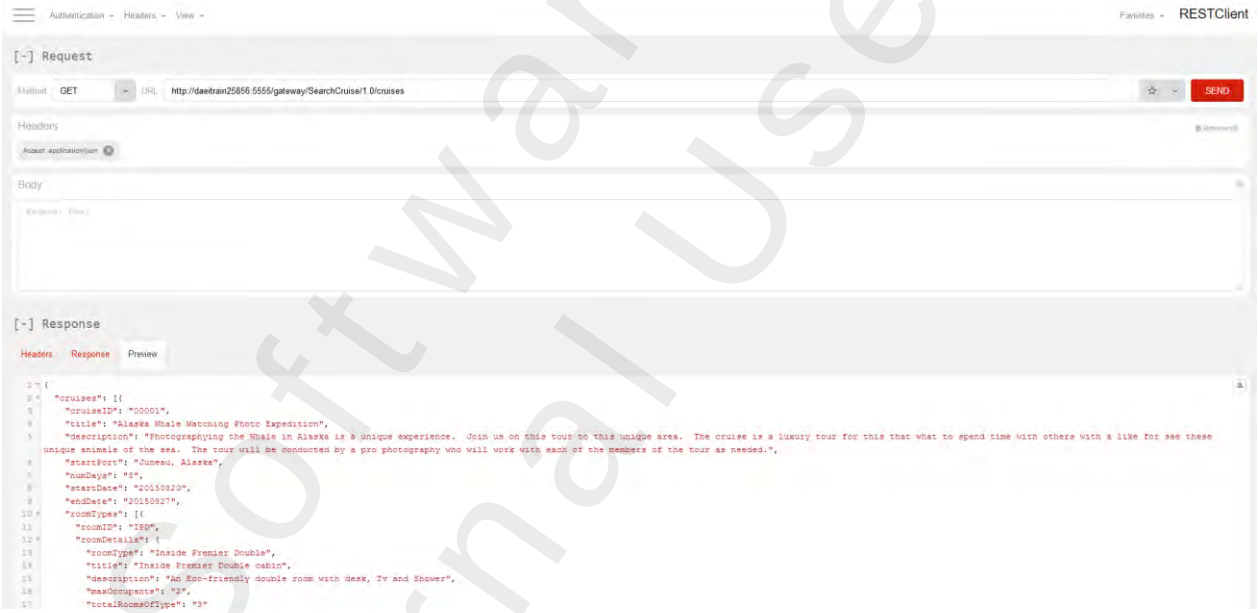
30. Switch over to edit mode and navigate to **Technical Information**. Provide the following properties:

    a. **Host:** localhost:53307

    b. **Base path:** /SAGTours




31. Select the **Policies** tab. Within **Routing** select the policy action **StraightThroughRouting**. Provide the change concerning the endpoint of the native service in Endpoint URI
    a. **Endpoint URI:** http//localhost:53307/SAGTours/${sys:resource_path}
32. Save the changes and activate the API.
33. Move back to **RESTClient.** Provide the following changes.
    a. Within the URL definition delete the appendix**?expectedStatusCode=201**
    b. Change Customer Header **Accept** from **text/plain** to **application/json**.
34. Verify the response **Response Body.**
35. In order to view the response body in correct JSON format use the **Preview** tab.



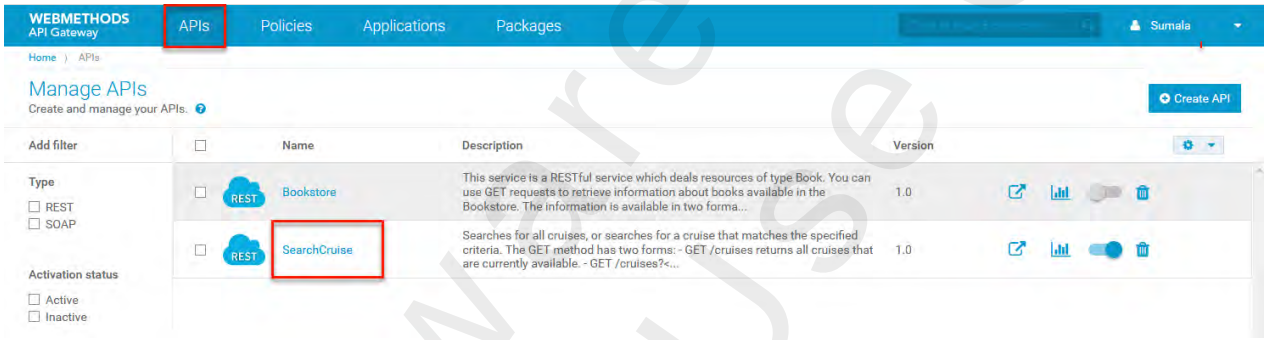
# Exercise 5:
# Adding a Logging Policy

## Objectives

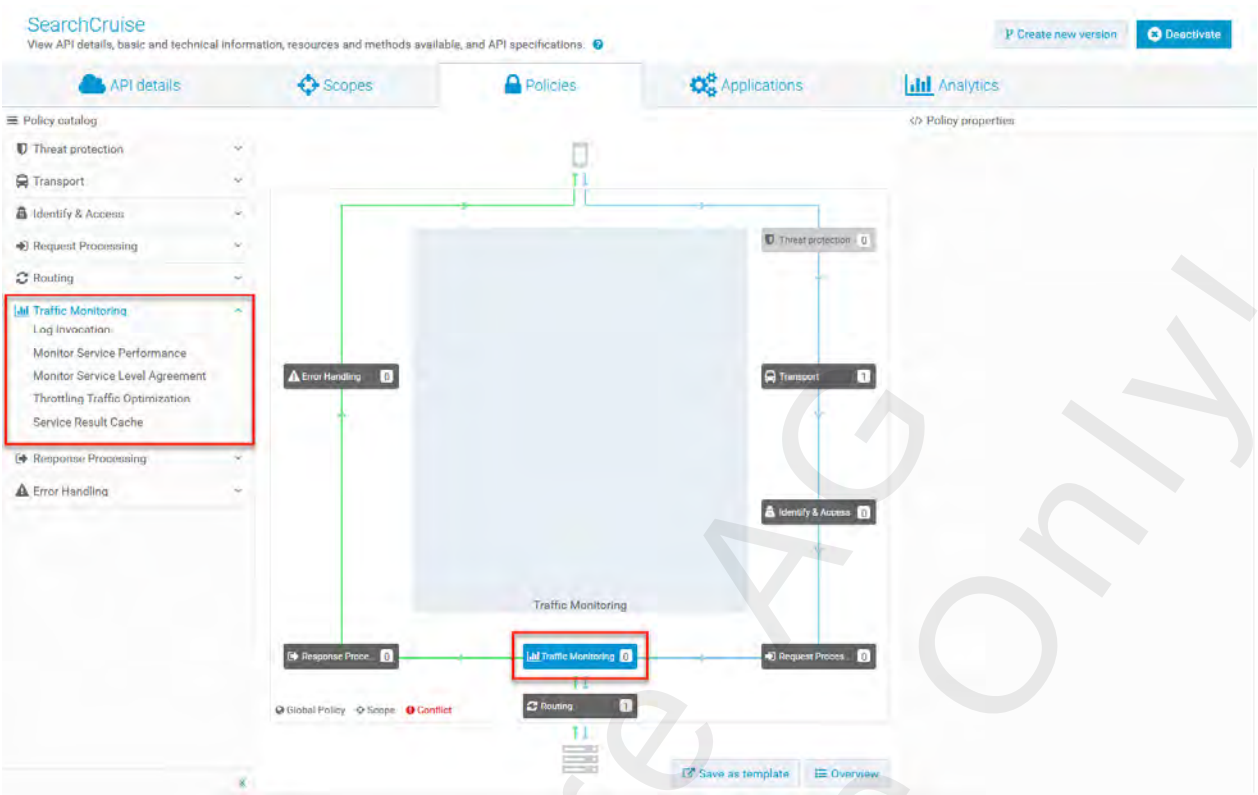
In this exercise, you will learn how to provide a global runtime policy which enforces the API Gateway to log the message request body.

## Steps

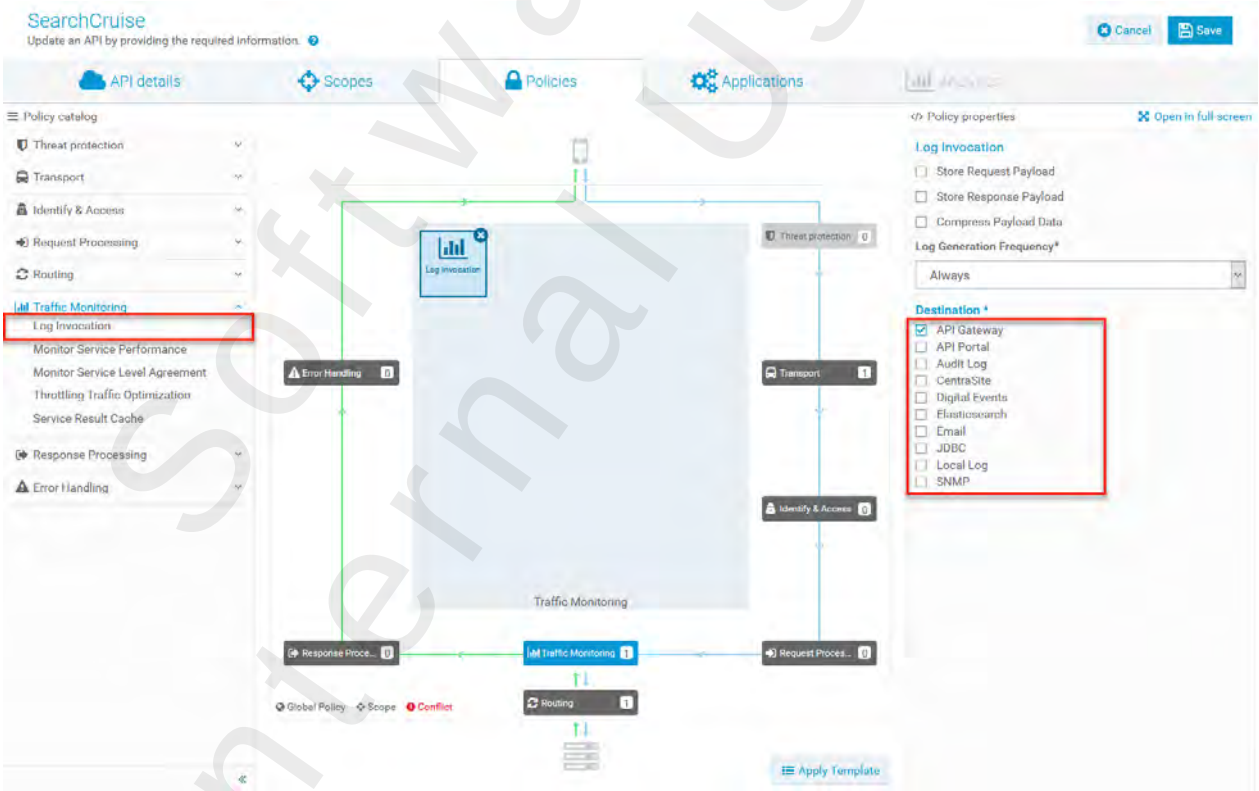
1. Open Windows Services UI to double check that the following services, needed for API Gateway, are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Event Data store 10.1 – should be started together with IS**
2. Login to the **API Gateway UI** as user **Sumala/manage** and configure monitoring policies for the service **SearchCruise**. On the **Manage APIs Page** select the service **SearchCruise**. This will open the Detail View of the **SearchCruise** service.




   a. Deactivate the API in order to change the policy configuration. Click the **Deactivate** button. Click **Edit** to open API in Edit mode.
3. Navigate to **Policies**. Select **Traffic Monitoring**.



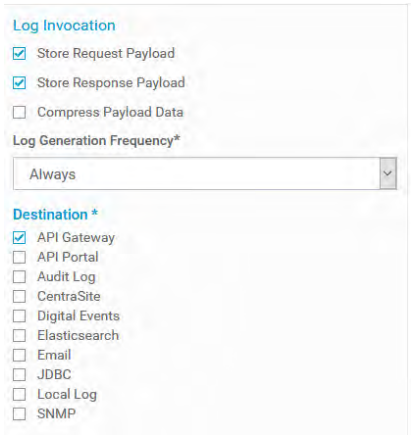

4. Click on **Log Invocation** in **Policy Actions catalog** section. This will move the policy action to the section **</> Policy properties**.



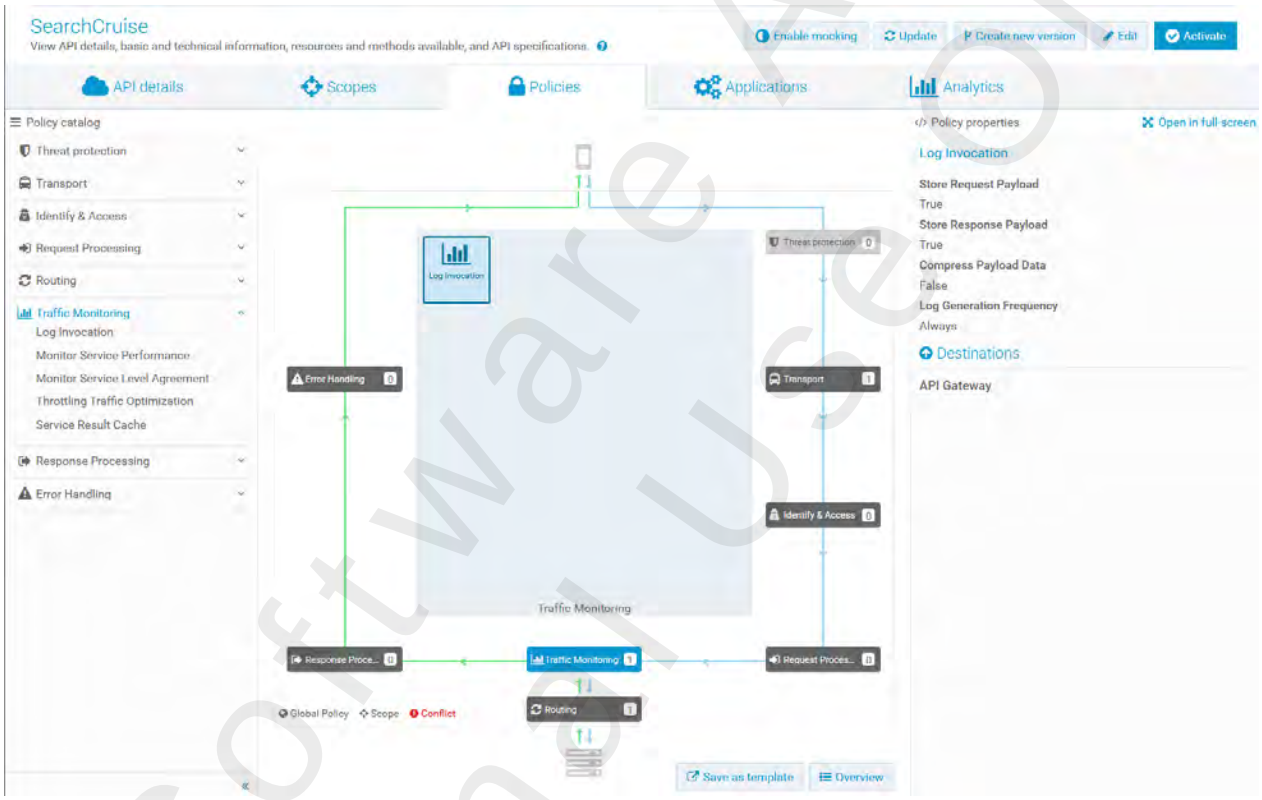

5. Provide the following properties

| | |
|---|---|
| **Store Request Payload:** | **-** select - |
| **Store Response Payload**: | - select - |
| **Log Generation Frequency**: | always |

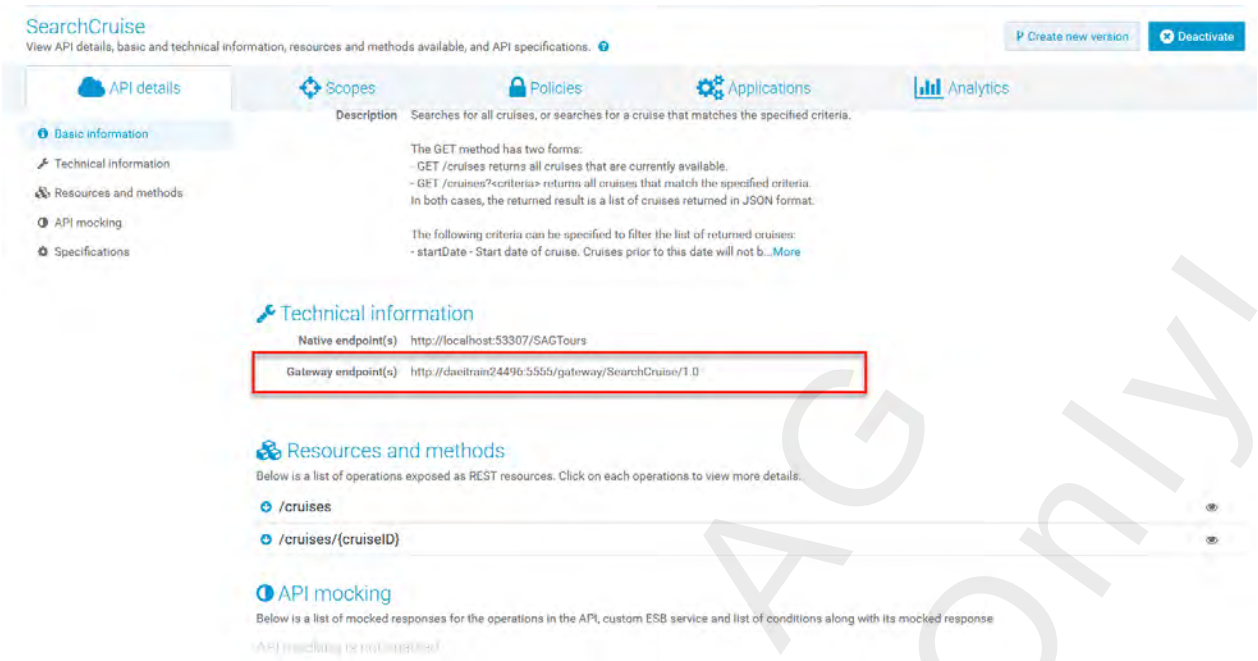
**Destination**: API Gateway




Click **Save**.




6. Click **Activate,** click **Yes** to enable the changes at runtime
7. Open **API Details** view to get **Gateway endpoint.**



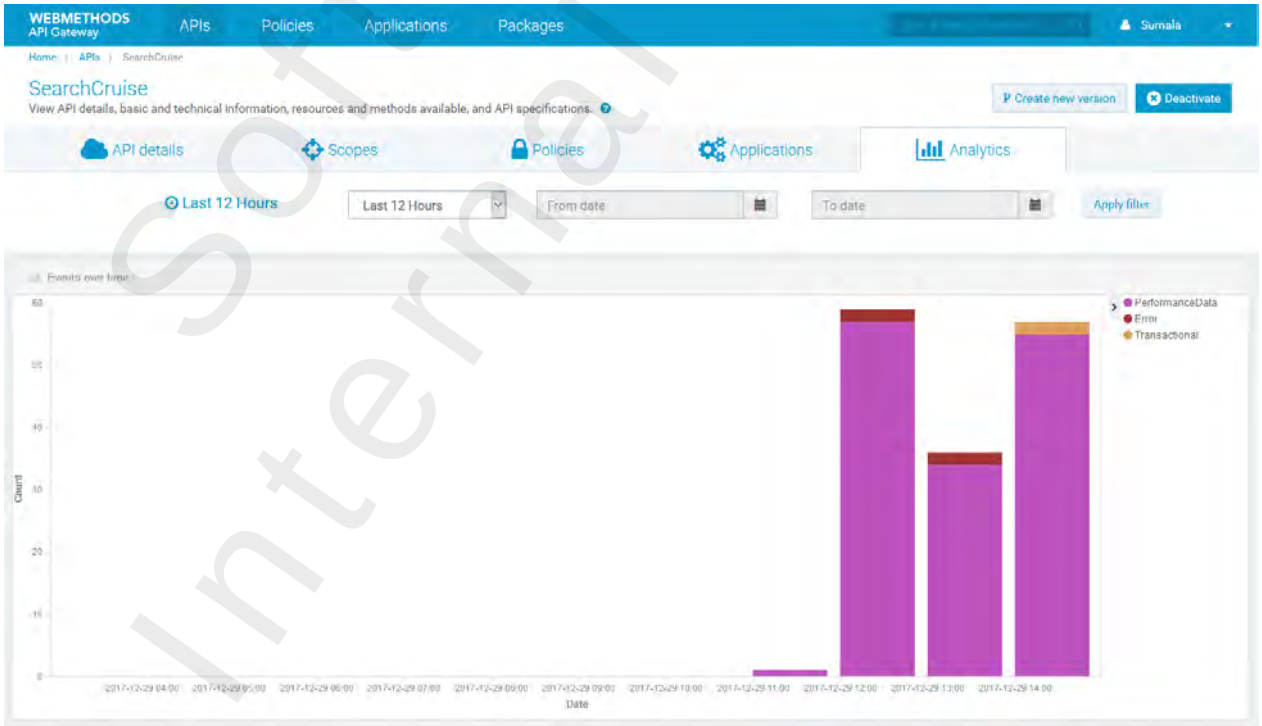

8. Open **Mozilla Firefox REST client** and configure a GET request against our newly created virtual service.

   a. **Methods:** GET

   b. **URL:** Copy and paste the Gateway endpoint out of the **API Gateway** Basic information profile and add resource name:
   http://DAETRAIN00749:5555/gateway/SearchCruise/1.0/cruises

9. Run the request pressing the **Send** button. In case of an error, make sure that you have set the desired custom header for Accept.

10. Now we look for the log data **API Gateway**. Go back to **API Gateway** and select the API **SearchCruise**. Open the tab **Analytics**.

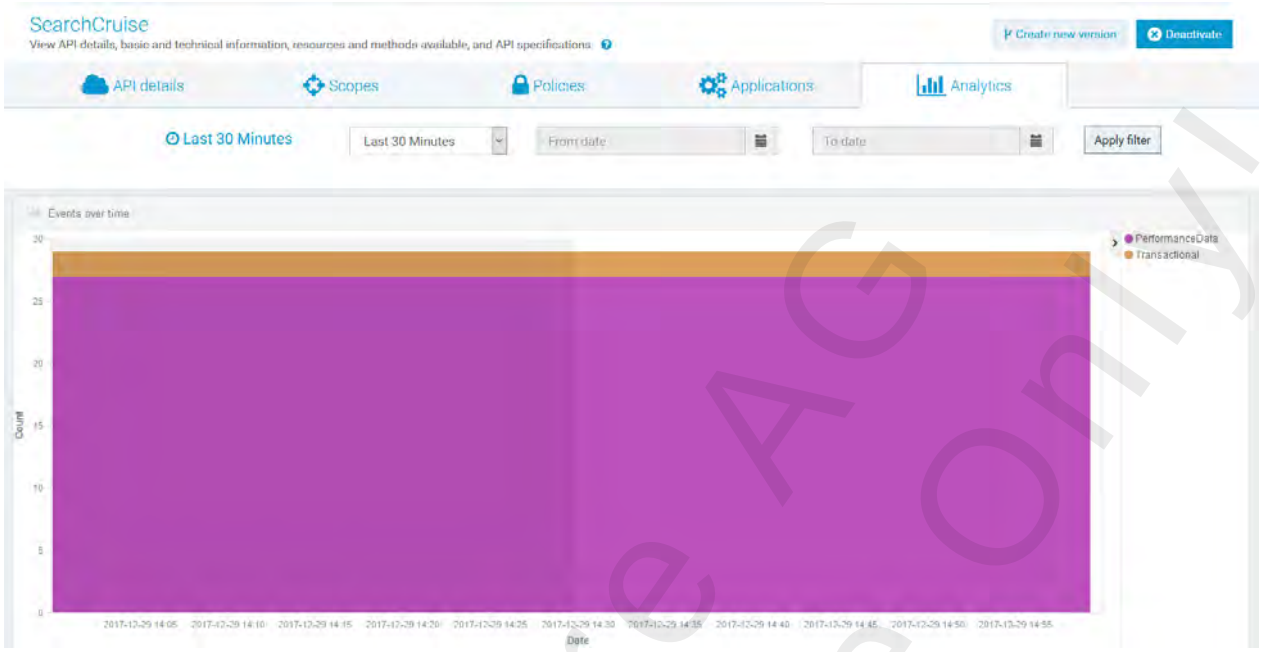



Your Graph probably will look somewhat different.

11. Refine the filter conditions

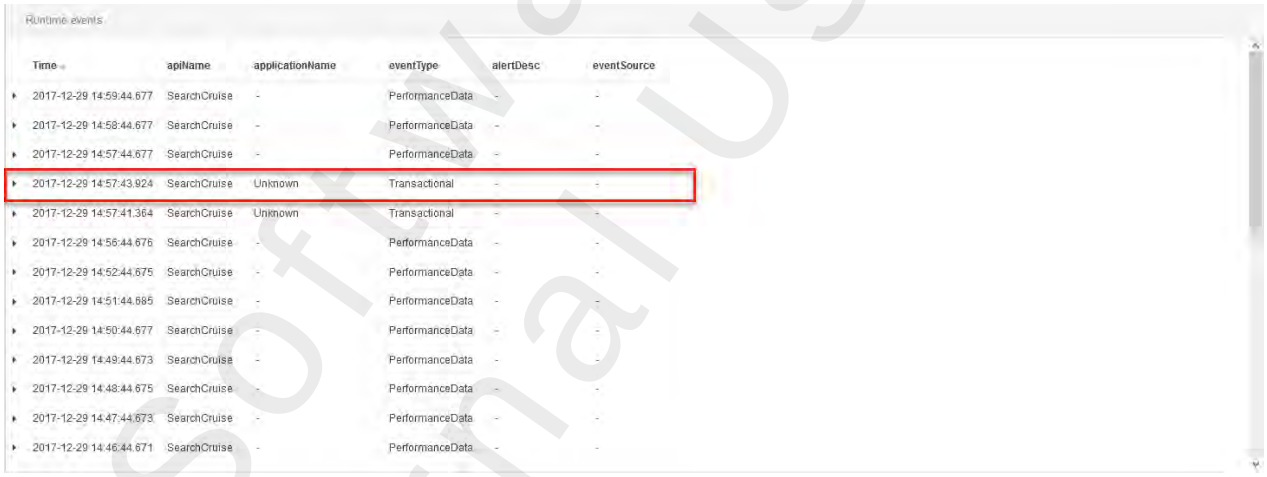
    a. **Time Range**: last 30 minutes

    Press **Apply Filter** Your Graph probably will look somewhat different.




12. Scroll down to view the provided data in the dashboards like **API trend by response**, **Native service performance**, …

13. Navigate to **Runtime Events**. Select an event of type **Transactional Data**.




14. Click the little triangle at the beginning of the row. This will open the details for this event.

This page intentionally left blank

# Exercise 6:
# Enforcing Security Policy

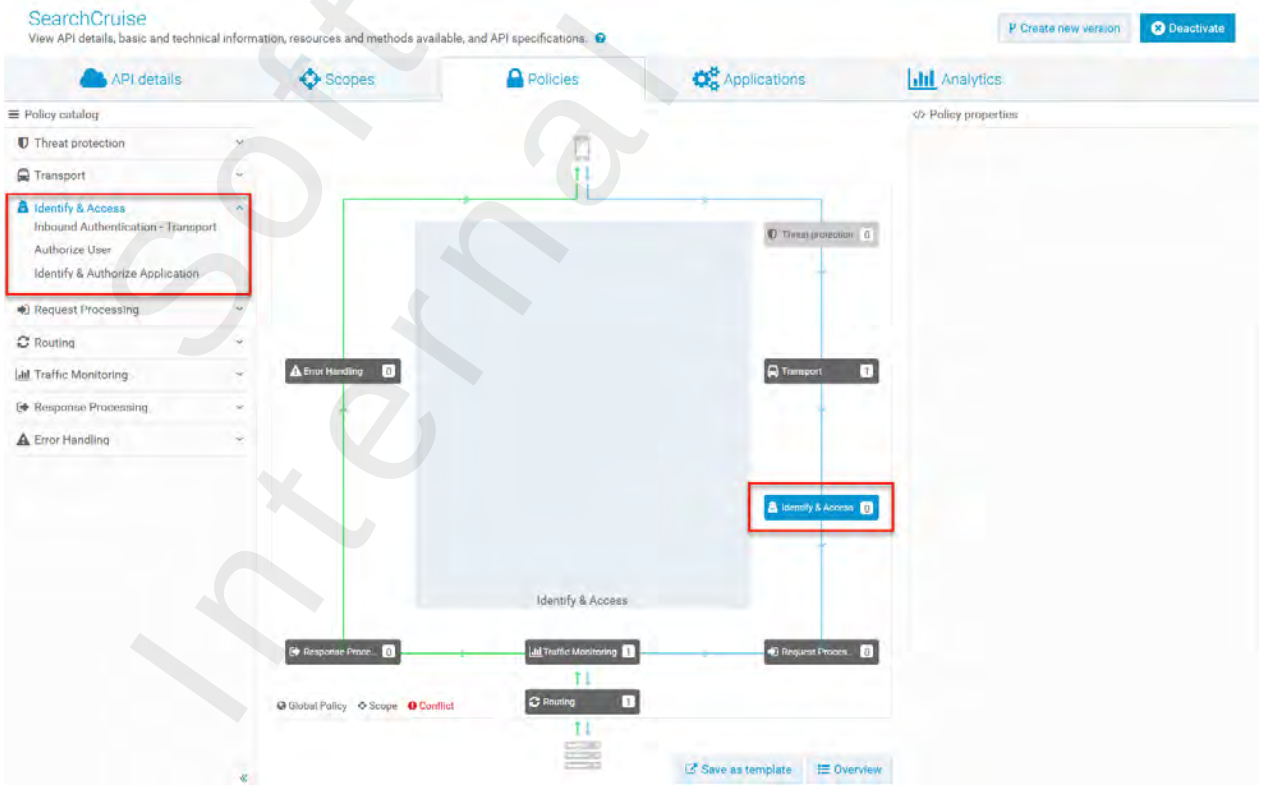
## Objectives

In this exercise, you will define a policy to enforce username and password when calling the API.

## Steps

1. Open Windows Services UI to double check that the following services, needed for API Gateway, are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Event Data store 10.1 – should be started together with IS**
   c. **Software AG Runtime 10.1**
2. Login to the **API Gateway UI** as user **Sumala/manage** and configure security polices for the service **SearchCruise**. On the **Manage APIs Page** select the service **SearchCruise**. This will open the Detail View of the **SearchCruise** service.


3. Navigate to **Policies**. Select **Identify & Access**.



4. Deactivate the API in order to change the policy configuration. Click the **Deactivate** button.

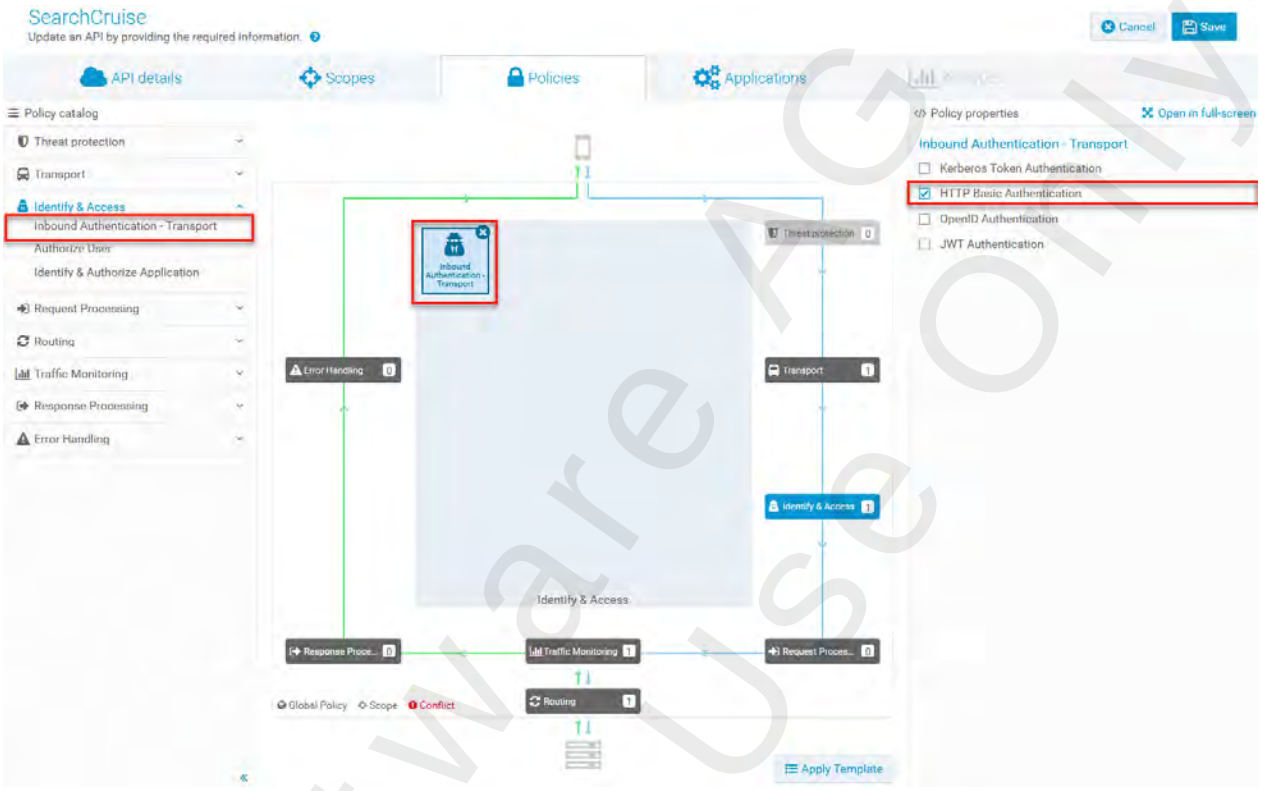
   Click **Yes**.

   a. Click **Edit** to open API in Edit mode.

5. Click on **Inbound Authentication _ Transport** in **Policy catalog** section. This will move the policy action to the section **Policy properties**.

   a. In **Policy properties** select:

      i. **Identification Type**: HTTP Basic Authentication




6. Click **Save**, click **Activate**. Click **Yes**.

7. Open **Mozilla Firefox REST client** and configure a GET request against our newly created virtual service.

   a. **Methods:** GET

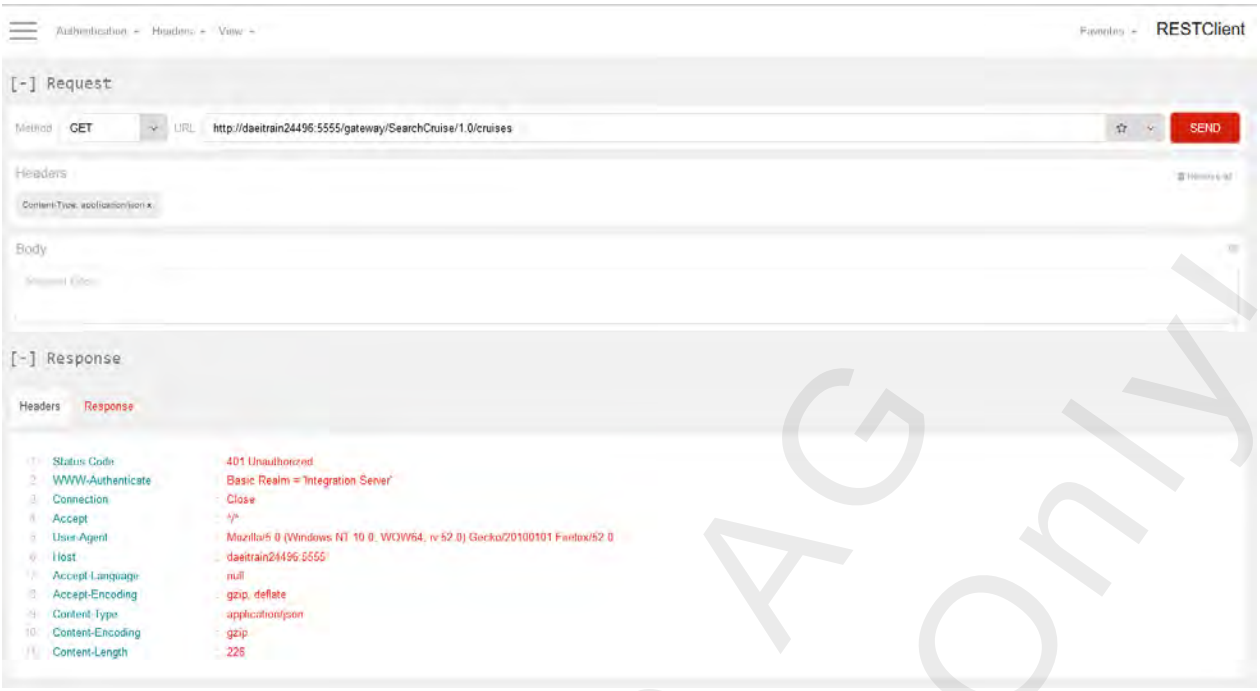
   b. **URL:** Copy and paste the Gateway Endpoint and resource from API Gateway UI: **http://DAETRAIN00729:5555/gateway/SearchCruise/1.0/cruises**

   c. **Custom Headers**:

      i. **Accept**: application/json

   Run the request pressing the **Send** button.

8. You will get an error response with stats code **401 Unauthorized.** In case an Authentication dialog pop-up, do not provide any credentials.




9. Provide an additional variable for Authentication.

   a. **Authentication**:

      i. **Basic Authentication**: Sumala/manage

   Run the request pressing the **Send** button.

This page intentionally left blank

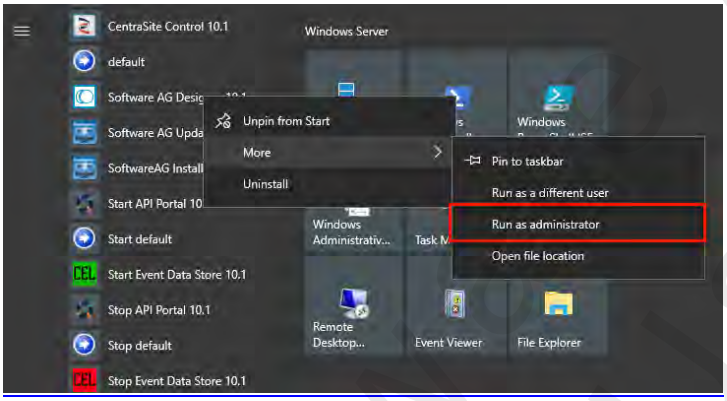
# Exercise 7:
# Using a WSS Security Token

## Objectives

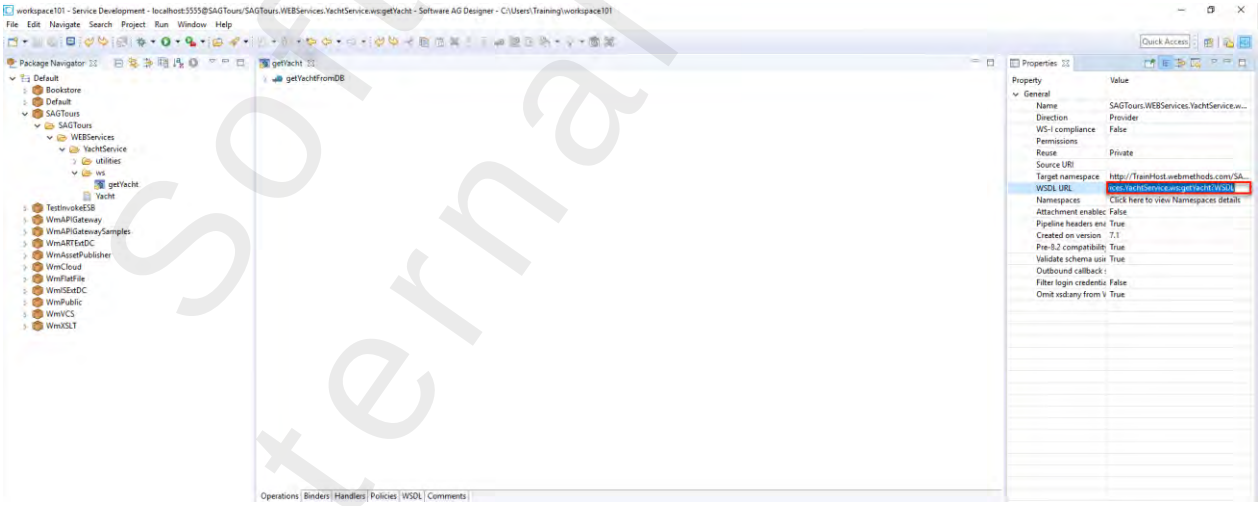
In this exercise, you will define a security run-time policy using **WSS Username token** and apply it to a virtual service running on API Gateway.

## Steps

1. Open Windows Services UI to double check that the following services, needed for API Gateway, are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Event Data store 10.1 – should be started together with IS**
2. Open **Software AG Designer 10.1** from the Windows **Start Menu > Software AG > Tools**. Use the right mouse key and use the option Run as administrator to start the Designer.




   In case you are asked for a **secure storage password** , provide the password **manage**.
3. In the **Package Navigator o**pen the package **SAGTours** and edit the **SAGTours.WEBServices.YachtService.ws:getYacht** Web Service Descriptor.



In the Properties, copy the Value of WSDL URL and paste it into the Import URL field in API Gateway. The value pasted should be similar to the following:

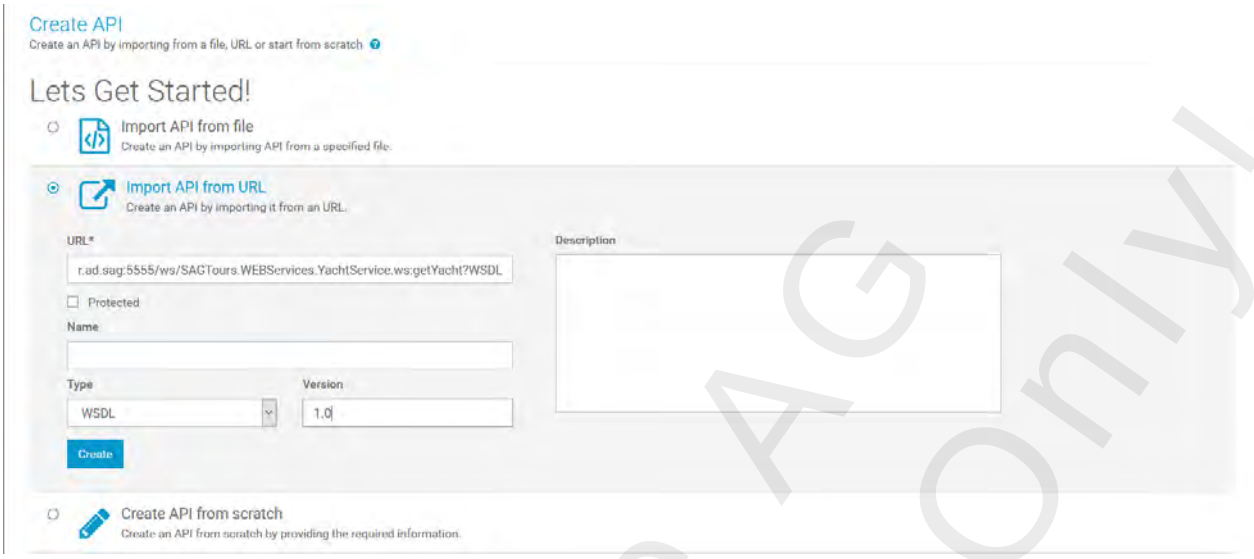
**http:// sagbase.eur.ag.sag:5555/ws/SAGTours.WEBServices.YachtService.ws:getYacht?WSDL**

1. Login to the **API Gateway UI** as user **Sumala/manage** and create a new SOAP based API. Use the option **Import API from URL**.

a. **URL**: - as mentioned above -

b. **Type**: WSDL

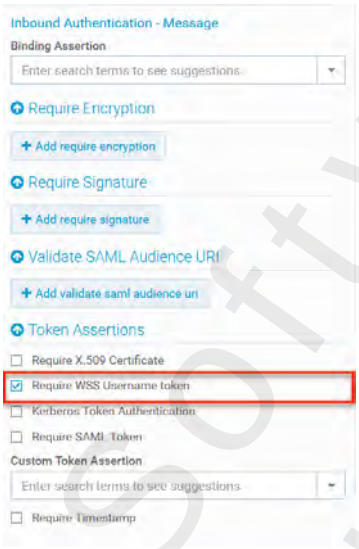
c. **Version**: 1.0



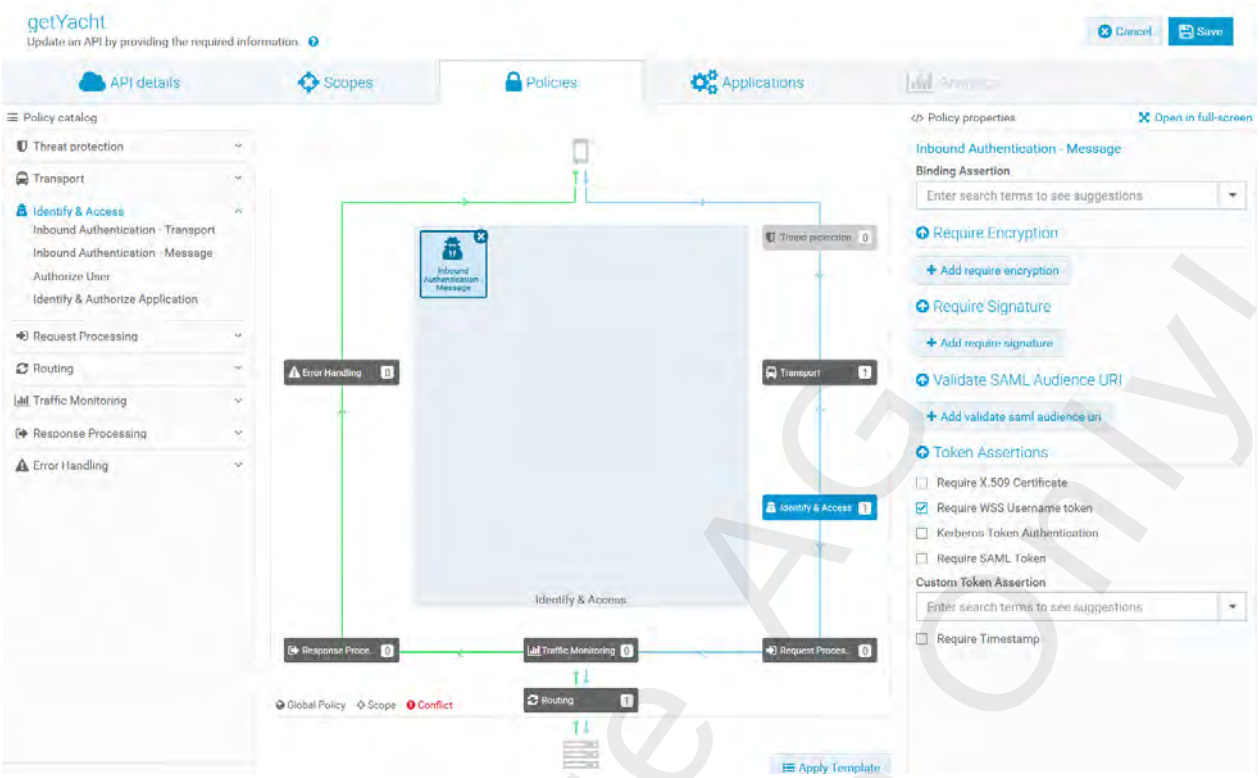

Click **Create.**

2. Naviagte to **Policies > Identify & Access**. Switch to **Edit** mode.
3. Select **Inbound Authentication – Message** navigate to **Token Assertions**

   a. **Require WSS Username token:** - select -



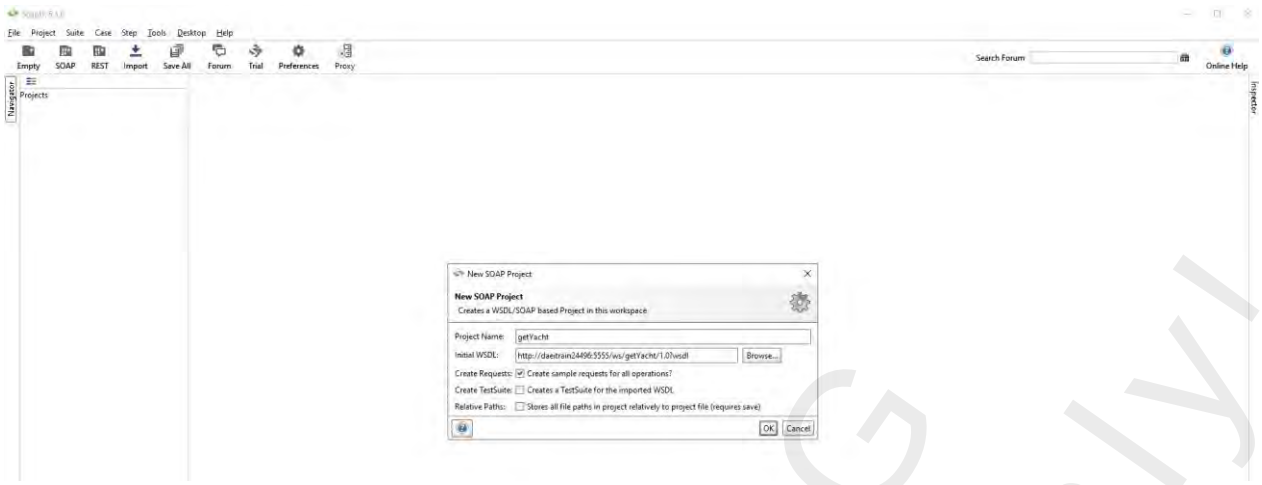

4. The policy action will identify the consumer from WSS Username




Click **Save** and **Activate**.

5. Navigate to **API details** and get the **Gateway endpoint**.


6. Open the **SOAP UI** web service test tool and create a new SOAP project. You will find SOAP UI within **Windows Start Menu > Tools > SOAP UI**. Select Projects, right mouse click, select New SOAP Project.
   a. **Name:** getYacht
   b. **Initial WSDL**: Copy the **Gateway Endpoint**. You will get this information out of **API Gateway > APIs > getYacht > Technical information**. And add to the URL: **?wsdl**

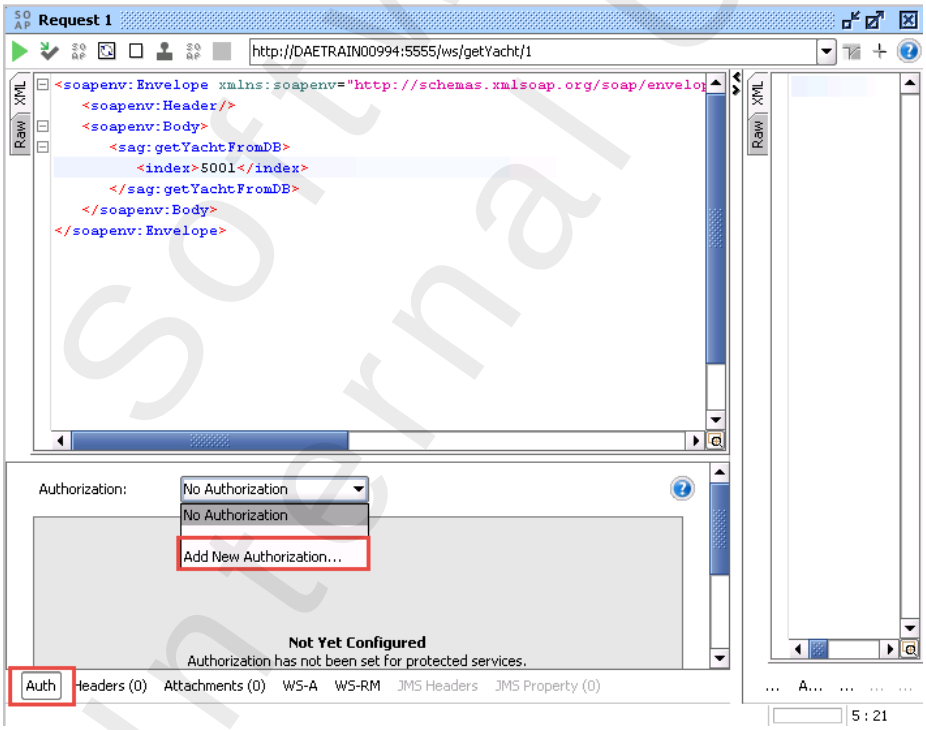


Click **Ok**.

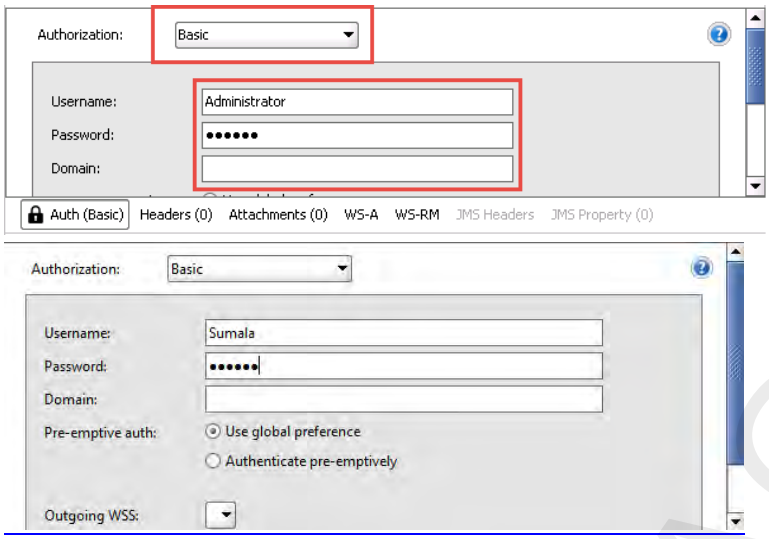
A new SOAP Project will be created. Browse to the operation **getYachtFromDB**. You will see a pre-generated request named **Request 1**. Open the request.

7. A new SOAP Project will be created. Browse to the operation **getYachtFromDB**. You will see a pre-generated request named **Request 1**. Open the request.

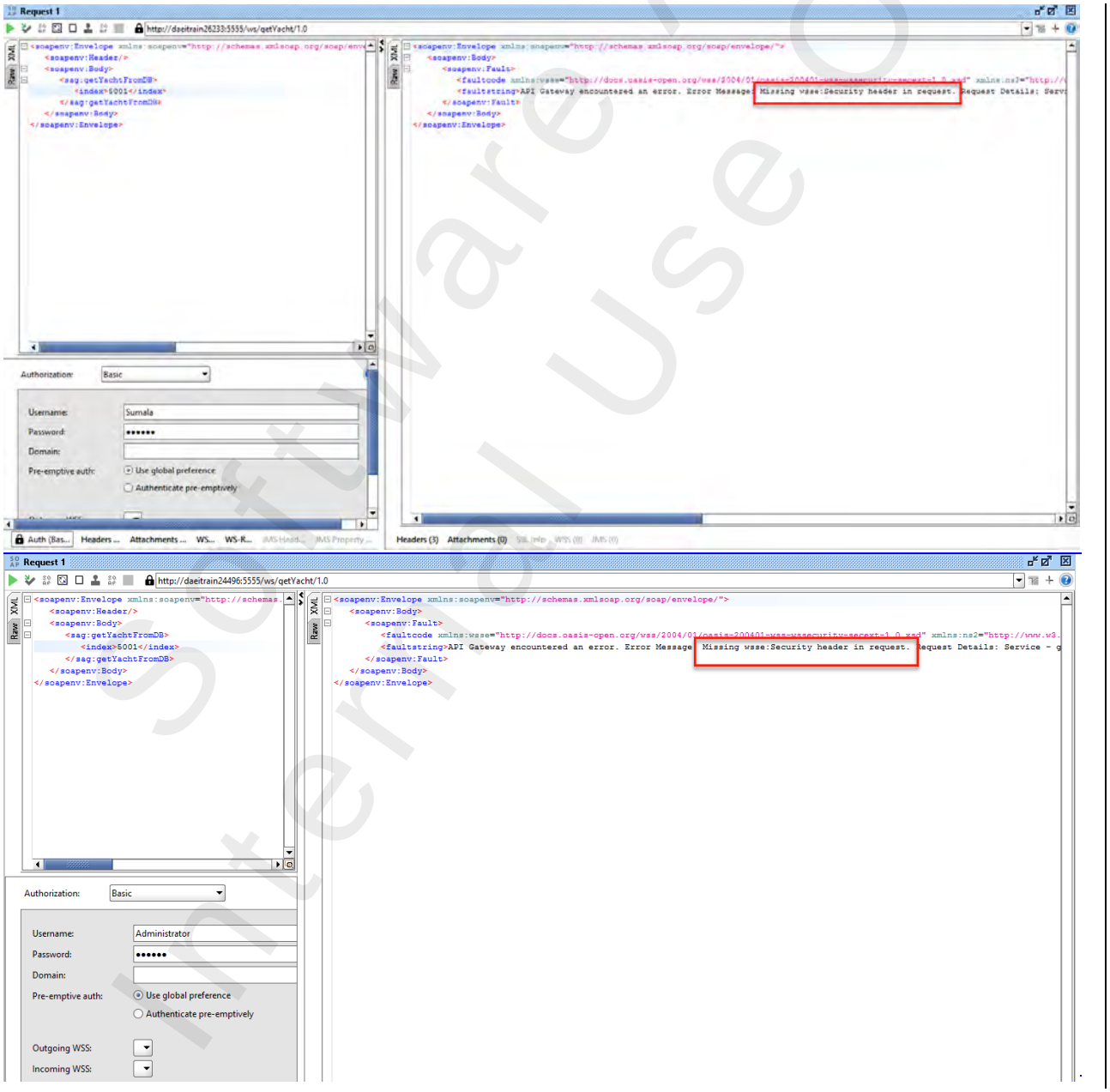
8. Within the SOAP envelope (soapenv) change the <index> value from ? to 5001. Add New Authorization.



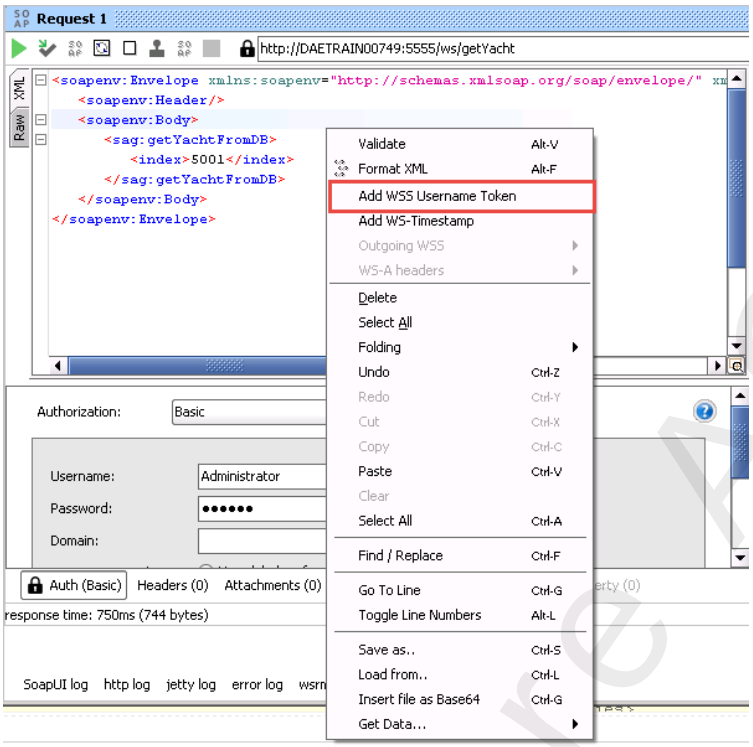

a. **Type**: Basic
b. **Username:** Sumala
c. **Password:** manage

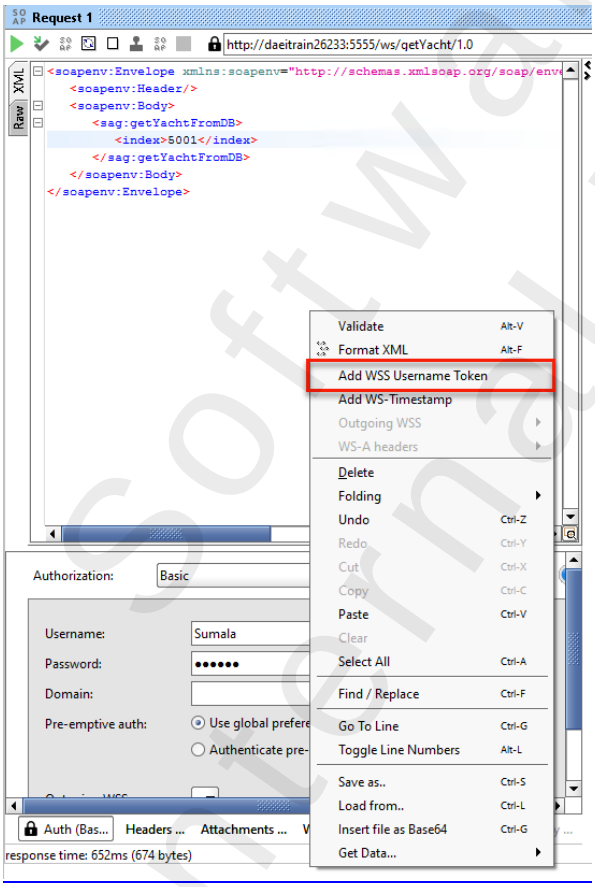



9. Run the request by hitting the green triangle in the header lien of the Request. It will fail. Look at the error message in the response.

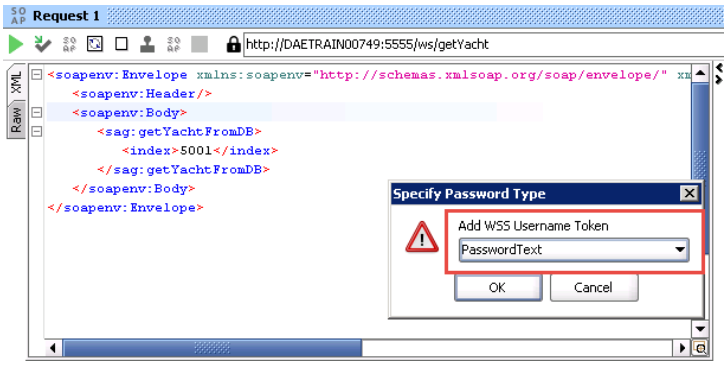


10. To run a service that has a policy using WS token based identification, you have to add WSS Username Token. Select the Request Body section and press right mouse button. From the list select **Add WSS Username Token**.

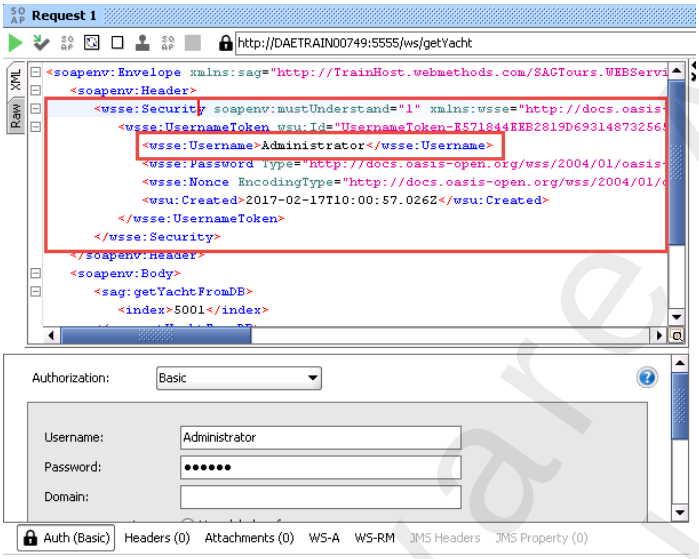


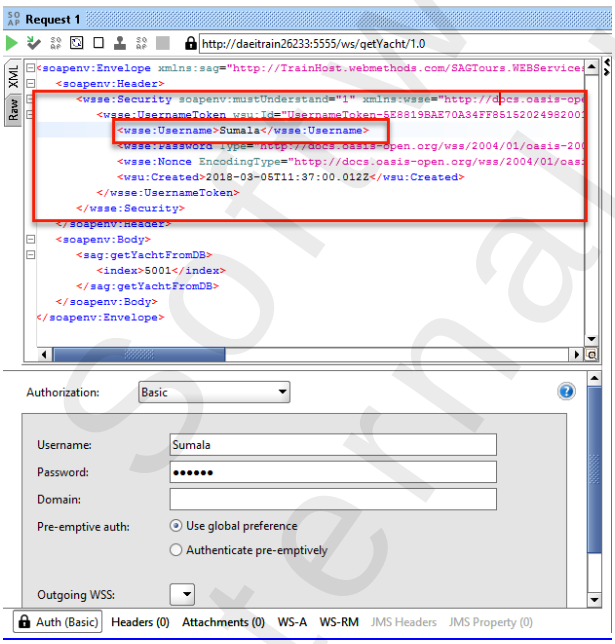




11. Select Password Type: PasswordText.

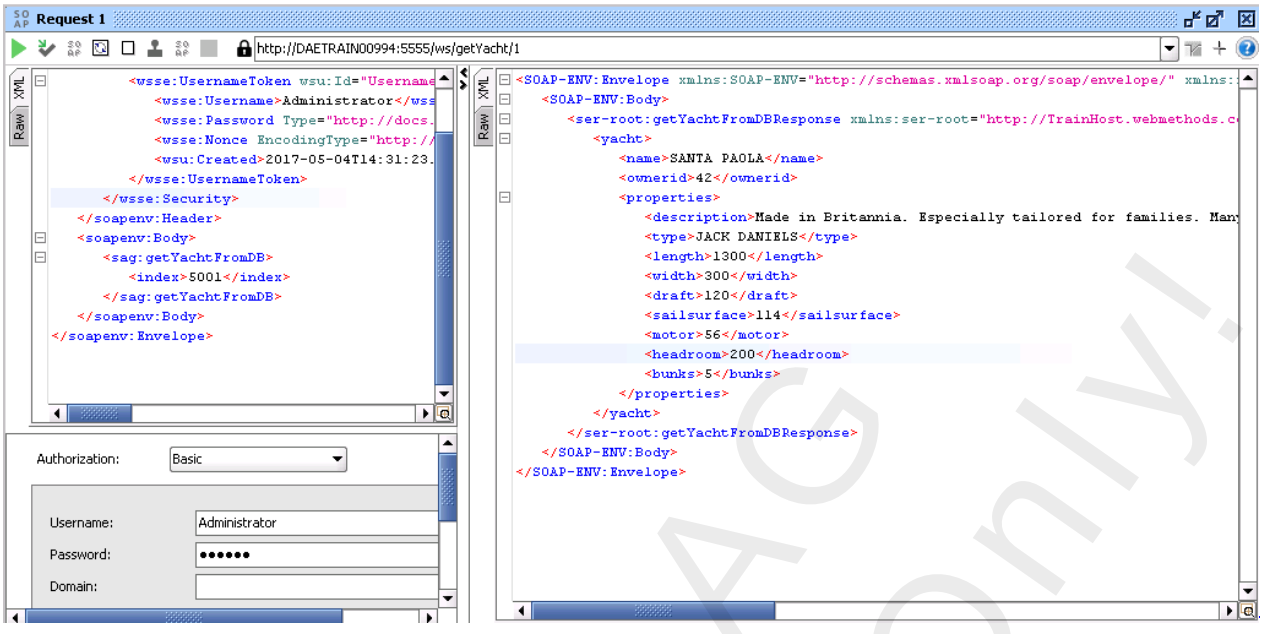


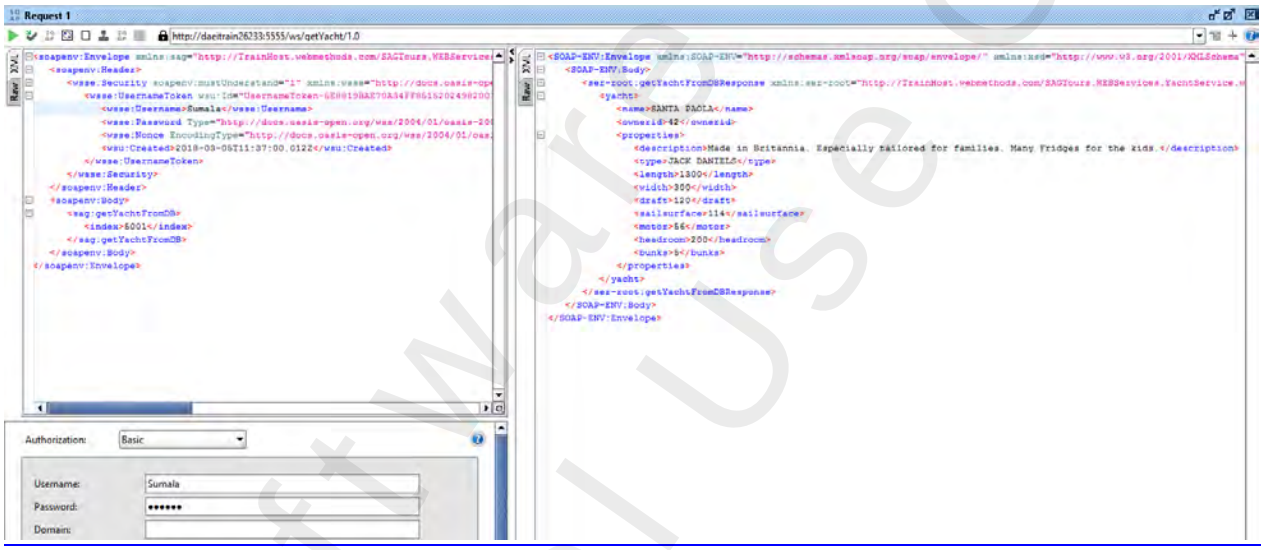

12. This will use credentials define in Basic Authentication to create a WSS Username Token. This wsse:security definition will be added to the Request Body in the soapenv:Header section.




13. Rerun the request. The SOAP response shows the result and is successful.



# Exercise 8:
# Creating a Consumer Application

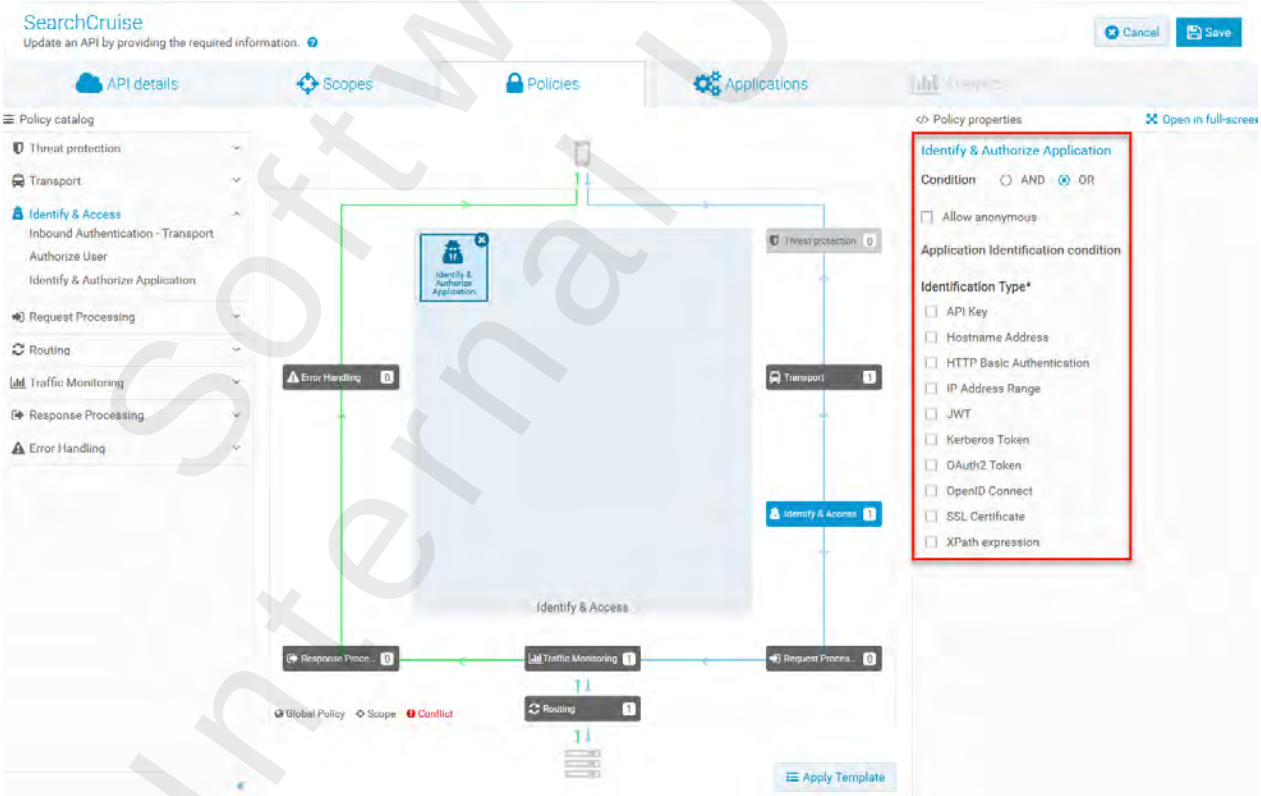
## Objectives

In this exercise, you will learn how to automatically provision and enforce a consumer relationship in API Gateway.

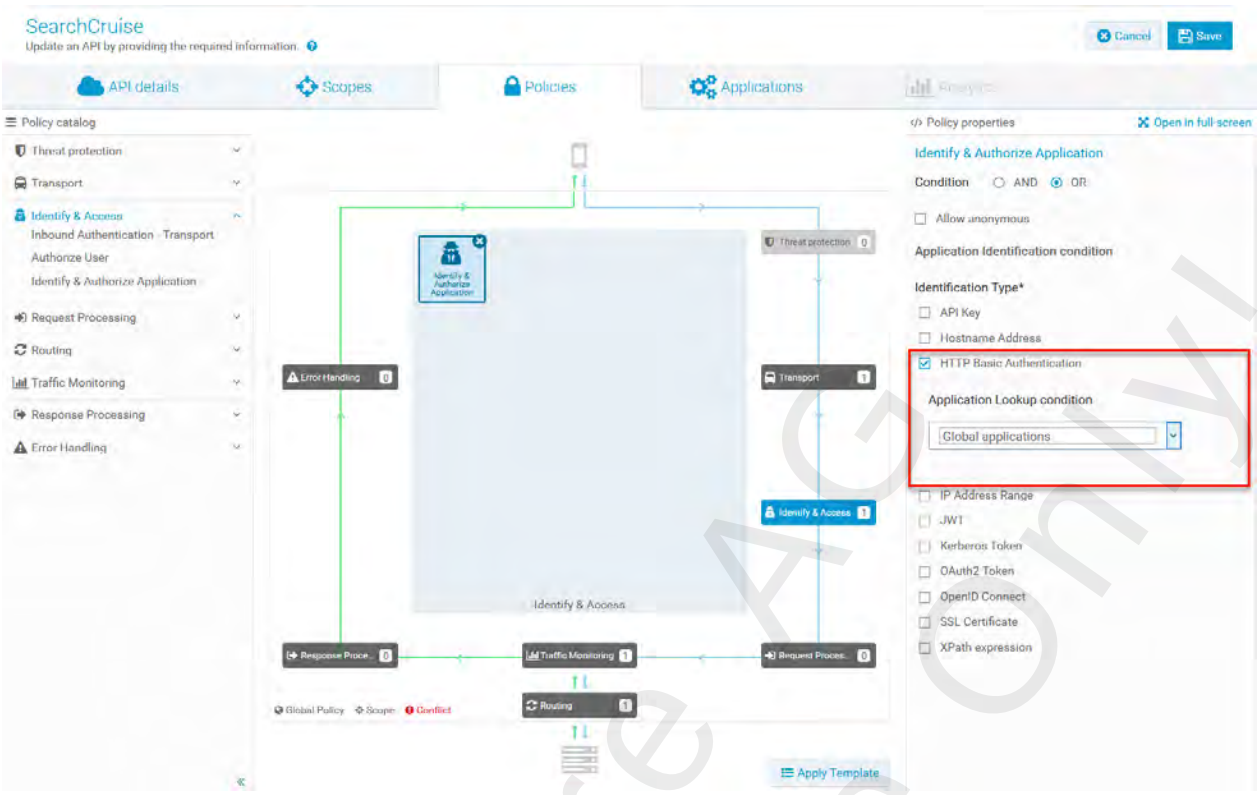
## Steps

1. Open Windows Services UI to double check that the following services, needed for API Gateway are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Event Data store 10.1 – should be started together with IS**
   c. **Software AG Runtime 10.1**
2. Login to the **API Gateway UI** as user **Sumala/manage.** Search for the **SearchCruise** API. Deactivate the API in order to change the policy configuration. Click the **Deactivate** button.

   Click **Yes**.
   a. Click **Edit** to open API in Edit mode.
   b. Navigate to **Policies > Identify & Access.**
   c. Remove the **Inbound Authentication – Transport** Policy action.
3. Click on **Identity & Authorize Application** in **Policy catalog** section. This will move the policy action to the section **Policy properties**.

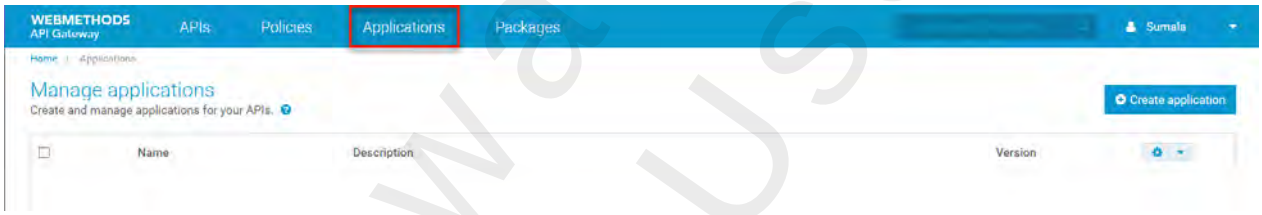



   d. In **Policy properties** select:
      i. **Identification Type**: HTTP Basic Authentication
      ii. **Application Lookup condition**: Global applications

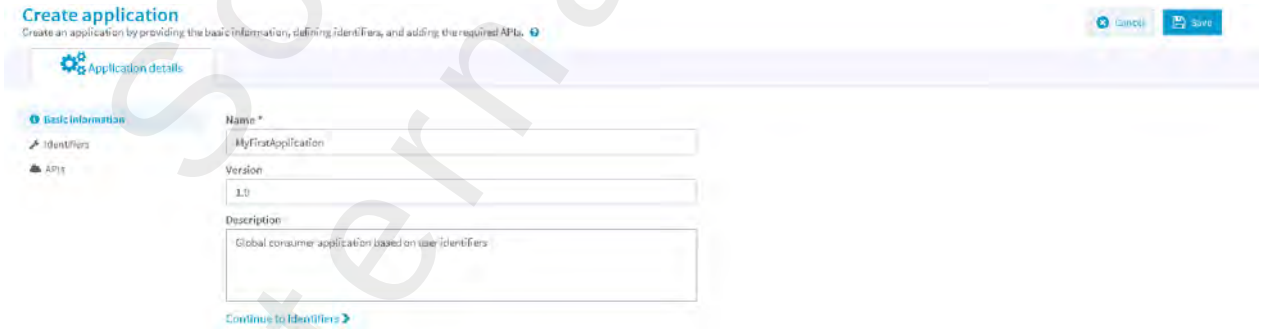



Click **Save**. Click **Activate.**

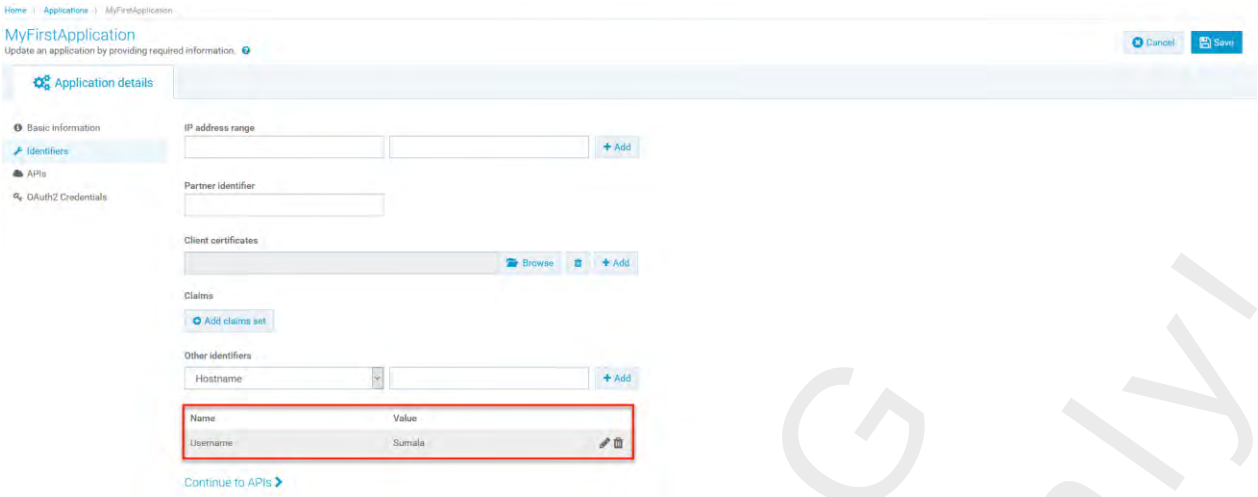
4. In the **API Gateway** header select the **Applications** to create a consumer application.




5. Click **Create Application**. Provide the following properties.
   d. **Name**: MyFirstApplication
   e. **Version**: 1.0
   f. **Description:** Global consumer application based on user identifiers




6. Click **Continue to Identifiers**. Within **Identifiers** select **Other identifiers** and do the following configuration:
   g. **Username**: Sumala
   Click **Add**.



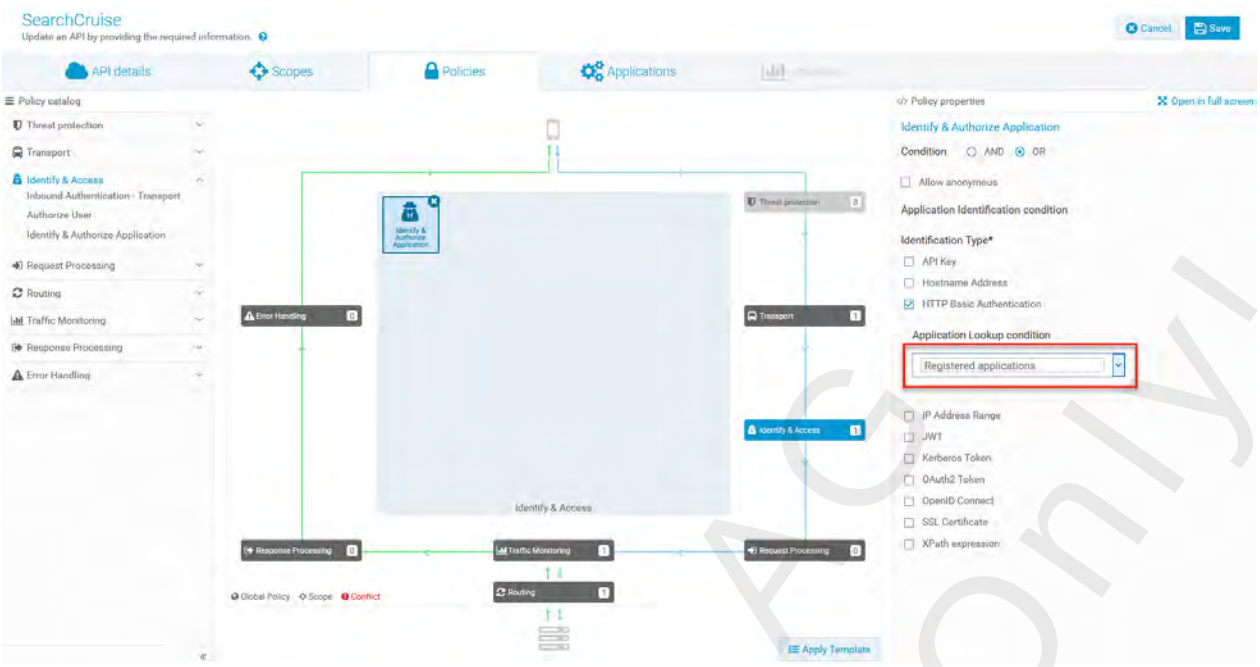

Click **Save**.

7. Open **Rest Client** and run request against **SearchCruise** using **Basic Authentication** set to **Sumala/manage**


8. Change the user credentials to **Administrator/manage**. Administrator is a valid user in Integration Server, but he is not member of the application we have just created. Therefore the request fails.

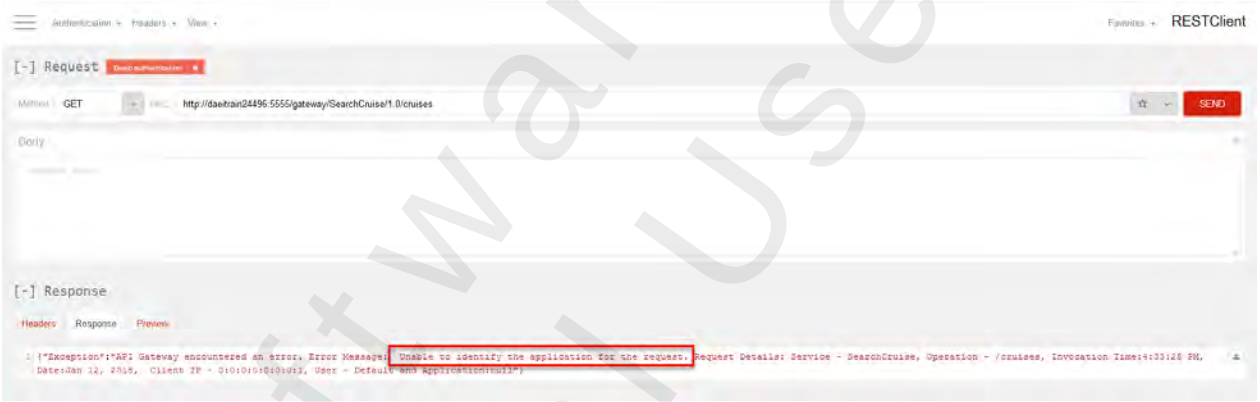

9. Now we want to enforce a close relationship between the API and the Consumer. Therefor we change the Application Lookup condition to Registered Application. Go back to **API Gateway**. Deactivate the **SearchCruise API** and switch to **Edit** mode. Navigate to the **Policies > Identify & Authorize Application**. Within the **Policy properties** change the **Application Lookup condition** as following:

   h. **Application Lookup condition:** Registered applications

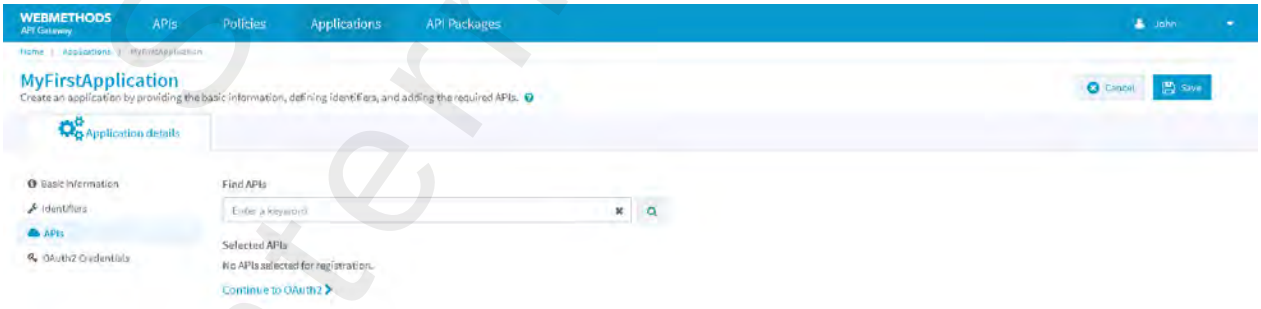



Save the changes and activate the API.

10. Go back to **REST Client** and try to rerun the **SearchCruise API** using credentials for **Sumala/manage**. The request will fail, because the application we have created is not registered or assigned to the **SearchCruise API.**

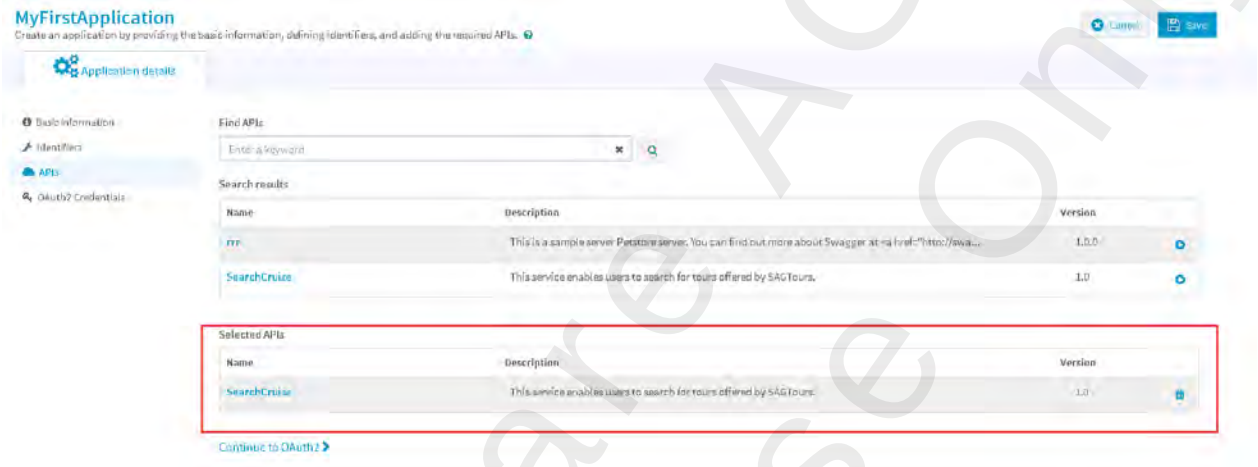



11. We change the global application to be an application for the **SearchCruise API**. Navigate to **Applications** in **API Gateway** header. Select the application **MyFirstApplication**. Switch to **Edit** mode.
12. From the navigation panel under **Application Details** select **APIs**

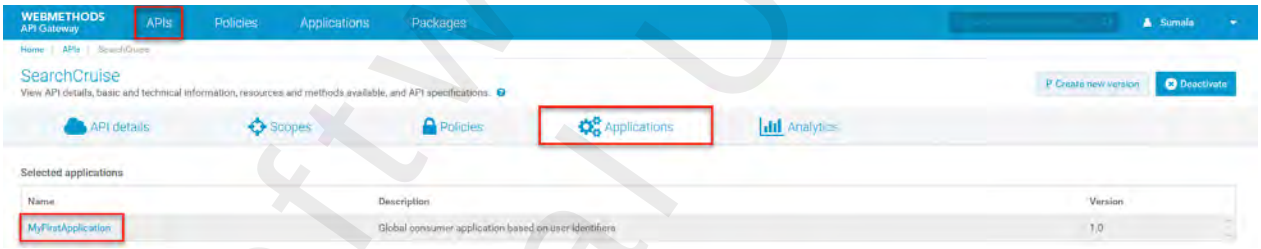



13. Click the Search icon and select the **SearchCruise API**.

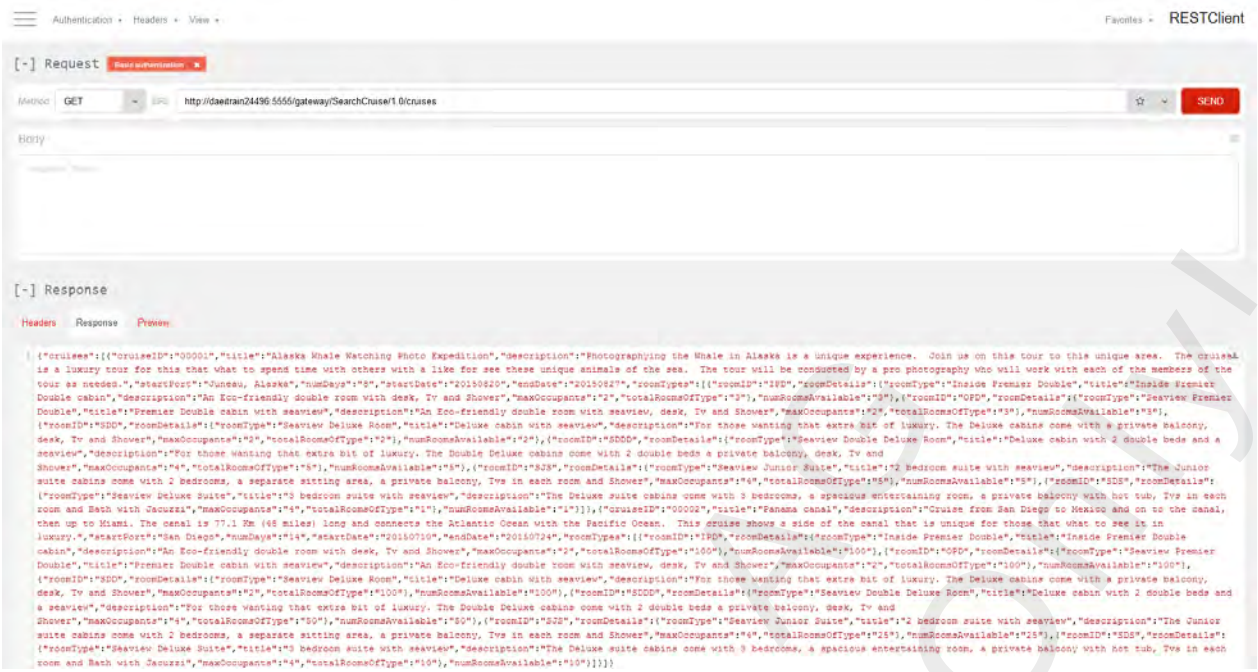

14. Now the **SearchCruise API** is added to the application.




15. Click **Save**. Go back to **SearchCruise** API and verify that the application is also seen from the **SearchCruise API.**




16. Rerun the request against **SearchCruise API** in **RESTClient**. Now the user **Sumala/manage** can run the request.



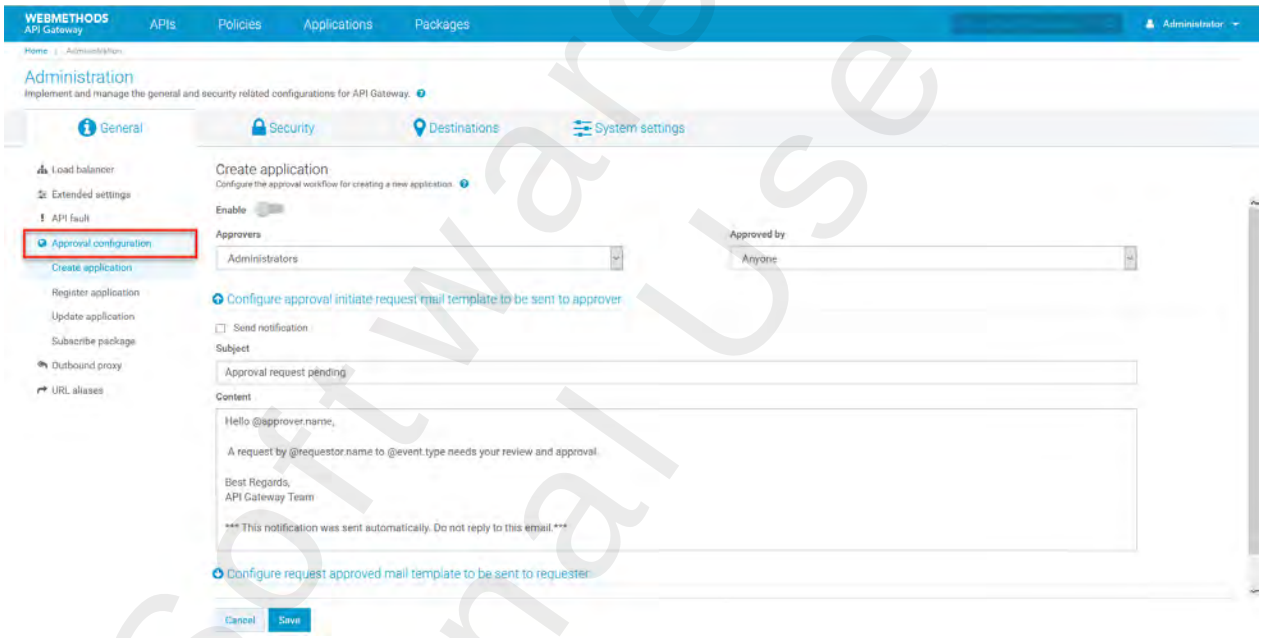
# Exercise 9:
# Enforcing a Consumer Relationship using API Key

## Objectives

In this exercise, you will learn how to enforce a valid APIKey for the SearchCruise API. For application creation and registering an application we will enforce the Approval process.

## Steps

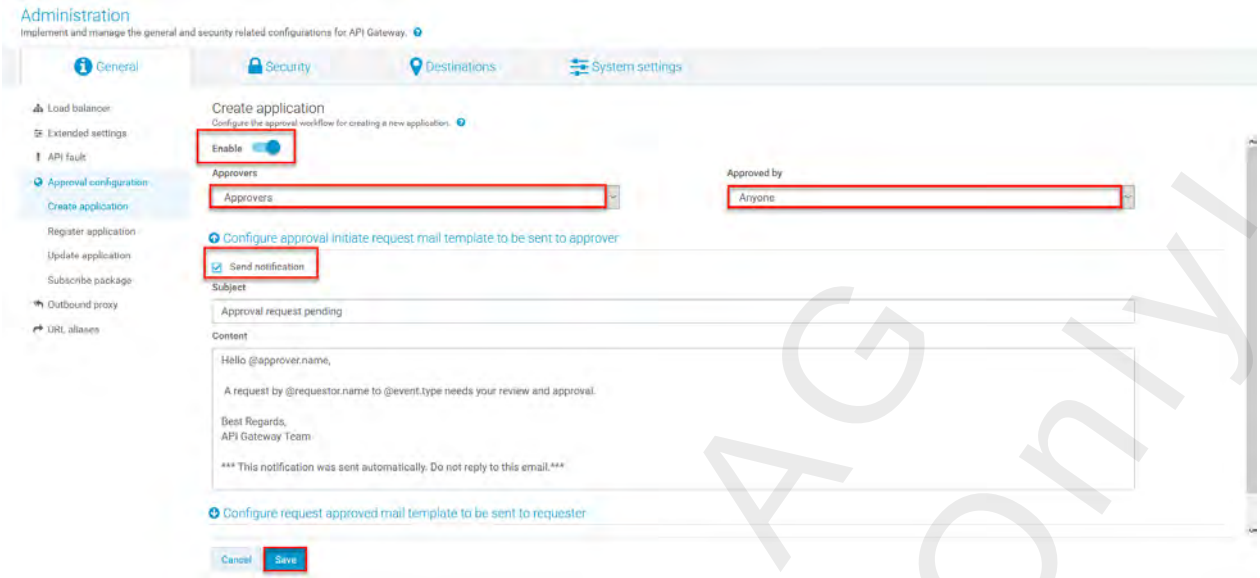
1. Open Windows Services UI to double check that the following services, needed for API Gateway, are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Event Data store 10.1 – should be started together with IS**
   c. **Software AG Runtime 10.1**
   d. **hMail-Server**
2. Login to the **API Gateway UI** as user **Administrator/manage** and navigate to **Administration > Approval Configuration**.




3. Within **Create application p**rovide the following properties
   a. **Enable**: - turned on -
   b. **Approvers:** Approvers
   c. **Approved by:** Anyone
   d. **Configure approval initiate request mail template to be sent to approver**
      i. **Send notification**: - select –
      ii. **Subject**: - keep the defaults -
      iii. **Content**: - keep the defaults –
   e. **Configure request approval mail template to be sent to requester**
      i. **Send notification**: - select –
      ii. **Subject**: - keep the defaults -
      iii. **Content**: - keep the defaults -
   f. **Configure rejection mail template to be sent to requester**
      i. **Send notification**: - select –

ii. **Subject**: - keep the defaults -

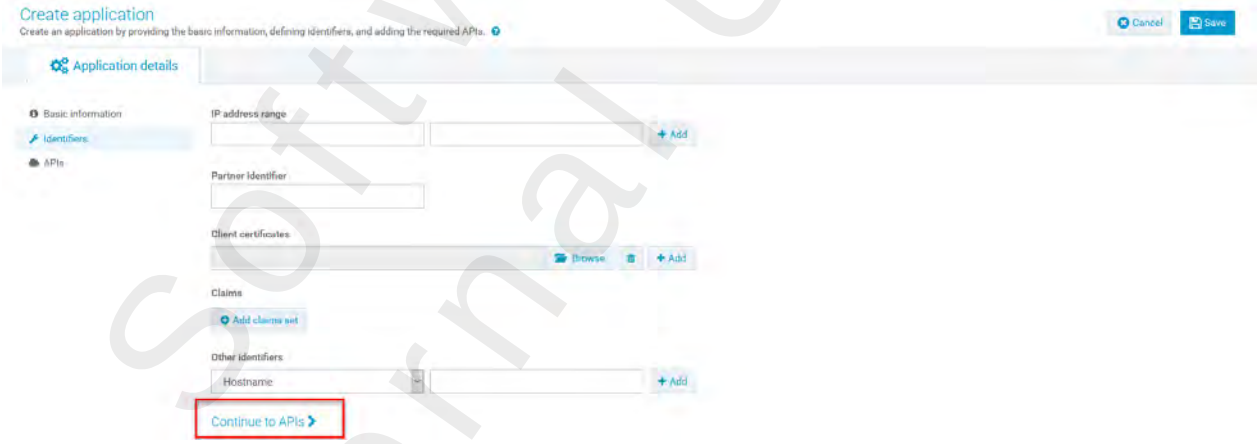
iii. **Content:** - keep the defaults -




Click **Save**.

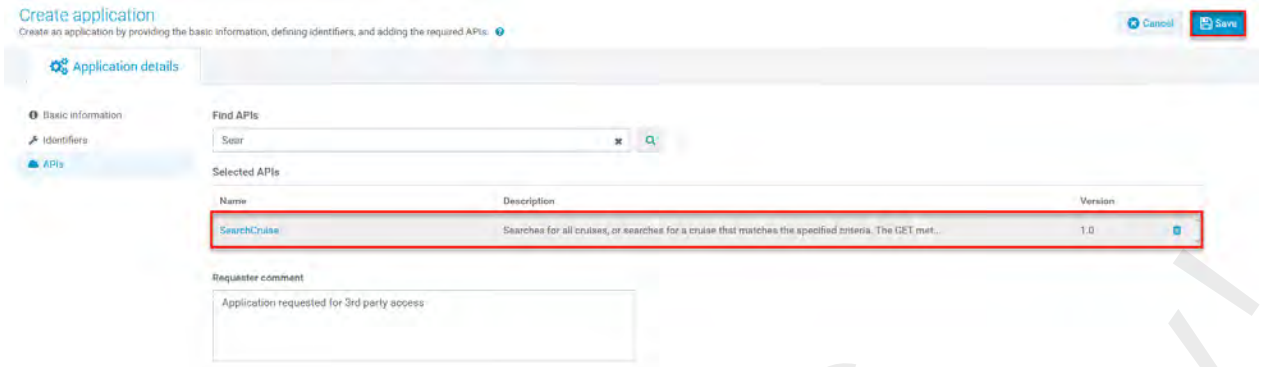
4. Do the same configuration for the workflows **Register application** and **Update application**.
5. Login to the **API Gateway UI** as user **Sumala/manage** and create a new application. Navigate to Applications, click **Create application**. Provide the following properties:
   a. **Name**: MyApplicationJustAPIKey
   b. **Version**: 1.0
   c. **Description**: Application following the Approval workflow and without identifiers. Just the API Key is identifying the consumer.
   d. **Requester comment**: Application requested for 3rd party access

   Click **Continue to Identifiers**.



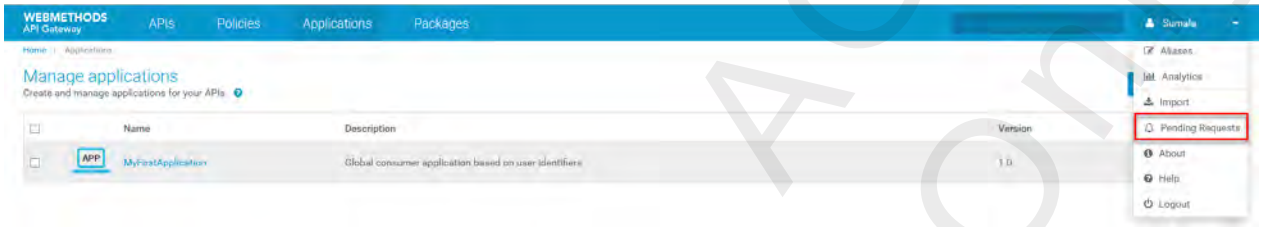

Click **Continue to APIs**. Provide the following properties:

   e. **Selected APIs:** SearchCruise

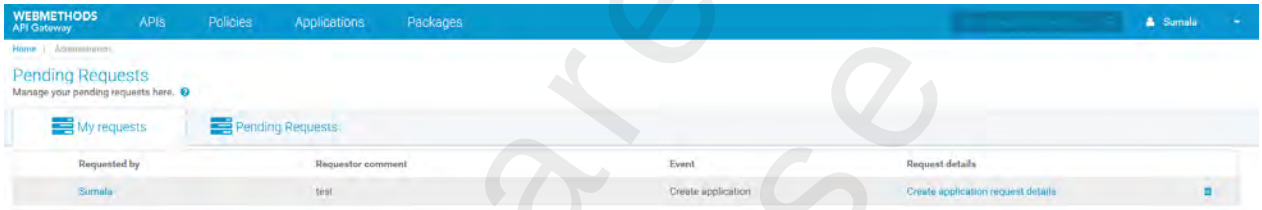



Click **Save.**

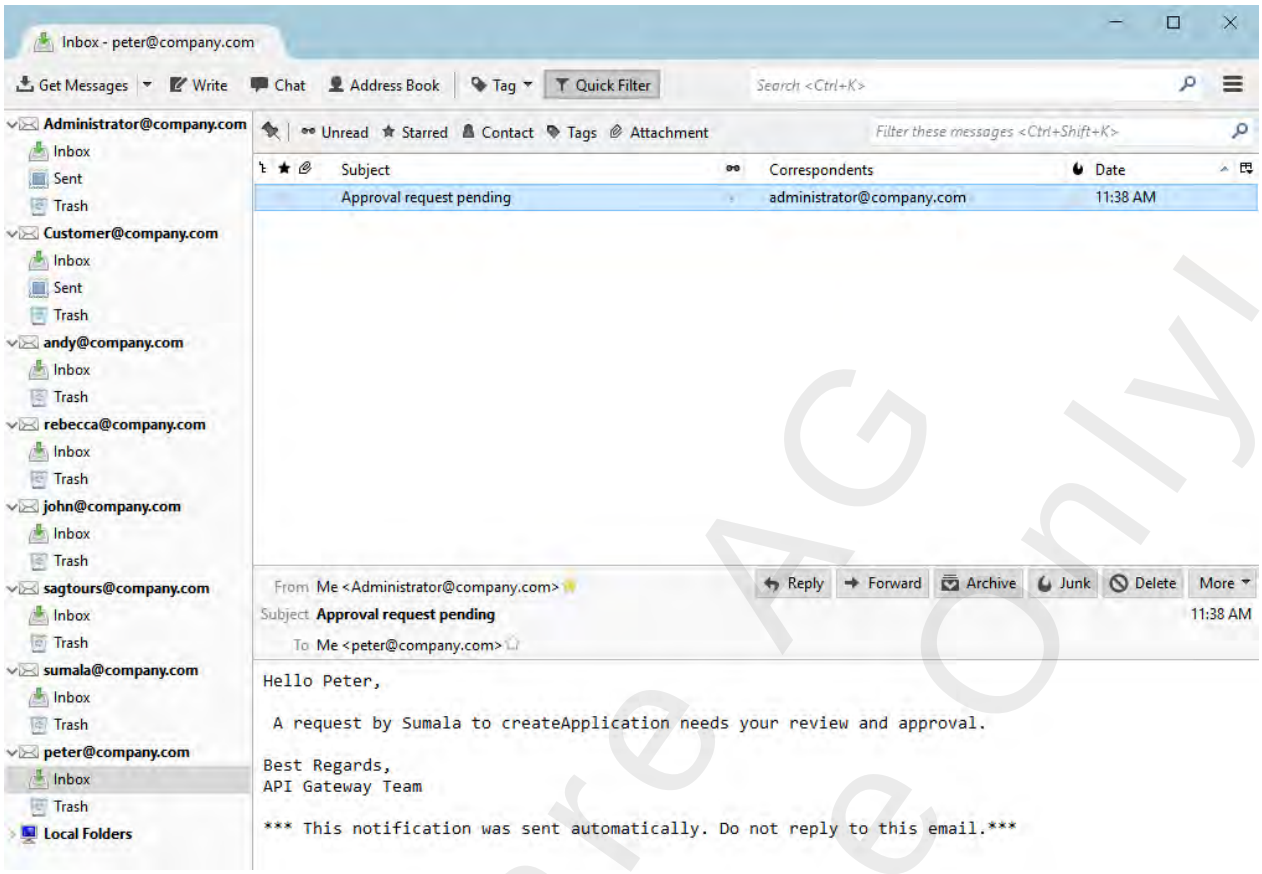
6. Review the request for creating this application. Navigate to **Sumala > Pending Requests**.



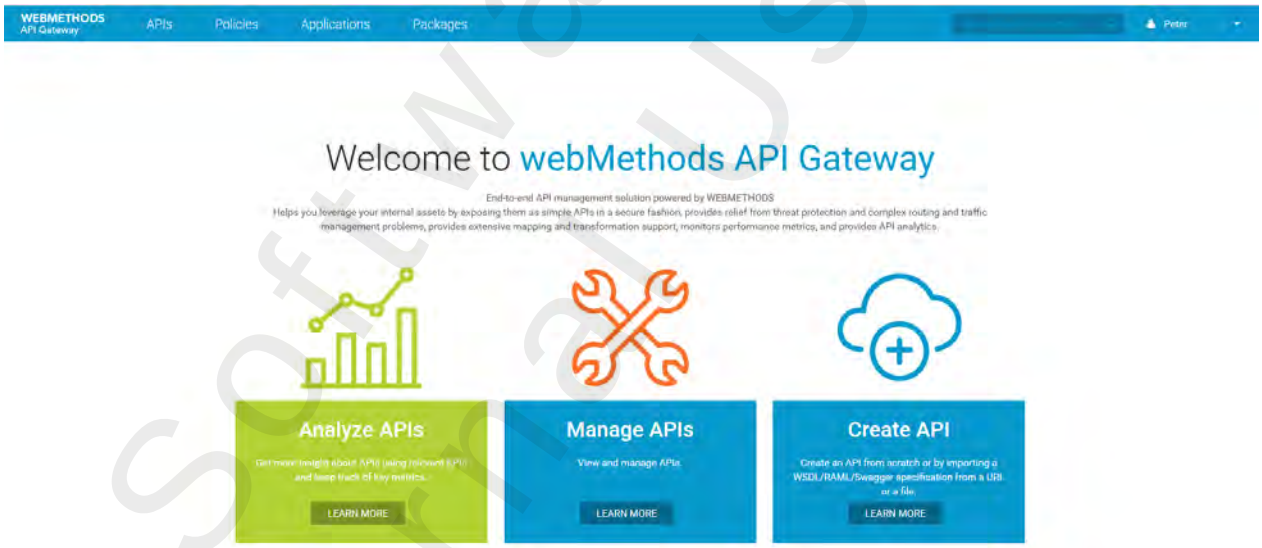

You will find **Sumuala**'s request within the tab **My requests**.



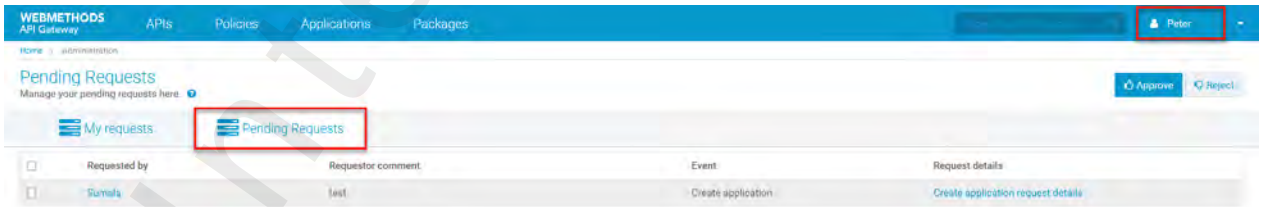

7. As configured in the approval workflow the user attached to the **Approver access profile** will get an email notification. In exercise 1 we have created the user **Peter** as user of the approver group with corresponding access profile. Open **Mozilla Thunderbird** and get messages for user **peter@company.com.**



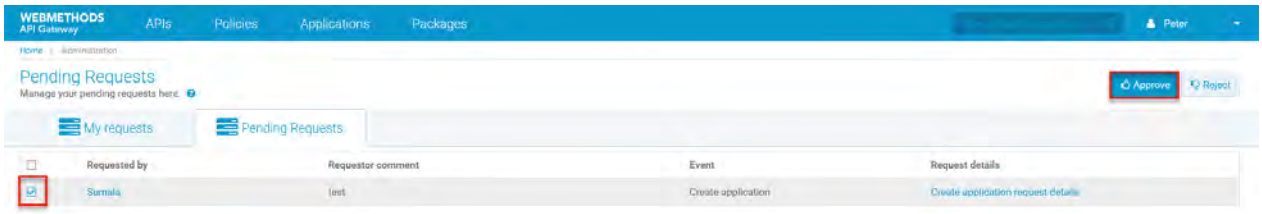

8. Logon to **API Gateway** as user **Peter/manage.**



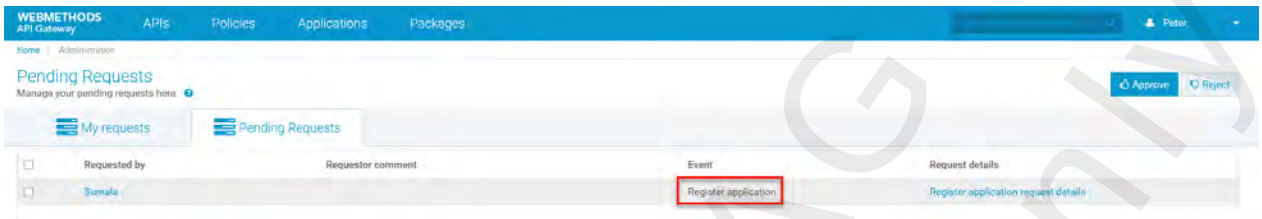

9. Navigate to **Peter > Pending** requests. You will find Sumala's request to create the application within the tab **Pending Requests.** The requests are listed with the events definition**.**




   a. Select the request **Create application** and click **Approve** to approve the create application request.

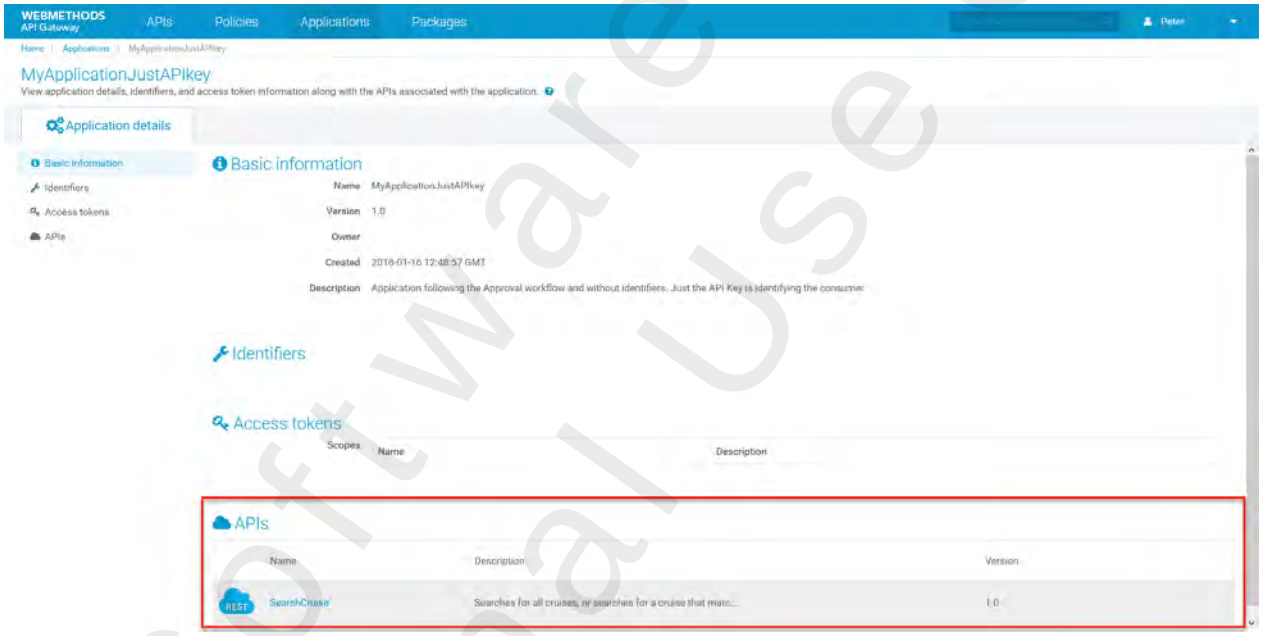



b. After the **Create application** request is approved, the **Register application** request is shown in the list of **Pending Requests**.

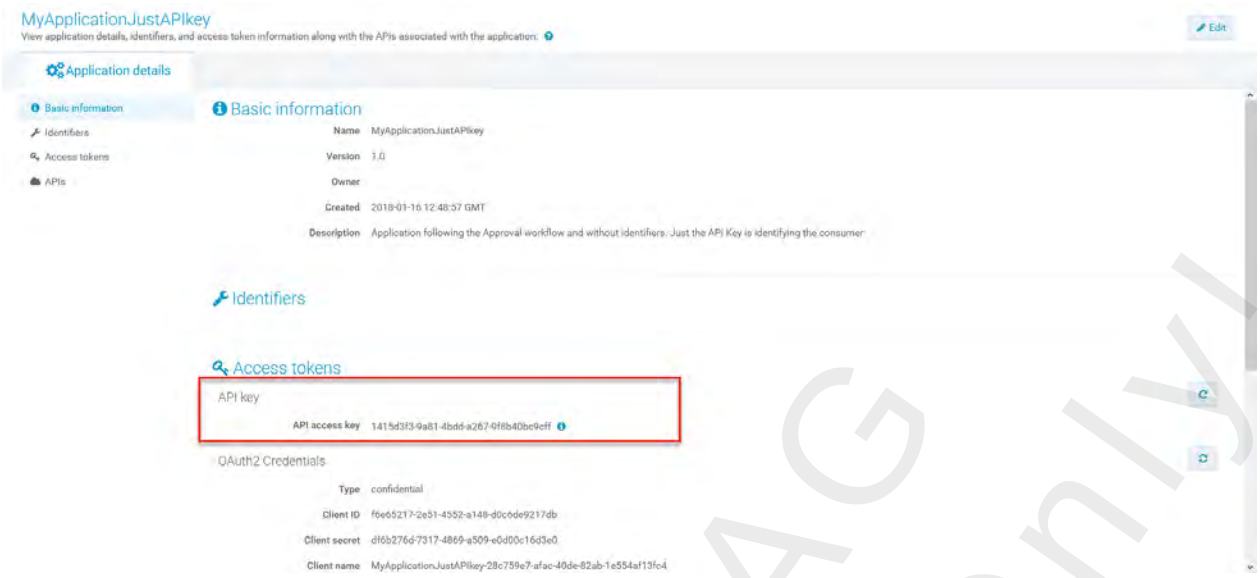



c. Approve this request.

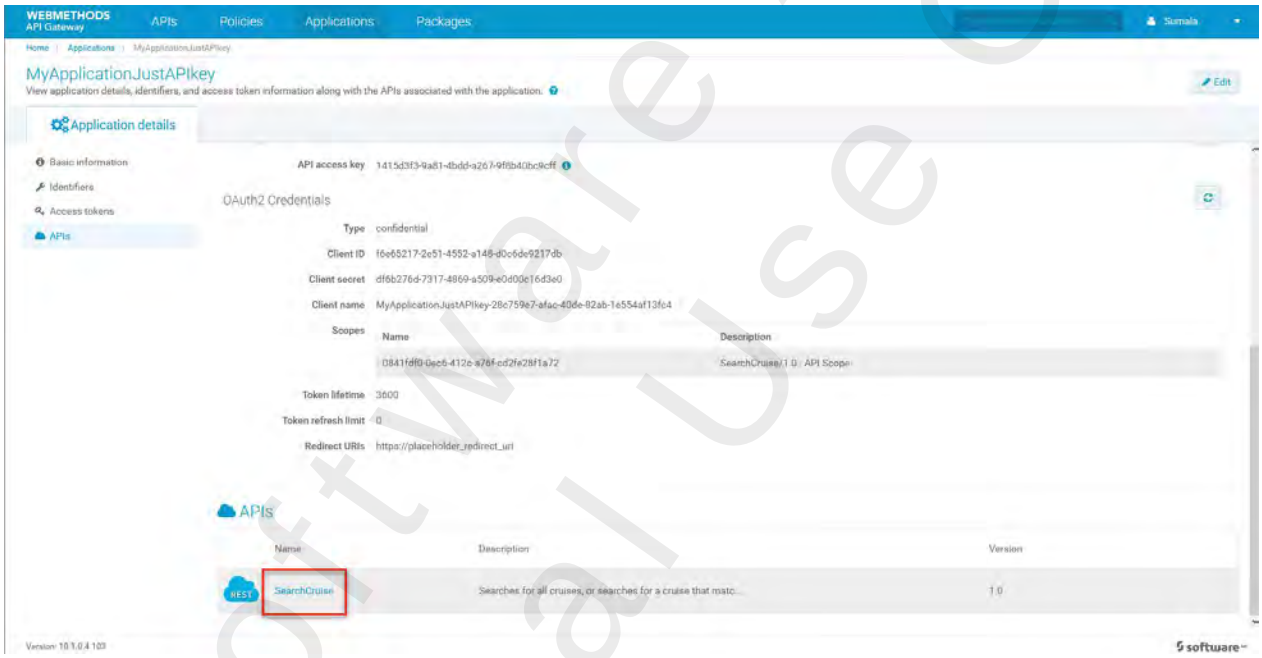
10. Navigate to **Applications.** The new created application is listed. Review the application **MyApplicationJustAPIKey**. Because Peter is not the owner of this application he doesn't see the acces token definitions.




11. Logon to **API Gateway** as user **Sumala/manage**. Navigate to the **Application Details** page of **MyApplicationJustAPIKey.** The **API Key** is created as part of the application.

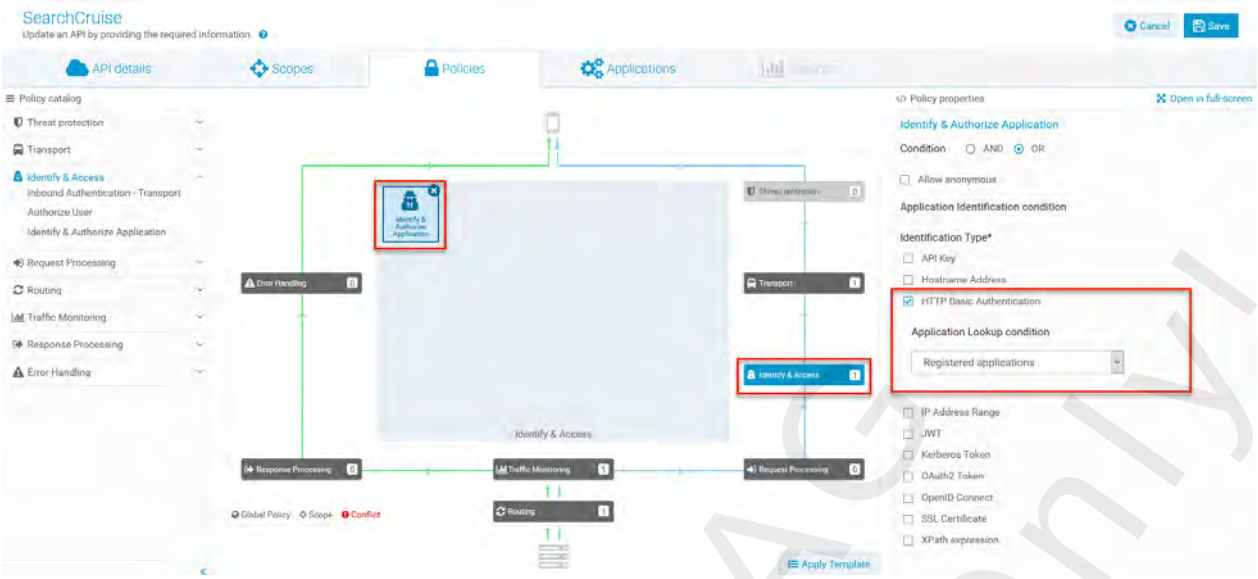



5. Now we need to configure the security policy action for the service **SearchCruise**. Navigate to APIs, which lists the service **SearchCruise**. Click on **SearchCruise.**

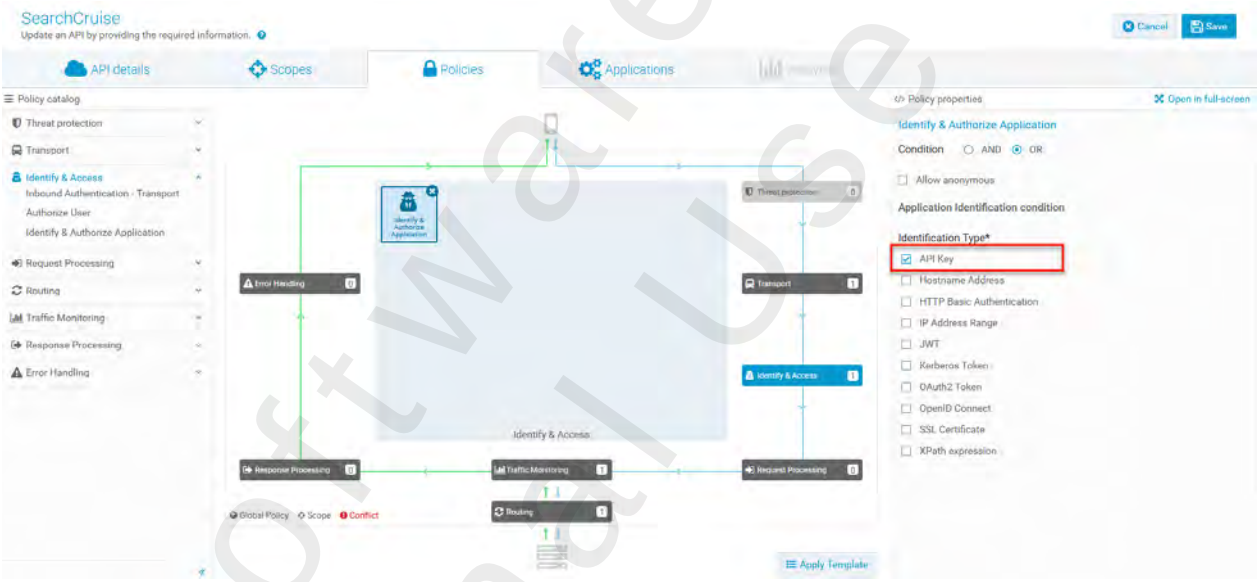



This will open the Detail View of the **SearchCruise** service.

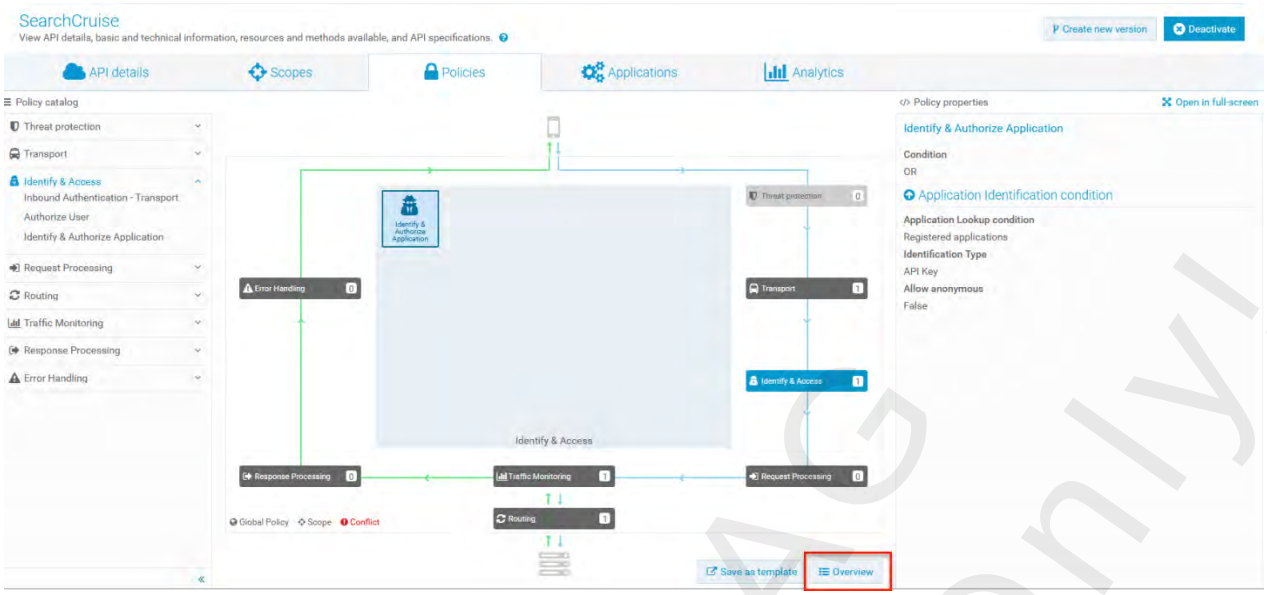
6. Deactivate the API and switch to **Edit** mode. Navigate to **Policies -> Identify & Access > Identify & Authorize Application**



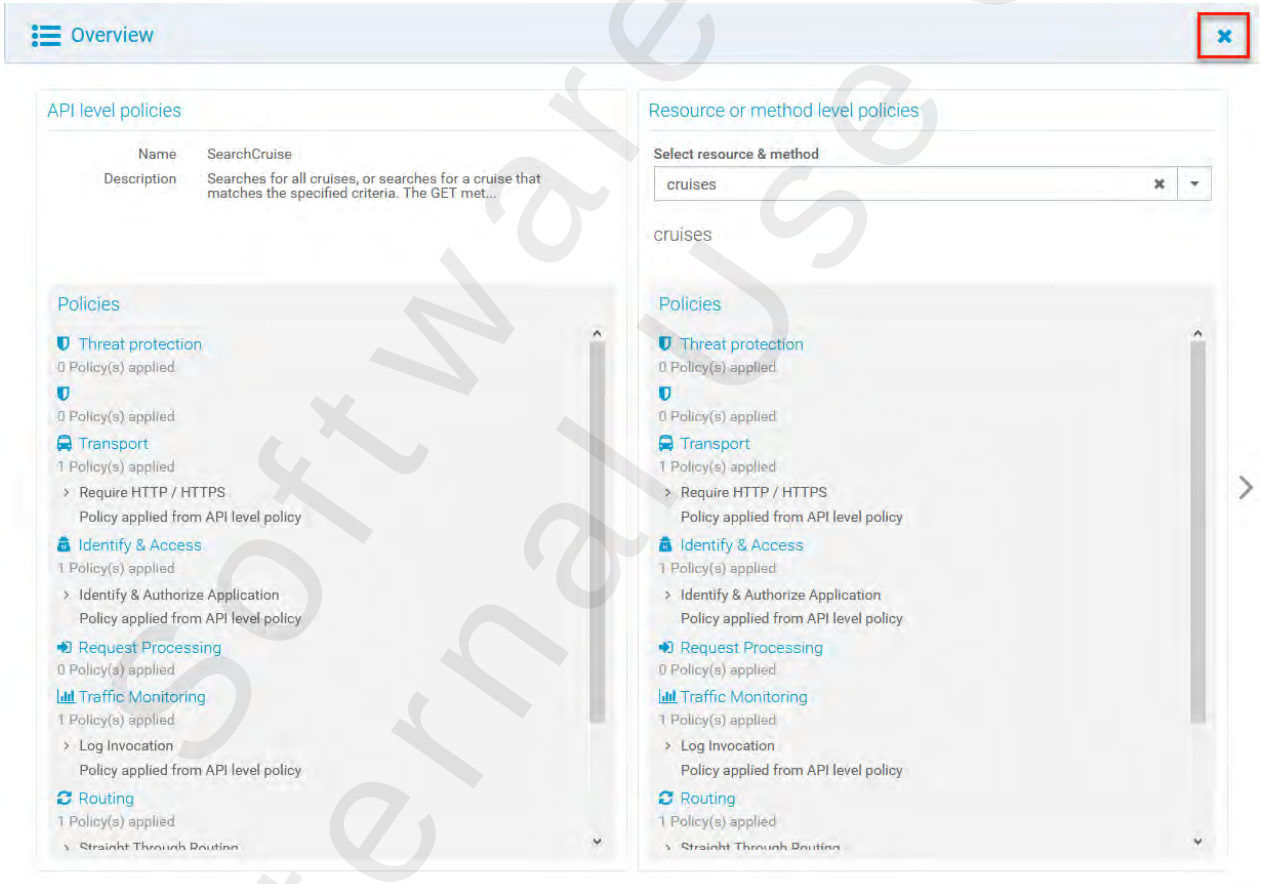

8. Within **Policy properties > Identify & Authorize Application > Identification Type**
   a. Deselect **HTTP Authentication**
   b. Select **API Key**




9. Save and activate the **SearchCruise** API.
10. Review all policy configurations by clicking the Overview icon.

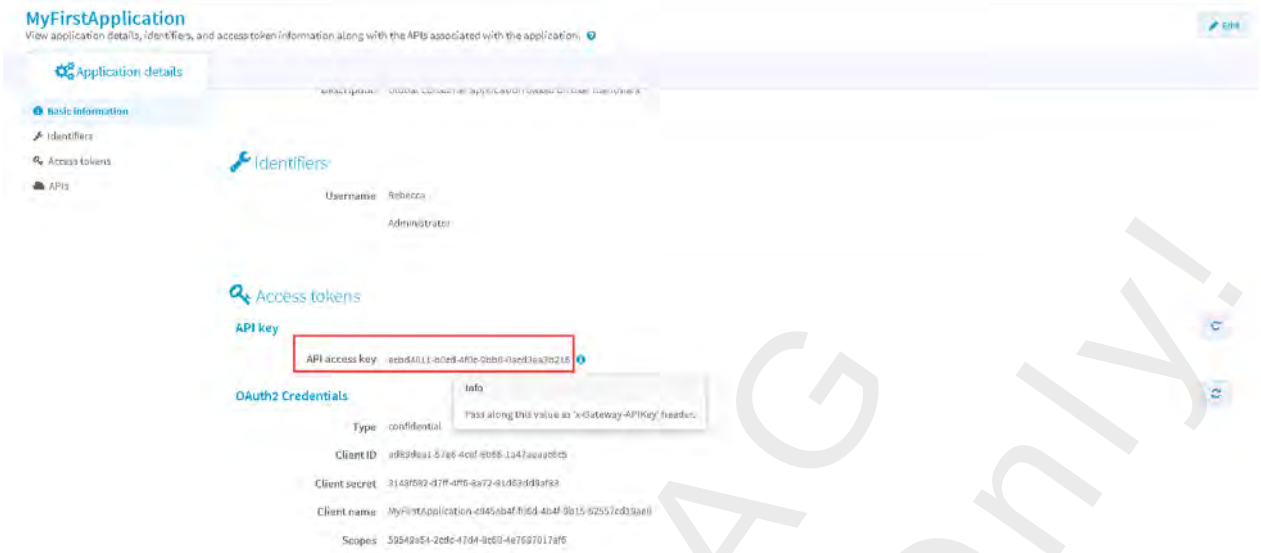



11. The **Overview** page shows a list of policies for the overall API and for a single resource or method. Close the page by clicking the blue cross in the right hand upper corner.




12. The API Key which need to be sent by the consumer when calling the **SearchCruise API** is part of the application registered to the **SearchCruise API**. Open **Application** tab to list all registered applications. Please open the application **MyFirstApplication** in Detail View. Hover your mouse over

the little blue icon next to the API Key value. You will see a note how to pass along the API key.




13. Open **REST Client**. Configure a **GET** request against our virtual service including the header parameter which provides the API Key. In addition to get the name of the header variable as mention in the previous step, you will find the name of the custom headers in the file C:\Training\APIGateway\Exercise9\SearchCruiseGET.txt.

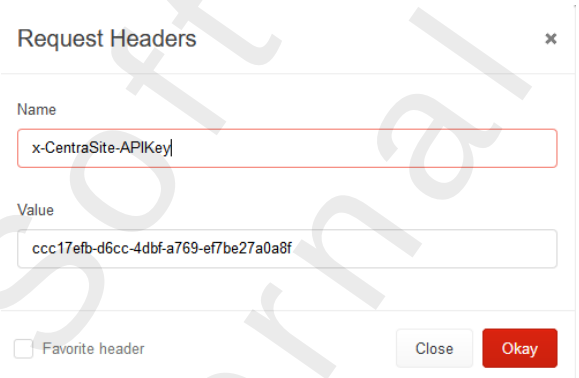
    a. **Methods**: GET

    b. **URL**: Copy and paste the Access URL out of the **Consumer Overview** profile: http://localhost:5555/gateway/SearchCruise/cruises

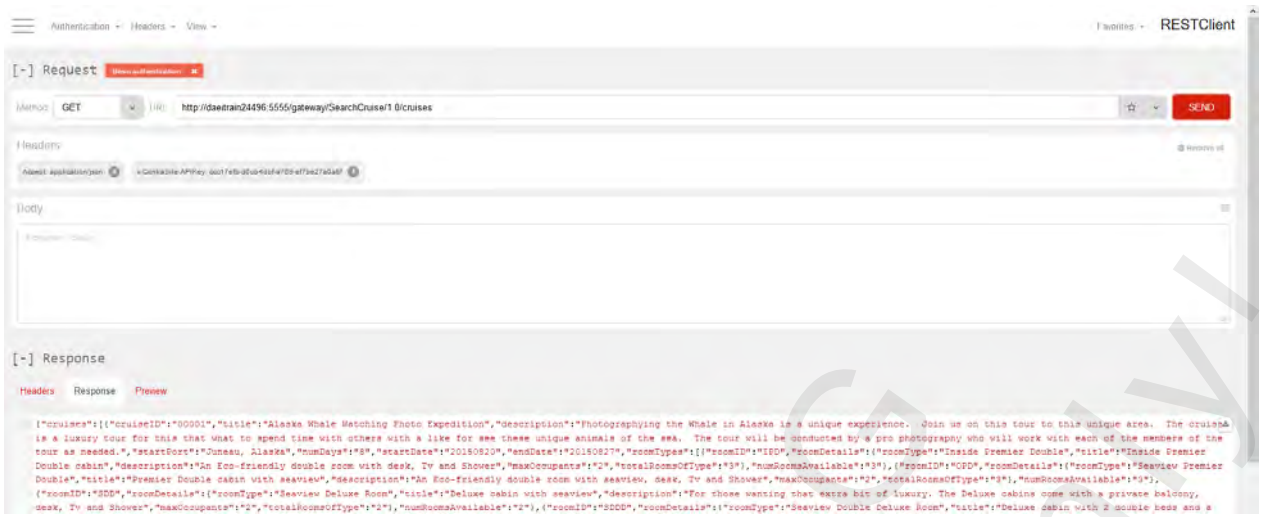
    c. **Custom Header:**

        i. **Accept**: application/json

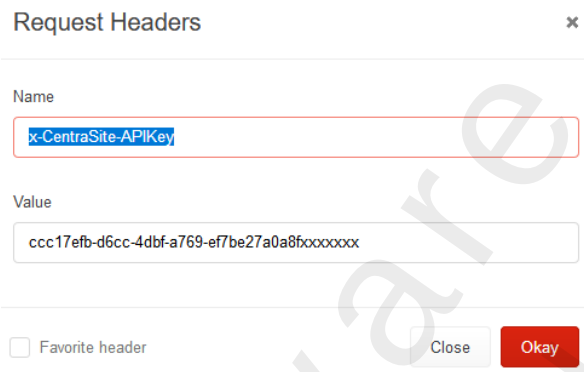
        ii. **x-Gateway-APIKey:** - value of API key as contained in the email or in API Consumer Profile, see above



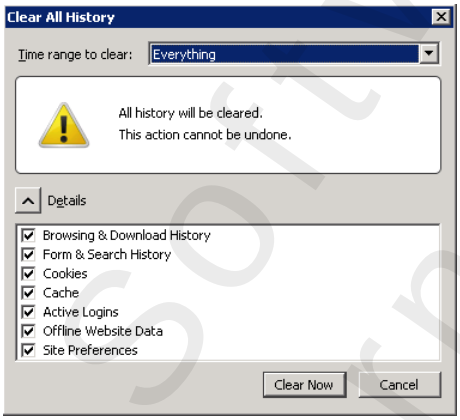

    d. Run the request using the Send button and verify the Response message. It should be the same result as before when running without **API Key** configuration.



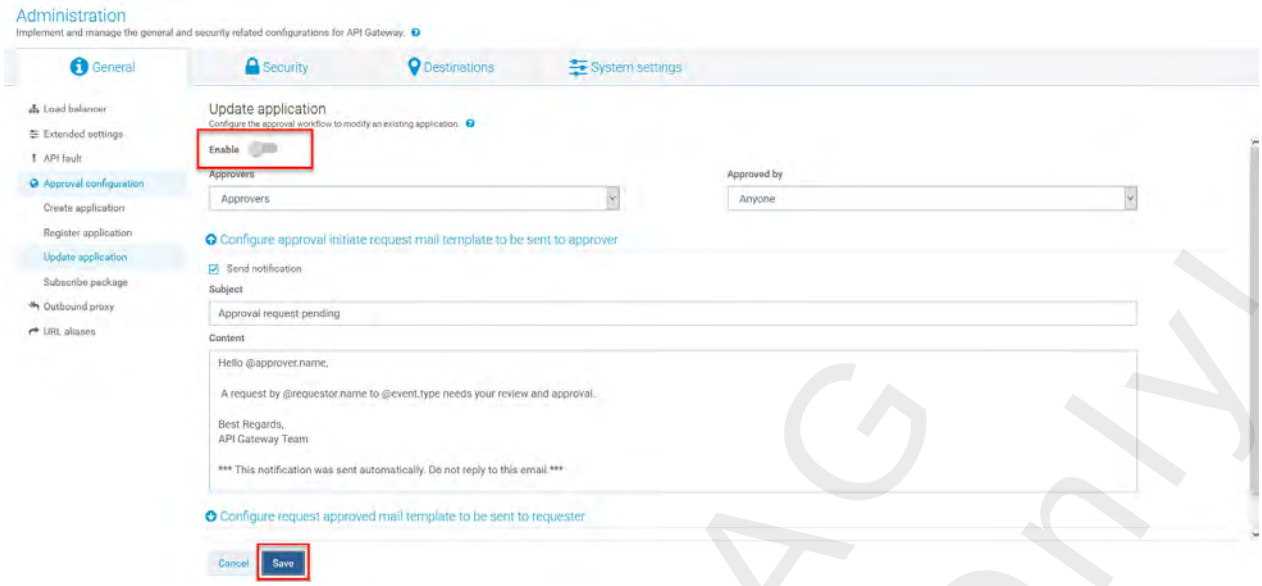

14. Verify that a valid API key is needed to call the service. Open the HTTP header defined to pass on the API key and change the valid API key to an invalid one by adding a number of xxxxxxx to it.




15. In case you don't get an error response, please clear the Firefox cache (press Ctrl + shift + delete). Set the time to clear to Everything.




16. Login to **API Gateway** as user **Administrator/manage**. Navigate to **Adminstration > Approval configuration.** Within Create application use the slider to disable the approval workflow.




Click **Save**.

17. Disable the workflow for the processes: R**egister application**.

This page intentionally left blank

# Exercise 10:
# Managing API Threat Protection

## Objectives

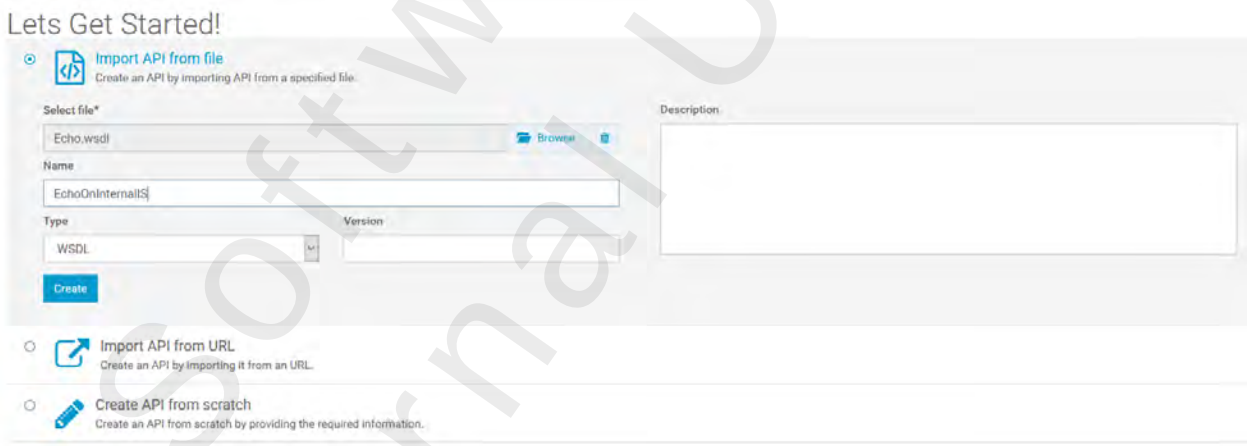
In this exercise, you will define some threat protection rules which will be imposed by API Gateway before the request reaches the internal applications. The internal application on our Virtual Machine is inside another installation of Integration Server running on port 7777.

The internal Application is the echo service. The URL to the WSDL file of this API is:

**localhost:7777/ws/EchoWS?wsdl**

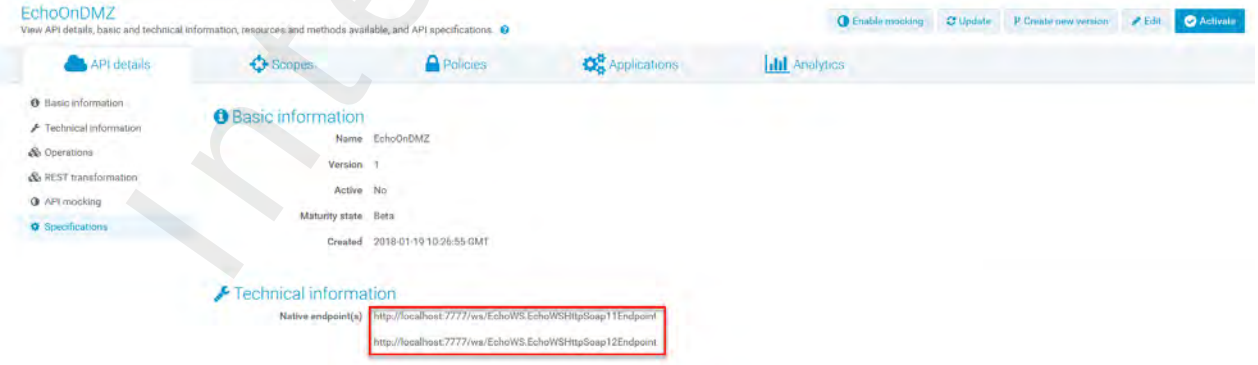
## Steps

1. Open Windows Services UI to double check that the following services, needed API Gateway, are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Internal Integration Server 10.1**
   c. **Software AG Event Data store 10.1 – should be started together with IS**
2. Logon to **API Gateway** as user **Administrator/manage.** Create a new API where the endpoint is on the preconfigured VM. Use the import function using a WSDL file.
   a. **Select File:** C:\Training\456-71E\Exercise10\Echo.wsdl
   a. **Name**: EchoOnInternalIS
   b. **Type**: WSDL
   c. **Version:**

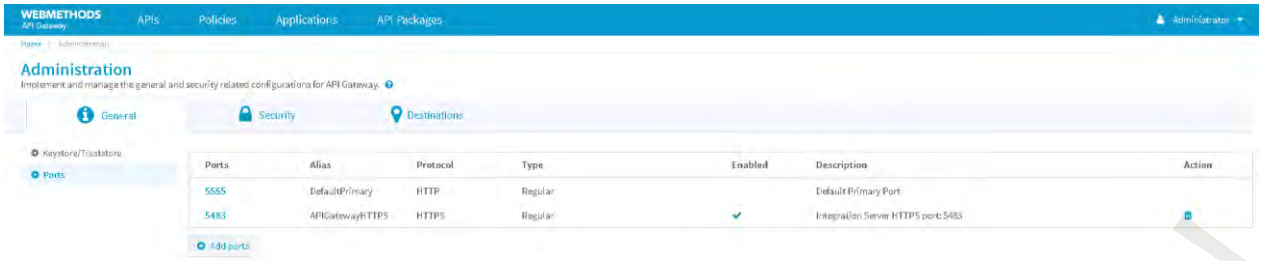



Click **Create.**

Navigate to Technical Information and verify that the input is on the specific Integration Server

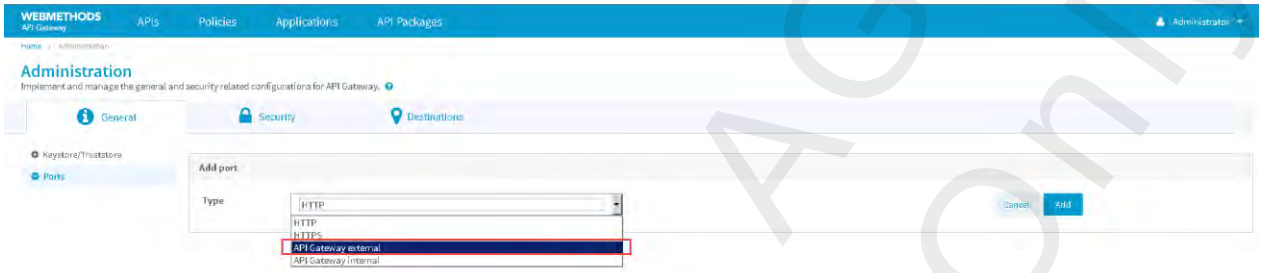


3. Navigate to the user menu (**Administrator** menu) and select **Administration > Security > Ports.**.



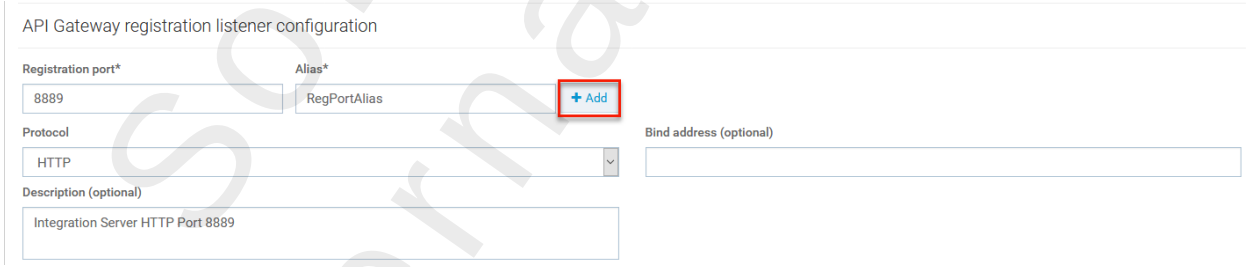

Click the **Add ports** button to create the External Port which is exposed to the consumer

4. Select type **API Gateway External**.



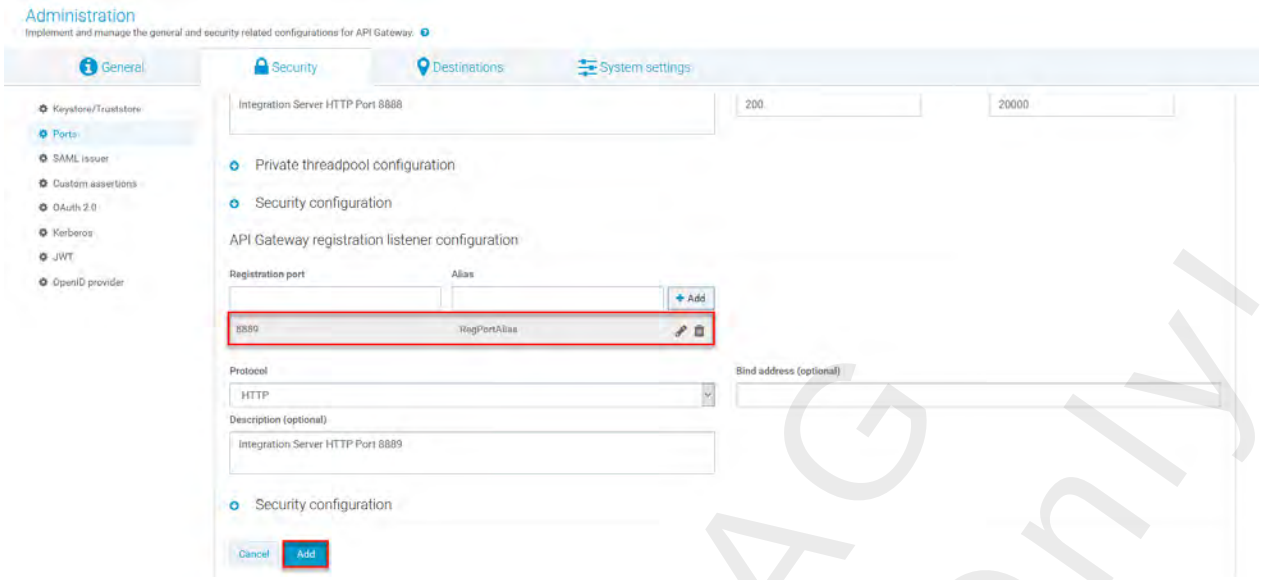

Click **Add.**

5. Provide the following properties
   a. **API Gateway external listener configuration**
      i. **External port**: 8888
      ii. **Protocol**: HTTP
      iii. **Alias:** ExtPortAlias
      iv. **Bind Address:** - leave the defaults –
      v. **Description:** Integration Server HTTP Port 8888
      vi. **Backlog:** - leave the defaults -
      vii. **Keep alive timeout**: - leave the defaults –
   b. **API Gateway registration listener configuration**
      i. **Registration port**: 8889
      ii. **Alias:** RegPortAlias
      Click **Add**

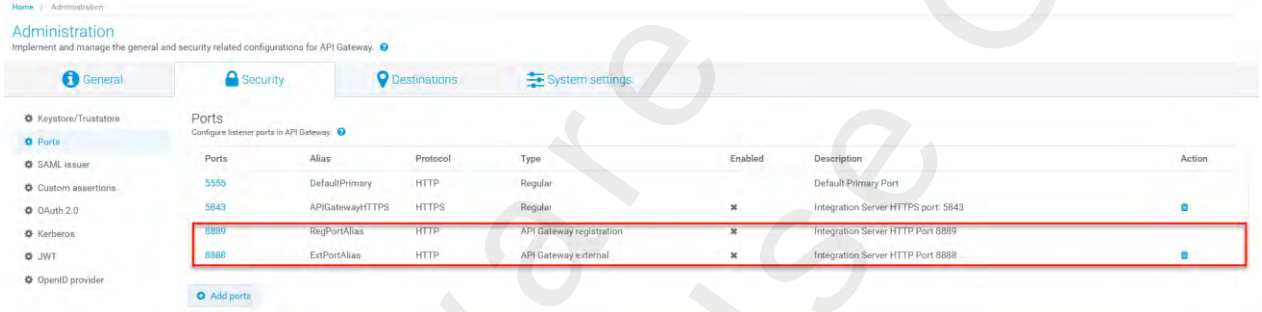



      iii. **Protocol**: HTTP
      iv. **Bind Address:** - leave the defaults –
      v. **Description:** Integration Server HTTP Port 8889

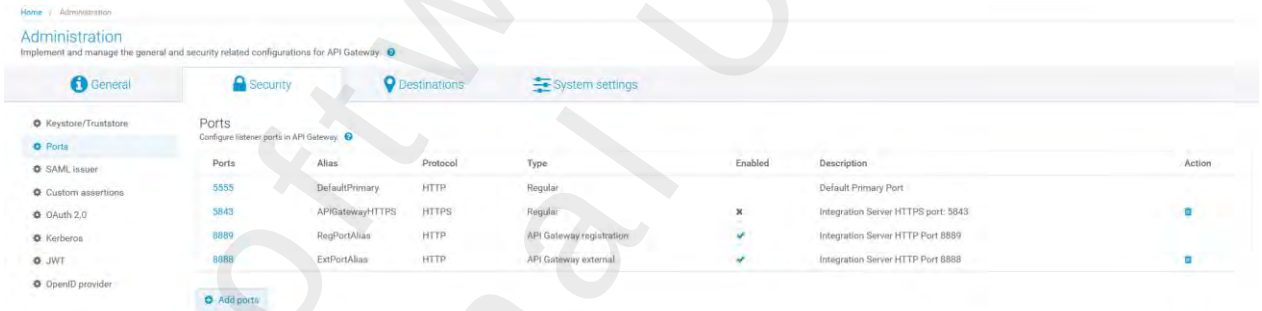
Click **Add** to add the external listener configuration.



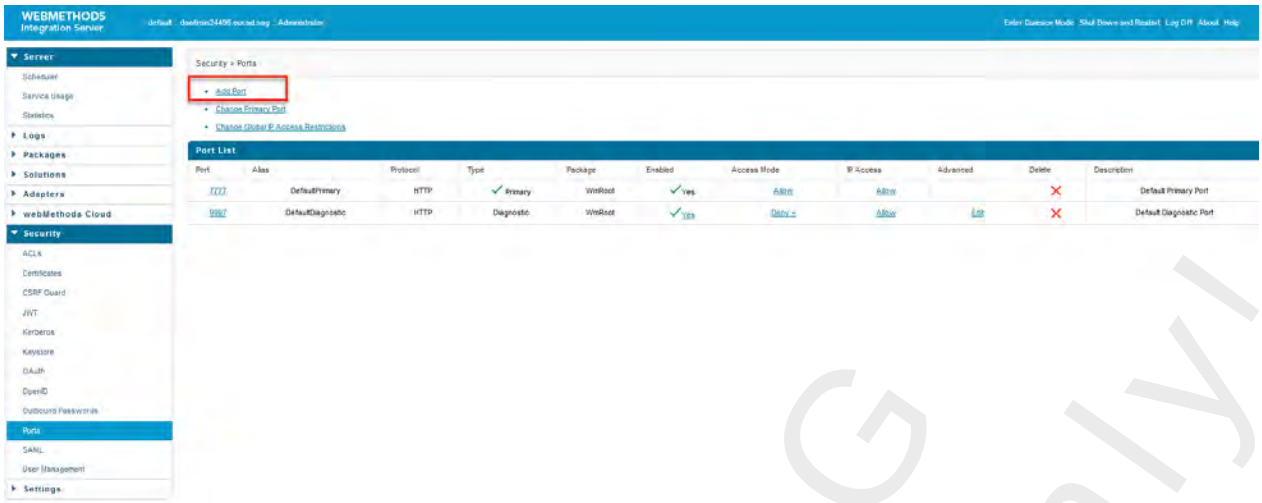

6. Activate the ports you have created in the previous step. Click the cross in the enabled column of the list of ports



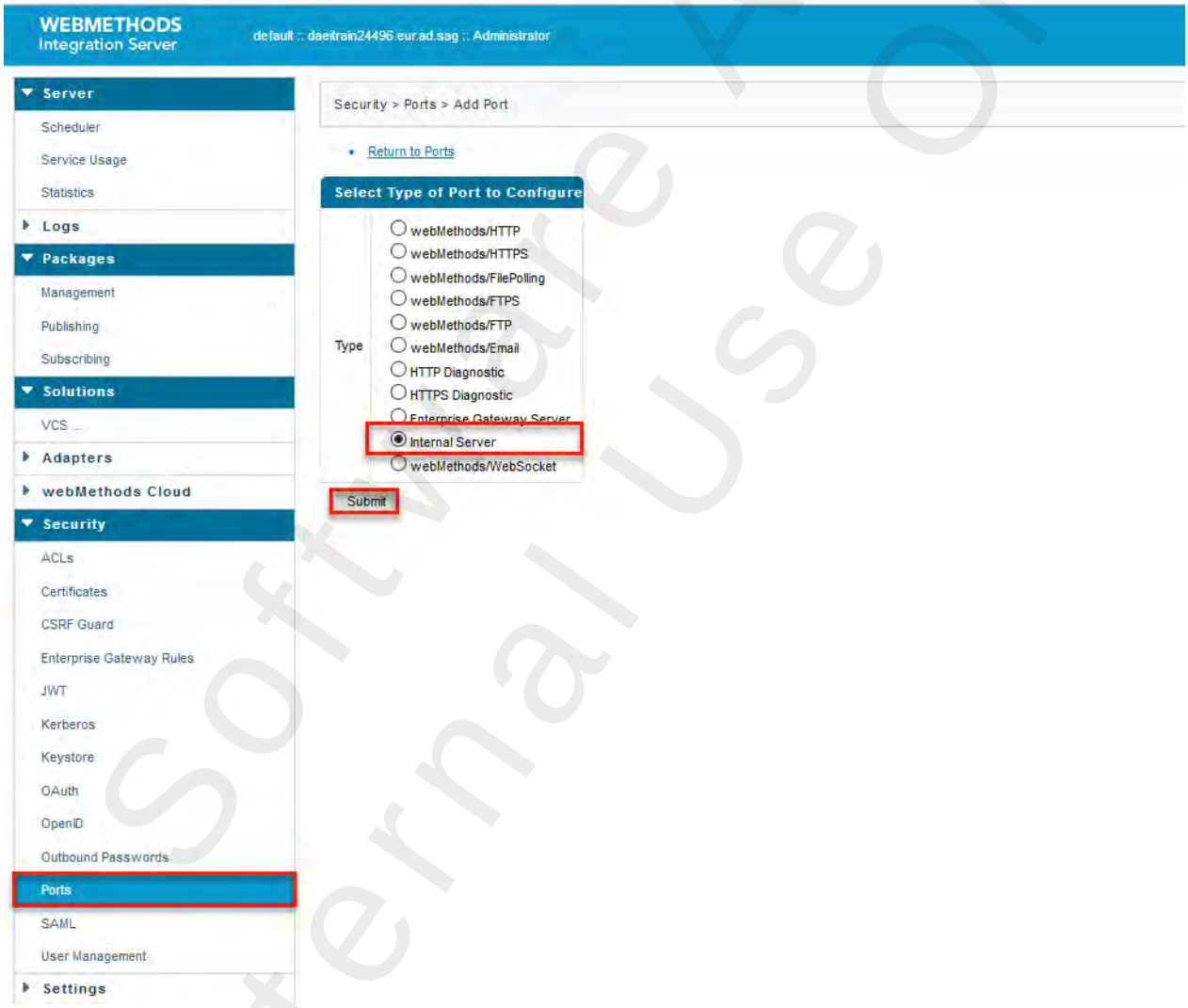

Click **Yes** to confirm to enable the port.




7. Within Firefox open the **IS Internal Administrator** for the installation of the **internal IntegrationServer** where the **Echo API** is running which we have already added **to API Gateway**.
   a. We need to configure the connection to **API Gateway**. Navigate to **Security > Ports > Add Port**.




b. Select **Internal Server** and click **Submit**.

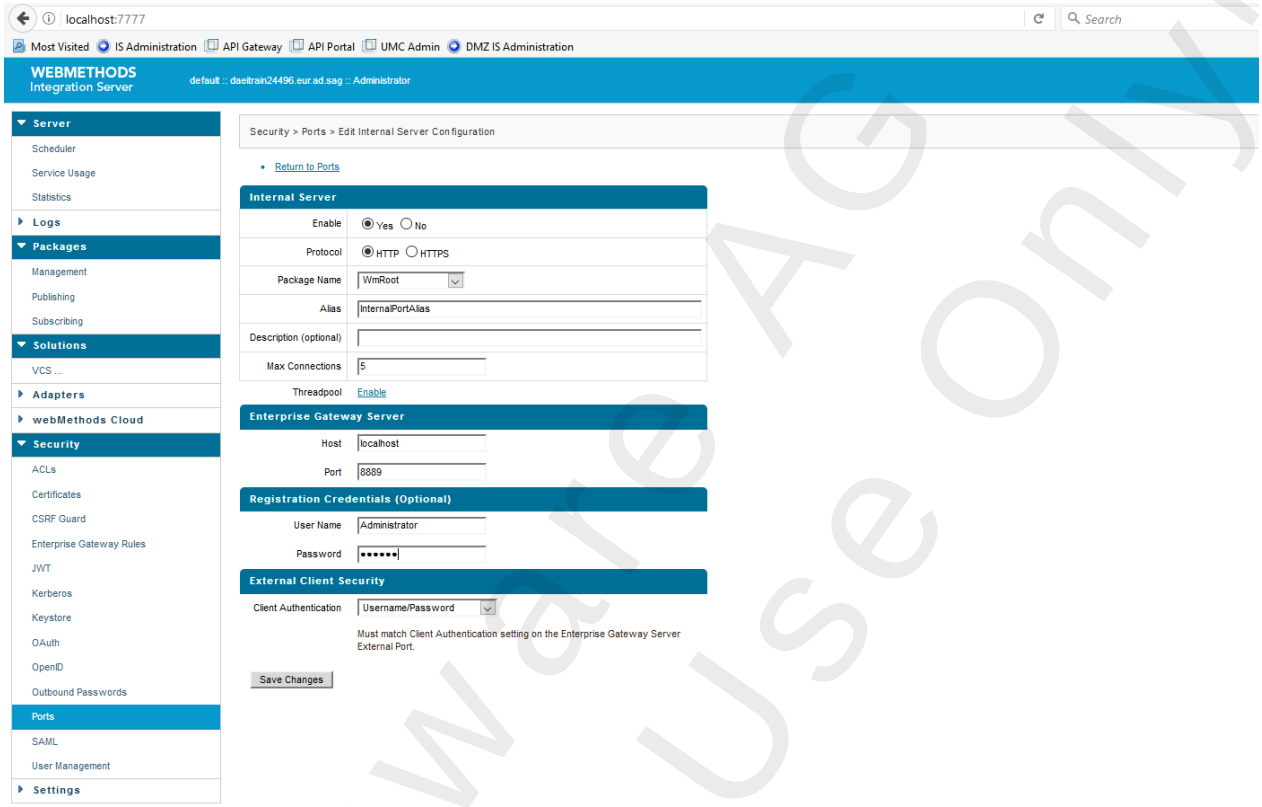



c. and configure the following properties:
   i. **Enable**: Yes
   ii. **Protocol**: HTTP
   iii. **Package Name:** WMRoot
   iv. **Alias**: InternalPortAlias
   **v. Description:**
   vi. **Max Connections**: 5
   vii. **Threadpool**: Enable

viii. **Enterprise Gateway Server**:
   1. **Host:** localhost
   2. **Port:** 8889

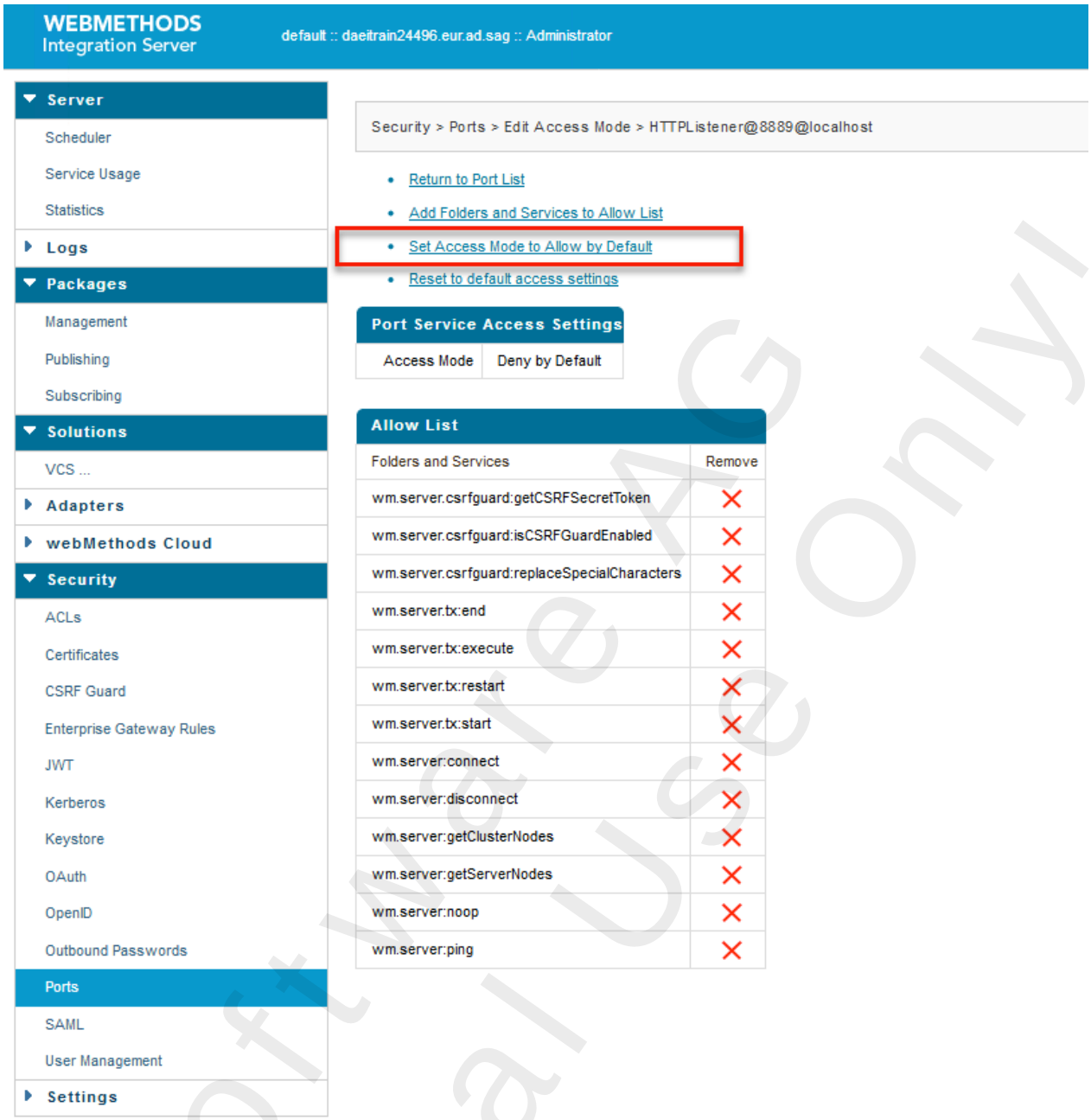
**ix. Registration Credentials (Optional)**
   1. **Name**: Administrator
   2. **Password**: manage

x. **External Client Security**:
   1. **Client Authentication**: Username/Password

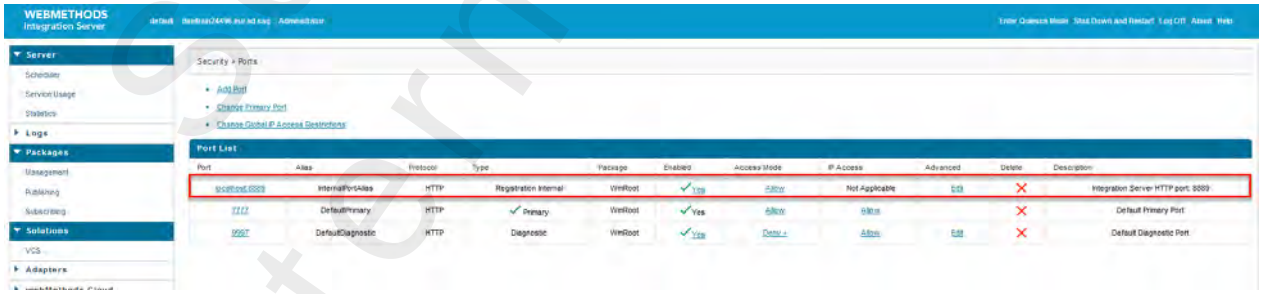



Click **Save Changes**.

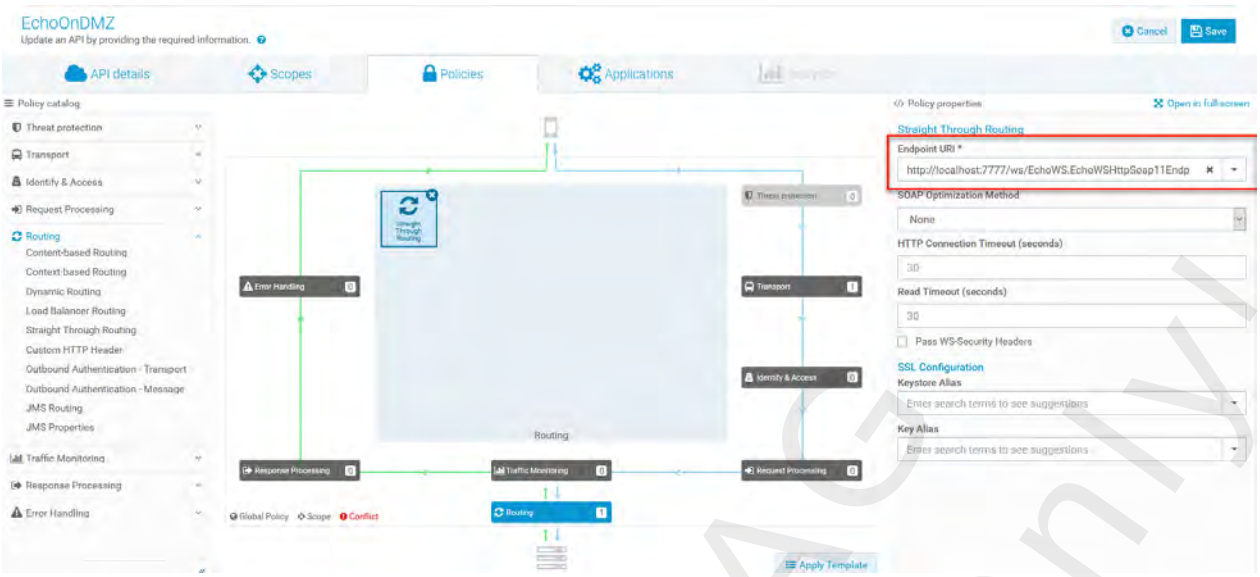
d. Change the **Access Mode** of the internal port. Select the new created port and click on **Deny+**




Click to **Set Access Mode to Allow by Default.** Return to port list.



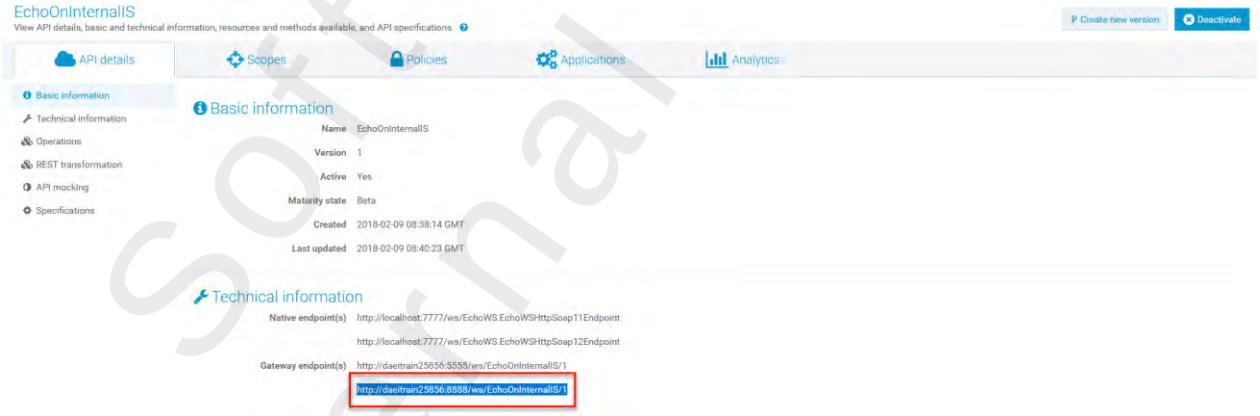
8. In order to use the **Reverse Invoke** functionality we have to modify the routing endpoint of the API we have created in one of the last steps. Logon to **API Gateway** as user **Administrator/manage**. Open the API **EchoOnInternalIS** in **Edit** mode. Navigate to **Policies > Straight Through Routing Endpoint URI**.



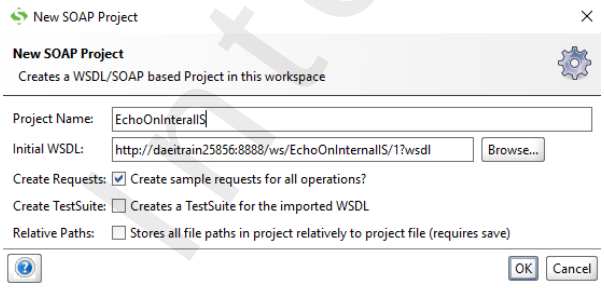

a. Modify the first part of the routing endpoint from **http://localhost:7777** to **apigateway://RegPortAlias**


Click **Save** and **Activate.**

9. Open **SOAP UI** and create a new **SOAP UI** project using the endpoint URL pointing to the external port 8888 of the Echo API as shown in **API Gateway**.

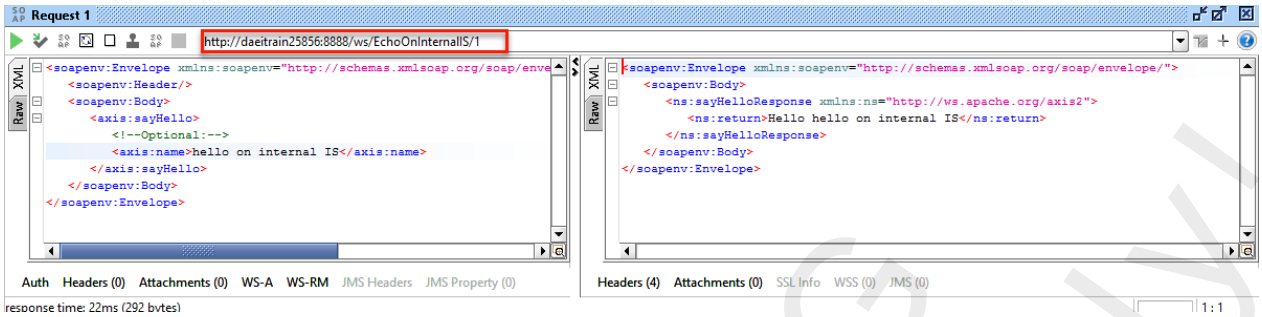



Provide **EchoOnInternalIS** as **Project Name**.

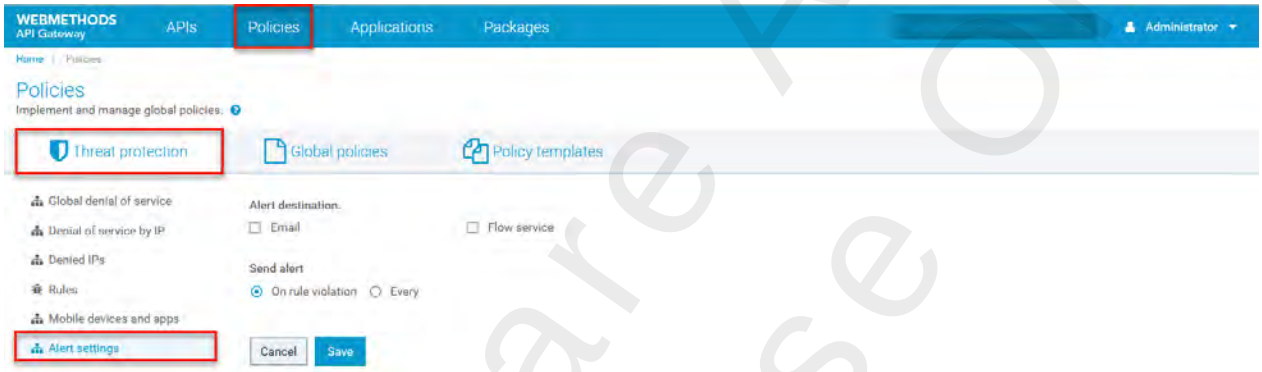



10. Within the new created **EchoOnInternalIS** project select one of the listed requests. Change the port-number in the endpoint from **5555** to **8888**.

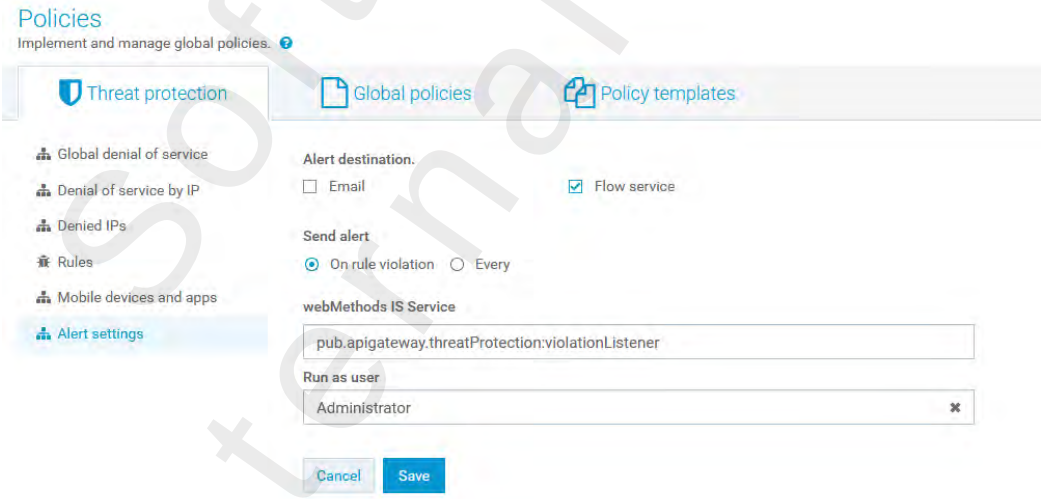

11. Provide some text for the name element. And run the request.




12. We want to use the option that whenever a rule is violated in **API Gateway**, an alert will be generated by **API Gateway**. We want to configure the alert setting on the global level. Within **API Gateway** navigate to **Policies > Threat Protection > Alert Settings.**



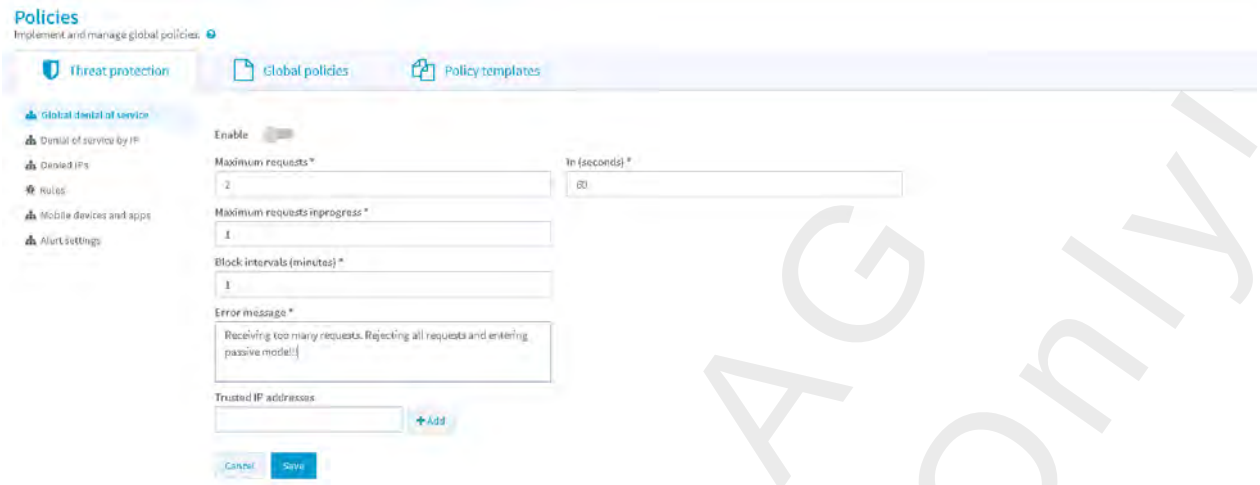

13. Provide the following properties.
    a. **Alert Destination**: Flow Service
    b. **Send alert:** On rule violation
    c. **webMethods Is Serivce**: pub.apigateway.threatProtection:violationListener
    d. **Run as user**: Administrator




Click **Save.**

14. Now we create a global denial of service policy. Navigate to **Policies > Threat protection > Global denial of service**. Provide the following properties:
    a. **Maximum requests**: 2
    b. **In (seconds):** 60
    c. **Maximum requested inprogress**: 1

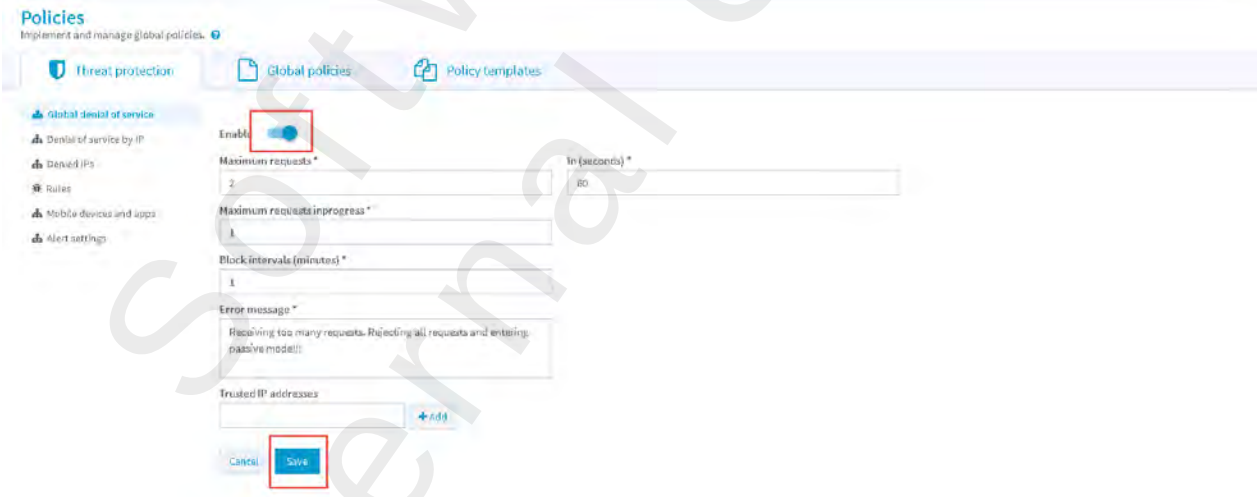
d. **Block interval (minutes):** 1
e. **Error Message:** Receiving too many requests. Rejecting all requests and entering passive mode!!!
f. **Trusted IP addresses**:



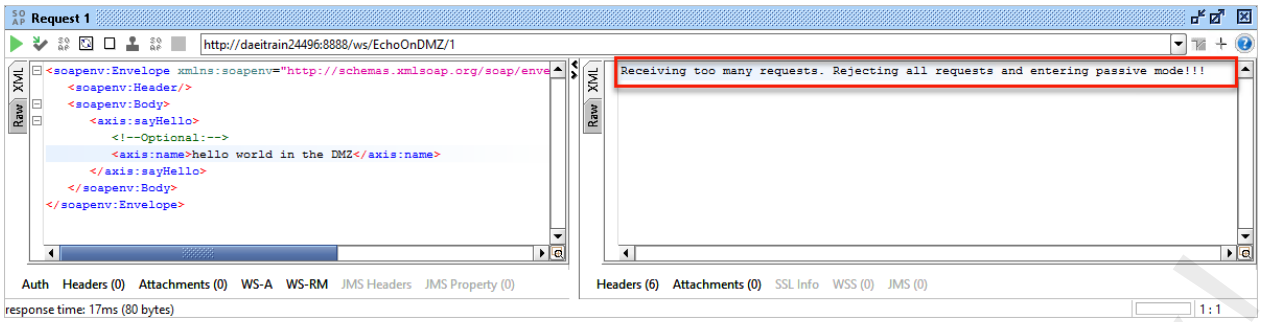

Click **Save.**

15. Enable the Policy

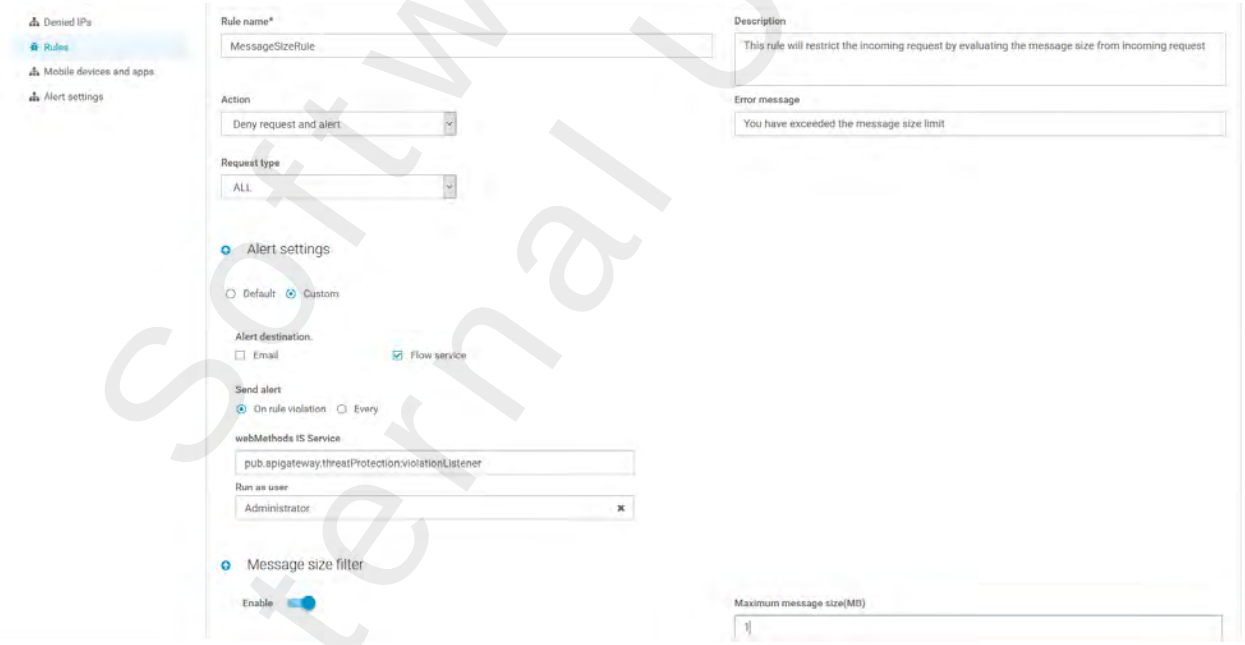

Click **Save.**




16. Go back to project **EchoOnInternalIS** in **SOAP UI**. Rerun the already configured request several times. After 2 times the request will respond the error message as defined in the global policy



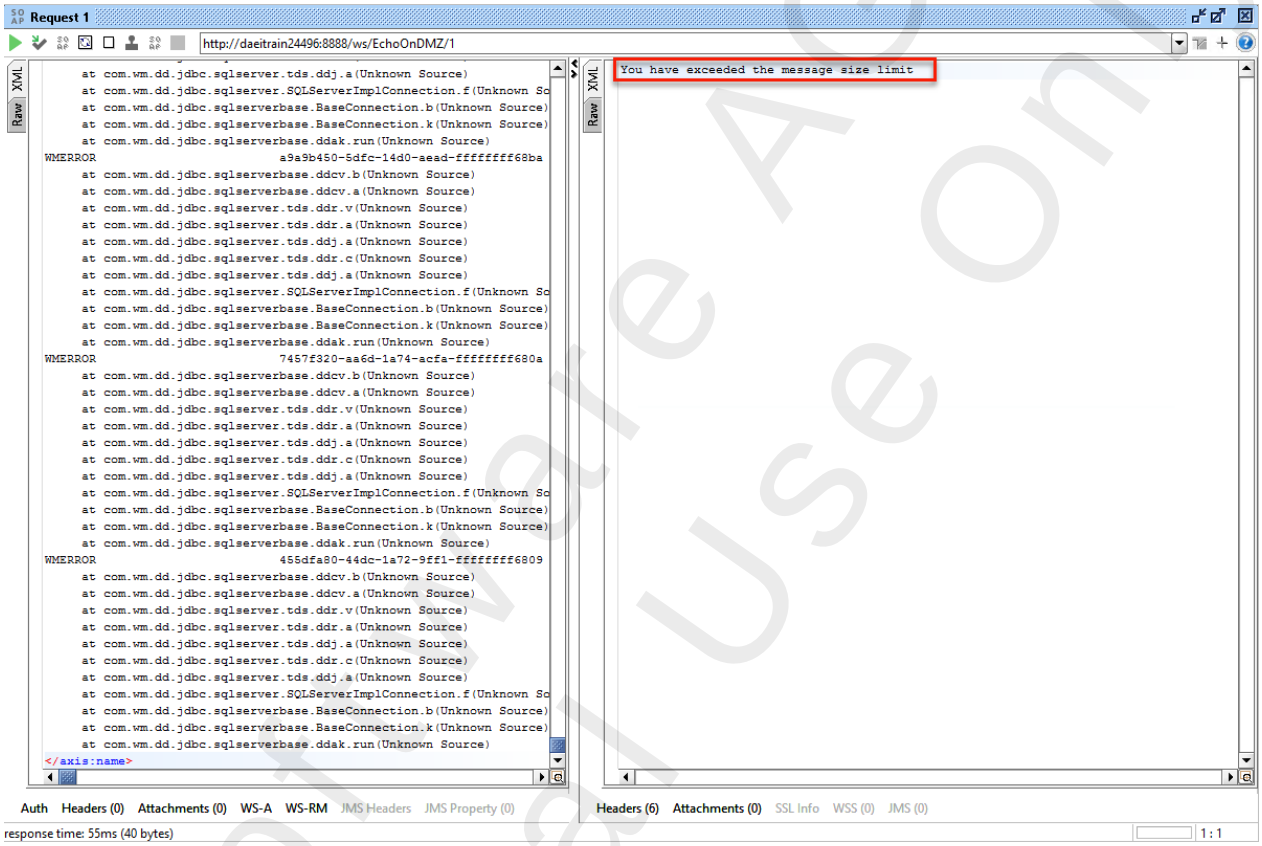

17. Now we want to use a rule to reject all incoming requests based on size of the request. Open **API Gateway** and navigate to **Policies**. Disable the global policy we have defined before.
18. Select Rules and click **Add rule** to create a new rule. Provide the following properties:
    a. **Rule name:** MessageSizeRule
    b. **Description**: This rule will restrict the incoming request by evaluating the message size from incoming request
    c. **Action**: Deny request and alert
    d. **Error message:** You have exceeded the message size limit.
    e. **Request type:** all
    f. **Alert Settings**
        i. **Custom**:
        ii. **Alert destination**: Flow service
        iii. **Send Alert**
            1. **On rule violation**
            2. **webMethods IS Service**: pub.apigateway.threatProtection:violationListener
            3. **Run as user**: Administrator
    g. **Message size filter**
        i. **Maximum message size (MB):** 1




Click **Save.**

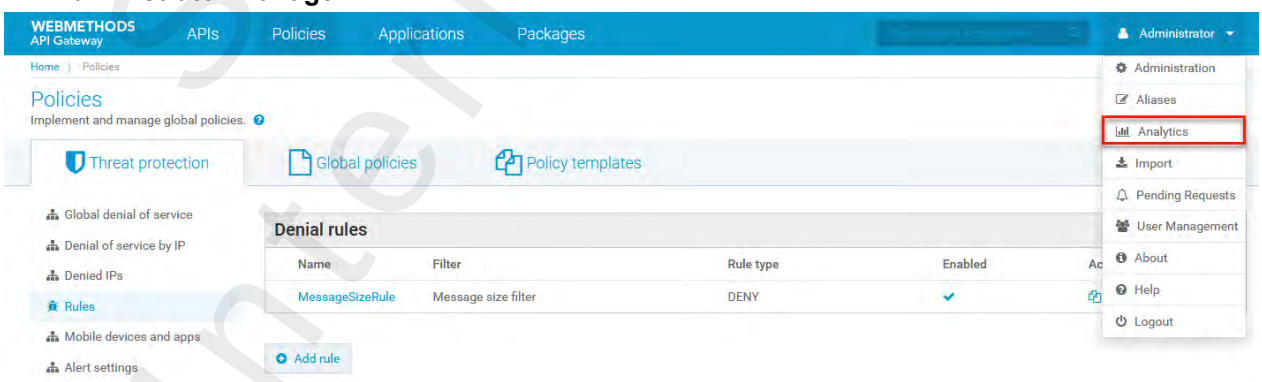
19. Enable the rule.


20. Open the file **OneMB.txt** in folder **C:\Training\APIGateway\Exercise10** in Notepad+.
21. Open **SOAP UI** and copy and paste the content of the **OneMB.txt** file as value to the **name** element.
22. Run the request, make sure that your endpoint points to port 8888. The response will be the error message you have just created.



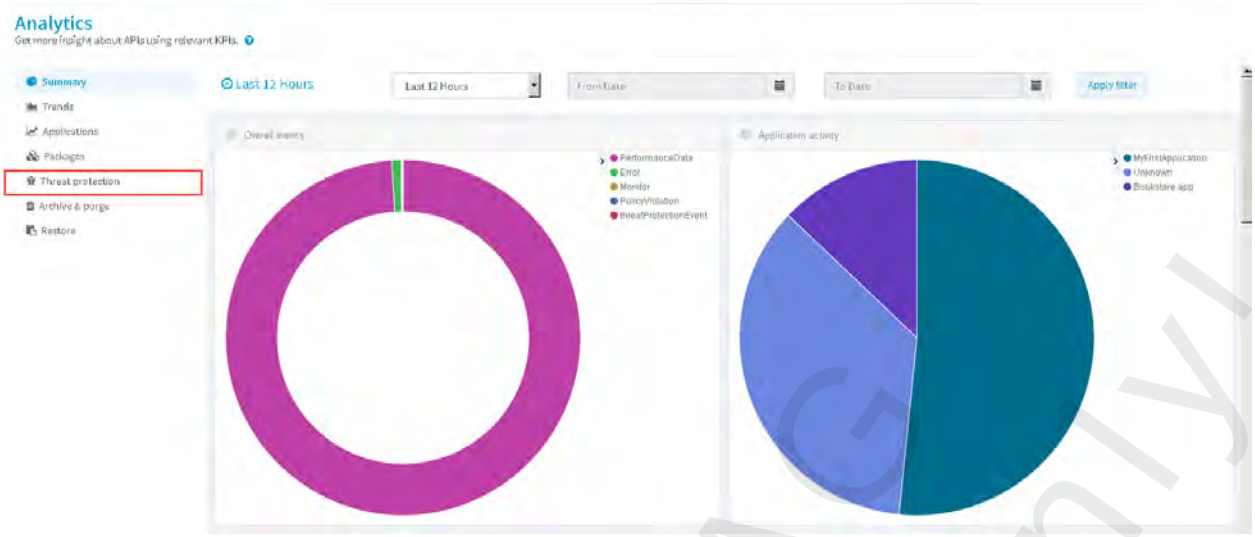

Run this request several time to produce some threat protection violations.

23. We want to review Dashboard information Threat Protection. Go back to **API Gateway** as user **Administrator/manage**.



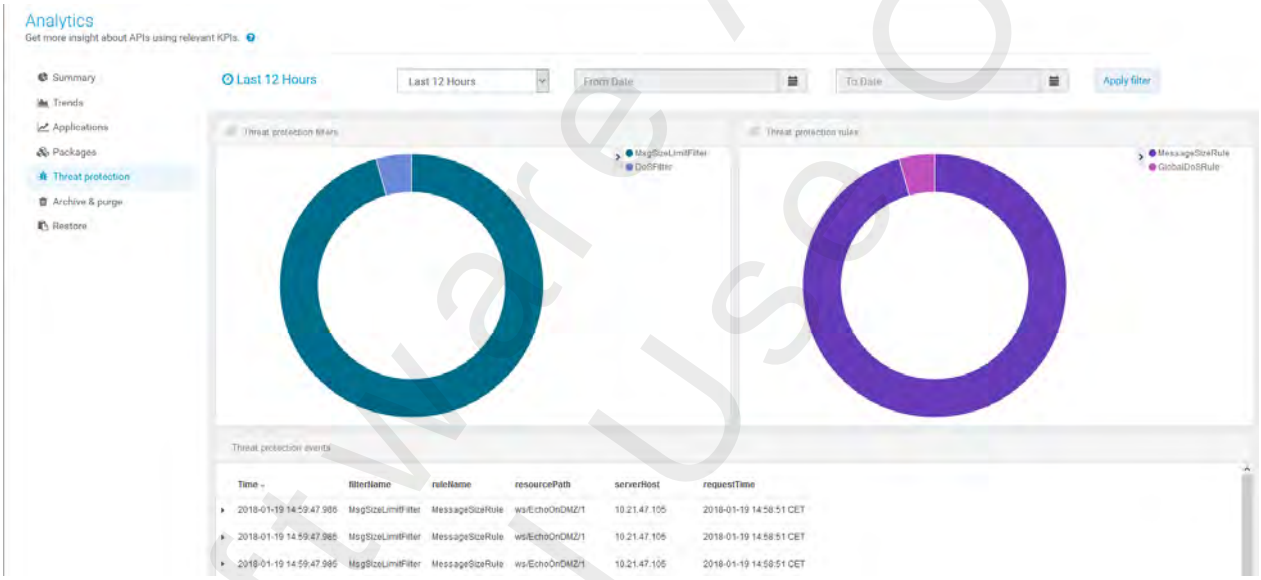

Open the User menu and select **Analytics**.

24. The **Summary** dashboards show **Overall events**, ….

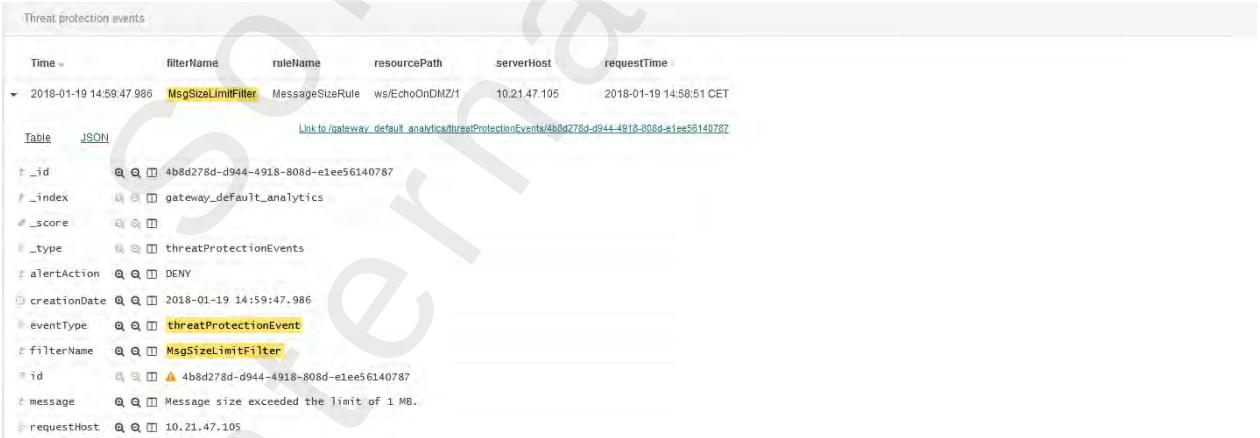



Within the left hand menu select **Threat Protection**.

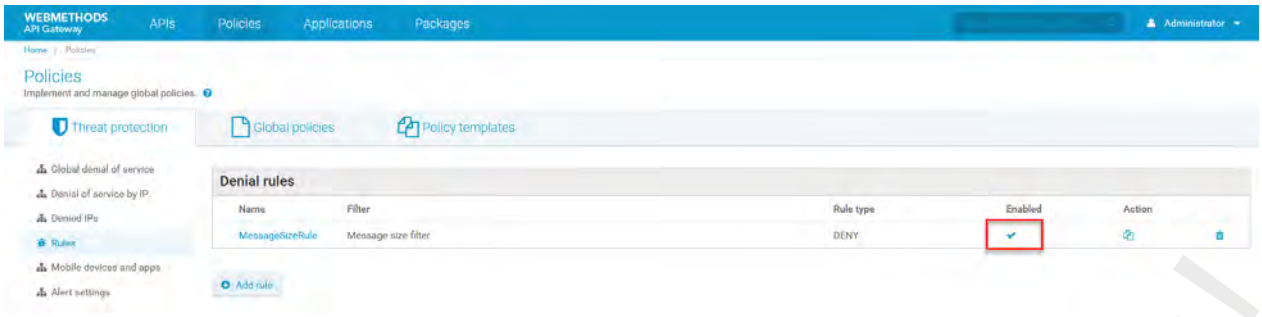
25. Review Threat protection filters and Threat protection rules.



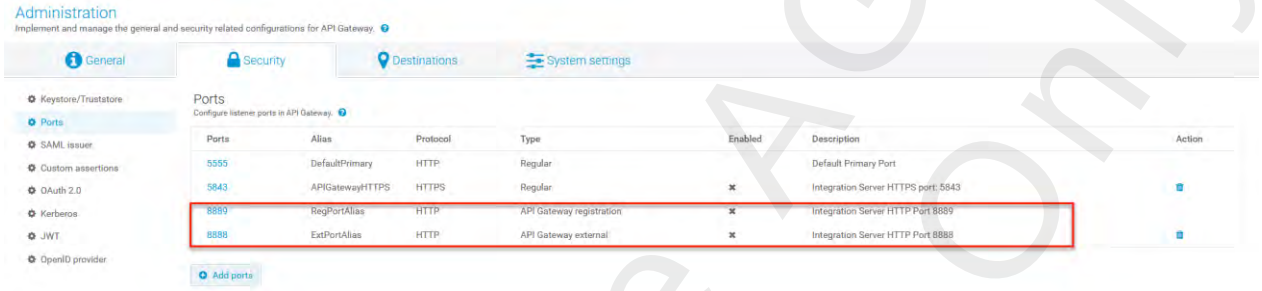

26. Click the triangle next to a threat protection event and review details.




27. Disable the Threat protection rule. Navigate to **Policies > Threat protection > Rules**. Within the list of rules disable the rule **MessageSizeRule** click the sign within the **Enable** column.




28. Disable the API Gateway registration port and the **API Gateway external** port. Navigate to **Administration > Security > Ports**. Within the list of ports disable these ports. Click the sign within the **Enable** column for each of the ports.



This page intentionally left blank

# Exercise 11:
# Manage Fine Granular Exposure of APIs

## Objectives

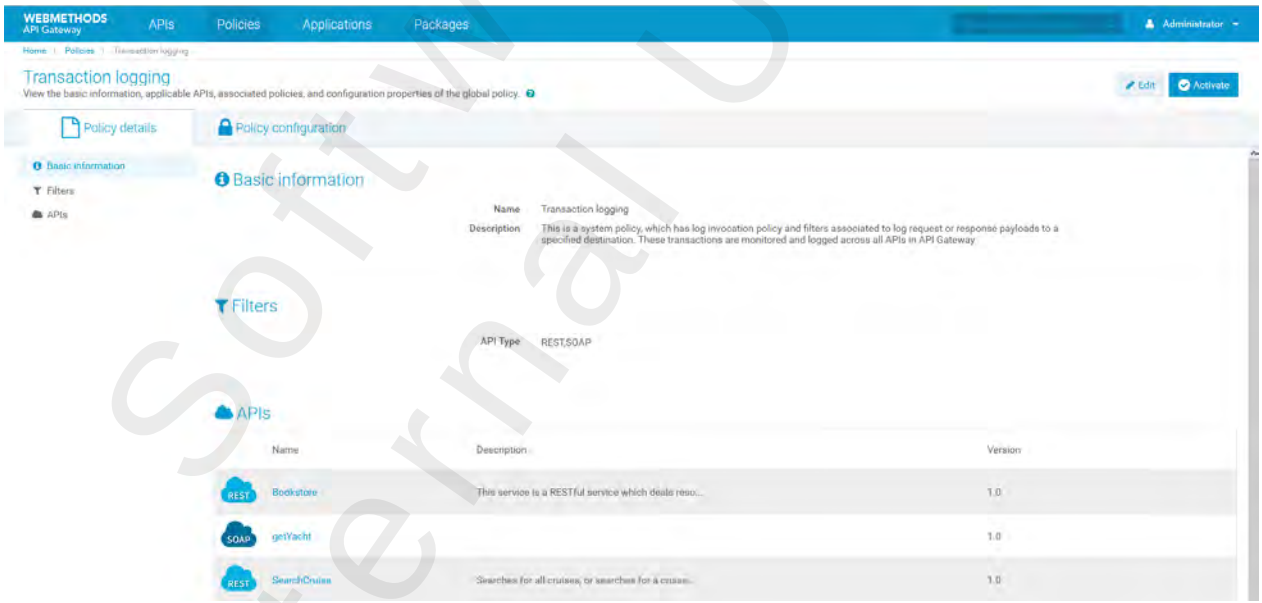
Define an APIs SAGTours application. An API for signing-up to SAGTours is available as well as an API for booking of a cruise. Before a cruise can be booked, the customer must sign-up to the SAGTours. The operations GET, PUT and DELETE of the Sign-up API are protected via Basic Authentication. The email you provide as user-email is the username you need for BASIC authentication. This fine granular security enforcement will be implemented using scope level policy actions.

All operations in the Bookings API are protected with Basic Authentication.

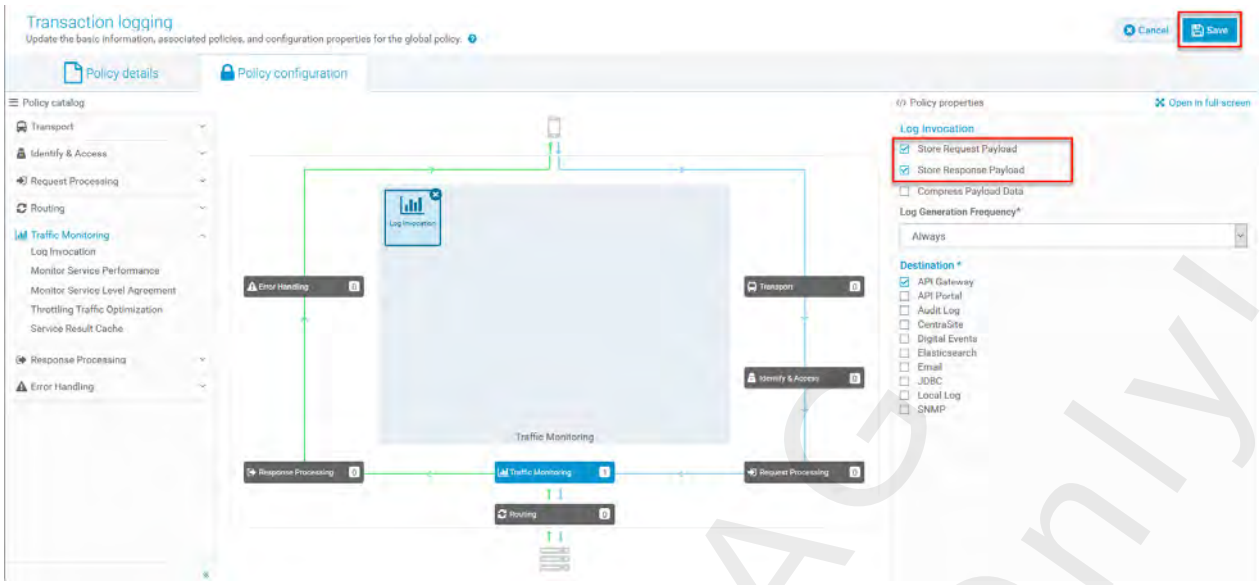
In addition to the enforced security layer for a set of operation we will enforce a general logging policy.

## Steps

1. Open Windows Services UI to double check that the following services, needed for API Gateway are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Event Data store 10.1 – should be started together with IS**
   c. **Software AG Runtime 10.1**
2. Login to the **API Gateway UI** as user **Administrator/manage.** We want to define a global policy for Logging. Naviagte to **Policies > Global policies**. A default **Transaction logging** policy is already predefined with the installation of **API Gateway.** Select the policy to open the detial view of this policy. Because it is a global policy all APIs you have created until now are listed within the **APIs** section.

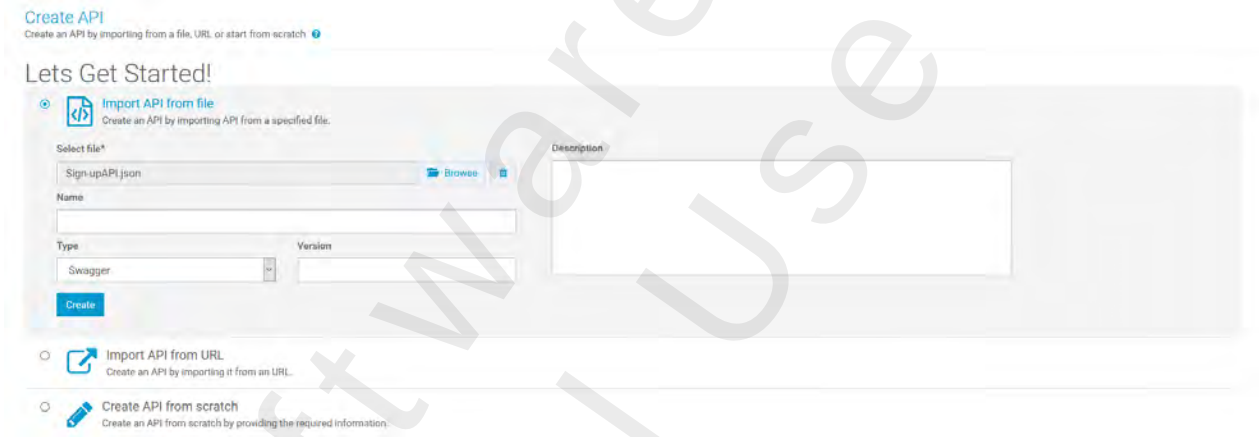



3. Navigate to **Policy configuration** and switch over to **Edit** mode. For the policy action **Log invocation** provide the following properties:
   a. Store Request Payload: - select -
   b. Store Respone Payload: - select –

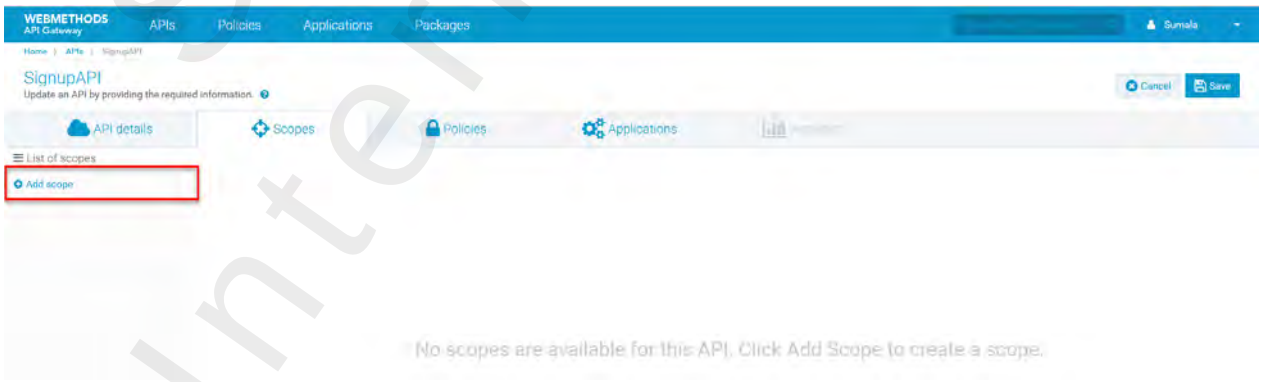



Click **Save**. Click **Activate**.

4. Login to the **API Gateway UI** as user **Sumala/manage.** Navigate to **APIs** and create a new S**ignUp API** using the import functionality. The swagger file **Sign-upAPI.json** is available in the folder **C:\Training\456-71E\Exercise11.**




5. Reviewing the **Resource and methods** section you find the resource **/customer** with operation **POST** to add a new customer. The result of this operation will provide a customerID in the response body. The 2nd resource is **/customer/{customerID}** to access a specific customer account. These operations GET, PUT and DELETE enforce **HTTP Basic Authentication**. In order to define a corresponding security action for this combination of resource and operations we will define a scope. Navigate to **Scopes** and switch on to **Edit** mode**.**




Click **Add scope.**

6. Within the **Scopes** page provide the following properties:

   a. **Name**: Customer Specific Data

b. **Description**: Accessing customer data in the SAGTours application, like GET, PUT, DELETE

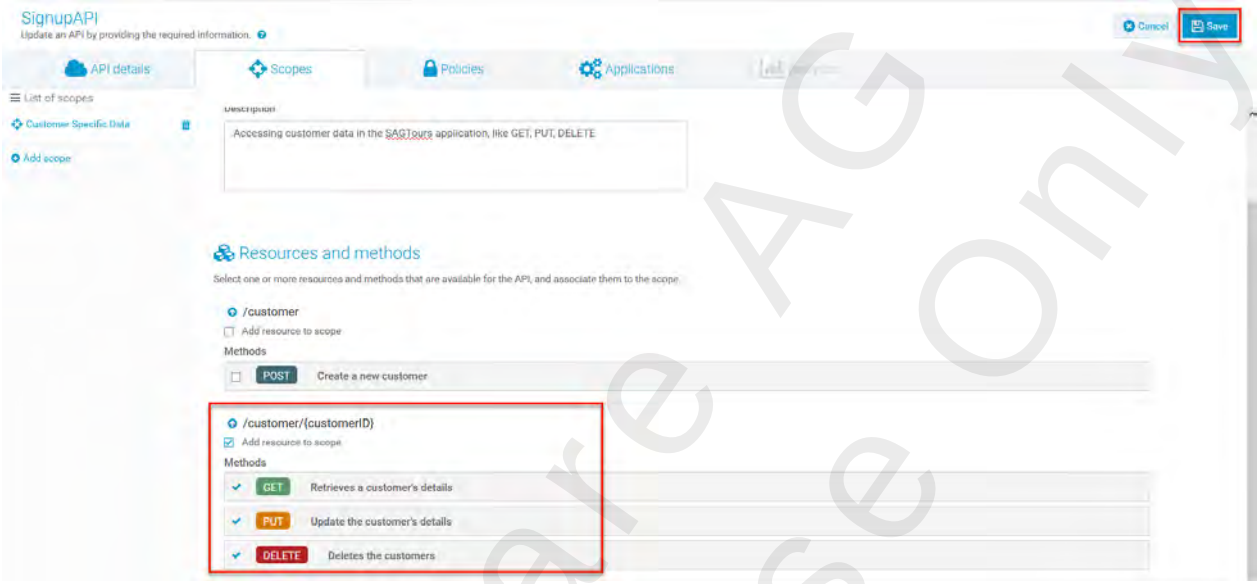
c. **Resources and methods**

i. **/customer**

1. **Add resource to scope**: - leave unselected -

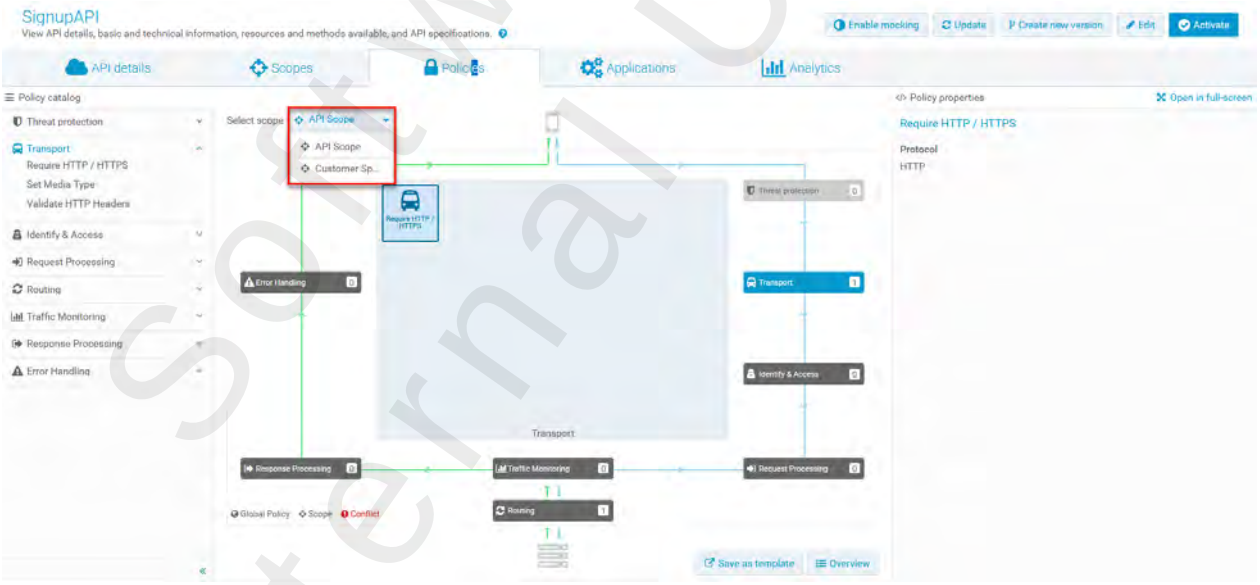
ii. **/customer/{customerID}:**

1. **Add resource to scope**: - select –
2. **Methods**: - leave methods GET, PUT, DELETE selected –




Click **Save.**

7. Navigate to **Policies.** Click Select **Scope > API Scope.**

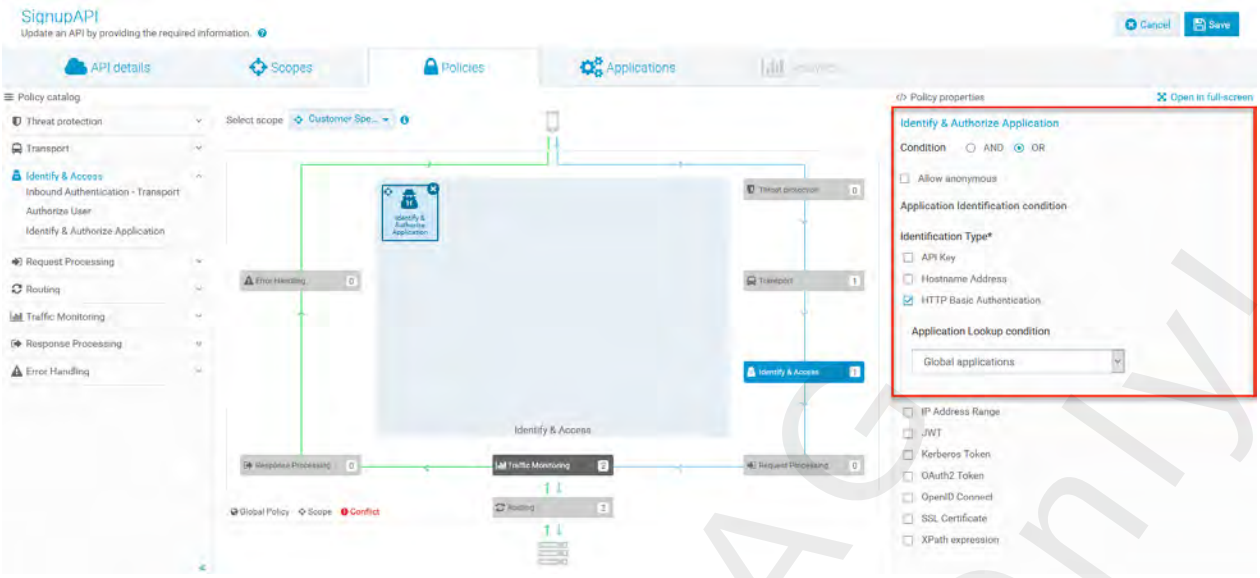



From the list select the just created scope **Customer Specific Data.**

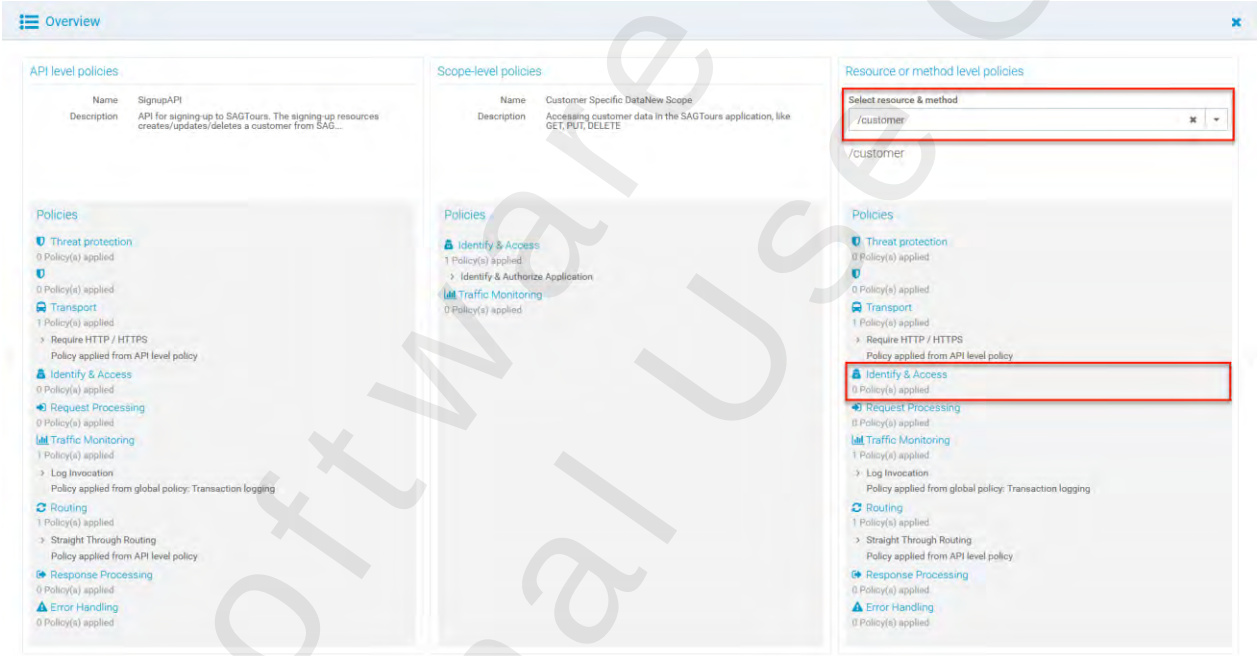
8. Switch to **Edit** mode. Add the polciy action **Identify & Access > Identify & Authorize Application.** Provide the following properties

i. **HTTP Basic Authentication**

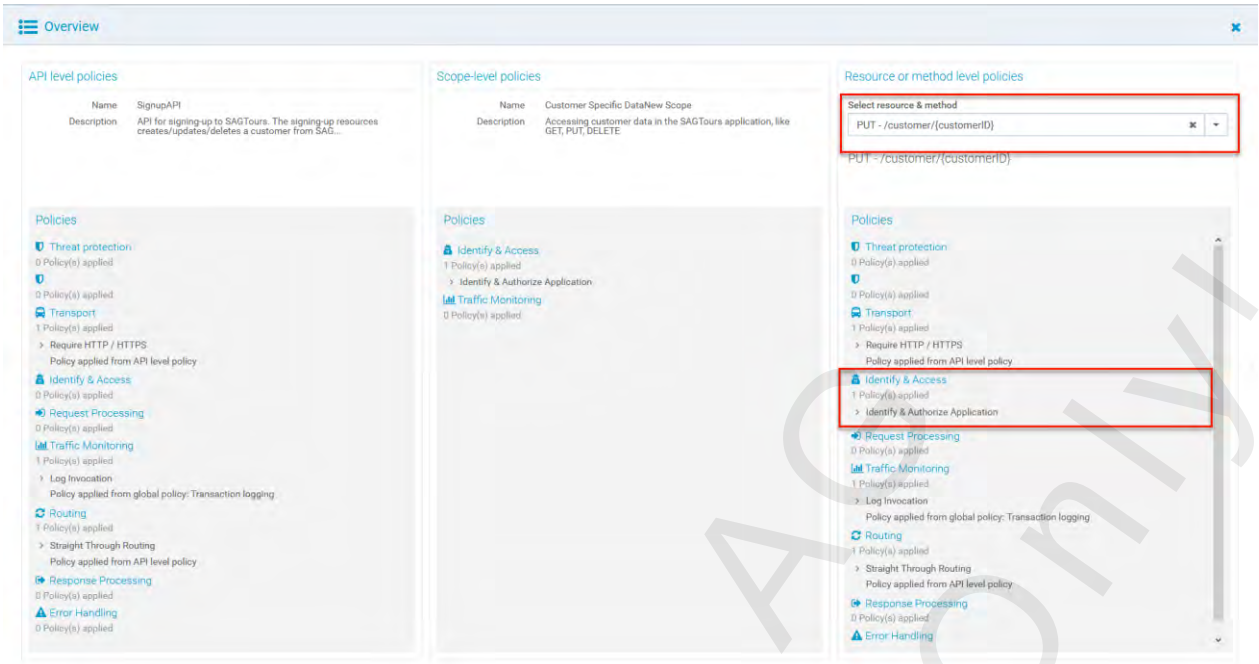
ii. **Application Lookup Condition:** Global applications




9. Click **Overview** on the bootom of the page to see the policy definitions structured by API level, Scope-level and Resource or method level.




Within the section **Select resource or method** scroll down to the resource & method **PUT - /customer/{customerID}**




Close the **Overview** page.

10. Activate the **SignupAPI.**
11. Import the **BookingsAPI** using the swagger file **BookingsAPI.json** available in the folder **C:\Training\456-71E\Exercise11.** Navigate to **Policies** and switch to **Edit** mode. Add the the policy **Identify & Access > Identify & Authorize Application.** Set the property **Identification Type** to to **HTTP Basic Authentication** with **Lookup Condition** set to **Global applications** as you have done for the **SignupAPI**. **Save** the API.
12. Open **Firefox**. Open a **Private Window**. Open **RESTClient** and create a new account in the SAGTours application. Use the API **SignupAPI** selecting the resource **/customer** and the operation **POST**.You will find a corresponding request body in the file **SignUp API**.txt in folder **C:\Training\456-71E\Exercise11.** You need to provide the following

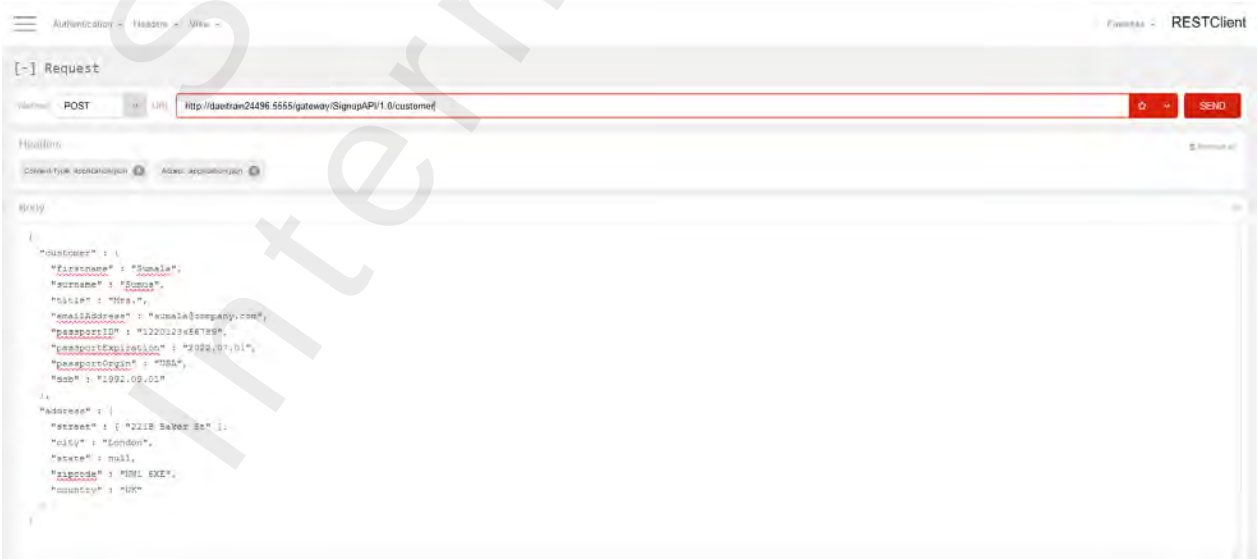
    a. **Headers**:

        i. **Accept:** application/json

        ii. **Content-Type**: application/json

    b. **Body**

        i. Copy and paste corresponding request body for method **GET** out of the file **SignUp API.txt** in folder **C:\Training\456-71E\Exercise11.**



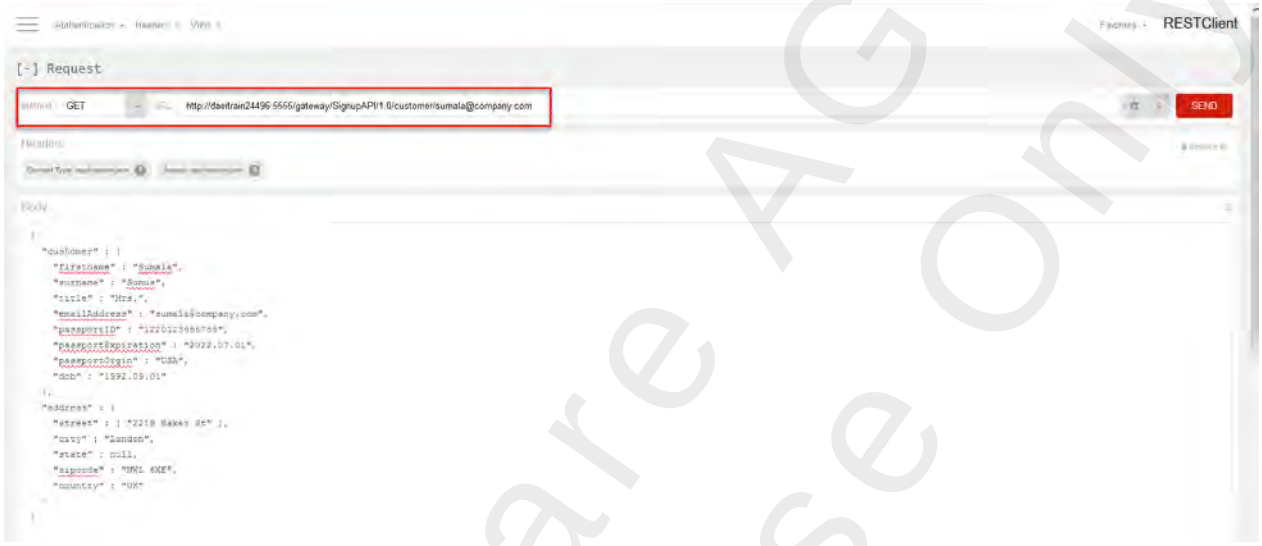
Click **Send**.

13. A new account with **customerID** set to **sumala@company.com** is created in the SAGTours application. To verify this we would llike to review the customer detail using the **GET** operation on the resource **/customer/{customerID}** within the API **SignupAPI.** Open **Firefox**. Open a **Private Window**. Open **RESTClient** and provide the following porperties

    a. URL: - copy and paste the Gateway endpoint. Append the the following -
    /customer/sumala@companay.com

    b. **Headers**:

        i. **Accept:** application/json



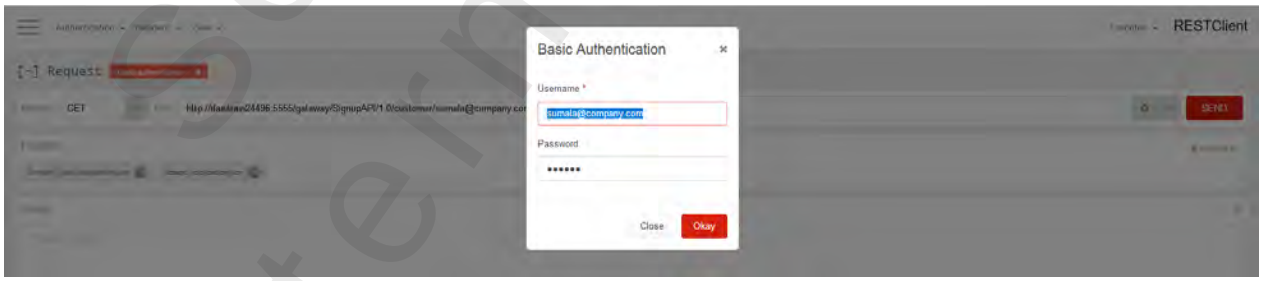

Click **Send**.

This leads to an error, because Basic Authentication is missing.


14. Provide **Basic Authentication** in RESTClient:

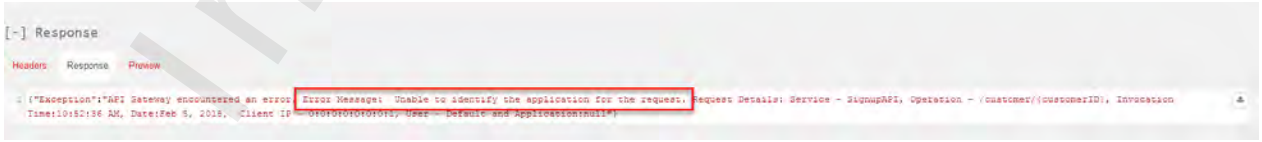
    a. **Username**: sumala@company.com

    b. **Password:** manage




Click **Send**.

This leads to an error, because API Gateway is unable to identify the user in any of the applications.

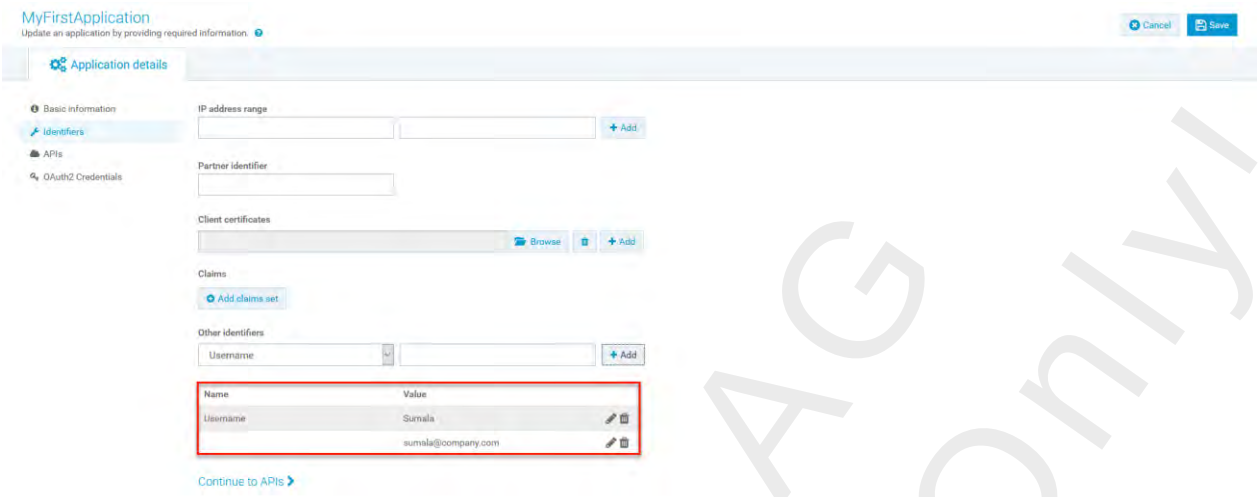



15. Within **API Gateway** navigate to **Applications**. Open the existing application **MyFirstApplication.** Switch to **Edit** mode and add another identifier. Provide the following propeties:

a. Identifier: Username

b. Token: sumala@company.com

Click **Add**.




Click **Save**.

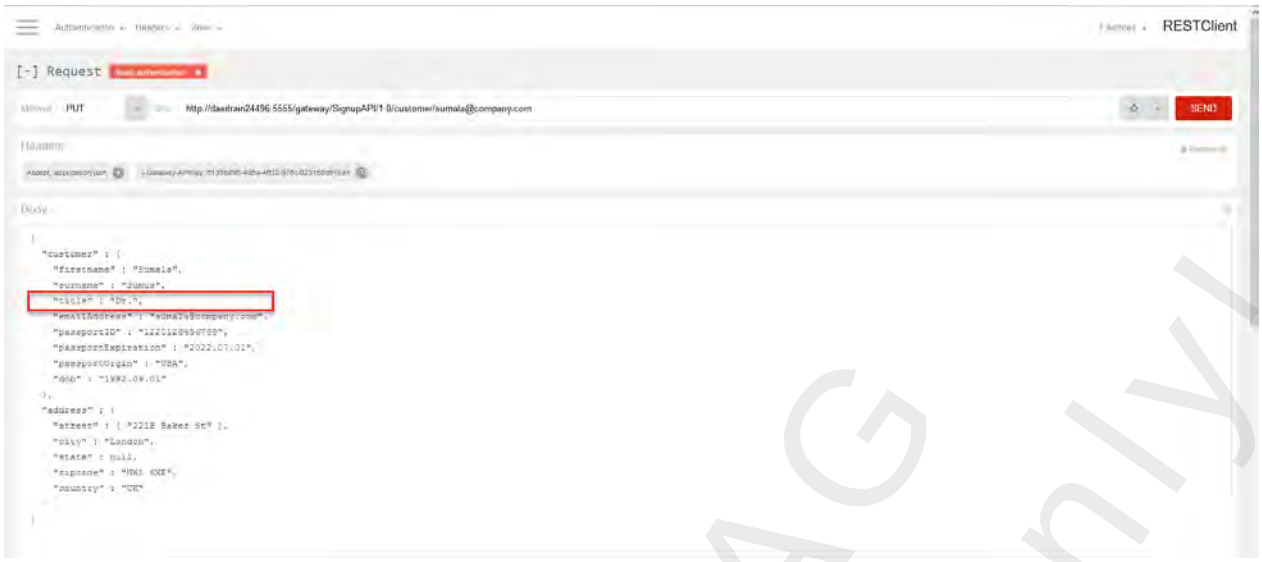
16. Within RESTClient run the request again. This time the request runs successfully.


17. Now we want to change the details of the customer sumala@company.com. Use the same seting as in step 16 and 17. Change the following properties:

a. **Method**: PUT

b. **Body:**

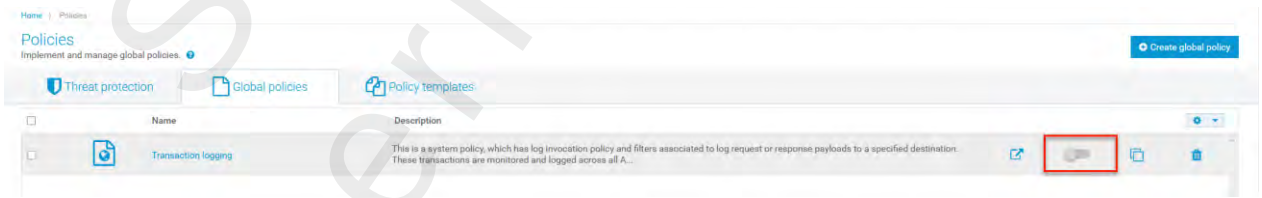
i. Copy and paste corresponding request body for method **PUT** out of the file **SignUp API.txt** in folder **C:\Training\456-71E\Exercise11.** This will change the attribute **title** from **Mrs.** To **Dr.**..




Click **Send**.

18. To verify the changes change the **Method** back to **GET**. Click **Send**.


19. Login as user Administrator/manage and ~~disbale~~ disable the Global Policy **Transaction logging**.



# Exercise 12:
# Monitoring the Virtual API

## Objectives

Scenario:

Create and use new Policy Action: Throttling Traffic Optimization and Logging

## Steps

1. Open Windows Services UI to double check that the following services, needed for API Gateway, are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Event Data store – should be started together with IS 10.1**
   c. **Software AG Runtime 10.1**
2. Logon to **API Gateway UI** as user **Sumala/manage**. Navigate to the **APIs**. Select the API **getYacht.** Deactivate the API and switch do Edit mode.
3. Navigate to **Policies > Traffic Monitoring.**
4. Switch to **Edit** mode. Within the section **Traffic Monitoring** select the action **Throttling Traffic Optimization**. A dialog box will ask you whether you want to add the depending policy action **Identify & Authorize Application**. Click **Yes**.


5. Provide the following parameters for the selected policy action. Use the button **Open in full-screen** in order to switch to full screen mode.
   a. **Limit Configuration**
      i. **Total Request**: - selected -
      ii. **Greater Than:** - selected -
      iii. **Value:** 2
   b. **Destination**:
      i. **API Gateway**: - select -

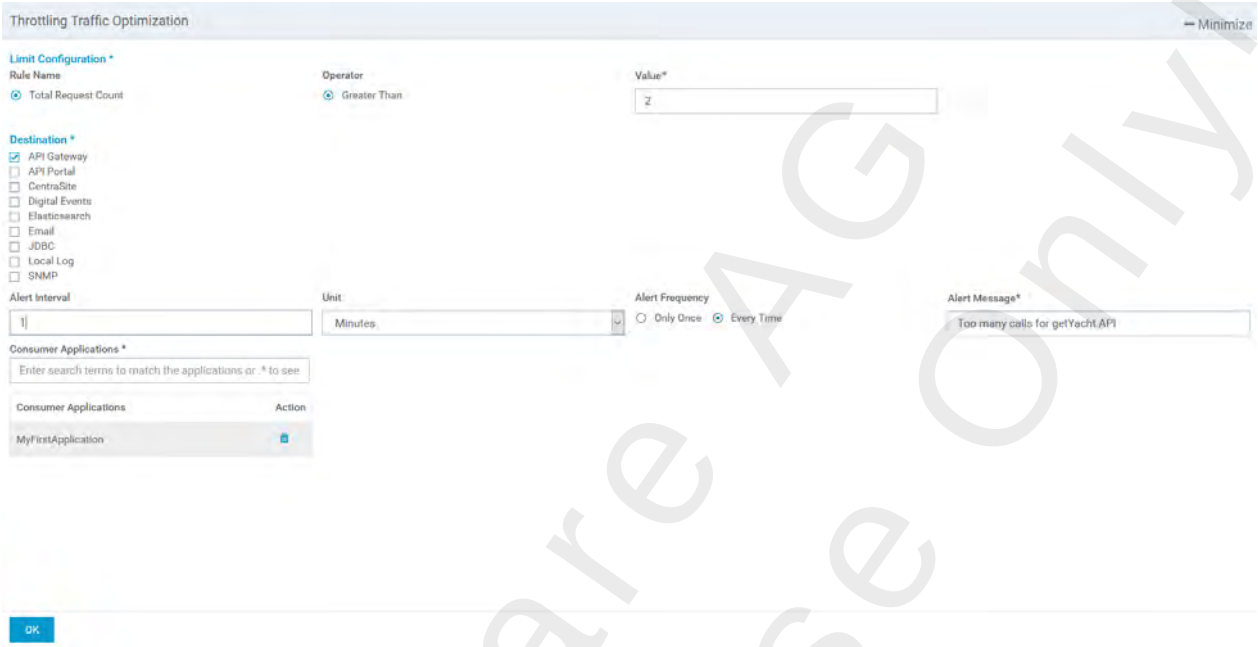
ii. **Alert Interval**: 1

iii. **Unit**: Minutes

iv. **Alert Frequency**: Every Time

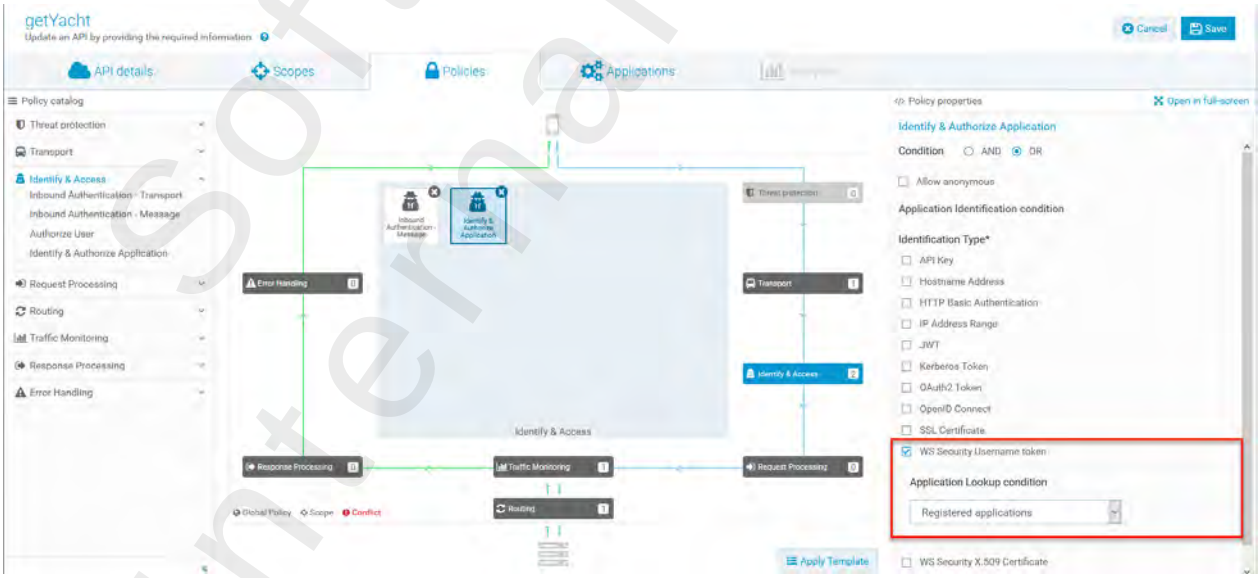
v. **Alert Message**: Too many calls for getYacht API

vi. **Consumer Applications**: MyFirstApplication

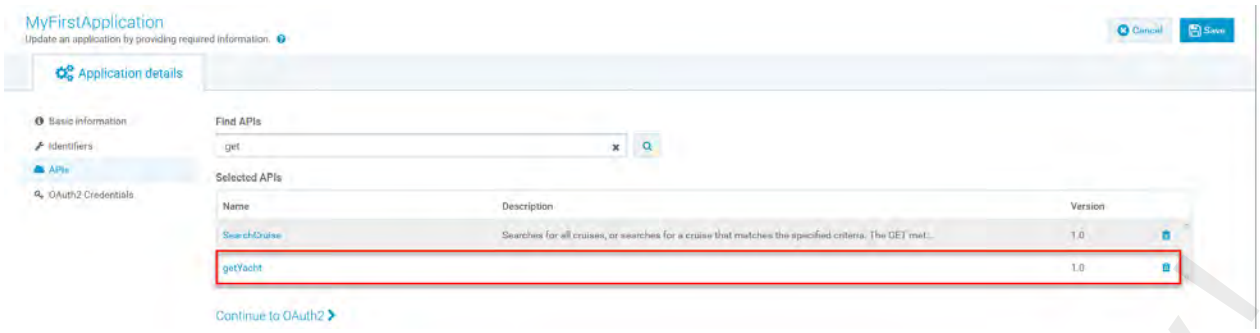



Click **OK**.

6. Navigate to **Policies > Identify & Access > Identify & Authorize Application**. Provide the following properties:
   a. **Application Identification condition**
      i. **Identification Type:** WS Security Username token
      ii. **Application Lookup condition:** Registered applications

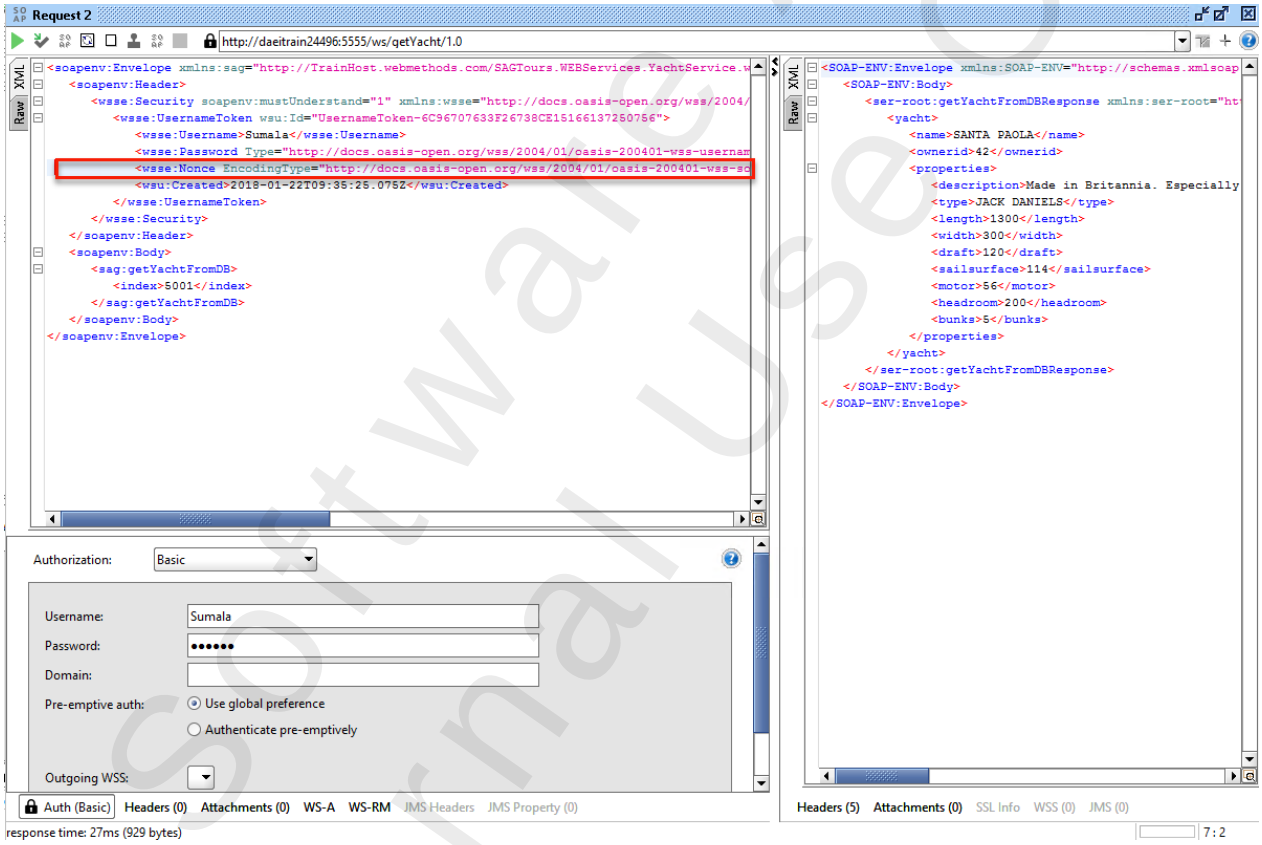



7. Click **Save** and **Activate.**
8. We need to add the **getYacht** API to the application **MyFirstApplication**. Navigate to **Applications**. From the list of applications select **MyFirstApplication.** Switch to **Edit** mode. Navigate to **Application Details > APIs**. Add the API **getYacht.**

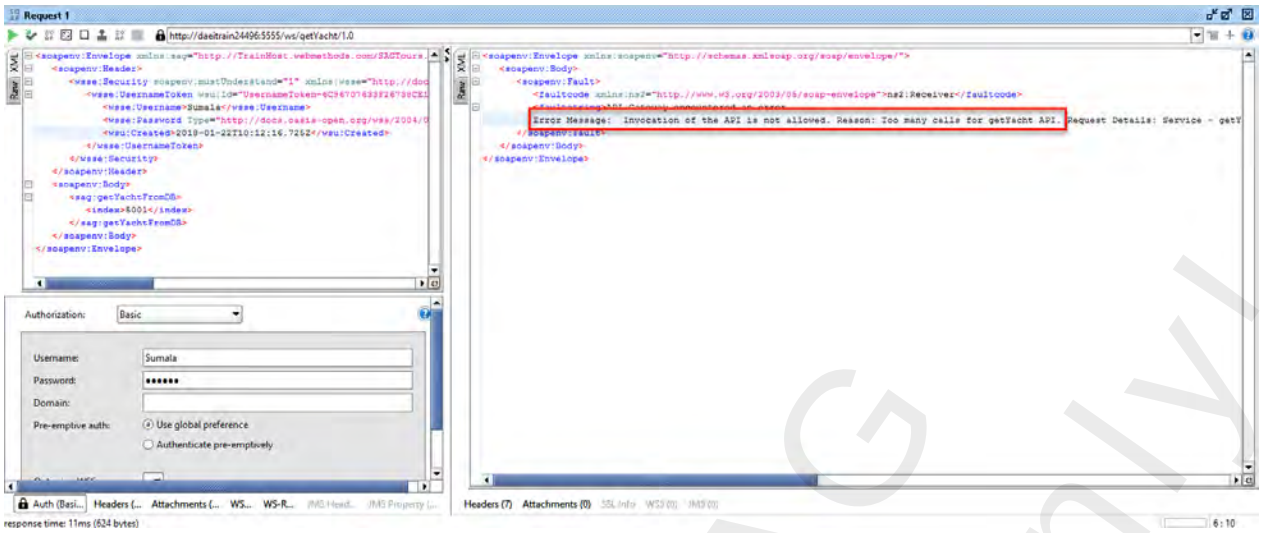



Click **Save.**

9. Open **SOAP UPI** and open the existing project **getYacht** which you have created in exercise 7 or create a new SOAP project. Provide **Basic Authentication** with
   a. **Username**: Sumala
   b. **Password**: manage
   Add **WSS Username token**.
10. Run the request.
11. Within the request body delete the line beginning with **<wsse:Nonce**

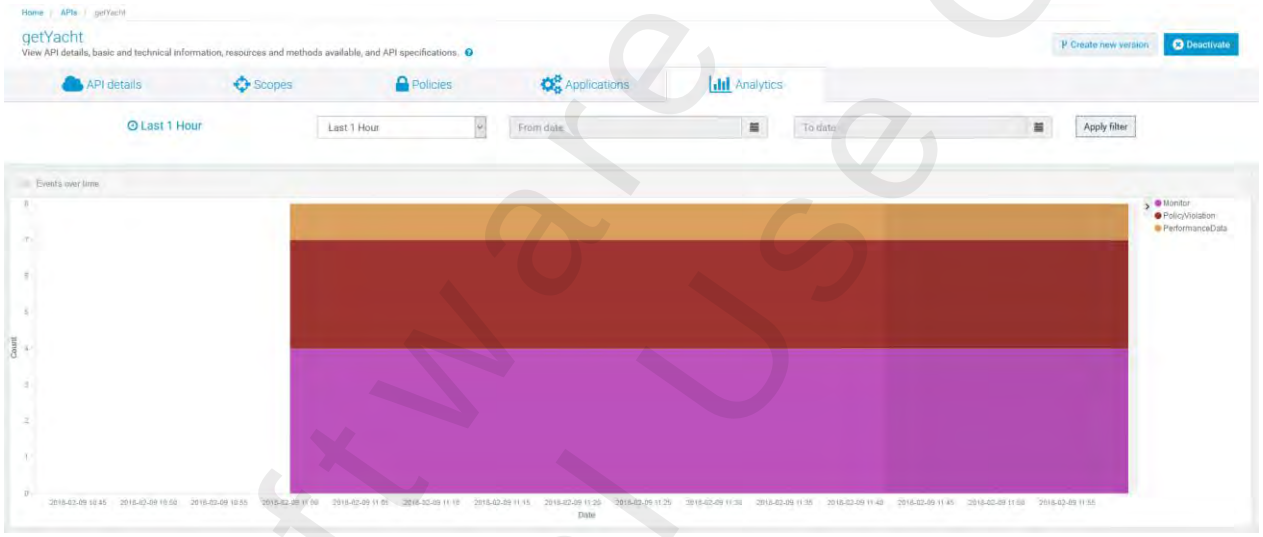



12. Run the request several time. An error will be returned because of the throttling policy mentioning of too many calls within the defined time limit.

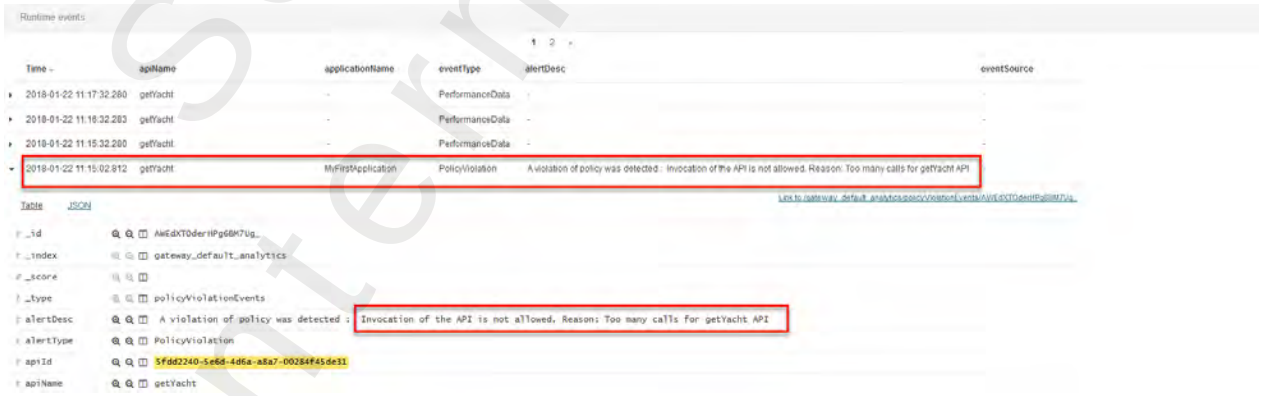



13. Now we try to find these error messages and also the warnings within **API Gateway**. Go back to **API Gateway UI** and select the service **getYacht**. Open the tab **Analytics**.

14. Set the filter to **Last 1 Hour**. Click **Apply filter**




15. Within **Runtime events** open the **PolicyViolation** event. Search for the corresponding alert description.



# Exercise 13:
# Using a Routing Policy Action

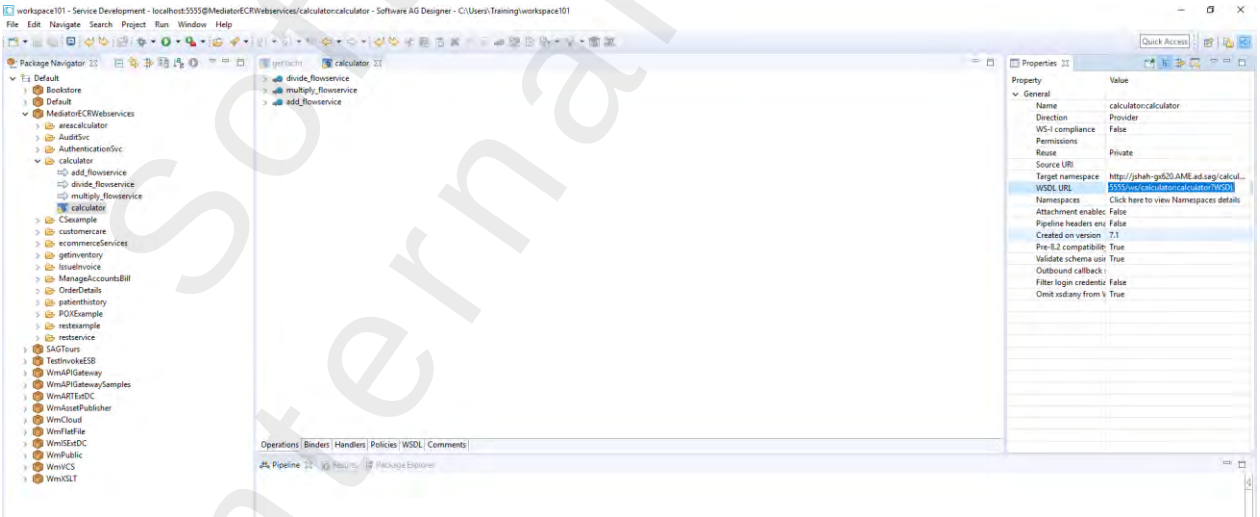
## Objectives

In this exercise, you will define a routing policy action using context based routing.

## Steps

1. Open **Windows Services UI** to double check that the following services, needed for API Gateway, are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Event Data store 10.1 – should be started together with IS**
2. Open **Software AG Designer 10.1** from the Windows **Start Menu > Software AG > Tools**. In case you are asked for a **secure storage password** , provide the password **manage**.


   Accept the default workspace .
3. Select **Window -> Open Perspective -> Service Development**. Within **Package Navigator** open the package **MediatorECRWebservices** and edit the **calculator:calculator** Web Service Descriptor.



In the Properties, copy the Value of WSDL URL and paste it into the Import URL field in **API Gateway**. The value pasted should be similar to the following:

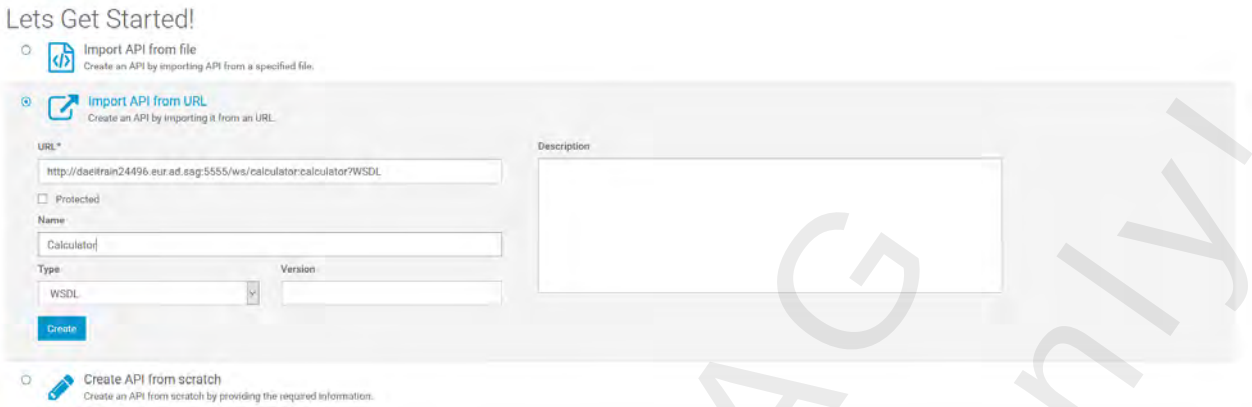
**http://DAETRAIN24496.eur.ad.sag:5555/ws/calculator:calculator?WSDL**

4. Login to the **API Gateway UI** as user **Sumala/manage** and create a new SOAP based API. Use the option **Import API from URL**.

a. **URL**: - as mentioned above –

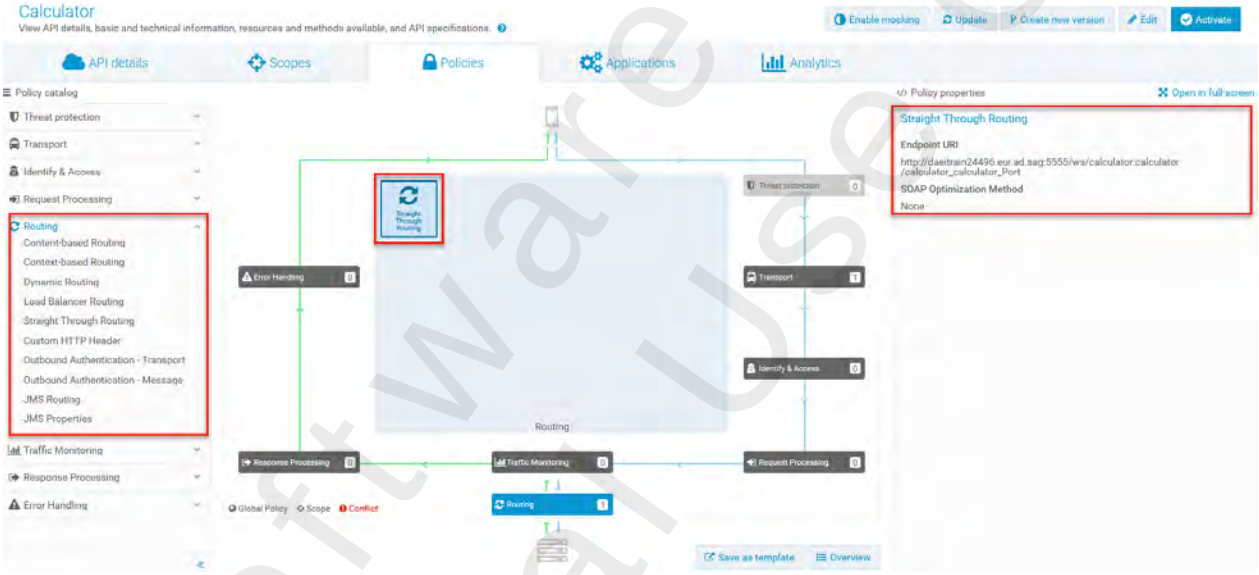
b. **Name**: Calculator

c. **Type**: WSDL

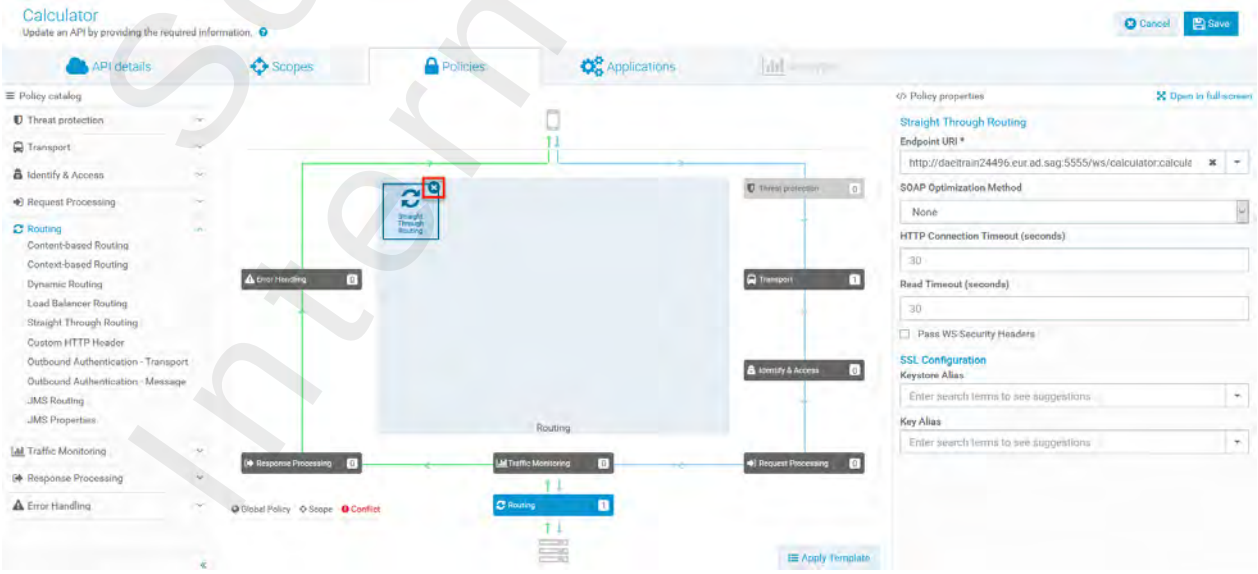



Click **Create.**

5. Navigate to **Policies > Routing**. Switch to Edit mode.




6. Switch to Edit mode. Remove the default configured Policy action **Straight Through Routing**.




7. Select **Content-based Routing** in the section **Policy catalog**.

8. To provide the routing definitions, open the text-file **CalculatorRouting.txt** which is located in folder **C:\Training\APIGateway\Exercise13**

9. Open the Content-based Routing policy action in **full-screen** mode. Provide the following properties:

   a. **Default Route To**.

      i. **Endpoint URI**: - copy and paste Native endpoint value –

      ii. **SOAP Optimization Method:** None

      **iii. HTTP Connection Timeout(seconds)** 30

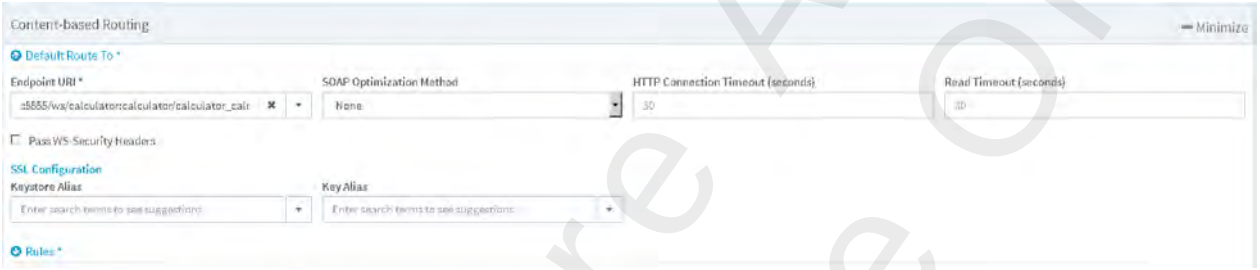
      **iv. Read Timeout (Seconds)** 30

      **v. Pass WS-Security Headers:** - do not check this -

   b. **SSL Configuration**

      i. **Keystore Alias:** - leave empty -

      ii. **Key Alias:** - leave empty –




10. Click the blue icon next to **Rules** and create the following Rules

    a. Rule to follow when calling the Add operation

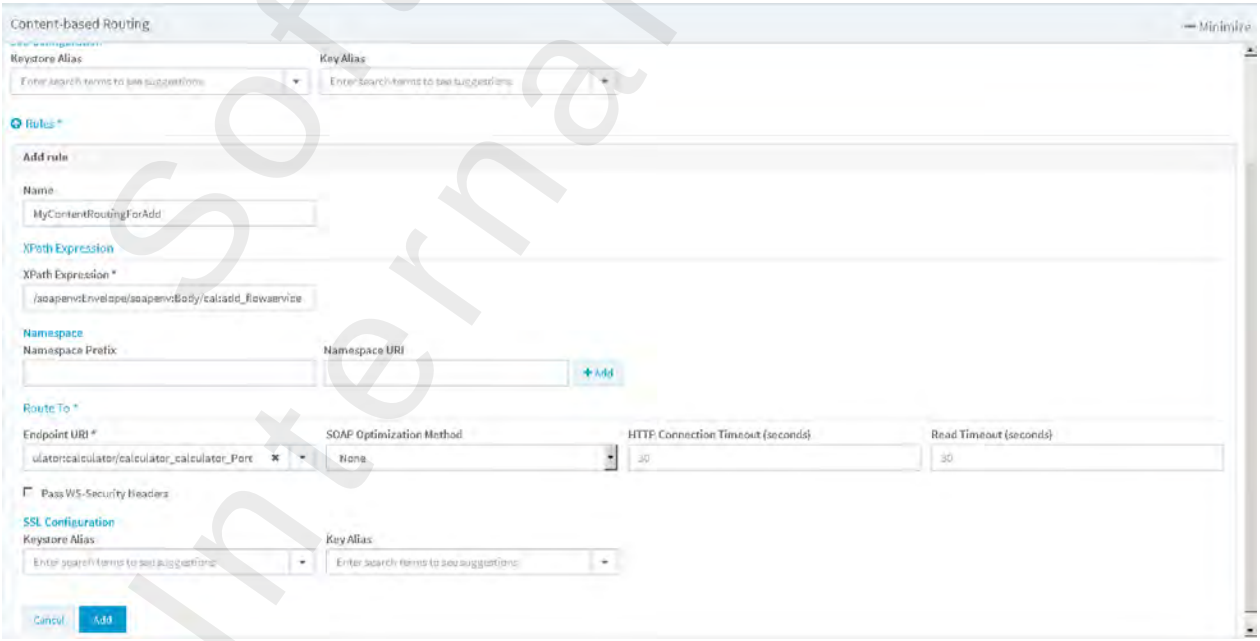
       **i. Name:** MyContentRoutingForAdd

       ii. **XPathExpression:** - copy X-Path expression for Add –

       **iii. Route To**

          **1. Endpoint URI:** - copy changed endpoint for Add operation -

       Leave all other default values.




Click **Add.**

11. Add a second rule for operation **multiply**. Provide the following properties:

a. Rule to follow when calling the **Multiply** operation

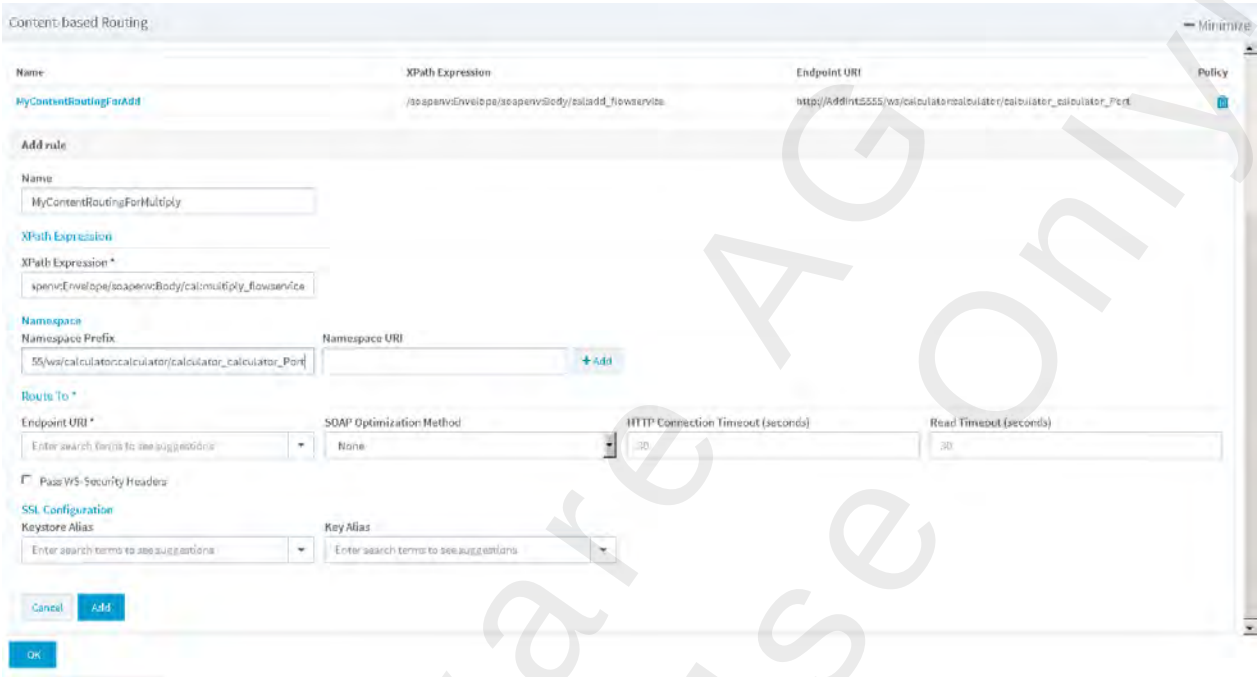
   i. **Name:** MyContentRoutingForMultiply

   ii. **XPathExpression:** - copy X-Path expression for Mulitply –

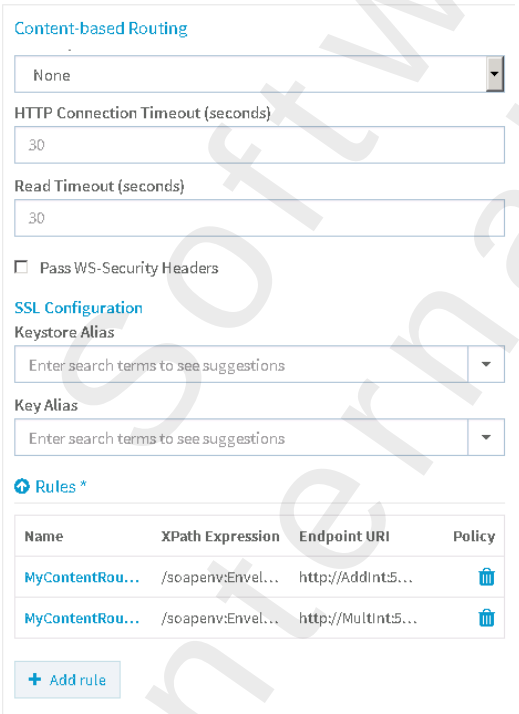
   iii. **Route To**

      1. **Endpoint URI:** - copy changed endpoint for Multiply operation –

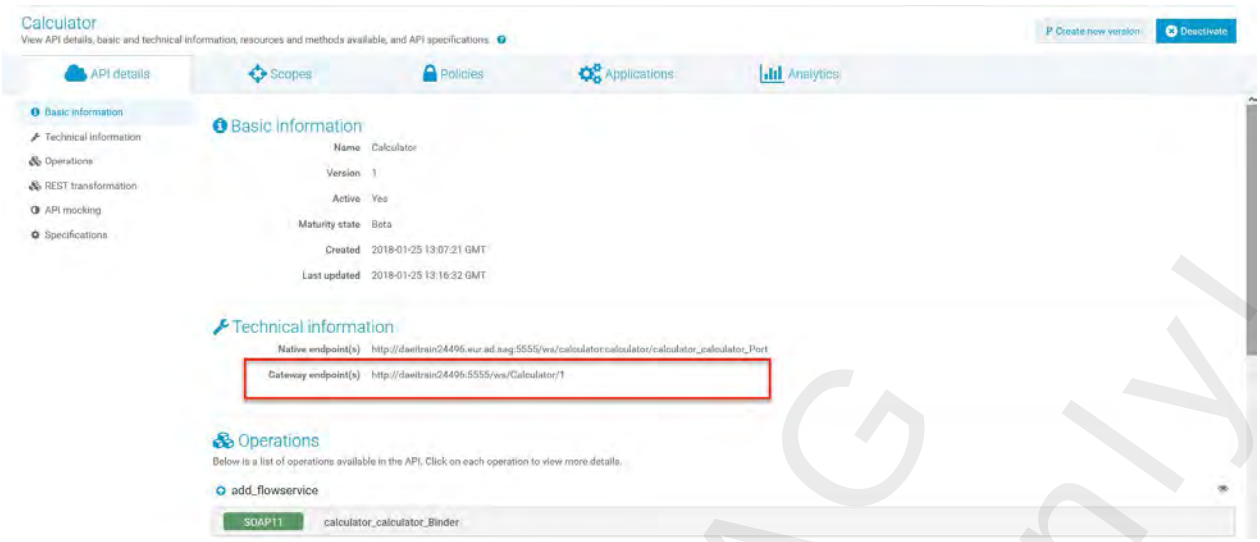
   Leave all default values.




Click **Add**. Click **OK.**

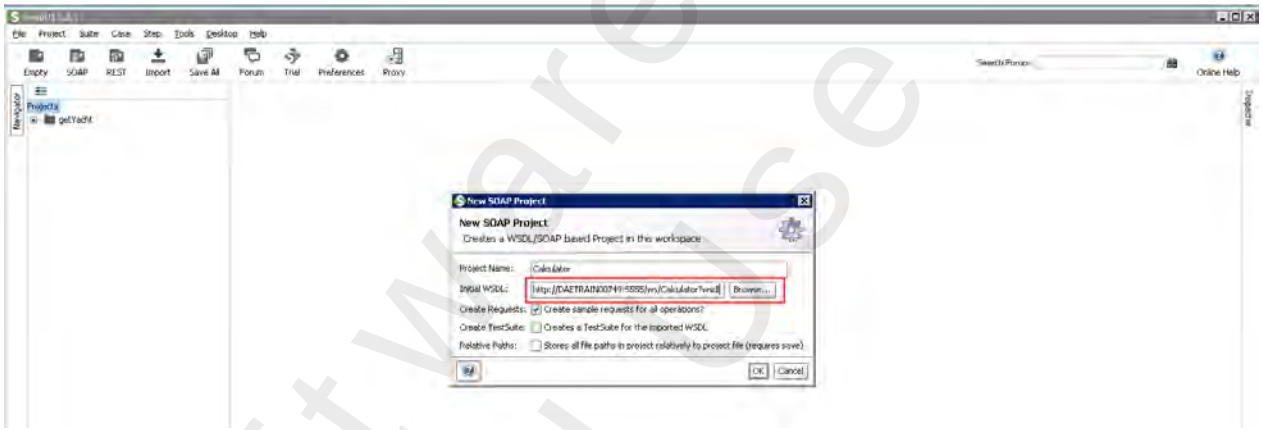



12. Save and activate the **CalculatorAPI**. Navigate to **API details > Technical information**.




13. Open **SOAP UI** and create a new SOAP project with name **Calculator** using the Gateway endpoint value and adding .?wsdl.

    **http://DAETRAIN00749:5555/ws/Calculator/1?wsdl**



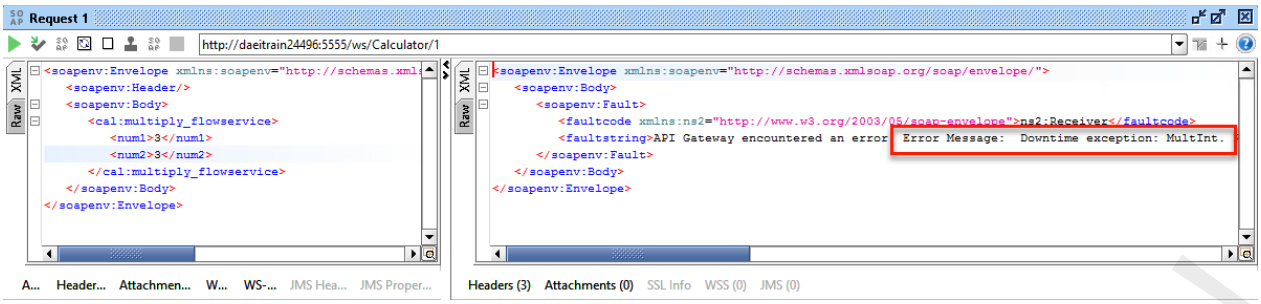

14. Within Projects navigate to **Calculator > calculator_calculator_Binder> add_flowservice > Request 1. Request 1** will be opened in a new window. Provide the following values:

    a. **Num1:** 3

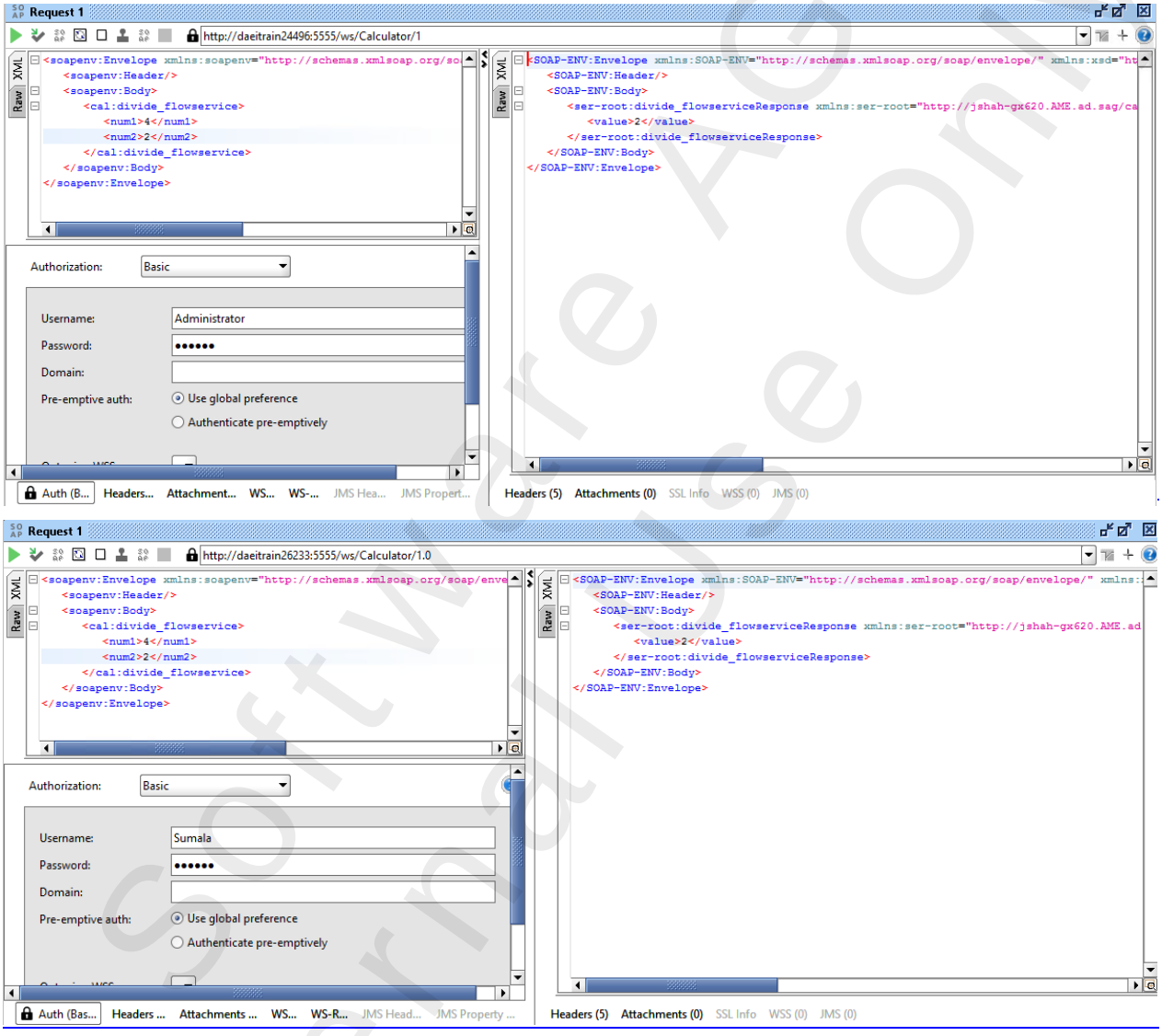
    b. **Num2:** 3

    Hit the run button. SOAP UI will return an error, because host **AddInt** does not exist.


15. Try to run operation **multiply_flowservice**. This will also fail as SOAP UI tries to run service on host MultInt.




16. Try to run **divide_flowservice**. This will fail with a different error message, because of missing credentials. Add **Basic Authentication** for user **Sumala/manage**.




NOTE: You can enable proper execution of the add_flowservice and multiply_flowservice by adding appropriate entries for the hosts “AddInt” and “MultInt” to your hosts file.

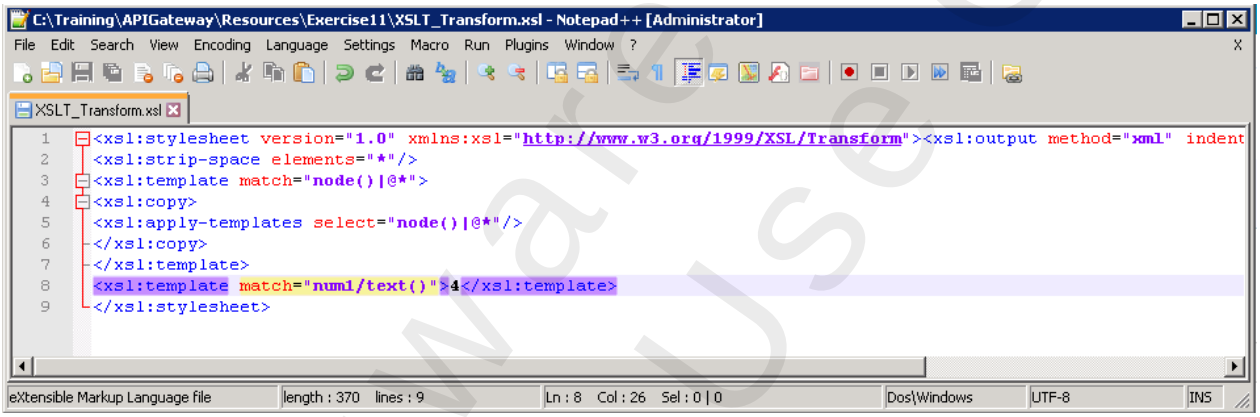
# Exercise 14:
# Updating Content Using XSLT Stylesheet

## Objectives

In this exercise, you will update the SOAP request content by concatenating a string to a particular xml element value, before sending request to the native Service.

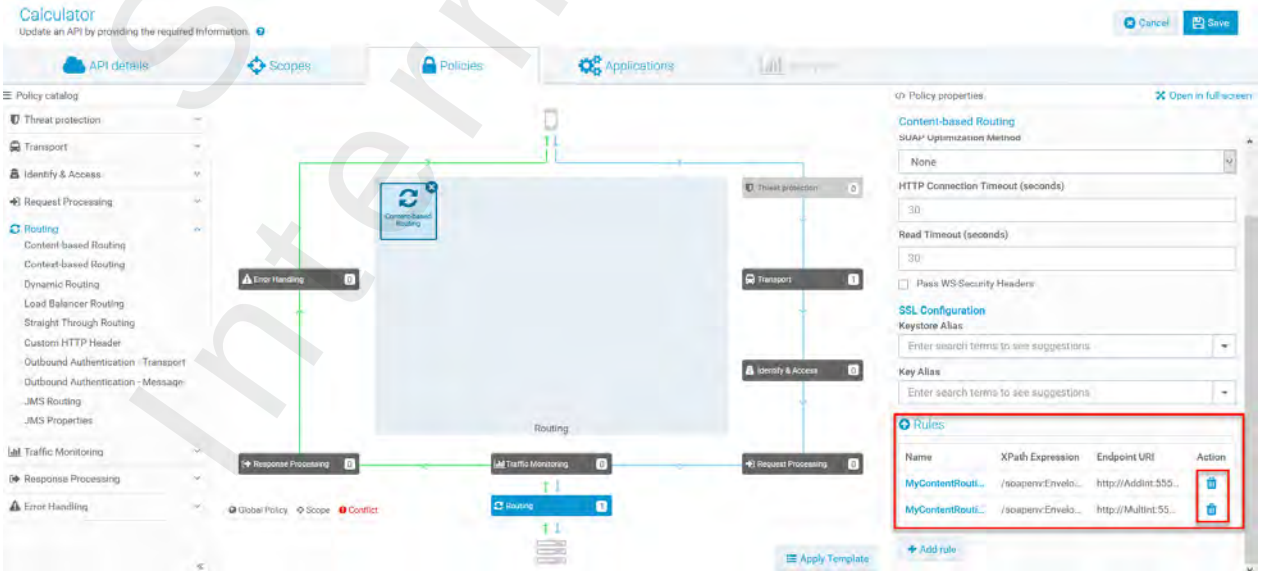
## Steps

2. Open Windows Services UI to double check that the following services, needed for API Gateway, are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Event Data store 10.1 – should be started together with IS**
3. Navigate to folder **C:\Training\APIGateway\Resources\Exercise14** and open the file **XSLT_Transform.xsl** in **Notepad+.**



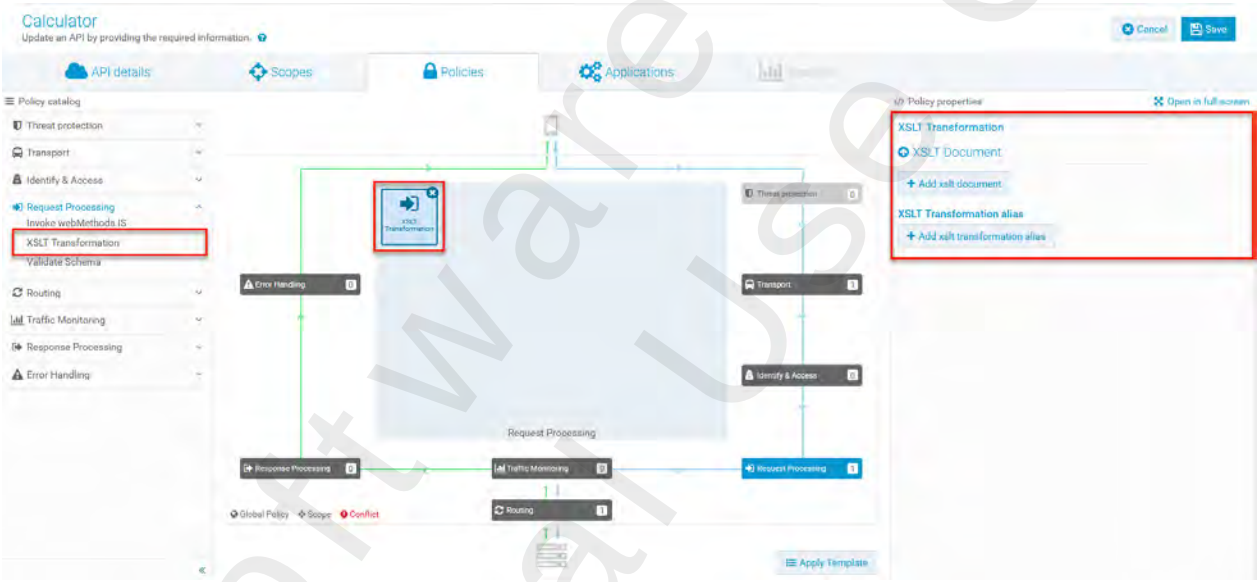

This XML stylesheet will replace whatever is in element “num1” with the value “4”.

4. Login to the **API Gateway UI** as user **Sumala/manage** and reconfigure the SOAP based API **Calculator**.
5. From the list of APIs select the API **Calculator.** On the APIs Detail Page, click **Deactivate.** Switch to **Edit** mode.
6. Navigate to **Policies > Routing.** Remove the rules you have defined in **Content-based Routing**.

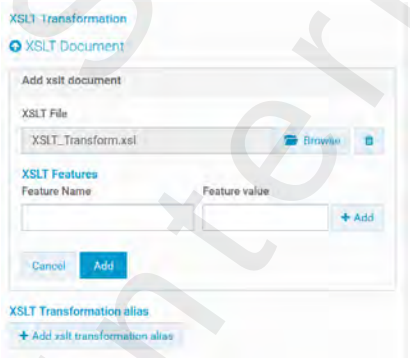


7. Navigate to **Policies > Request Processing.**


8. Select **XSLT Transformation** in the **Policy catalog.**




9. Within the Policy Properties click **Add xslt document** and provide the following values:
   a. Click Browse and search for the file **XSLT_Transform.xsl** in folder **C:\Training\APIGateway\ Exercise14.** Do not provide properties for **XSLT Features.**




   Click **Add.**

10. Click **Save** and click **Activate**.

11. Open **SOAP UI** and naviagte to the project **Calculator** you have configured in the last exercise.

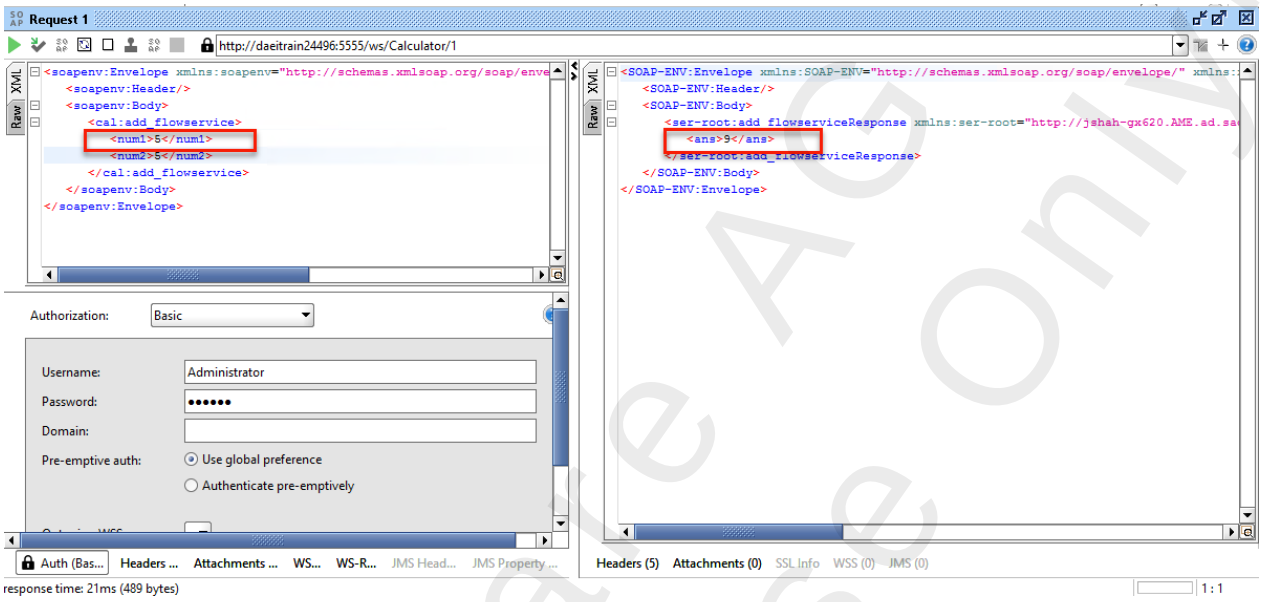
12. Open the request **Request 1** for the operation **add:flowservice.** Run the service with the following properties:

    a. **num1**: 5

    b. **num2** 5

    c. **Basic Authentication:** Sumala/manage

13. The SOAP response shows the result. During execution the XSLT Transformation changed <num1>5</num1> to <num1>5</num1> , making the result to <ans>9</ans>.



This page intentionally left blank

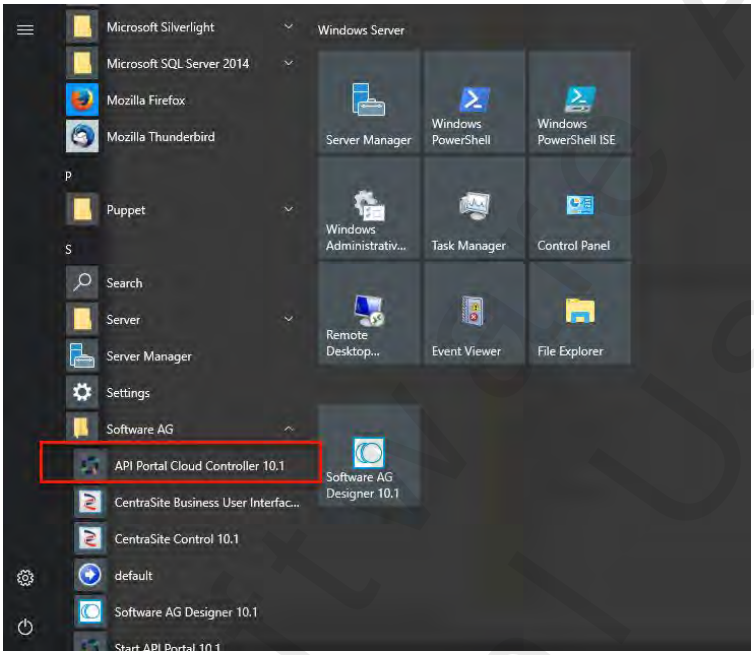
# Exercise 15:
# Setting up Integration with API Portal

## Objectives

In this exercise, you will create a Package and a Plan to allow a set of Quality definitions like costs, availability of the service for a set of APIs.

## Steps

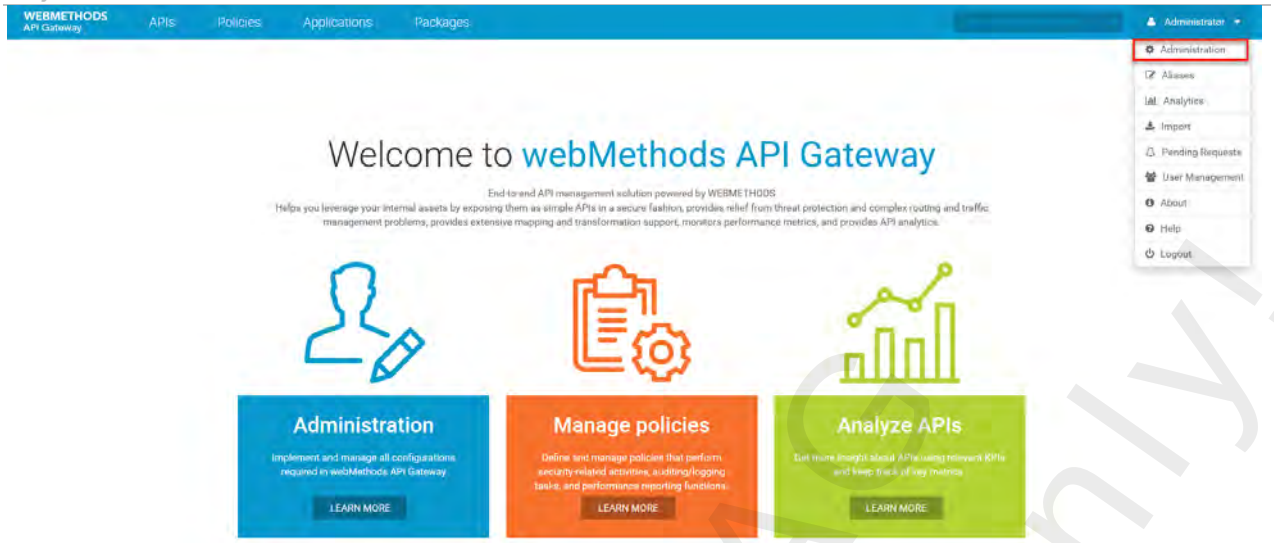
1. Open Windows Services UI to double check that the following services, needed API Gateway, are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Event Data store 10.1 – should be started together with IS**
   c. **Software AG API Portal 10.1**
2. Start **API Portal Cloud Controller** using Windows Start Menu




3. Provide the command **list** to verify the state of all runnables. Wait until all runnables are started. In case of problems use the command stopall and then the command startall.
   NOTE: IT MAY TAKE A FEW MINUTES FOR ALL RUNNABLES TO START.

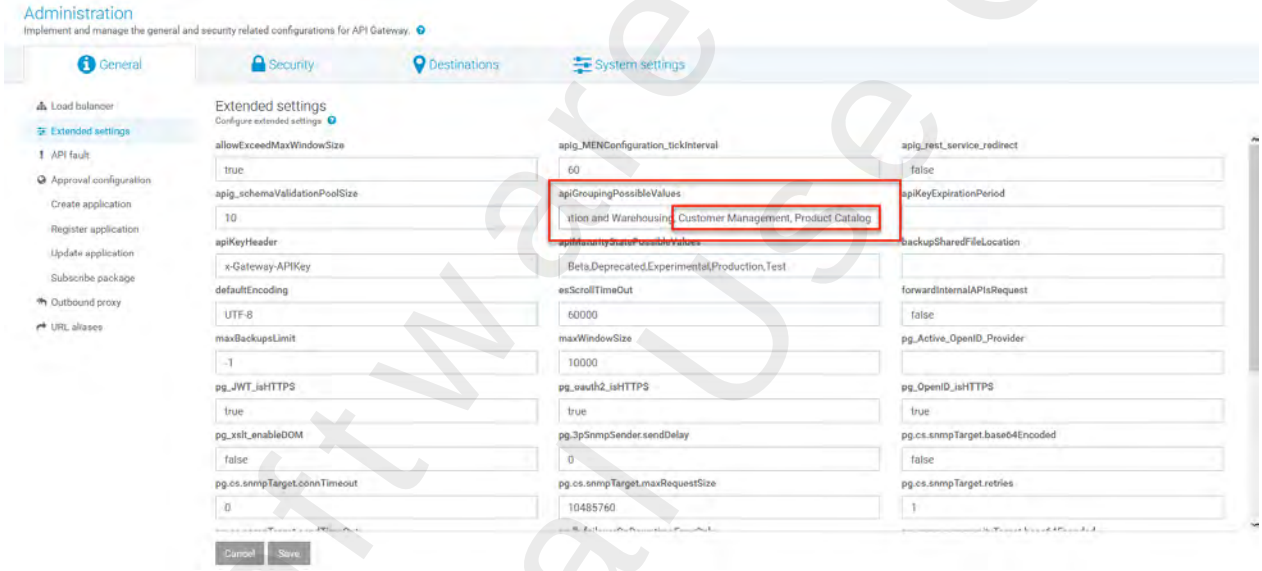
```
ACC+ localhost>list
Node localhost - 7 installed runnables.
ID                State    Version   Type
zoo_s             STARTED  10.0.2.1  com.aris.runnables.zookeeper-run-prod
postgres_s        STARTED  10.0.2.1  com.aris.runnables.PostgreSQL-run-prod
cloudsearch_s     STARTED  10.0.2.1  com.aris.cip.y-cloudsearch-run-prod
elastic_s         STARTED  10.0.2.1  com.aris.runnables.elasticsearch-run-prod
kibana_s          STARTED  10.0.2.1  com.aris.runnables.kibana-run-prod
apiportalbundle_s STARTED  10.0.2.2  com.aris.bundles.apiportalbundle-run-prod
loadbalancer_s    STARTED  10.0.2.1  com.aris.runnables.httpd.httpd-run-prod

ACC+ localhost>_
```

4. Login to **API Gateway** as user **Administrator/manage. S**elect **Administration**.



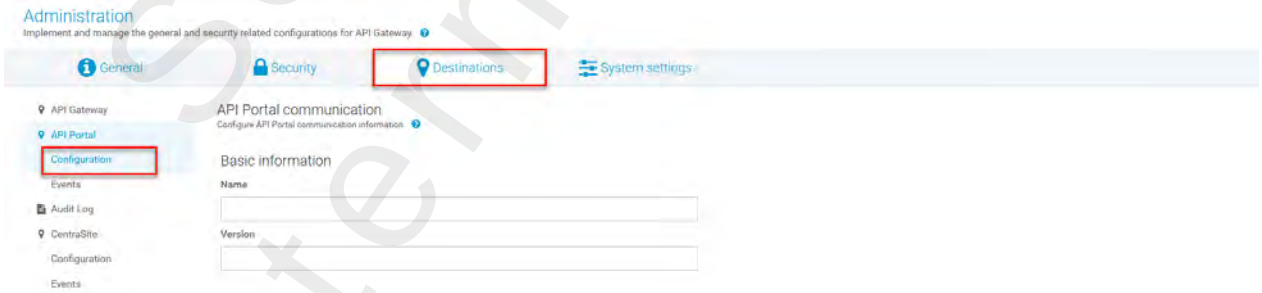

4.5. Navigate **to General > Extended settings**. Adapt the following setting by adding following values to the comma separated list:

a. **apiGroupingPossibleValues**: Customer Management, Product Catalog




Click **Save.**

5.6. Navigate to **Destinations > API Portal > Configuration**




6.7. Provide the following properties:

a. **Name:** MyPortal
b. **Version** 1.0
c. **Portal Configuration**
   i. **Base URL**: http://<your hostname as predefined in the Base URL of the Gateway configuration> :18101
   ii. **Tenant**: default
   **iii. Username:** Andy

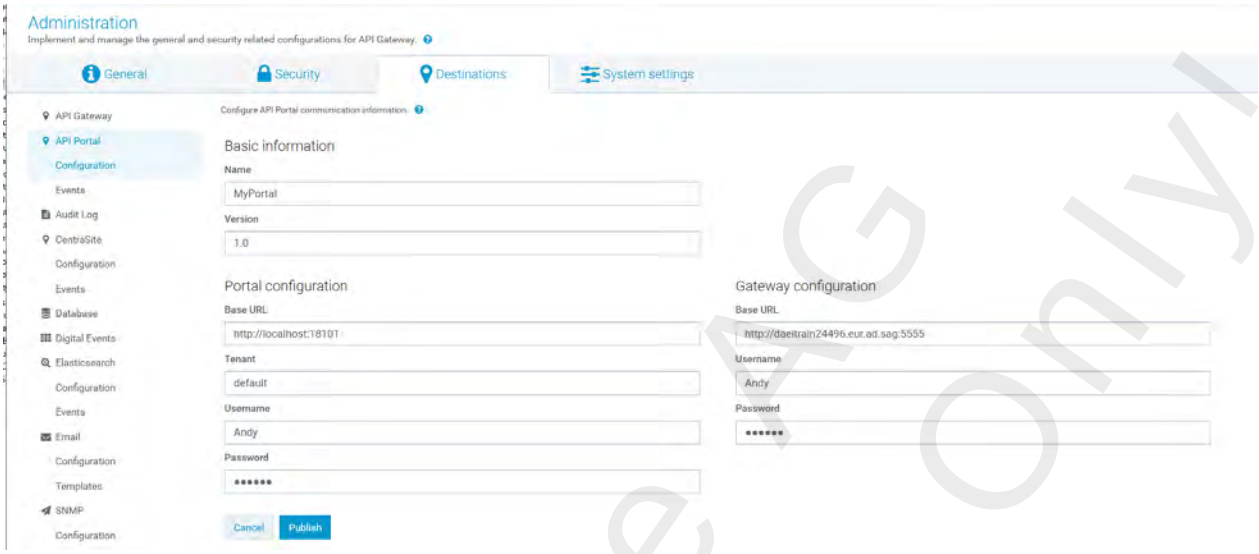
iv. **Password**: manage

d. **Gateway Configuration**

**i. Base URL**: - leave defaults -

**ii. Username:** Andy

iii. **Password**: manage




Click **Publish**. You will get a success message as response.

10. Navigate to **Destinations > API Portal > Events.** We want to send **Performance Metrics** and **Events** to **API-Portal.** Set the following properties

a. **Event Types**:

i. **Error:** - select _

ii. **Lifecycle:** - select –

iii. **Policy violation:** **-** select -

b. **Performance Metrics Data:**

i. **Report performance data:** - select -

ii. **Publish Interval (minutes)**: 2


Click **Save**.

This page intentionally left blank

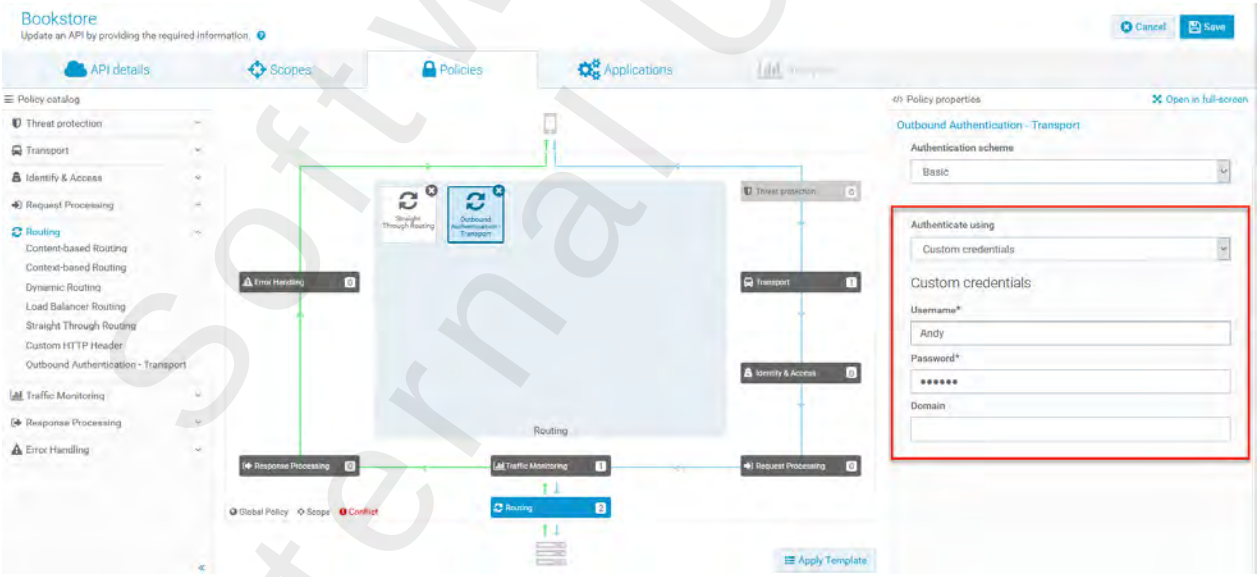
# Exercise 16:
# Publishing API to API-Portal

## Objectives

In this exercise, you will learn how to publish the virtual REST APIs to the API Portal. Andy as API Provider will publish the APIs belonging to the SAGTours application to API Portal in order to expose them to external consumers.

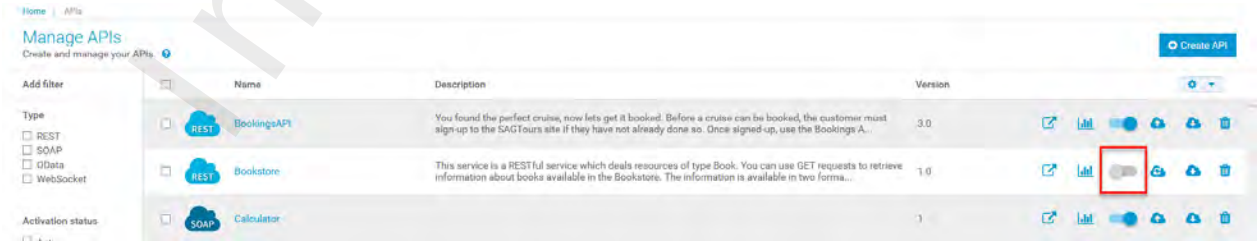
## Steps

1. Open Windows Services UI to double check that the following services are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Event Data store 10.1 – should be started together with IS**
   c. **Software AG API Portal**
2. Use **the API-Portal Cloud Controller** to verify that **Software AG API Portal** is running.
3. Login to **API Gateway** as user **Andy/manage.** Navigate to **APIs** and select the API **Bookstore.** Deactivate the API and switch over to **Edit** mode. Navigate to **Policies > Routing.** Add the **Outbound Authentication – Transport** policy. Set the following properties:
   a. **Authentication Scheme**: Basic
   b. **Authenticate using:** Custom Credentials
      i. **Custom Credentials**
         1. **Username:** Andy
         2. **Password:** manage
         3. **Domain:** - leave empty –

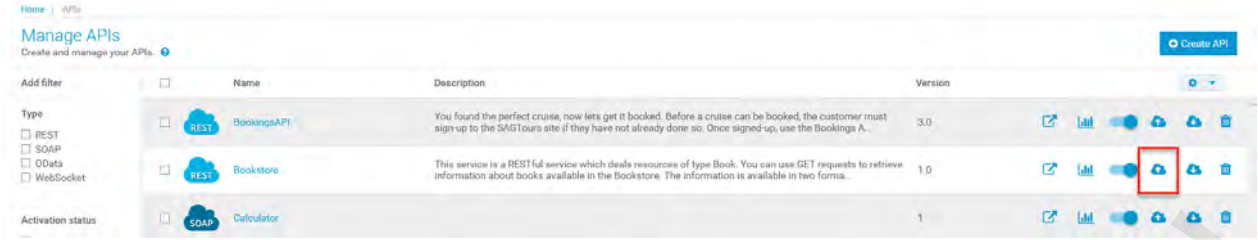



Click **Save**.

4. Navigate to **APIs.** Use the slider to activate the API.

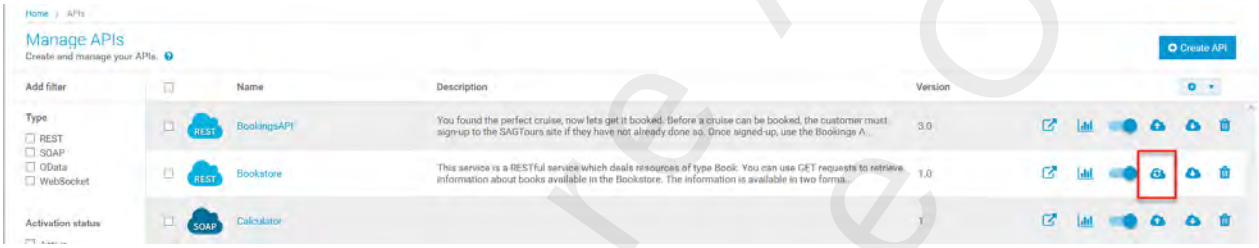


5. Click the cloud icon to publish the API.

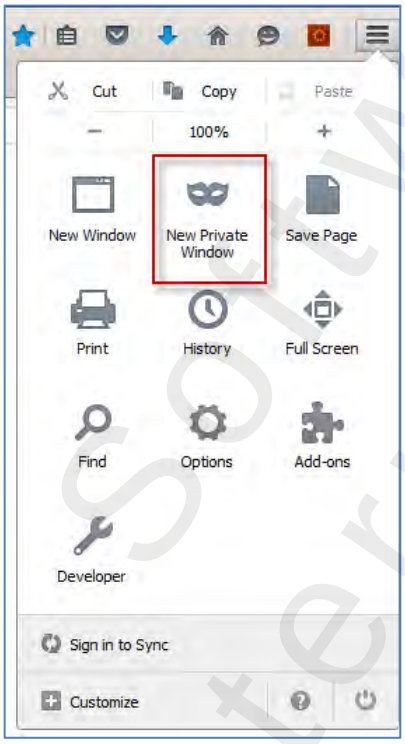



   a. Within the next dialog keep the default settings and click **Publish**.


   You will get a success message and the publish icon is changed to a republish icon.

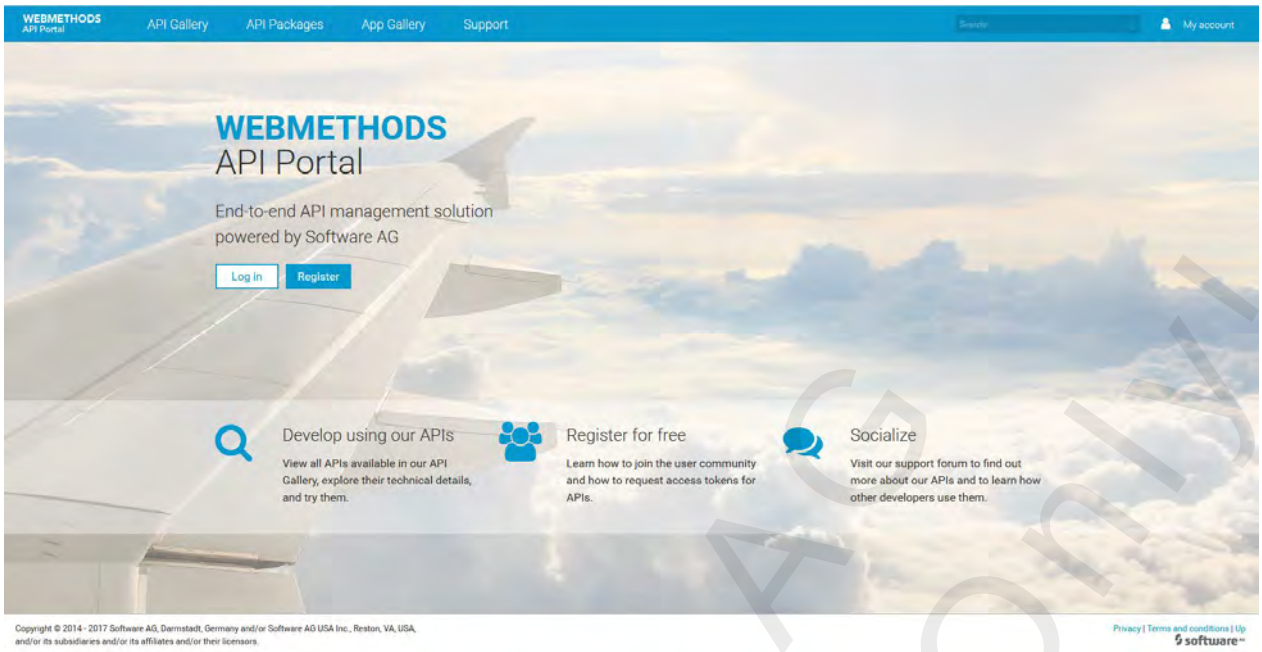



6. In your **Firefox Browser** create a new Windows as **New Private Window**




   This helps to avoid caching problems.

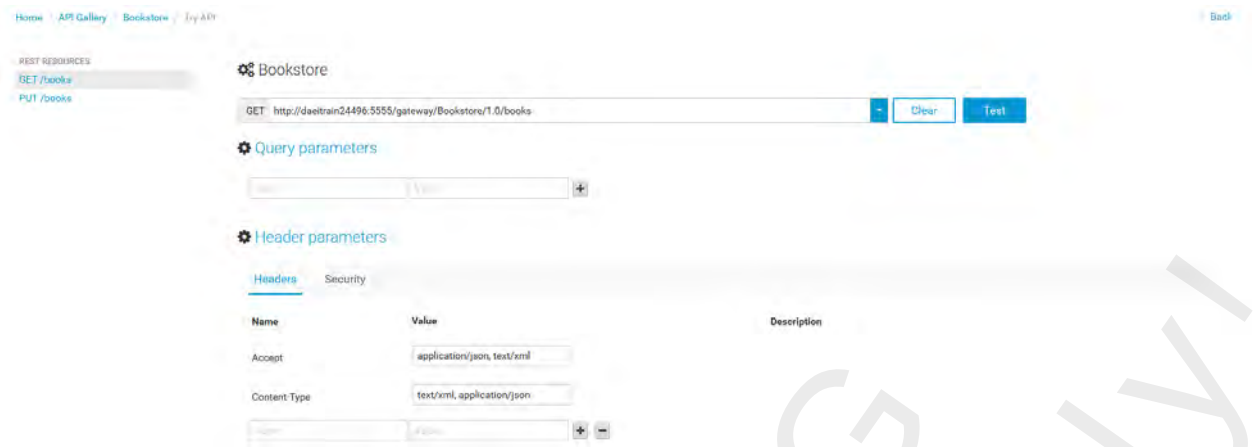
7. Open **API Portal** using the URL **http://localhost:18101/#default/home**. A shortcut **API-Portal** is configured for the Browser Mozilla Firefox.



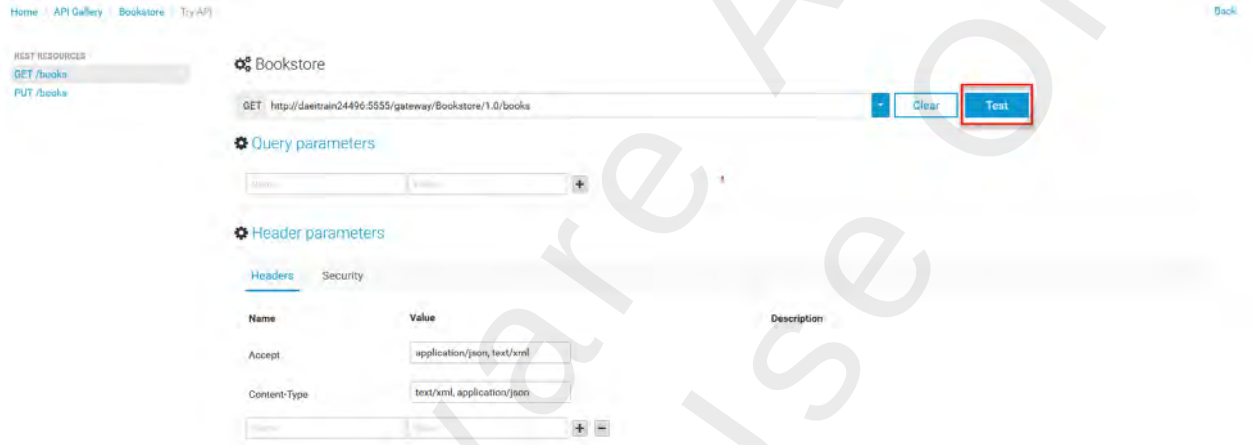

8. Click the **API Gallery** button in the header to see the registered APIs.


9. Select the **Bookstore** API to look for the detailed information._Click **View details**. On the left hand side you find an overview listing the different sections of the API definition/description.

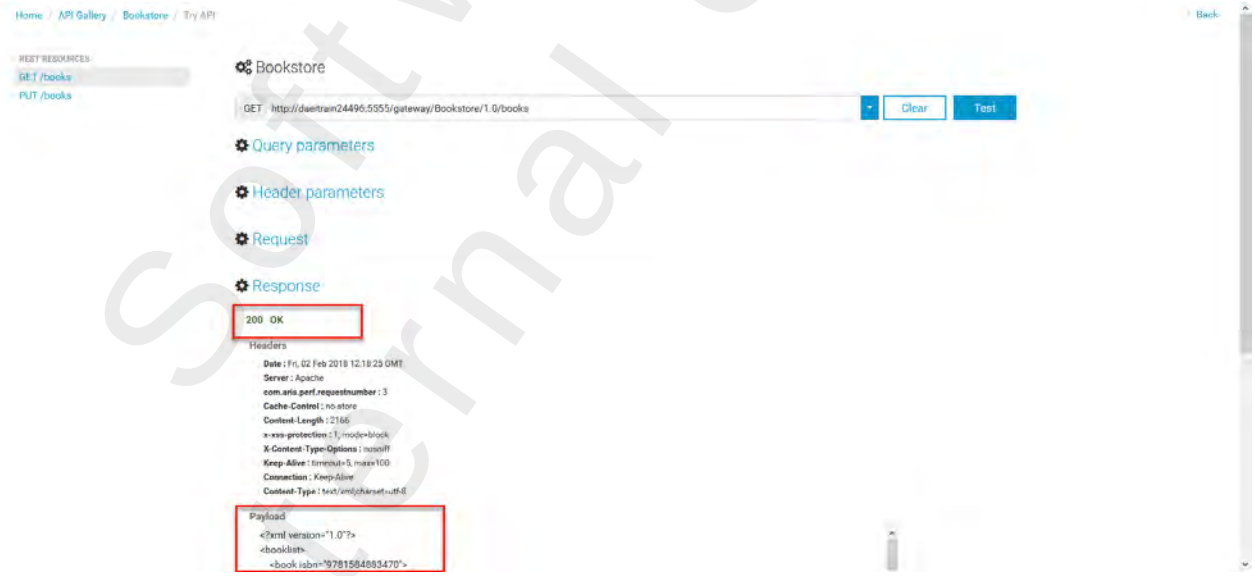

10. Navigate to **Try API>>**




11. Click **Test** to run the API with the default settings. Remember the Basic authentication requested by the native Bookstore API is provided with the **Outbound Authentication – Transport** policy we have defined on the API on API Gateway.




12. Scroll down to the section **Response.** You will find the response code **200 OK** and within the **Payload** section you will find the list of book.



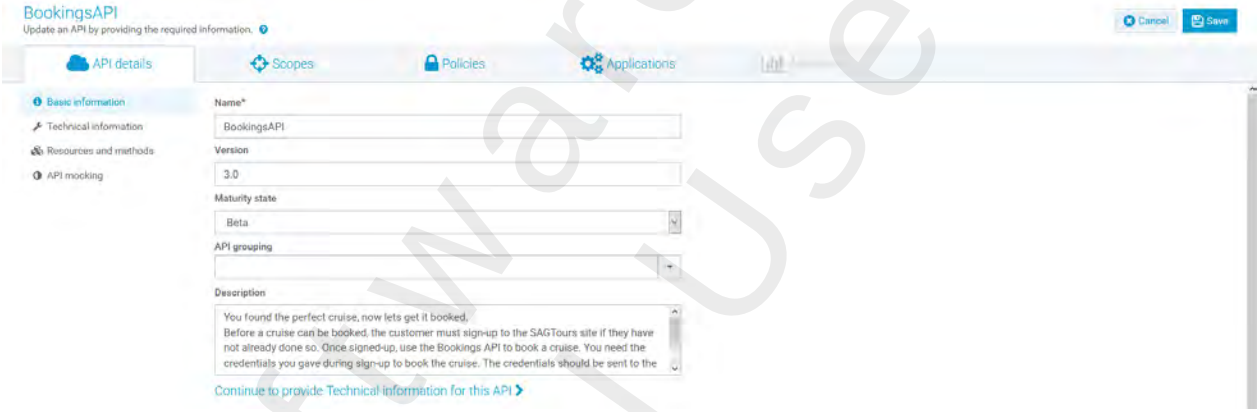
# Exercise 17:
# Publishing SAGTours APIs to API-Portal

## Objectives

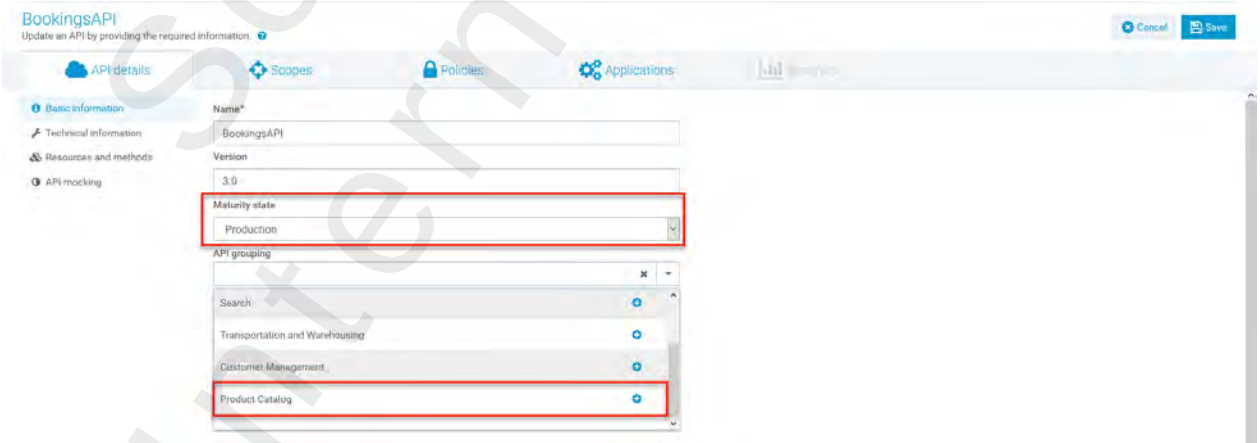
In this exercise, you will learn how to publish the virtual REST APIs to the API Portal. Andy as API Provider will publish the APIs belonging to the SAGTours application to API Portal in order to expose them to external consumers.

## Steps

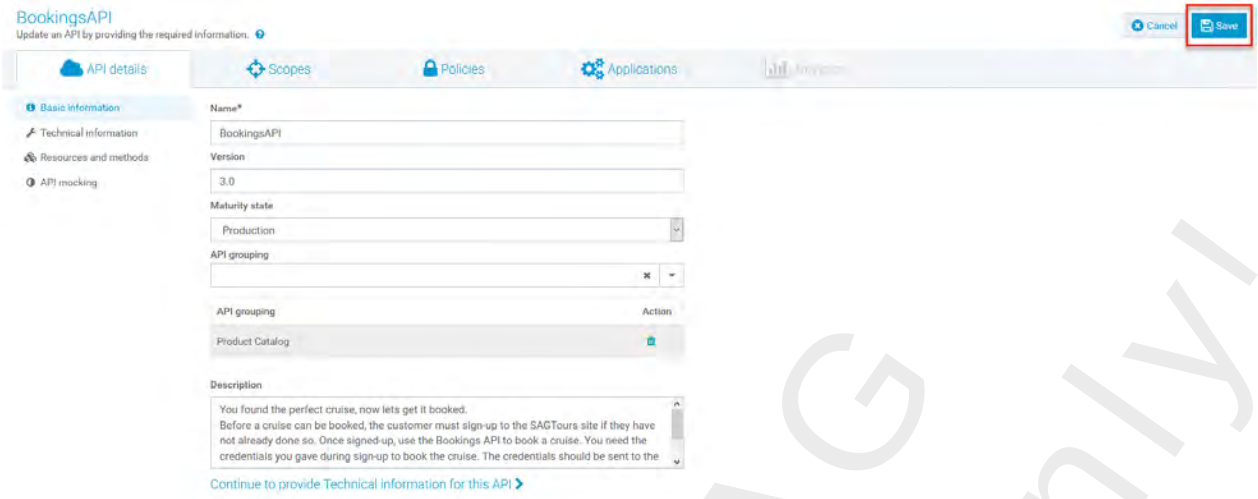
13. Open Windows Services UI to double check that the following services are up and running. If service is not running, start the service.
    a. **Software AG Integration Server 10.1**
    b. **Software AG Event Data store 10.1 – should be started together with IS**
    c. **Software AG API Portal 10.1**
14. Use **the API Portal Cloud Controller** to verify that **Software AG API Portal** is running.
15. Login to **API Gateway** as user **Andy/manage.** Navigate to **APIs** and select the API **BookingsAPI.** Deactivate the API and switch over to **Edit** mode. Navigate to **API Details > Basic information.**



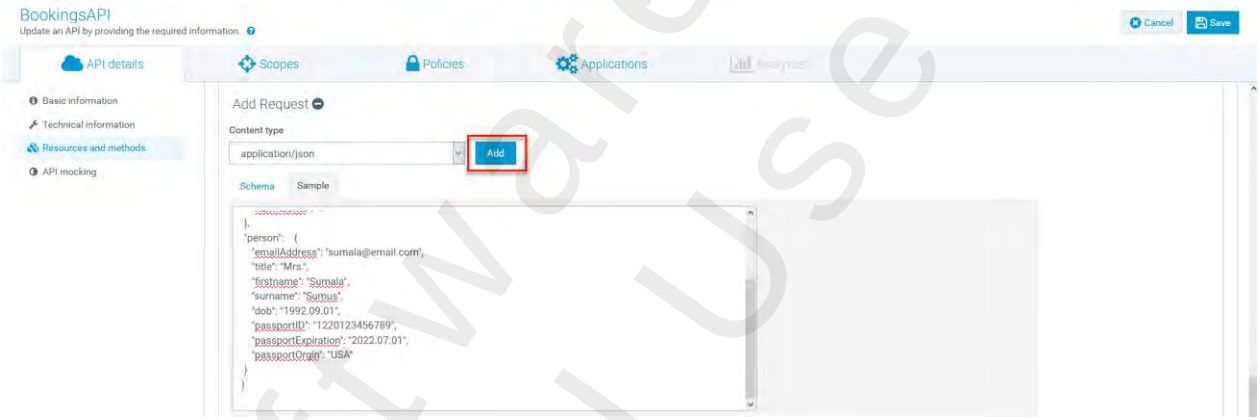

16. Provide the following properties and scroll down to the profile **API Portal Information**.
    a. **API Maturity Status**: Production
    b. **API Grouping**: Product Catalog




From the list of **API grouping** values select **Product Catalog** and click the + sign.

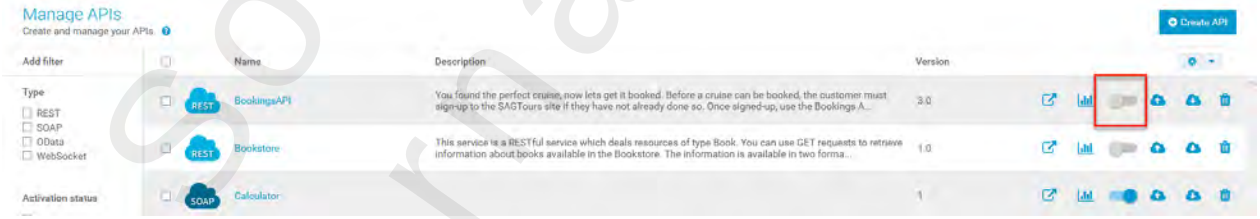



17. Provide a sample request body. Navigate to **API Details > Resources and methods > /bookings > POST**. Click **Add Request**. Provide the following properties
    a. **Content type**: application/json
    b. **Sample:** copy and paste the request sample out of the file **BookingAPI.txt** available in folder **C:\Training\456-71E\Exercise16**

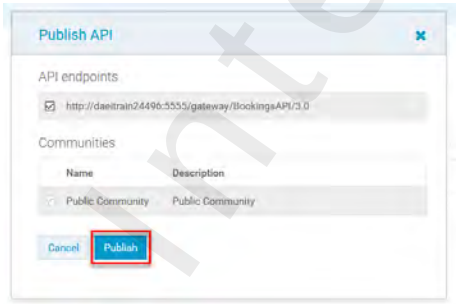



Click **Add.** Click **Save.**

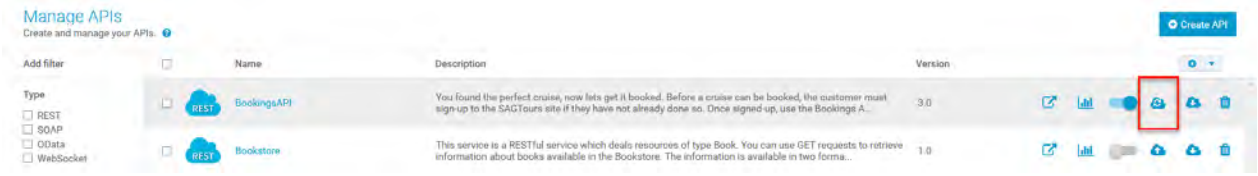
18. Navigate to **APIs.** Use the slider to activate the API.




19. Click the cloud icon to publish the API. Within the next dialog keep the default settings and click **Publish**.




You will get a success message and the publish icon is changed to a republish icon.

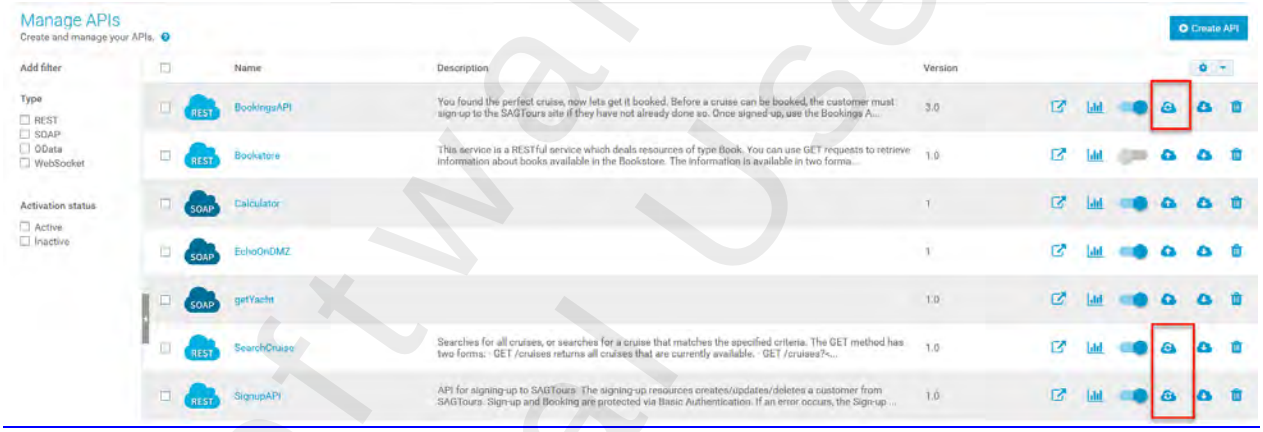


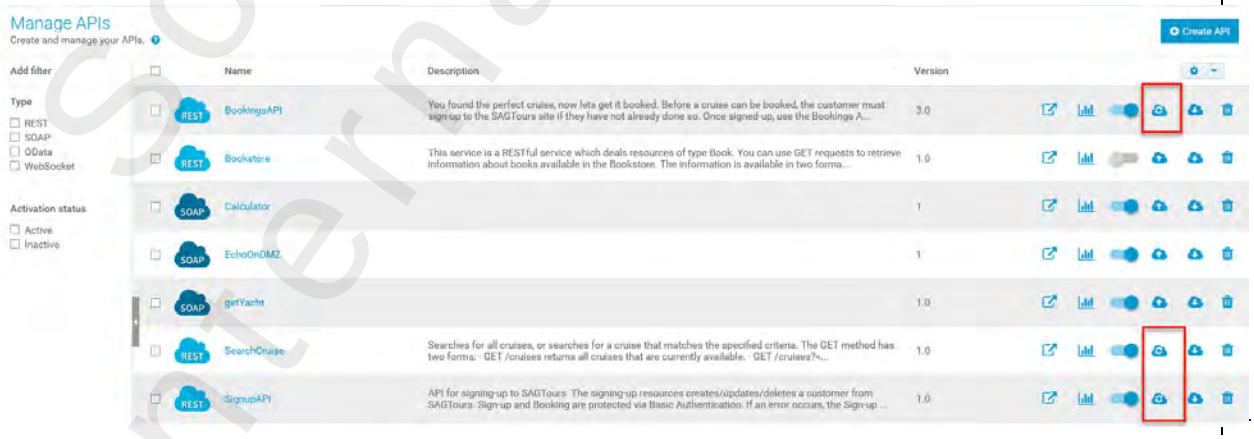

20. Open **SearchCruise** and provide the following properties:
    a. **API Maturity Status**: Production
    b. **API Grouping:** Search

    Save your changes. Activate and publish the API to **API-Portal.**
21. Open **Sign~~u~~ U~~p~~pAPI** and provide the following properties:
    a. **API Maturity Status**: Production
    b. **API Grouping**: Customer Management
    c. **API Details > Resources and methods > /customer > POST:** copy and paste the request sample for POST out of the file **SignUpAPI.txt** available in folder **C:\Training\456-71E\Exercise16**
        i. Click **Add.**
    d. **API Details > Resources and methods > /customer/{customerID} > PUT:** copy and paste the request sample for PUT out of the file **SignUpAPI.txt** available in folder **C:\Training\456-71E\Exercise16**
        i. Click **Add.**

    Click **Save.**

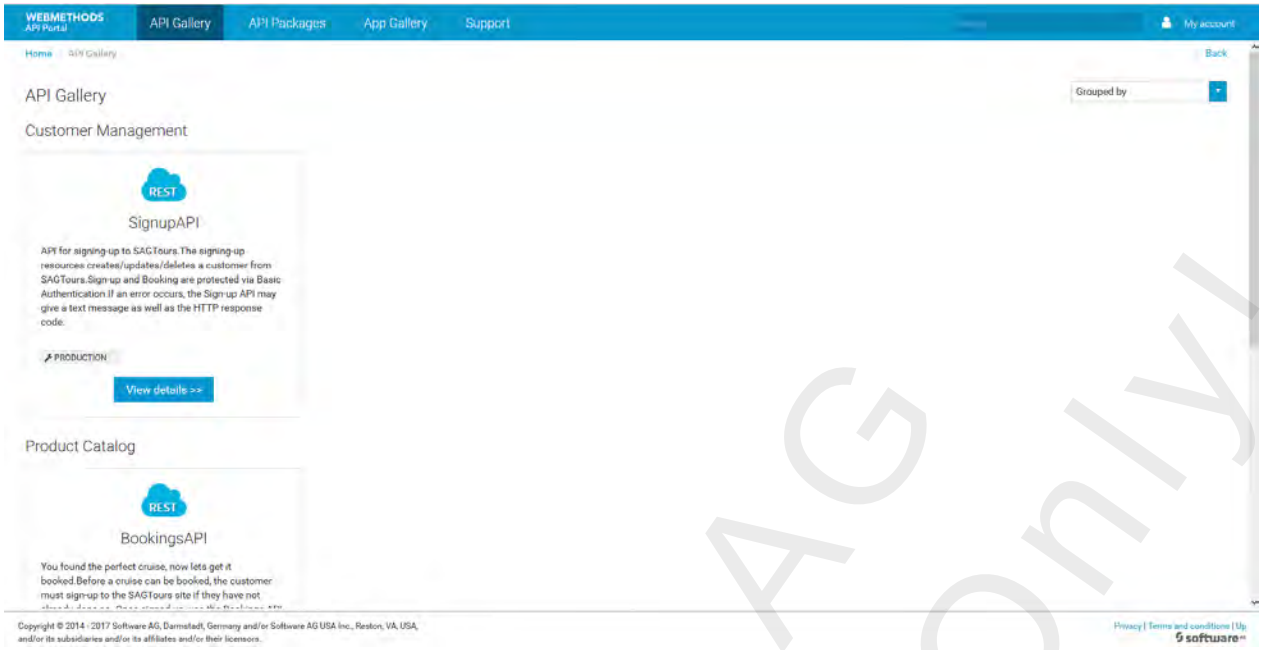
    Activate and publish the API to **API-Portal.**




22. In your **Firefox Browser** create a new Windows as **New Private Window** and open **API Portal.**

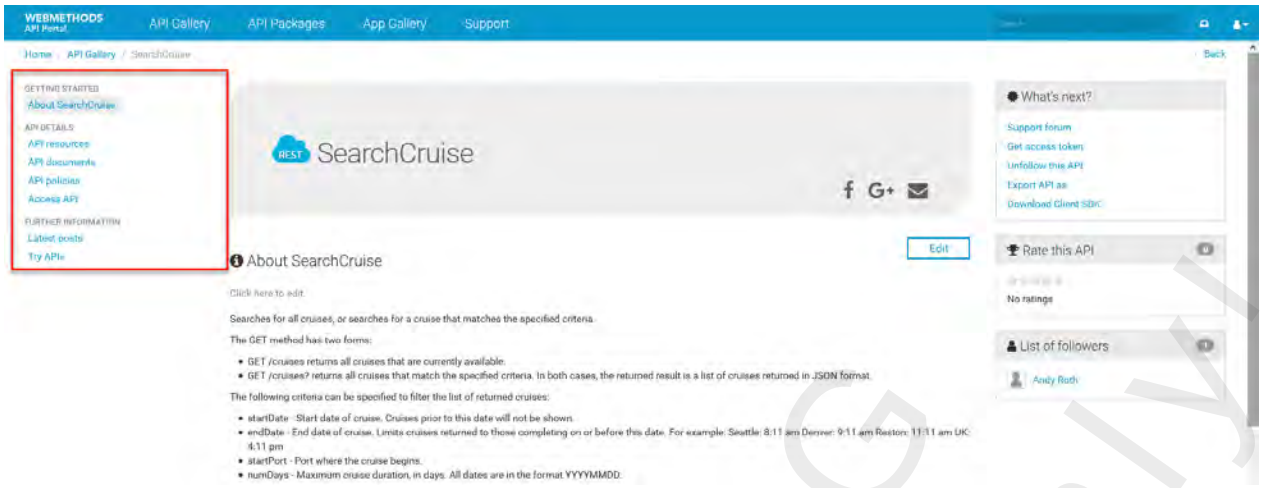
    This helps to avoid caching problems.
23. Open the **API Portal** using the URL **http://localhost:18101/#default/home**. A shortcut **API-Portal** is configured for the Browser Mozilla Firefox.
24. Click the **API Gallery** button in the header to see the registered APIs.



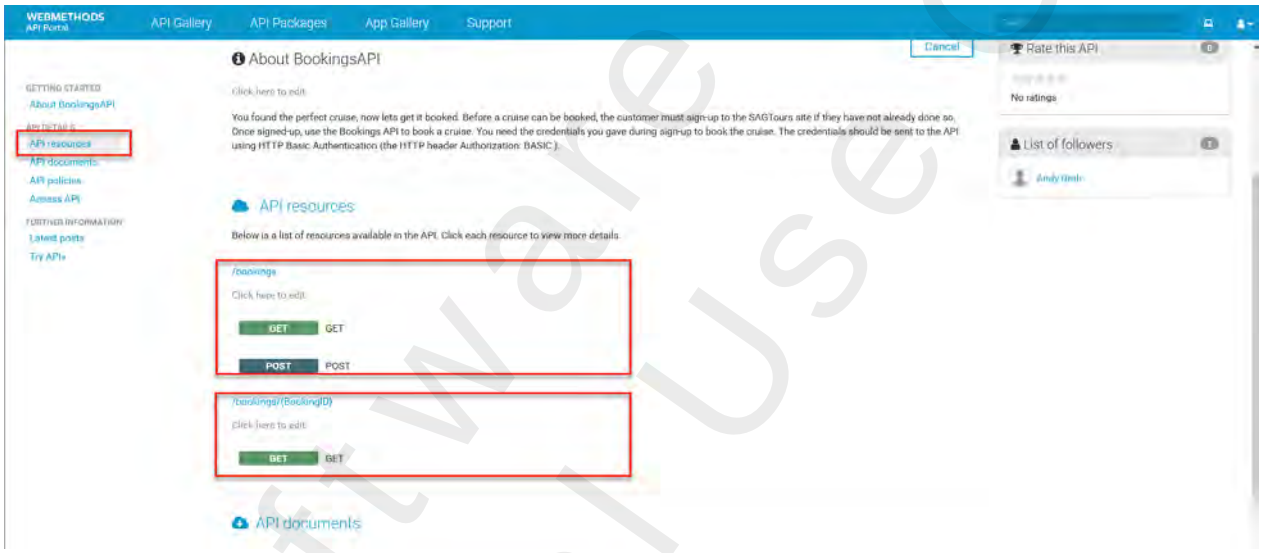

25. The default grouping is **API group** like **Customer Management**, **Product Catalog**, ….
26. We would like to review the APIs of the **SAGTours** application as API Provider. Login to **API Portal.** Navigate to **My account > Log in**. Login as user **Andy/manage**


27. Try to find the APIs you have deployed to the **API Portal**. Use the **Search** location in the header bar and provide the following characters
    a. replace **Search** with **Search** and wait or click the **Search** icon

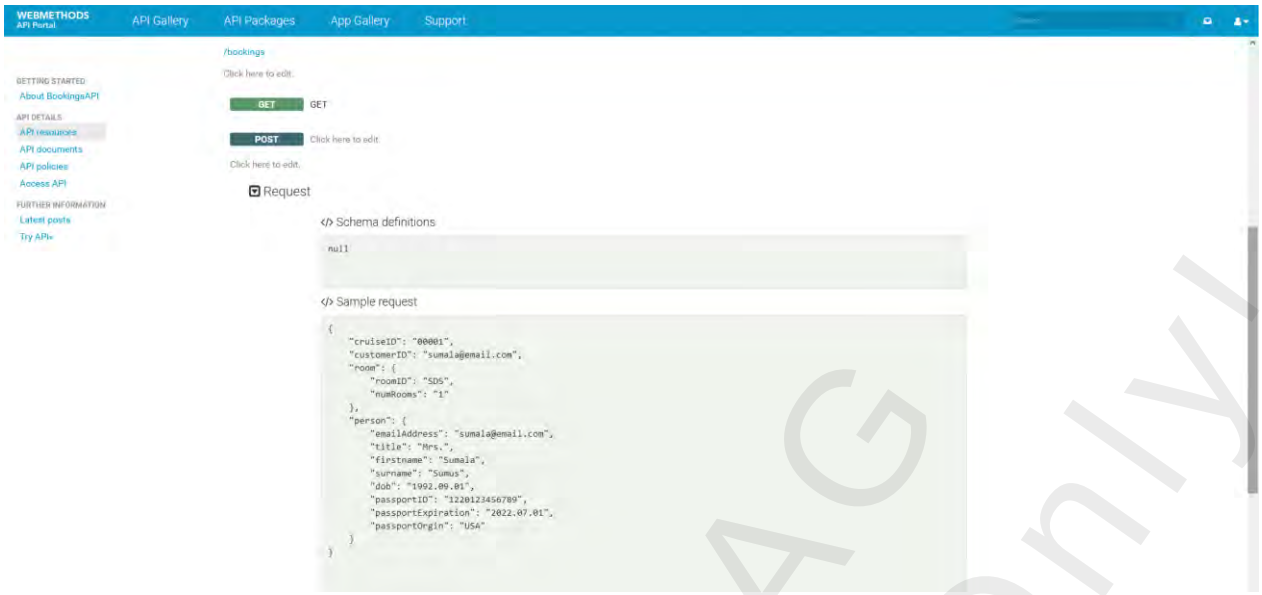

28. From the list of APIs select the API **SearchCruise.** Hover the mouse over the API and click. This will open the **SearchCruise** in Detail view. On the left hand side you find an overview listing the different sections of the API definition/description.




29. Search and open **BookingsAPI**. Use the **API Resources** link to list the resources available within the **BookingsAPI**. Within **API Resources** you find a list of the Resources and Methods. Click on the paths to show details about a particular resource invocation. The combination of Methods and Resources as defined in **API Gateway**.



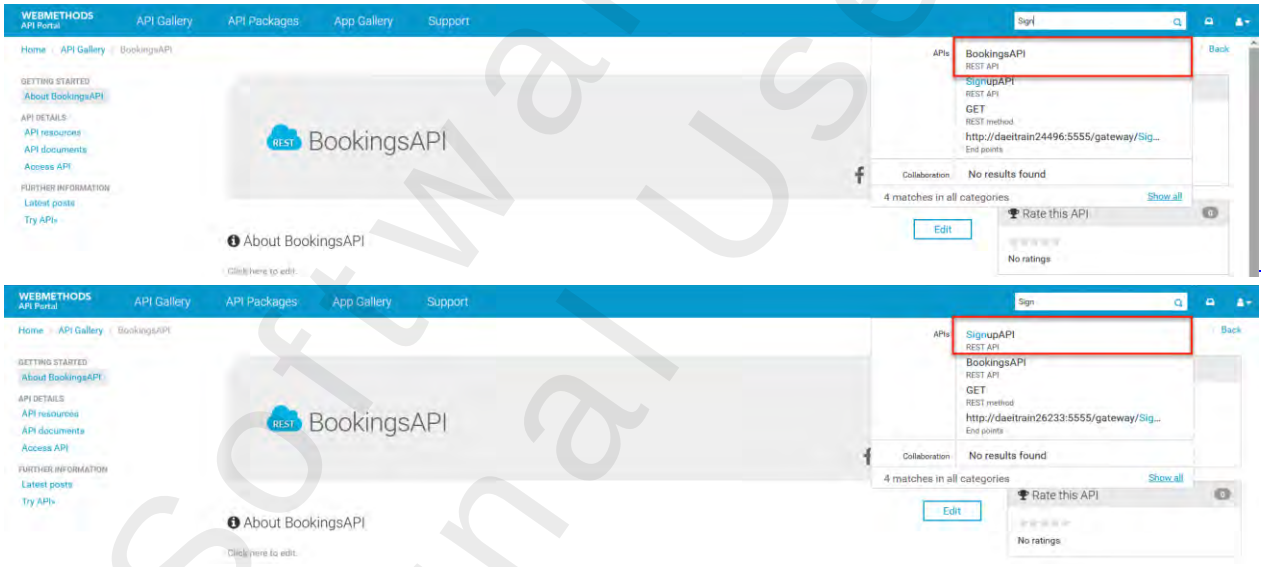

30. Click on the Method to show details about a particular resource invocation. The combination of Methods and Resources as defined in **API Gateway** including
    a. Parameters
    b. Request



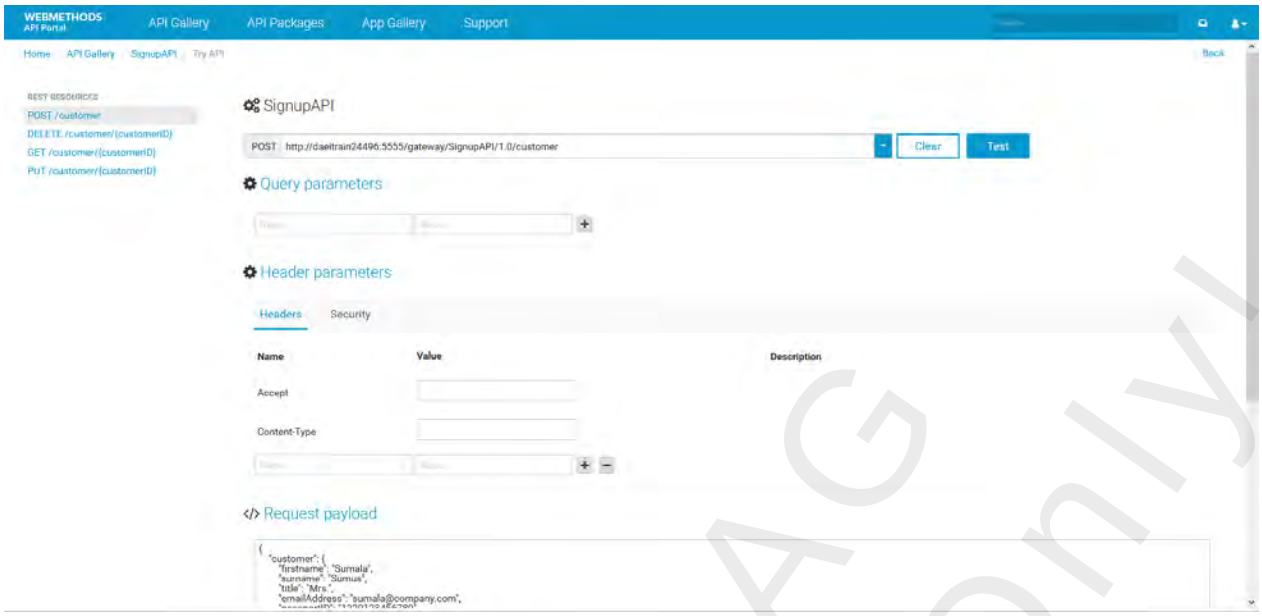

31. In order to use the Test functionality of the API Portal navigate back to the section **About the BookingsAPI**. The content of this section mentions that the customer needs to sign-up to the SAGtours before he can book a cruise.

    Before a cruise can be booked, the customer must sign-up to the SAGTours site if they have not already done so.

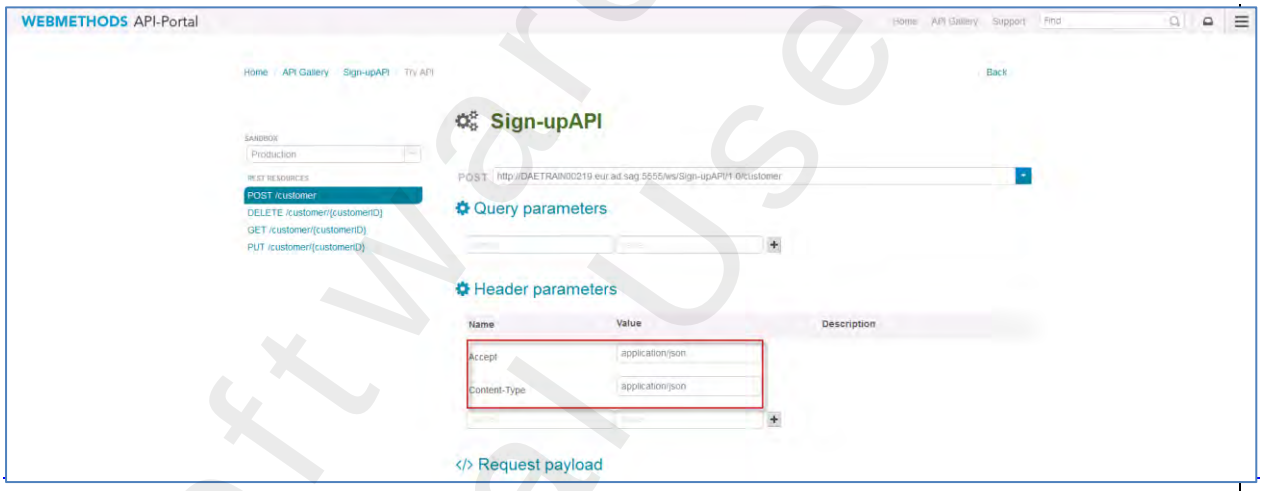
32. Use the **Search** functionality again and search for the **SignupAPI**.

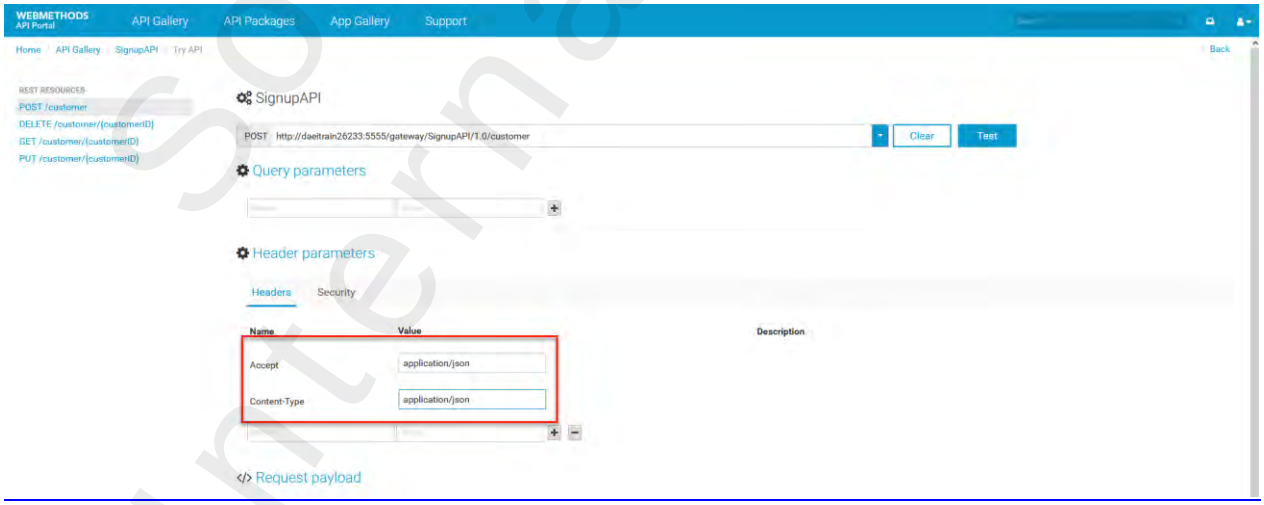



33. Navigate to the section **Try API >>** and select the REST Resource **POST/customer**.




34. Provide header parameters. Open the section **Header Parameters** and make sure that values for Accept and for Content-Type are set
    a. **Accept**: application/json
    b. **Content-Type**: application/json



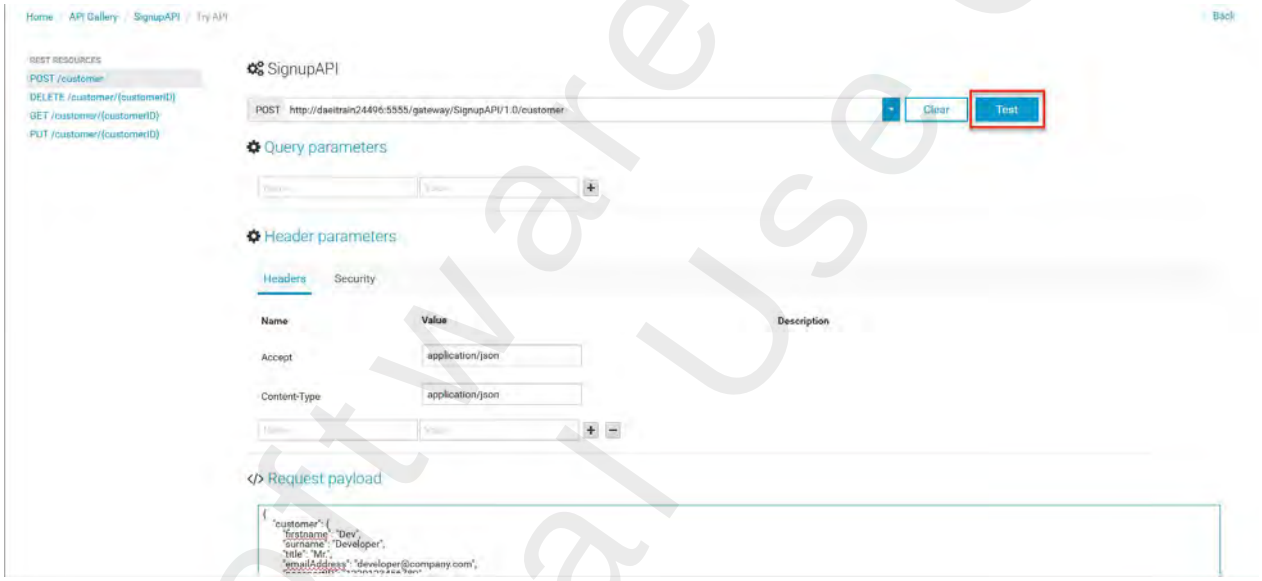

35. Replace the values in **the Request Payload** with dummy-data:
    a. **customer**
        i. **firstname**: Dev
        ii. **surname**: Developer

iii. **title**: Mr

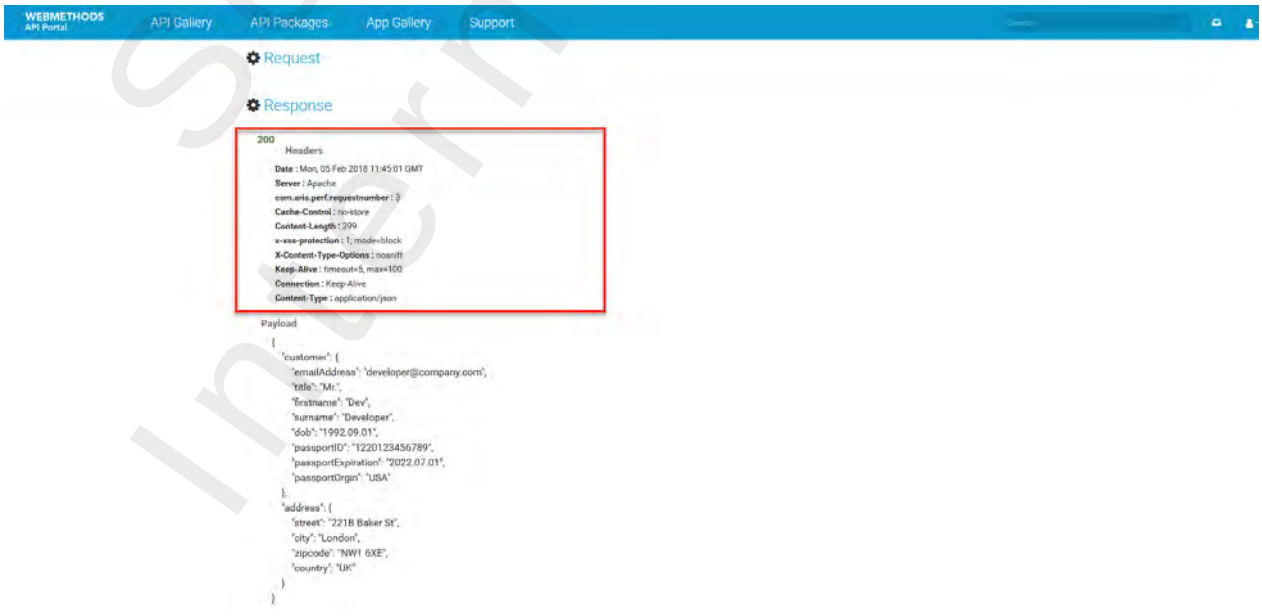
iv. **emailAddress**: developer@company.com

b. leave all the other values


36. Click the **Test** button.




37. Review the **Response** and verify that the Response Code is 200.



38. We skip the step to verify that the customer has been added using the REST Resource **GET /customer/{customerID}** from the navigation section. You have done this in one of the former exercises using RESTClient.

This page intentionally left blank

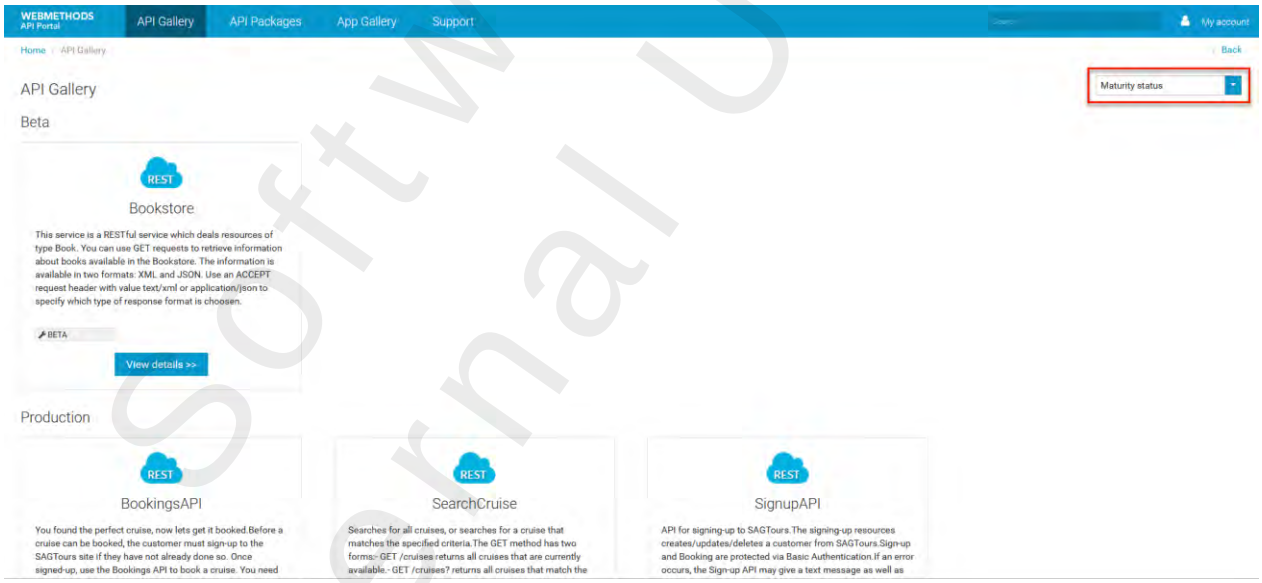
# Exercise 18:
# Using API Portal as Consumer

## Objectives

In this exercise, you will learn how to use API Portal as consumer.

## Steps

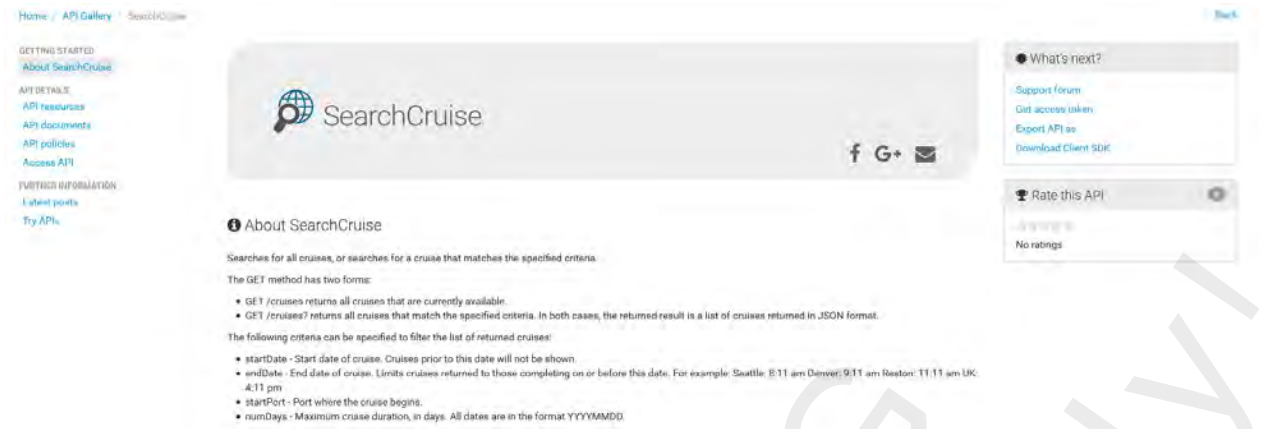
1. Open Windows Services UI to double check that the following services are up and running. If service is not running, start the service.
   a. **hMailServer**
   b. **Software AG Runtime 10.1**
   c. **Software AG Integration Server 10.1**
   d. **Software AG Event Data store 10.1 – should be started together with IS**
   e. **Software AG API-Portal 10.1**
2. Use the **API Portal Cloud Controller** to verify that **API Portal** is running. Use the **list** command to verify that all services are up and running.
3. In your **Firefox Browser** create a new Windows as **New Private Window**
4. Open the **API Portal** and use the **API Gallery** to see all APIs provided by API Gateway. In case you are still logged as user **Andy**, logout.
5. Change the listing of the APIs by changing the ordering which is per default ordering by **API Group**. Click the **Grouped by** link and you will get a list of predefined grouping criteria. From the list select the value **Maturity status**.



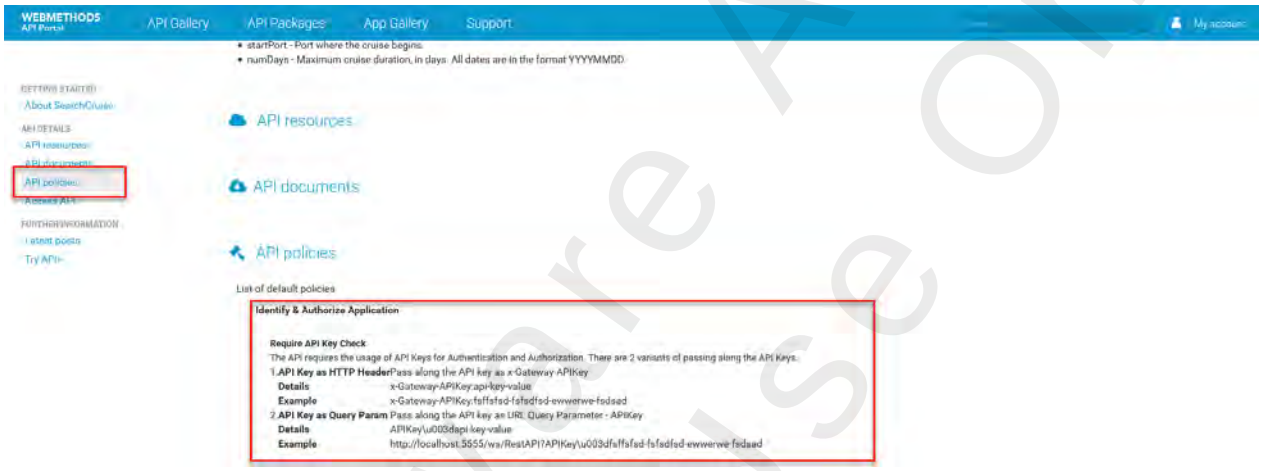

6. Select **SearchCruise,** click **View Details**. You will find the same set of information as you already have seen in the last exercise. The **About SearchCruise** informs about the procedure how to use the **SearchCruise API** and which search options are provided.



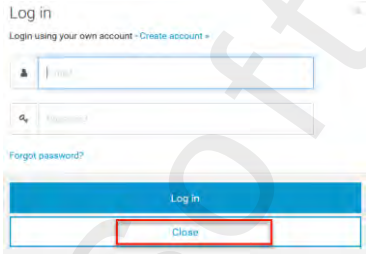
 



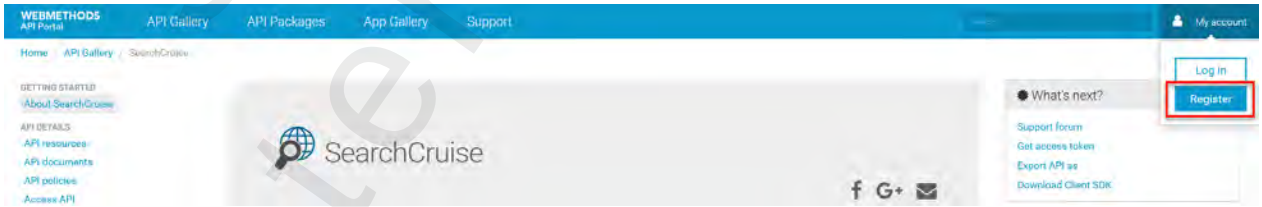

7. Within the **SearchCruise API** page click **API policies**. You will find out, that the API requires the usage of API Keys for Authentication and Authorization. We have done this configuration in Exercise 9.




8. Within the **SearchCruise API** page click **Try API>>** available within **Further Information**. A **Log in** page will pop up. In order to use APIs which are API Key protected you must have your own account. Right now you don't have an account. Therefore, click **Close**.



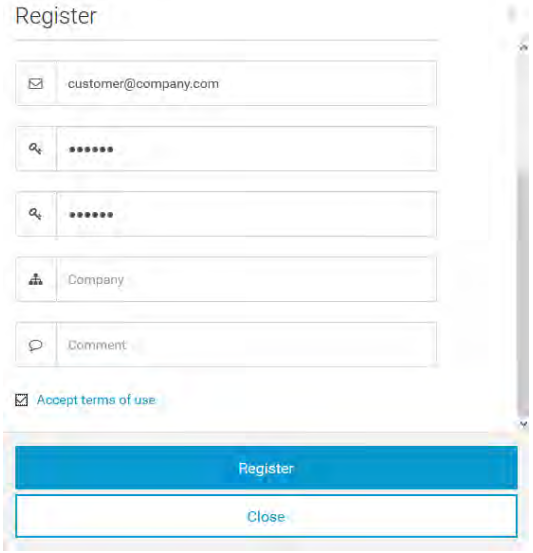

9. In order to request an **API Key**, the guest user needs to be registered to the **API Portal** . Navigate to **My account** in the header line.




Click the **Register** button.

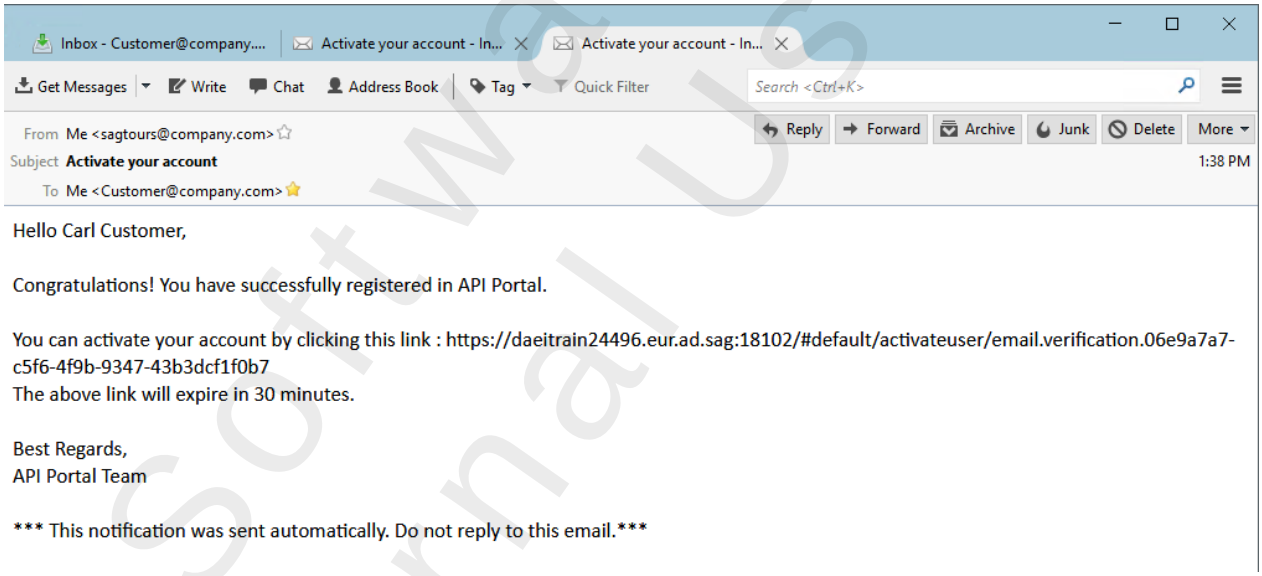
10. Make sure that **hMailServer** is running. If not running start **hMailServer**.
11. The popup will ask you for user details in order to create a new account for the **API-Portal**. Provide the following information:
    a. **First Name**: Carl
    b. **Last Name**: Customer
    c. **Email**: customer@company.com

d. **Password**: manage
e. **Confirm Password**: manage
f. **Company**: - leave empty-
g. **Comment**: - leave empty-
h. **Accept terms and conditions of site**: - select –




Click **Register**. Click **Close.**

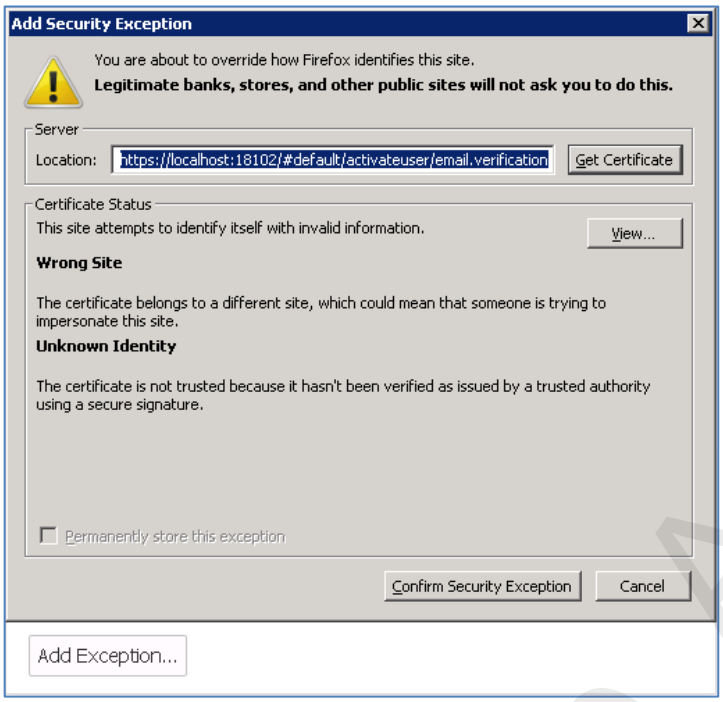
12. Open **Mozilla Thunderbird** and verify that an email has been sent by **API-Portal** to user **customer@company.com**, which says that the user account in **API-Portal** has been activated.



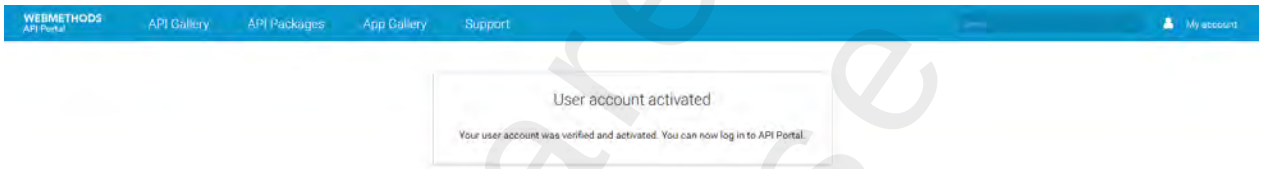

13. Copy the link in your email notification, open a new Browser Window as **New Private Window** and paste the URL. Agree that you understand the Risks.

Click **Advanced.**
Click **Add exception**

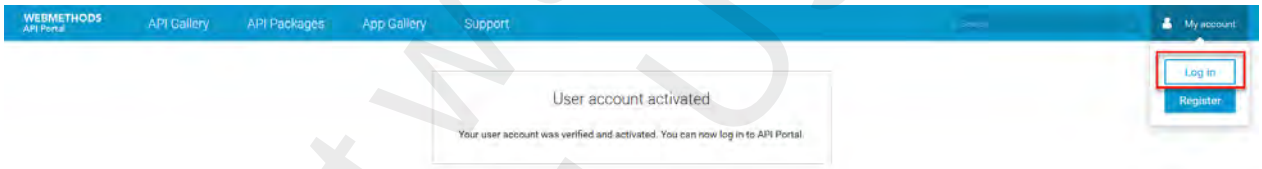
14. Click the button to **Confirm Security Exception**.




15. Now your user account is activated.

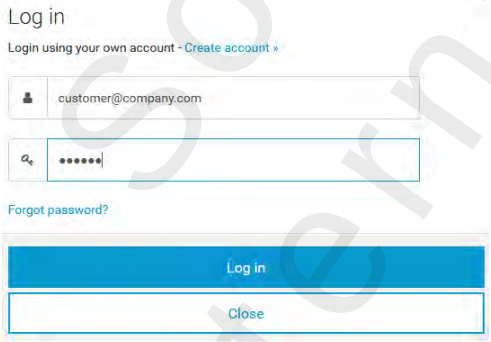



16. Use the **Log in** button to log in to **API-Portal** with your user credentials.

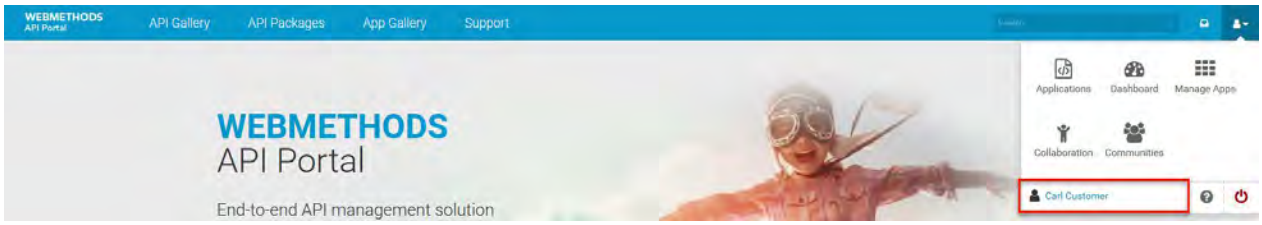



Use:

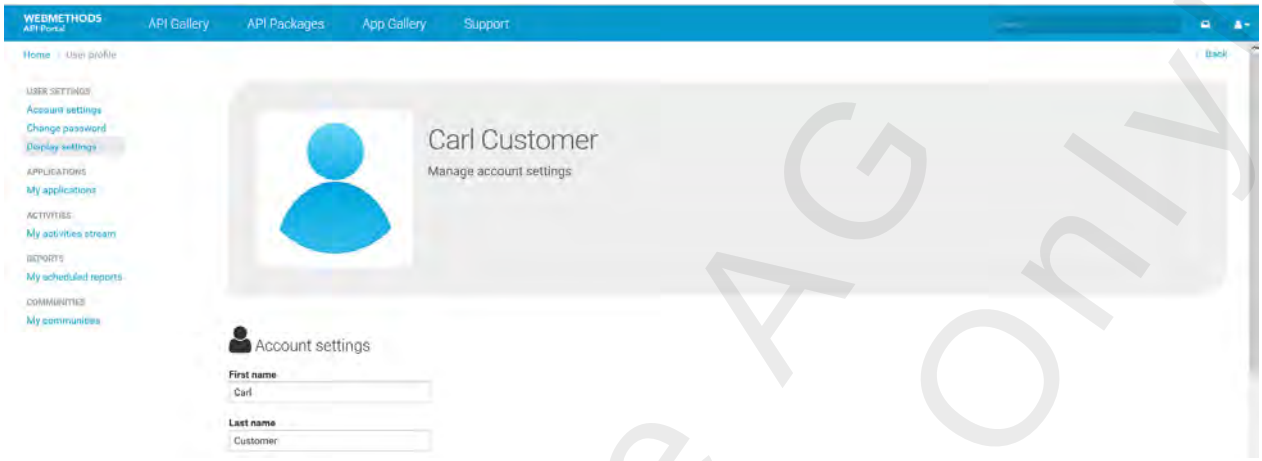
a. **Email**: customer@company.com
b. **Password**: manage




17. After successful login the API-Portal header line looks different than it does for the unknown user. You can review your profile etc. clicking the icon on the very right hand side in the header.

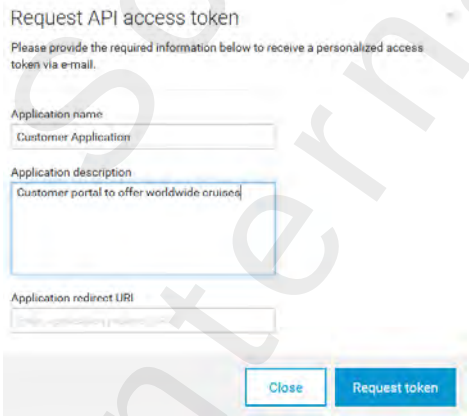



18. You can review your profile by clicking the user name link.



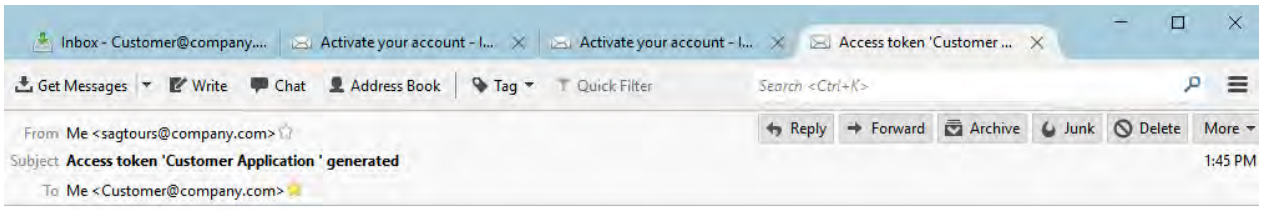

19. Search and open the **SearchCruise API**. Navigate to **What's next?** > **Get access token.**


20. **In the Request API access token** dialog provide the following details.
    a. **Application Name**: Customer Application
    b. **Application Description**: Customer portal to offer worldwide cruises
    c. **Application redirect URI**: - leave empty -




Click the **Request Token** button.

21. You will get a Response Message that the Access Token has been requested successfully.
22. Use **Mozilla Thunderbird** and verify that an email has been sent by **API Gateway** which references the generated API token.




Hello Carl Customer,

Your request for an access token has been processed. The details of the token is as follows

Application name: Customer Application
Application description: Customer portal to offer worldwide cruises
APIs: SearchCruise

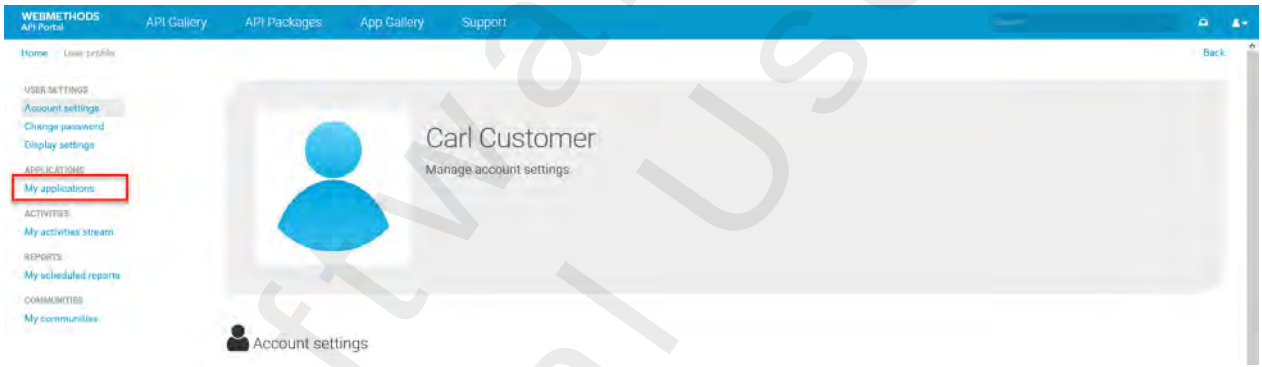
API Key
API access key: 6624d207-f411-441b-8176-c2a1e580feeb
Expiry date: Unlimited

OAuth2 Credentials
Client ID: 2e103f4d-dadb-4d18-93a1-7942bdde163c
Client secret: 173a240b-09d7-4f21-9629-2cf14deb5498

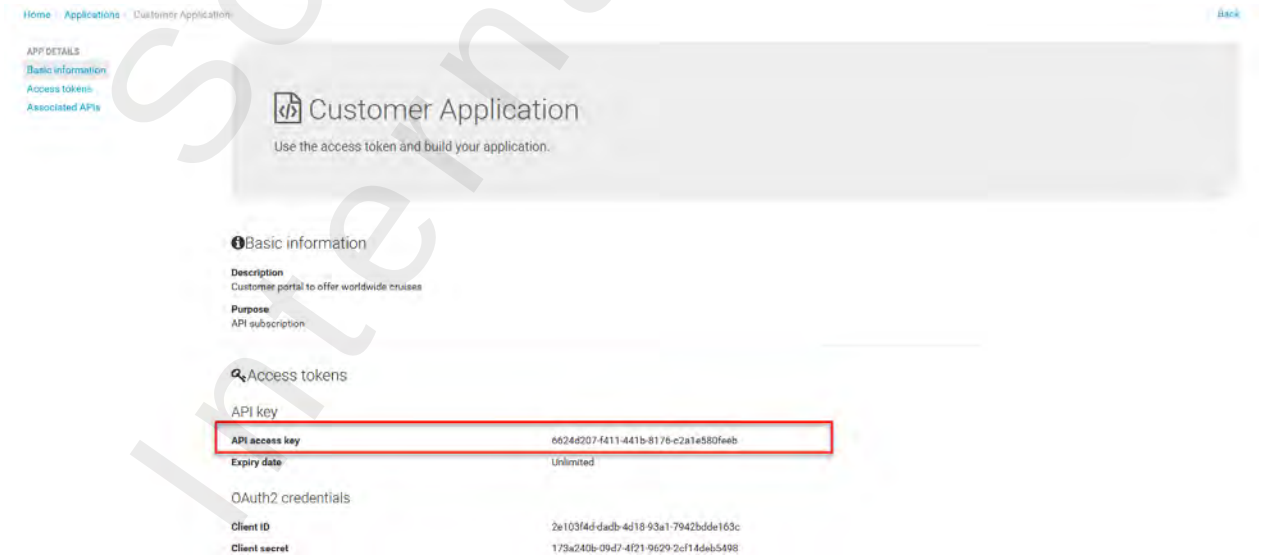
Best Regards,
API Portal Team

*** This notification was sent automatically. Do not reply to this email.***

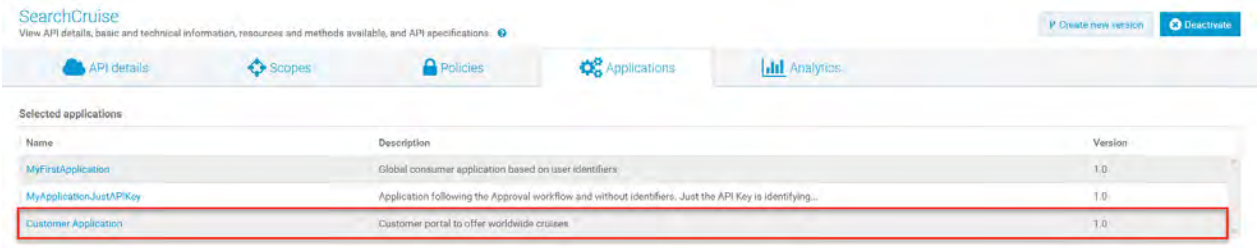
23. The generated access token is also available in your **API Portal** user account. Open **Carl Customer -> Application > My applications**.



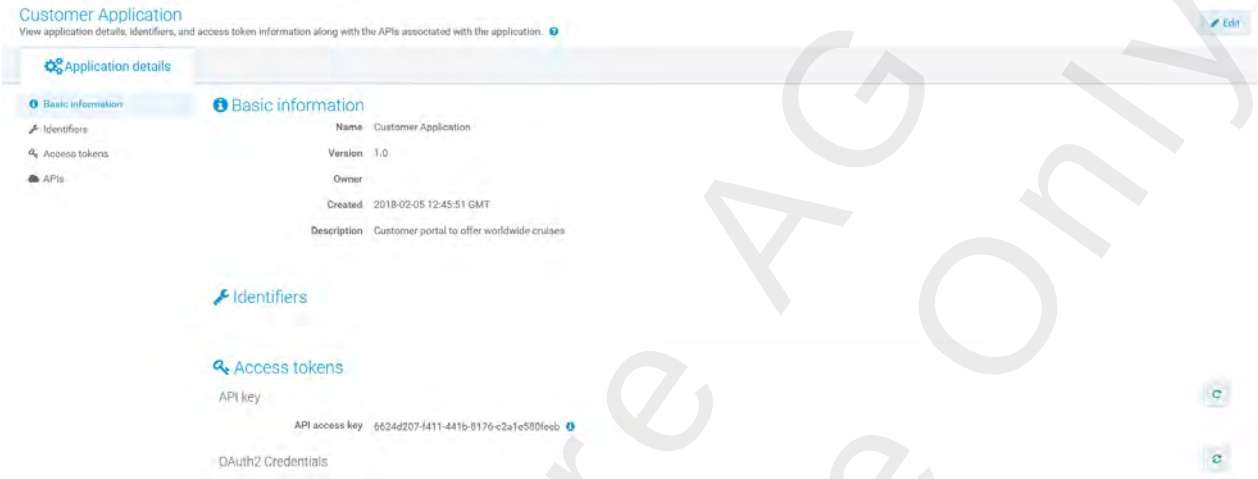

a. From the list of applications select the new created application **Customer Application.** Review the generated **Access Token**.



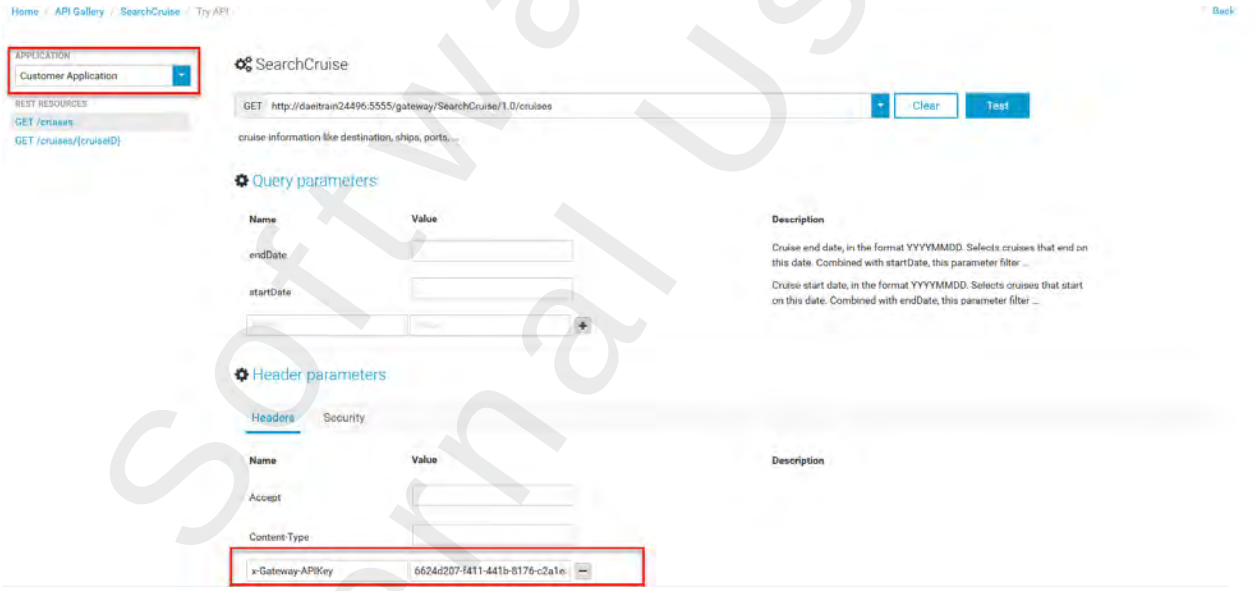

24. Go back to **API Gateway**. Login as User **Administrator/manage**. Verify that a new application has been added to **SearchCruise**. Navigate to **APIs > SearchCruise > Applications**.



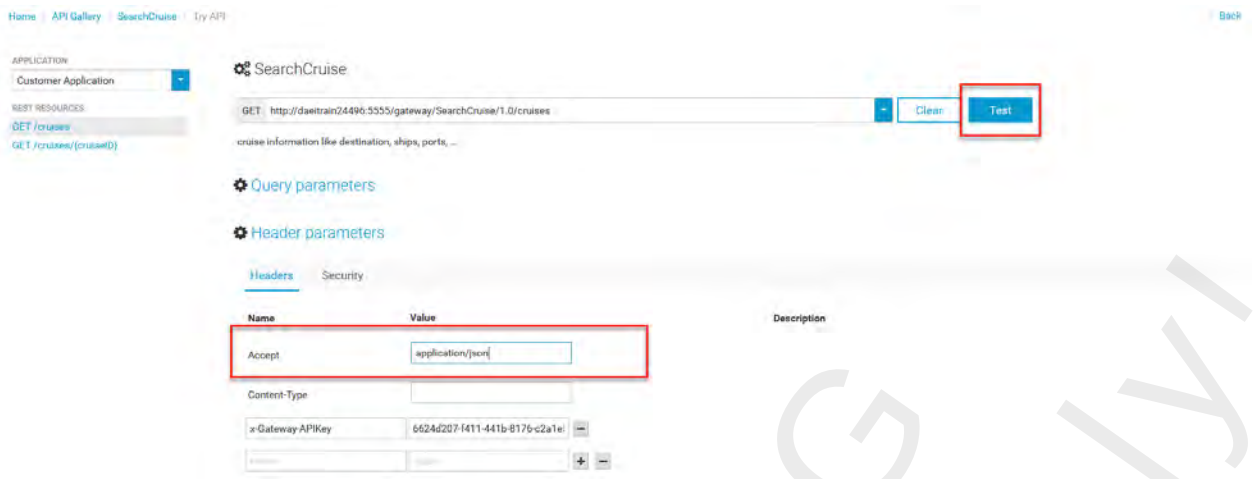

25. Select the application **Customer Application**. Verify that the API Key is listed and **SearchCruise** is attached.




26. Go back to **API Portal** with credentials **customer@company.com/manage**. Search for **SearcCruise API** and navigate to **Further Information > Try API>>**. The new created Application is listed for the **SearchCruise.**

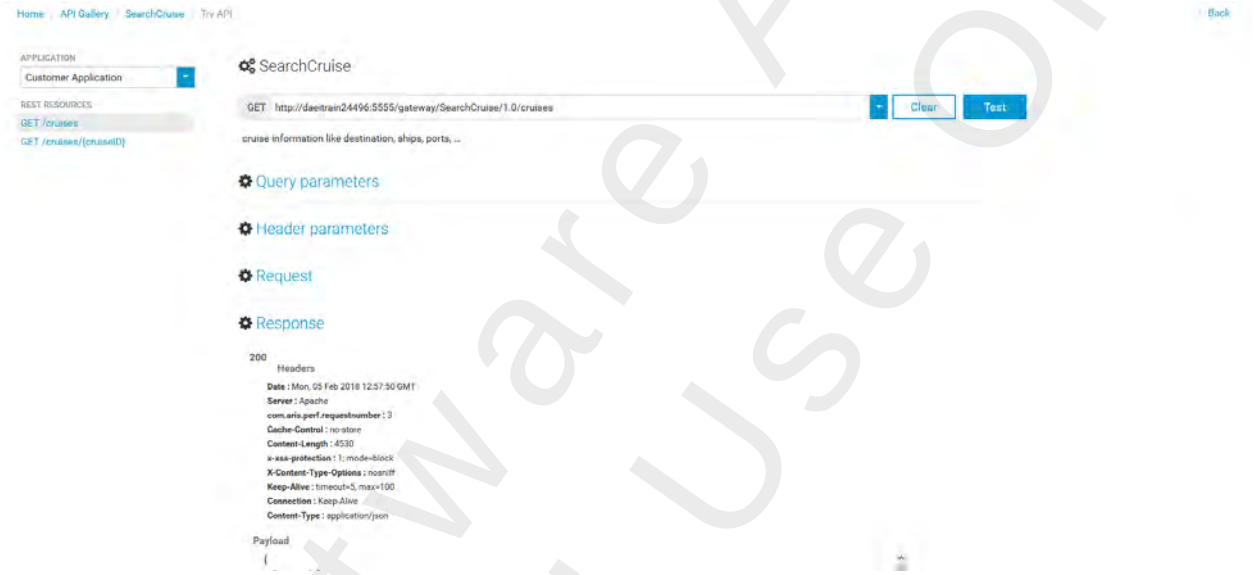



27. Select the resource **GET /cruises**. Run API with the default Application named **Customer Application**.
    a. Verify that the Header Variable is set :
        i. Accept application/json




Click **Test**.

b. Review the **Response.**




c. Rerun the request several times in order to get **API Gateway** to collect some performance data.

d. Mark in the payload section one of the listed cruiseIDs for copy and paste.

Response

200 OK

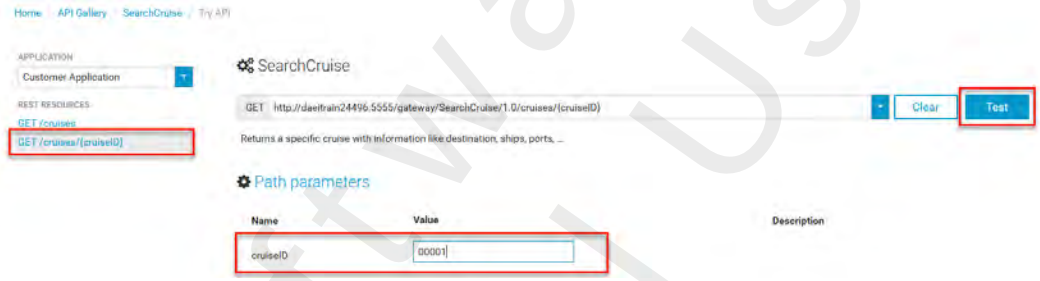
Headers

**Date :** Fri, 23 Dec 2016 14:33:04 GMT
**Server :** Apache
**Content-Length :** 4530
**Keep-Alive :** timeout=5, max=93
**Connection :** Keep-Alive
**Content-Type :** application/json;charset=UTF-8

Payload


```
{
  "cruises": [
    {
      "cruiseID": "00001",
      "title": "Alaska Whale Watching Photo Expedition",
      "description": "Photographying the Whale in Alaska is a unique experience.  Join us on this tour to this unique area.  The cruis
e is a luxury tour for this that what to spend time with others with a like for see these unique animals of the sea.  The tour will be cond
ucted by a pro photography who will work with each of the members of the tour as needed.",
      "startPort": "Juneau, Alaska",
      "numDays": "8",
      "startDate": "20150820",
      "endDate": "20150827",
      "roomTypes": [
        {
          "roomID": "IPD",
          "roomDetails": {
            "roomType": "Inside Premier Double",
            "title": "Inside Premier Double cabin",
            "description": "An Eco-friendly double room with desk, Tv and Shower",
```

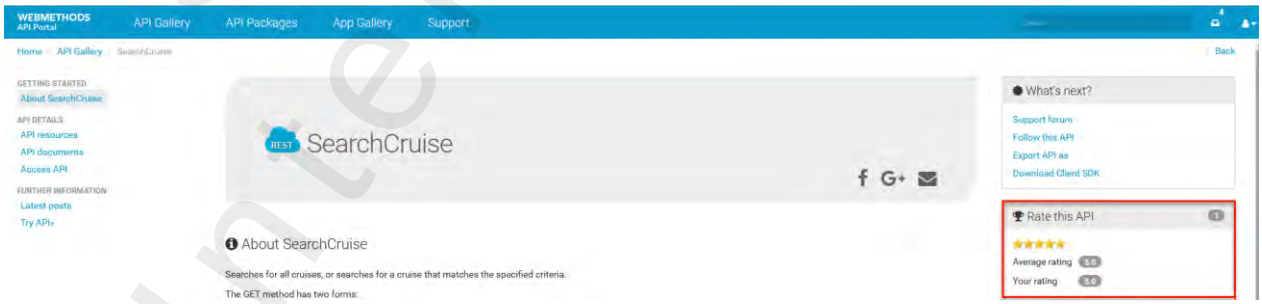

28. Navigate to test the resource **GET /cruises/{cruiseID}**. Provide all values and run the request to get 1 specific cruise




Run the request using the **Test** button and verify the result.

29. Use the **Back** link to go back to the landing page for the **SearchCruise API**.


30. Rate the API by selecting as many of the stars as you think it is.




31. Click the **Collaboration** link in the user menu to communicate with others.


Review the functionality of the collaboration inside the **API-Portal**.


32. Logout and review the ratings of the **SearchCruise API** as guest user. Open the **Menu** using the icon on the very right hand side. Within the menu, select the **Logout** icon in the lower left hand corner.

# Exercise 19:
# Send Events to API Portal

## Objectives

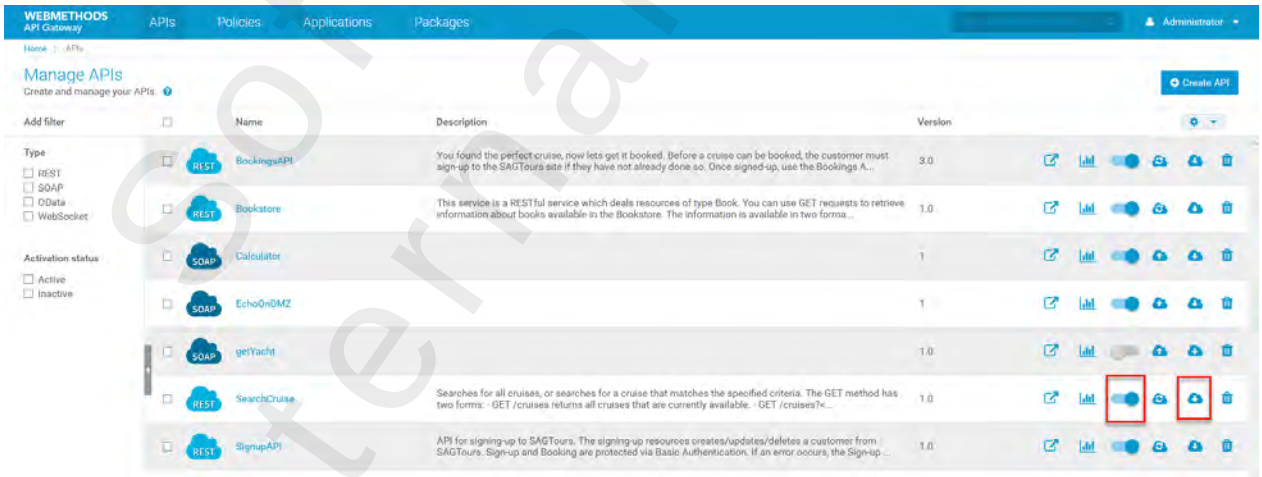
Scenario:

Login to API-Portal as API Administrator and try the API-Portal dashboard functionality to monitor the usage of API Portal. To monitor the runtime behavior of the API use API Gateway to review the performance profile of the API.

## Steps

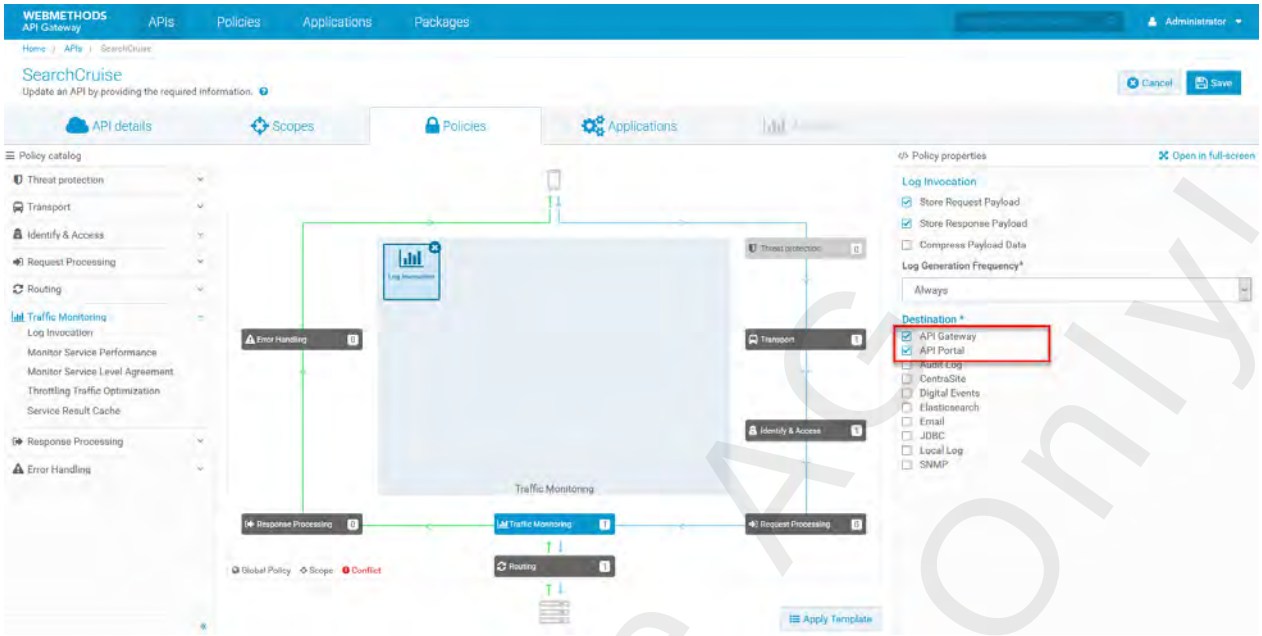
1. Open Windows Services UI to double check that the following services are up and running. If service is not running, start the service.
   a. **Software AG Runtime 10.1**
   b. **Software AG Integration Server 10.1**
   c. **Software AG Event Data store 10.1 – should be started together with IS**
   d. **Software AG API-Portal 10.1**
   e. **hMail-Server**
2. Use **API Cloud Controller** to verify that **API Portal** is up and running.
3. Logon to **API Gateway** as user **Administrator/manage.** In case the global policy **Transaction logging** is active, use the slider to deactivate the policy.


4. Navigate to APIs. Unpublish and deactivate the **SearchCruise API.**



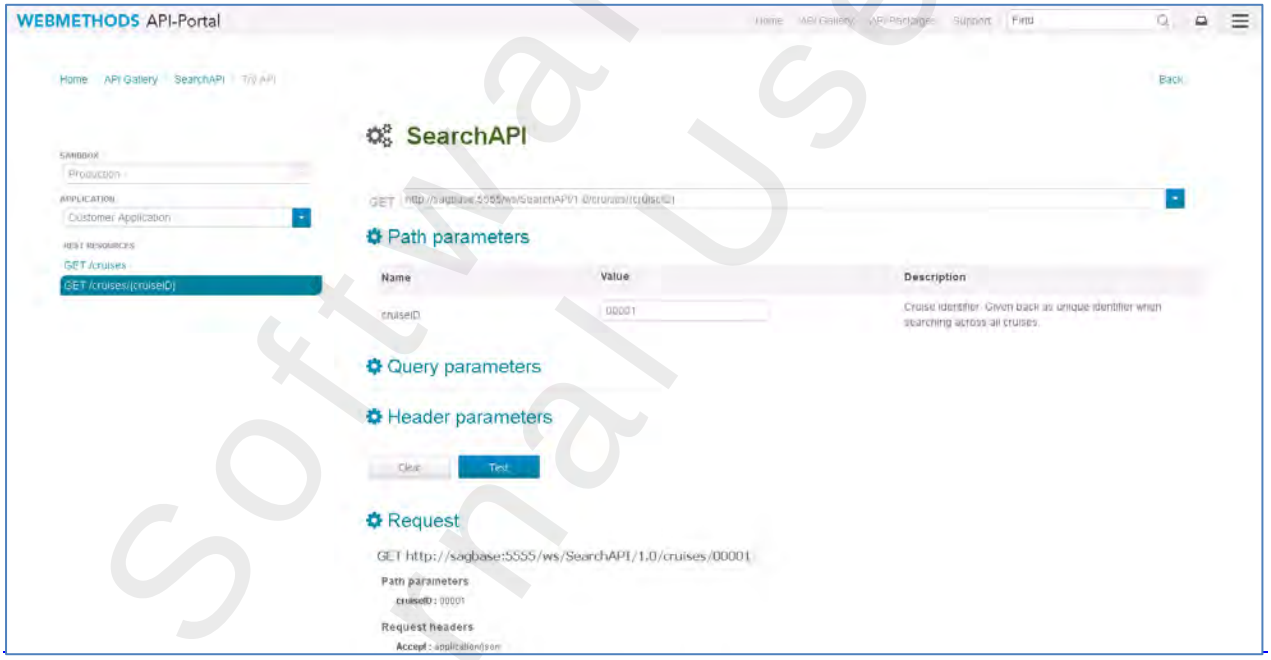

5. Switch to Edit mode. Navigate to **Policy Enforcement > Logging and Monitoring.** In case not already added to the API, select the policy **Log Invocation.**
   a. Provide the following details:
      i. **Payloads**
         1. **Request** and **Response**
      ii. **Send Log Data**
         **1. API Gateway**

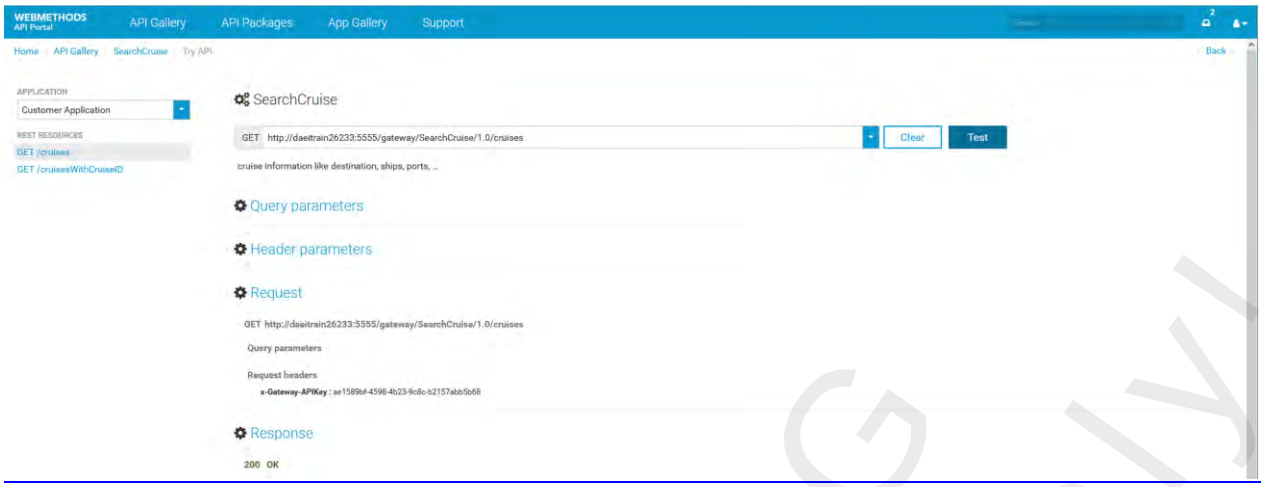
2. **API Portal**




Click **Save.** Activate and publish **SearchCruise** to **API Portal.**

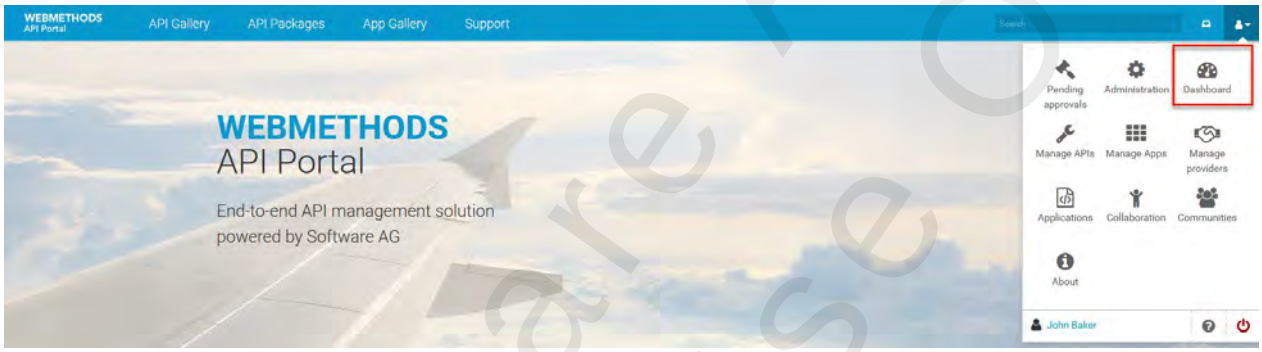
6. Login to **API Portal** as user **customer@company.com/manage** and run **SearchCruise** several times, with and without search or path parameter.





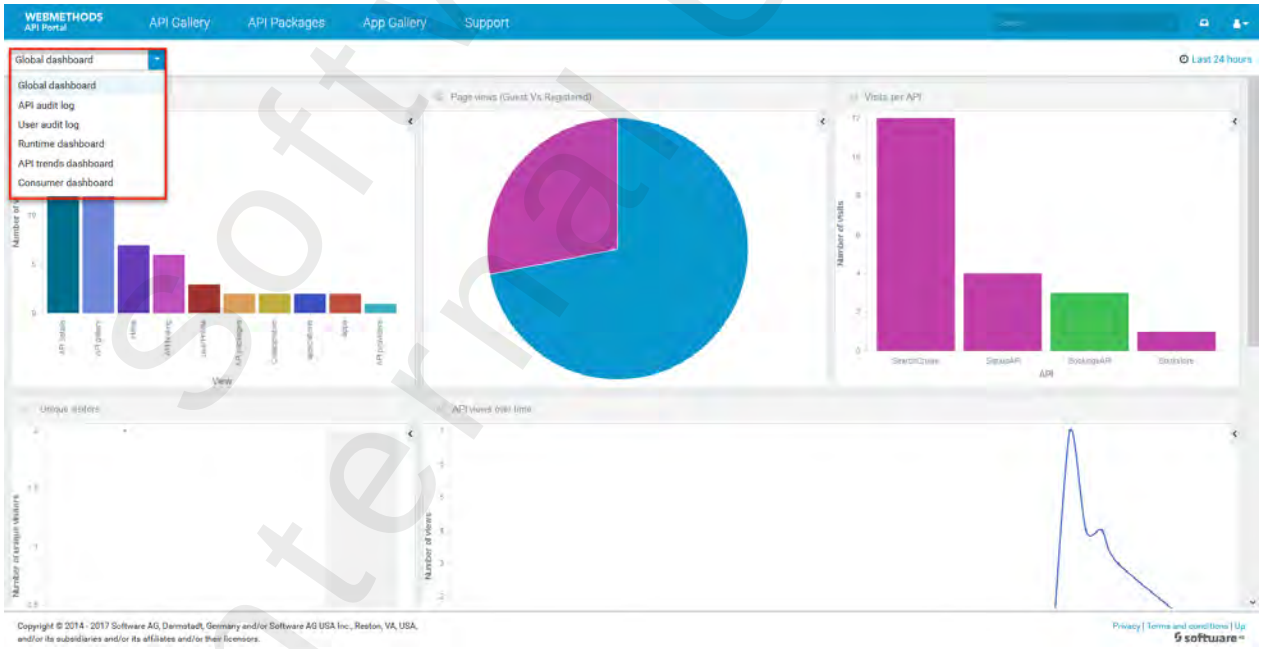

Logout from **API-Portal**.

7. Login to **API-Portal** as user **john / manage**. Navigate to the User menu. **John** has additional entries in the user menu.

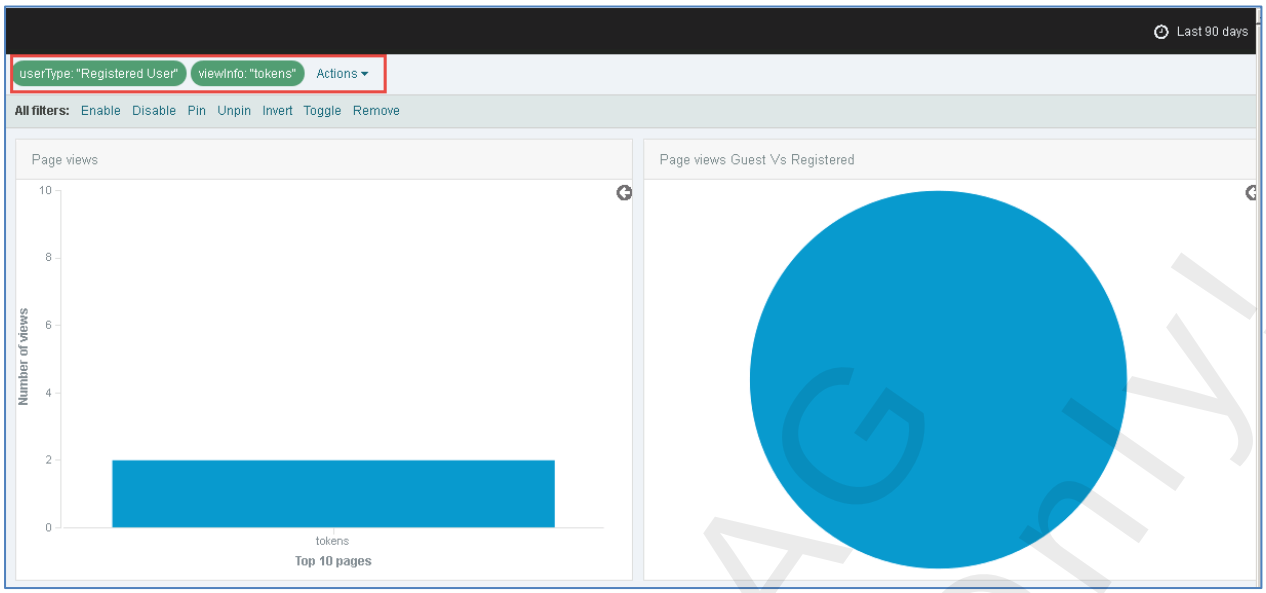



Click **Dashboards**.

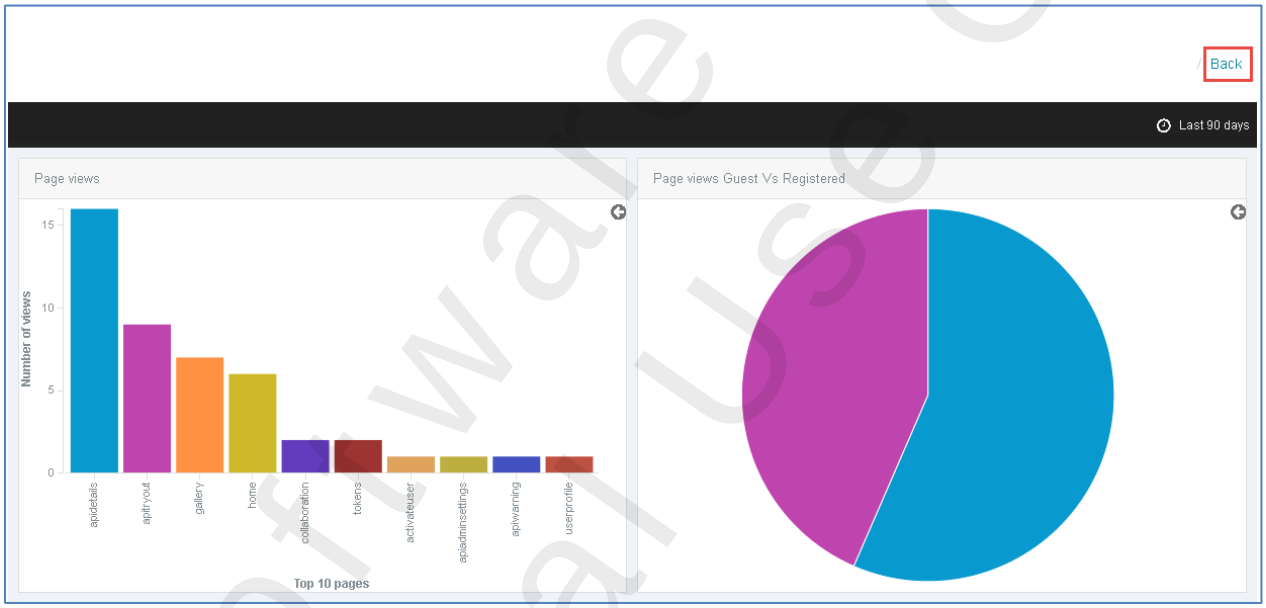
8. On the **Dashboards** page you can use different groups of dashboards. The default dashboard **Global dashboard** is a collection of 6 different dashboards.



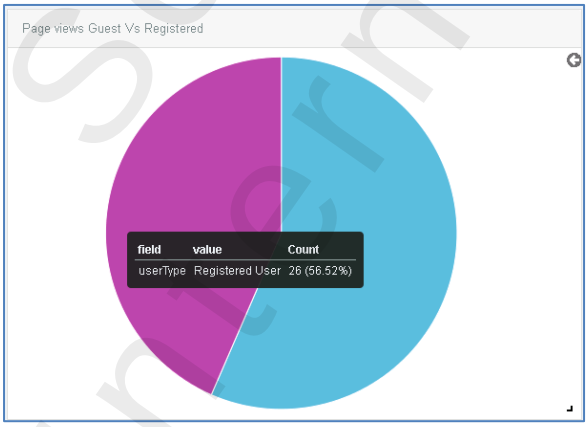

9. Hover the mouse over one of dashboards.
   a. Within Page views this will show how often this page is viewed.
   b. Click the page **API details.** This will drill down to just this page.



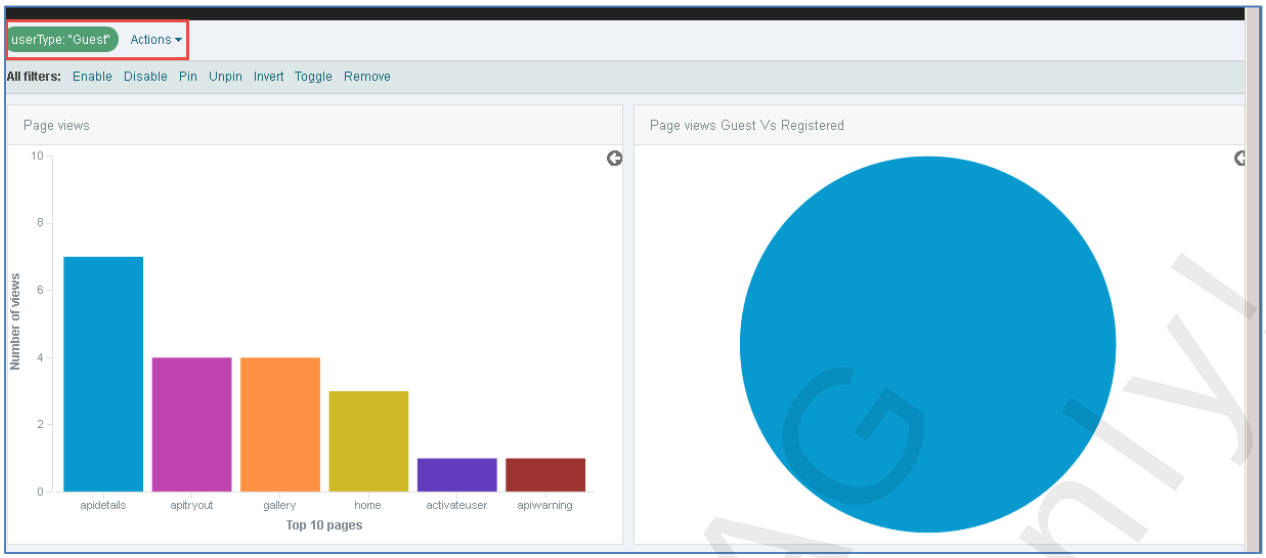

c. Navigate to initial Configuration of **Global dashboard**. Now select **Page views (Guest vs. Registered)**




d. Hover the mouse over one of the sections and review the details.

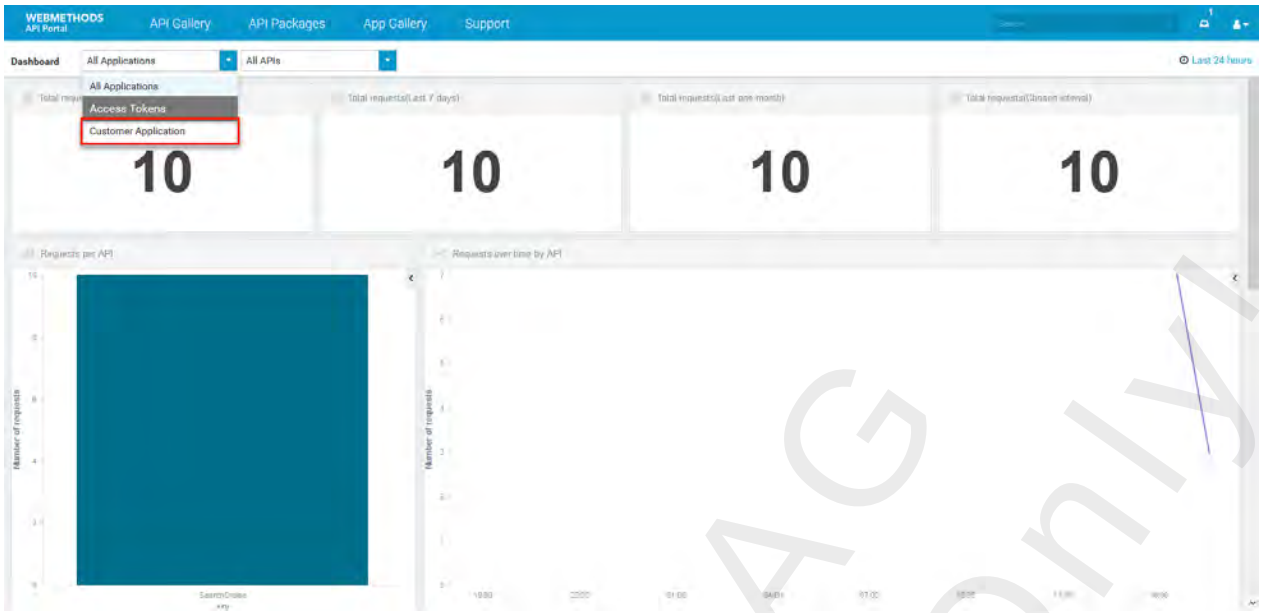



e. Click one of the sections. Only information on views is displayed, see header line navigation.



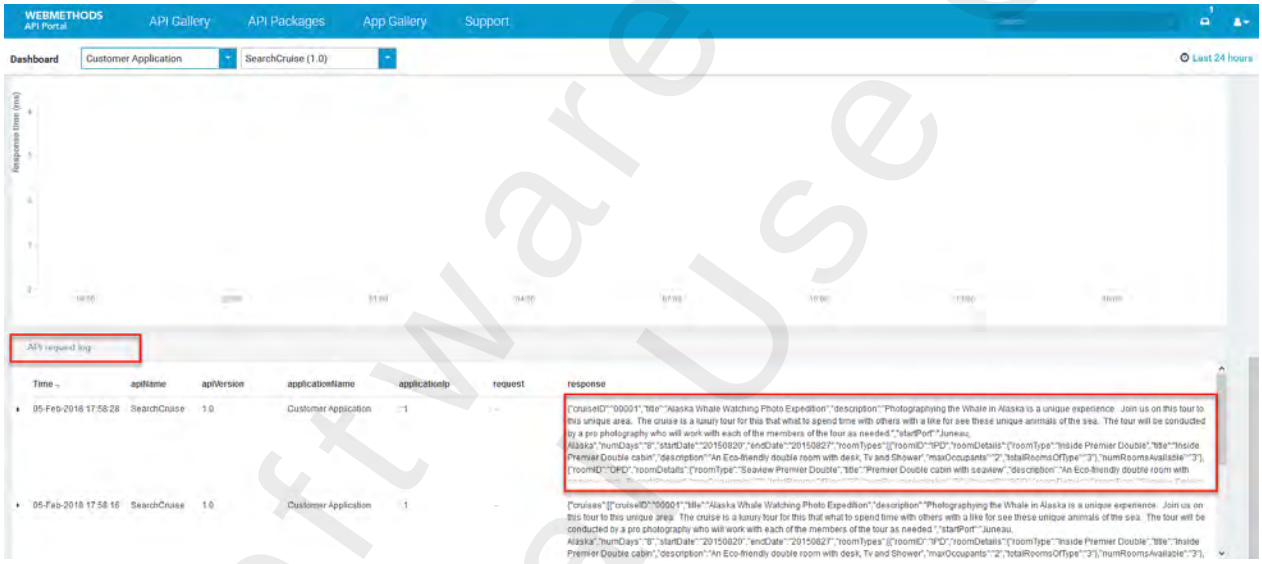

10. Navigate to **Runtime dashboard**. Scroll down to the **Runtime events** dashboard. You will find **PerformanceMetrics** information.


11. Logout and login to **API Portal** as user API Consumer **customer@company.com/manage** and review metrics. Navigate to **User menu > Dashboards.**

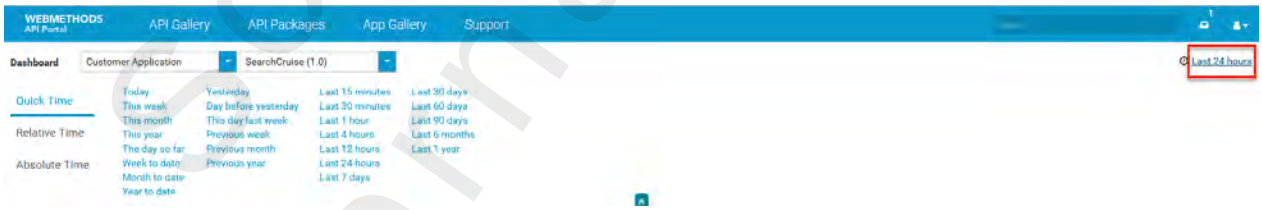



Select **Customer Application**.

12. Scroll down to **API request log**.




13. You can use the **Last 24 hours** link to change the time interval.



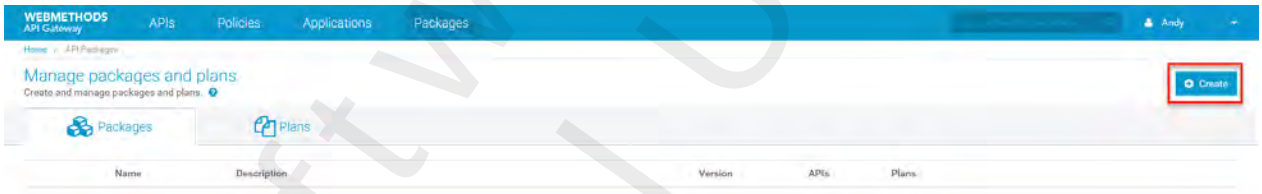
# Exercise 20:
# Managing API Packages and Plans

## Objectives

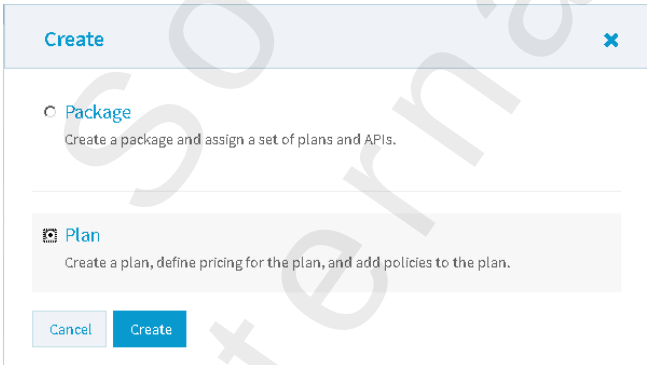
In this exercise, you will create a Package and a Plan to allow a set of Quality definitions like costs, availability of the service for a set of APIs. We would like to offer all APIs around the SAGTours application as a package.

## Steps

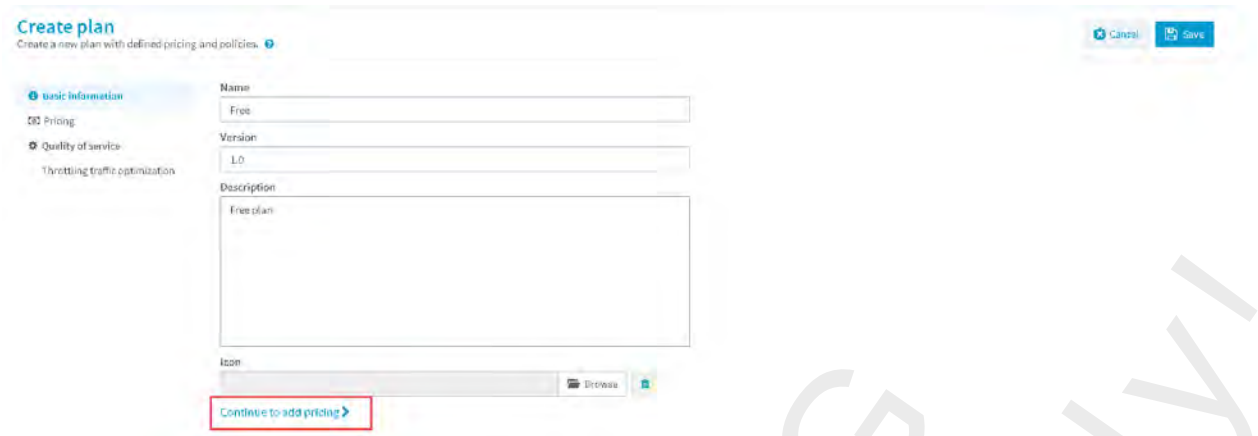
1. Open Windows Services UI to double check that the following services, needed API Gateway, are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Event Data store 10.1 – should be started together with IS**
   c. **Software AG API Portal**
2. Login to **API Gateway** as user **Andy/manage**.
3. When subscribing to a package an application is generated and access to the APIs is controlled via API Key. Therefore we can remove the existing **Identify & Access** policies we have defined on the **SAGTours APIs**.
   a. Deactivate and edit **SearchCruise**. Remove the policy **Identify & Access > Identify & Authorize Application**. Save the changes. Edit BookingsAPI and SignUpAPI as well and remove any Identify & Authorize policies that you may have.
4. Navigate to **APIs**. Activate and publish/republish **SearchCruise, BookingsAPI, SignUpAPI.**
5. Navigate to **Packages**. Click **Create.**




6. Select **Plan** and click **Create.**




7. Create a new Plan
   a. **Name**: Free
   b. **Version:** 1.0
   c. **Description:** Free plan

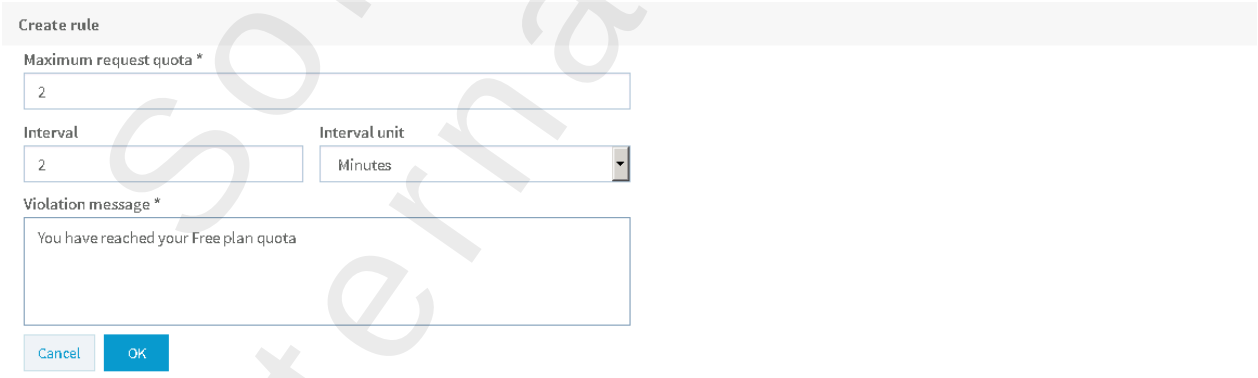



Click **Continue to add pricing**

d. **Pricing**
   i. **Costs:** 0
   ii. **Terms:** Terms
   iii. **License:** License

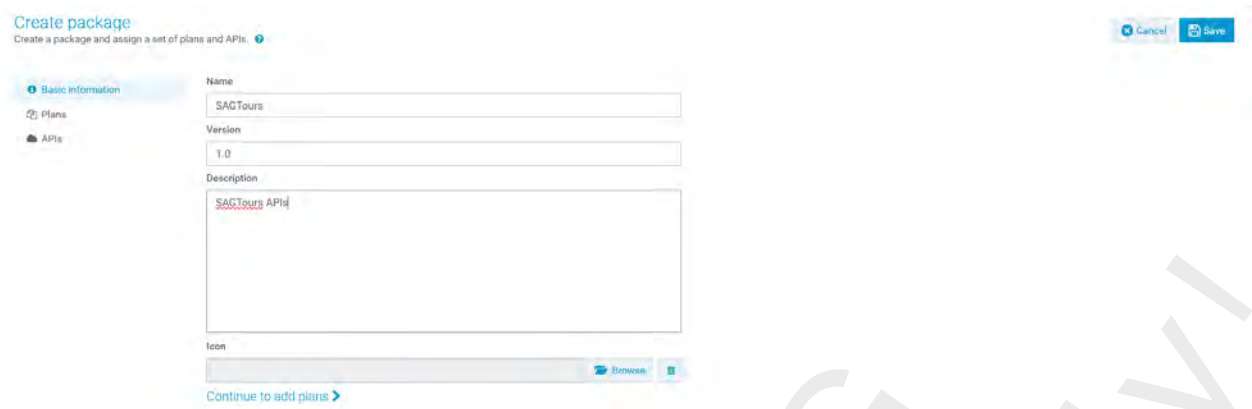

Click **Continue to Quality of Service**

e. **Quality of Service -Throttling Traffic optimization**
   i. **Maximum request quota:** 2
   ii. **Interval:** 1
   iii. **Interval unit:** Minutes
   iv. **Violation message:** You have reached your Free plan quota




Click **OK.** Click **Save.**

8. Navigate back to **Packages** and click **Create** to create a new **Package**
   a. **Name**: SAGTours
   b. **Version**: 1.0
   c. **Description**: SAGTours APIs

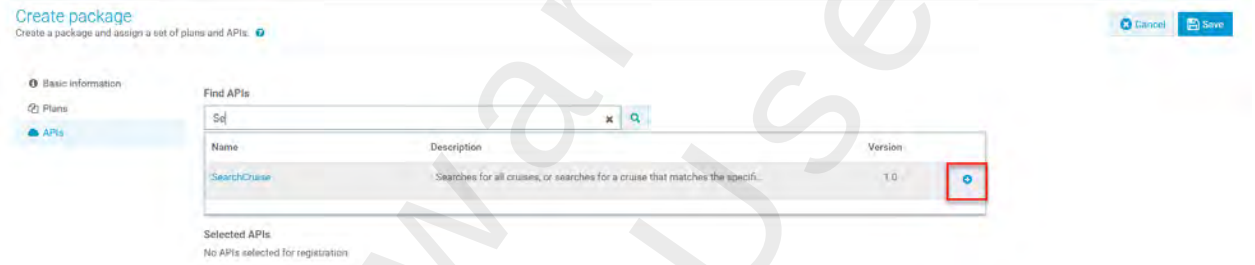



Click **Continue to add plans**:

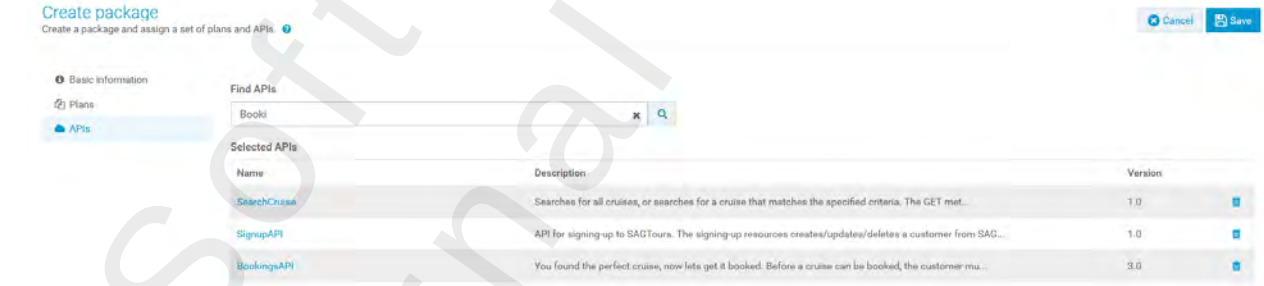
d. **Select plans**: Free


Click **Continue to add APIs>**

e. Search for **SearchCruise.** Click the API or the + sign to add the API.

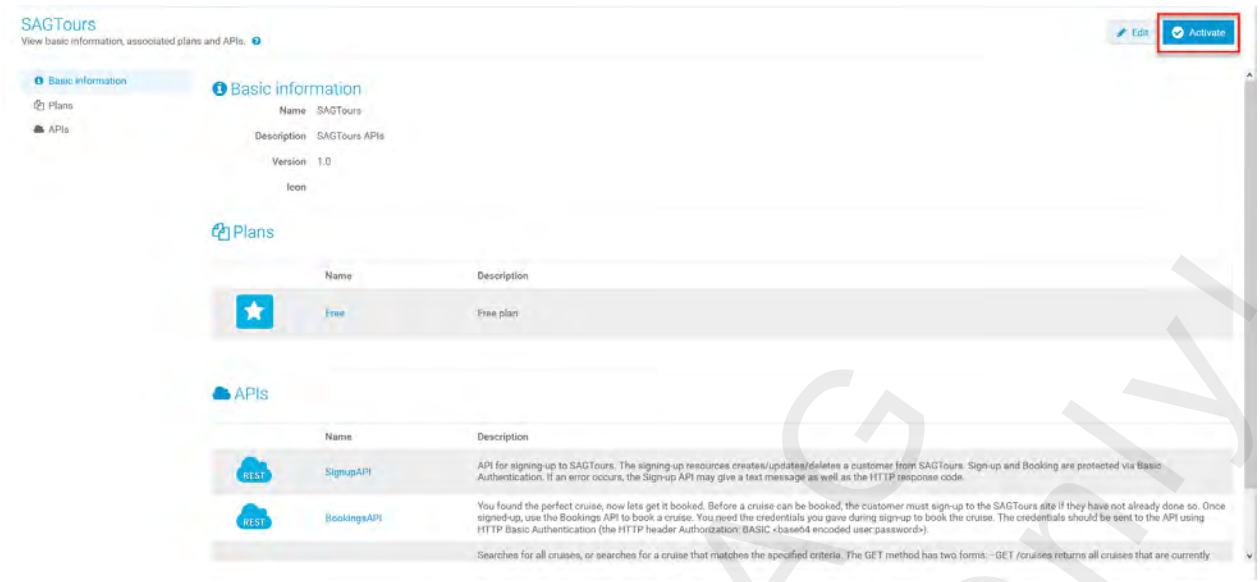



Add the APIs **SignupAPI** and **BookingsAPI.**



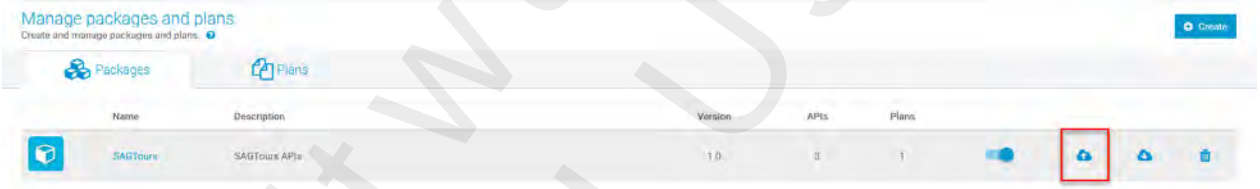

Click **Save.**

9. Click **Activate.** Click **Yes.**



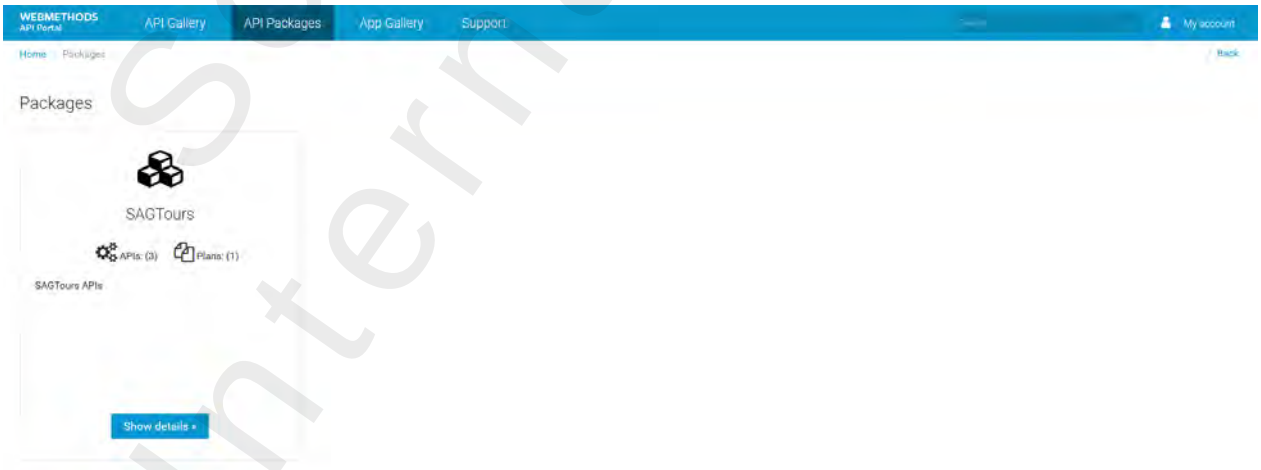

Review the just created Package.


10. In order to publish the new created Package we need to first publish the allocated APIs. This we have already done in one of the former exercises. Select the new created package **SAGTours** and publish the package**.**

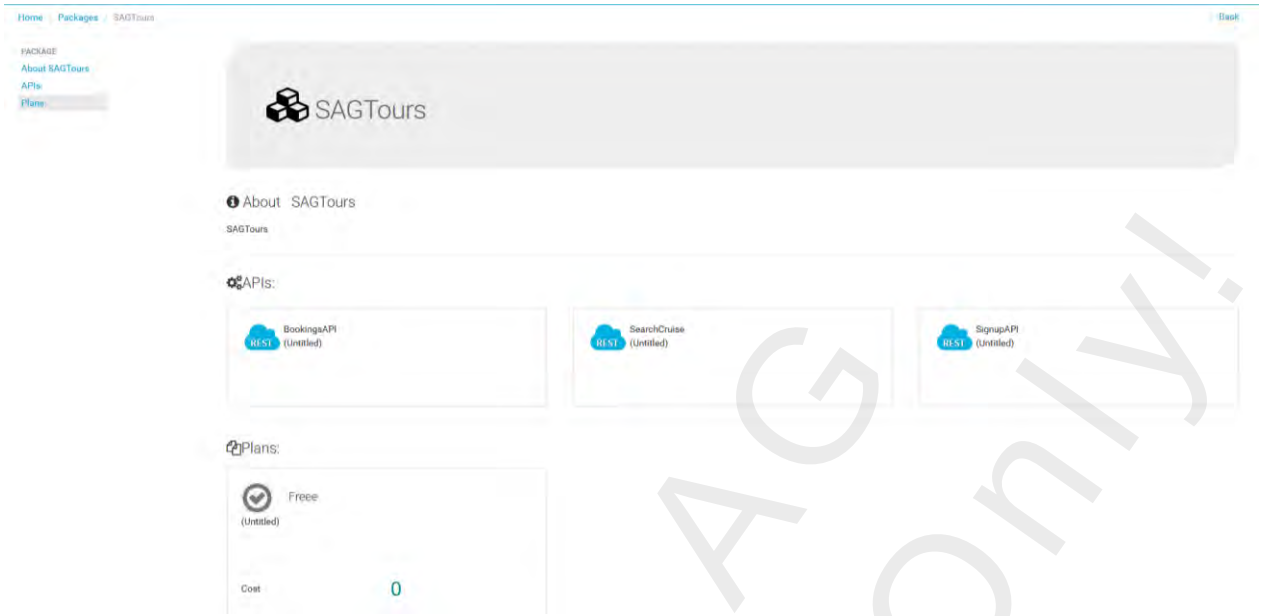



Click **Yes**.

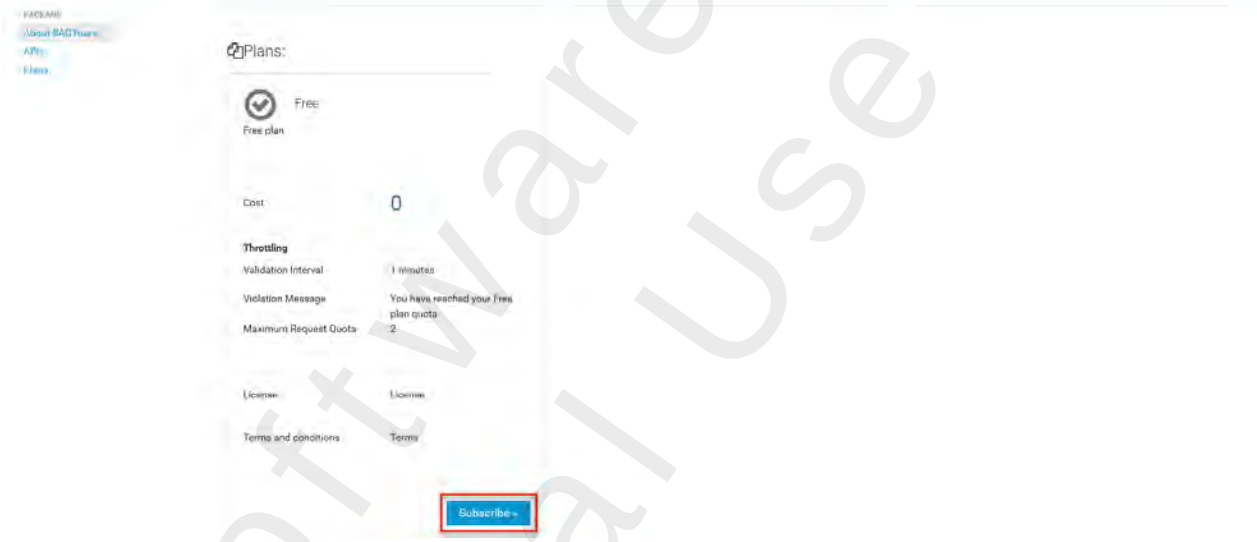
11. Open the **API Portal** as **New Private Window**
12. Navigate to **API Packages**



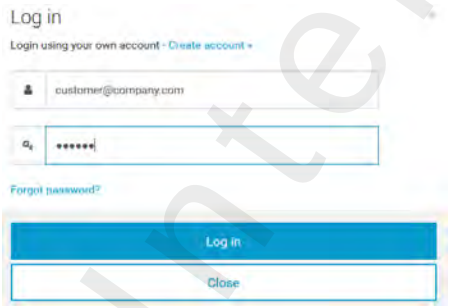

13. Select the package **SAGTours** and click **Show details.**
    Review. The listed **APIs** and **Plans** to be the same as configured in API Gateway.




Within the plan **Free** scroll down click **Subscribe.**



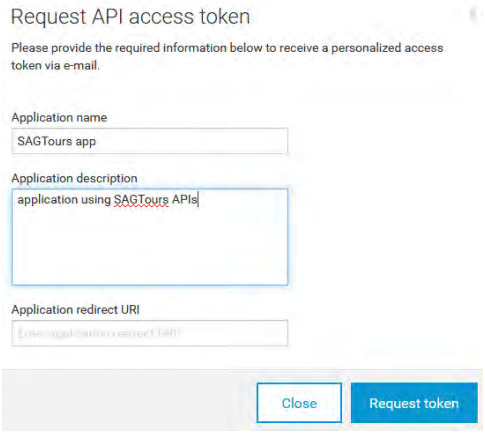

14. Subscribing to a plan creates an **API Access token** which requires to have a valid user account in **API-Portal**. A user account for user **customer@comapny.com** you have already created in **API-Portal**. On the screen **Log in** provider the following credentials
    a. **E-mail**: customer@company.com
    b. **Password**: manage




Click **Log in.**

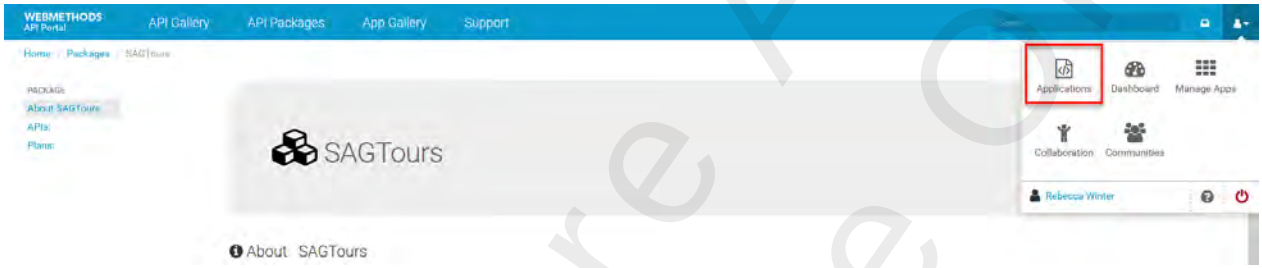
15. Click again **Subscribe**. On the screen **Request API access token**, provide the following Parameters.
    a. **Application name**: SAGTours app

b. **Application description**: application using SAGTours APIs




Click **Request token.** Click **Close.**

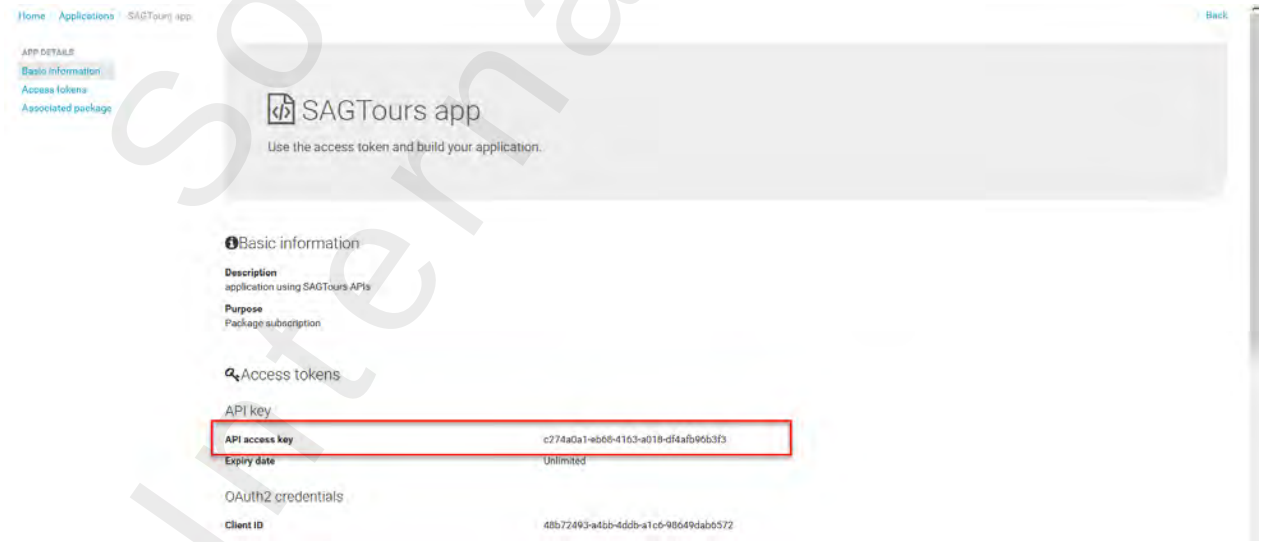
16. Open the user's menu and select **Applications.**



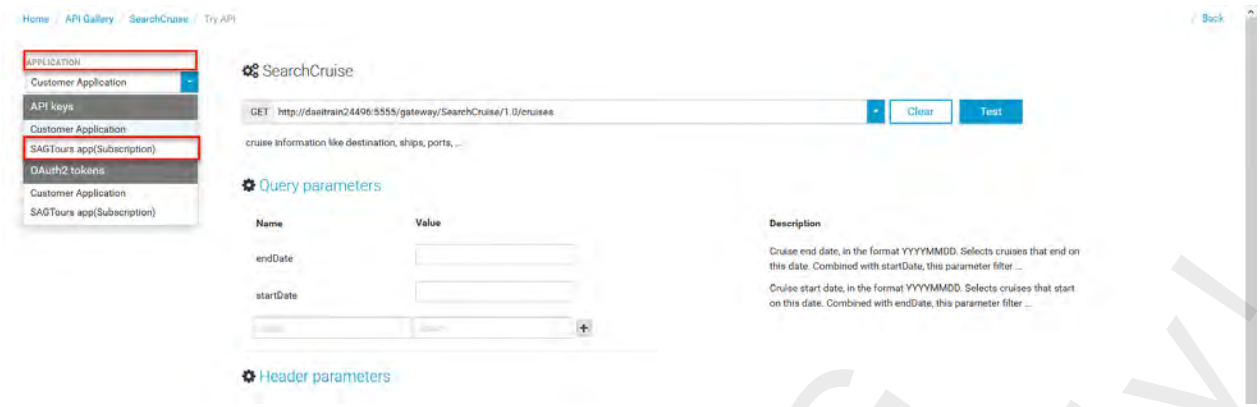

17. You will find the just created application in the list of **Applications**.


Select the application **SAGTours** app.

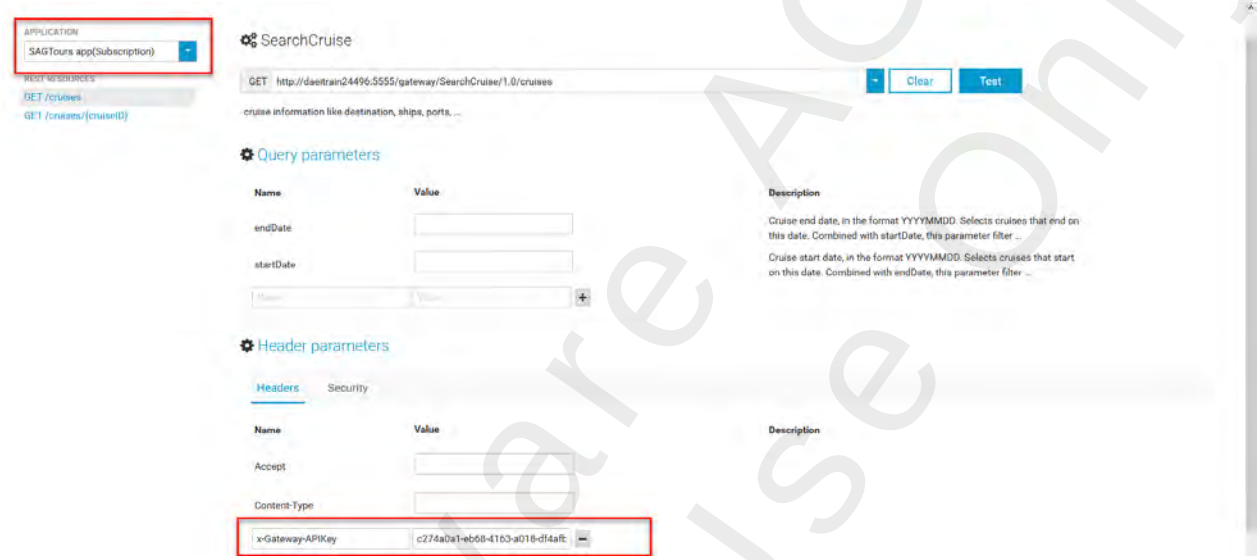
18. The generated **Access tokens** is listed.



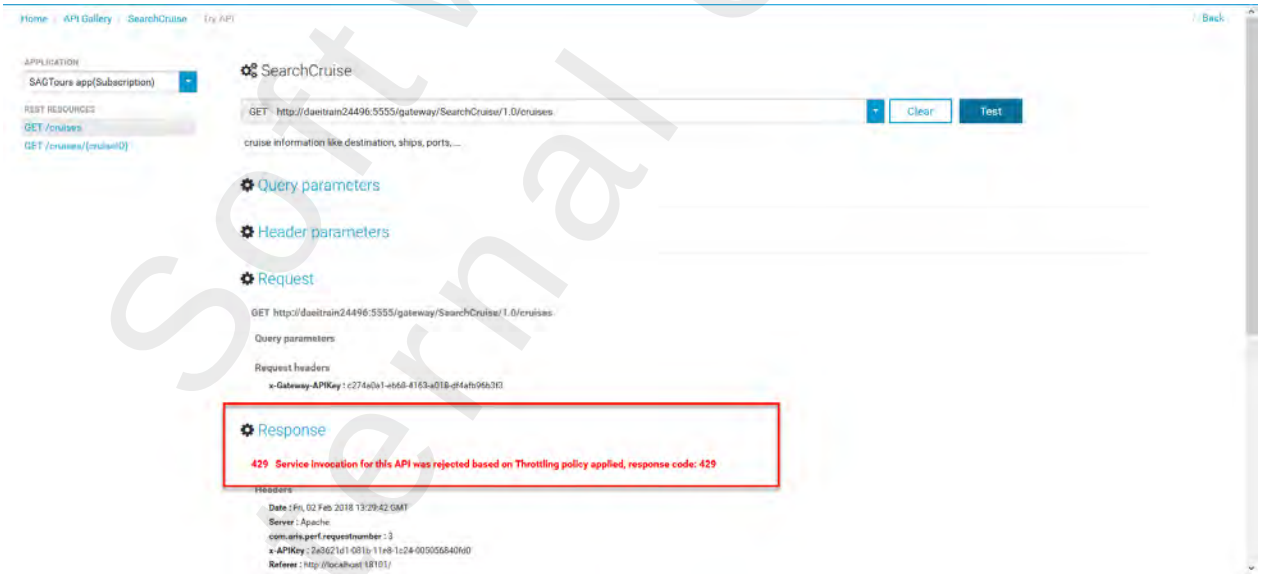

19. Navigate to **Associated packages**. Select the package **SAGTours.** Select the API **SearchCruise**. This will open the API in Detail view. Navigate to **Try API>>.** Within **Application** you find the API Keys and OAuth2 tokens for **SAGTours a**pp and **Customer Application**, which we have created already.




Select **SAGTours app (Subscription)** with the list of **API keys**.

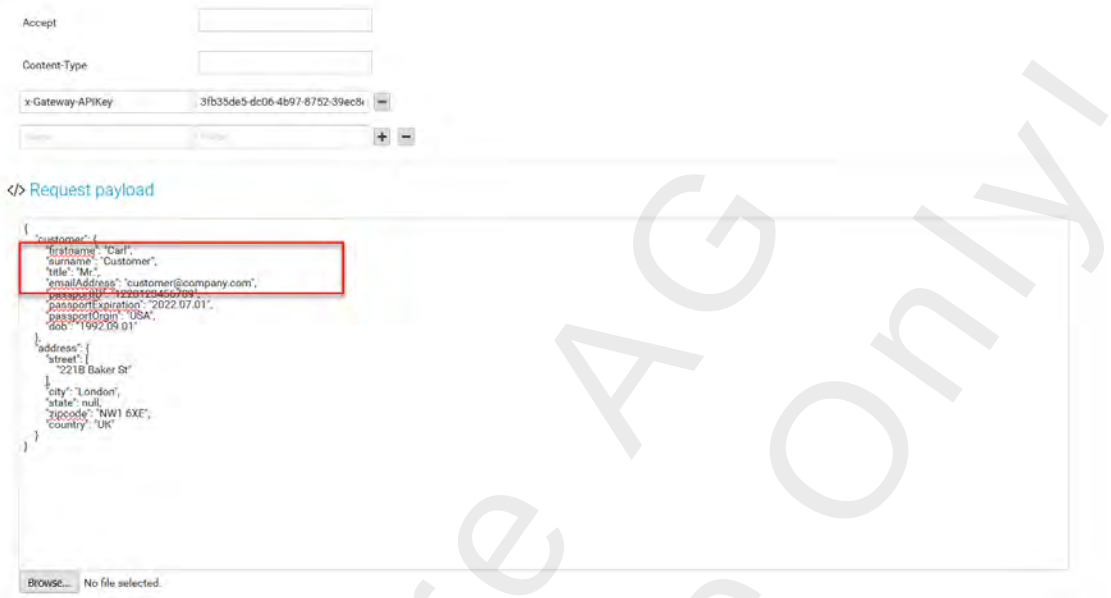



20. Run the request clicking the **Test** button. Run the request several times until you see the expected error message.




21. Navigate to **Packages > APIs**. Select the **SignupAPI.** Navigate to **Try API>>.** You will find the **SAGTours app** listed, as well as the **API key** added to request as header parameter. We want to add a new customer to the SAGTours application. From the list or resources select the resource **POST /customer.**
    a. Provide following Header parameters
        i. **Accept:** application/json
        ii. **Content-Type:** application/json

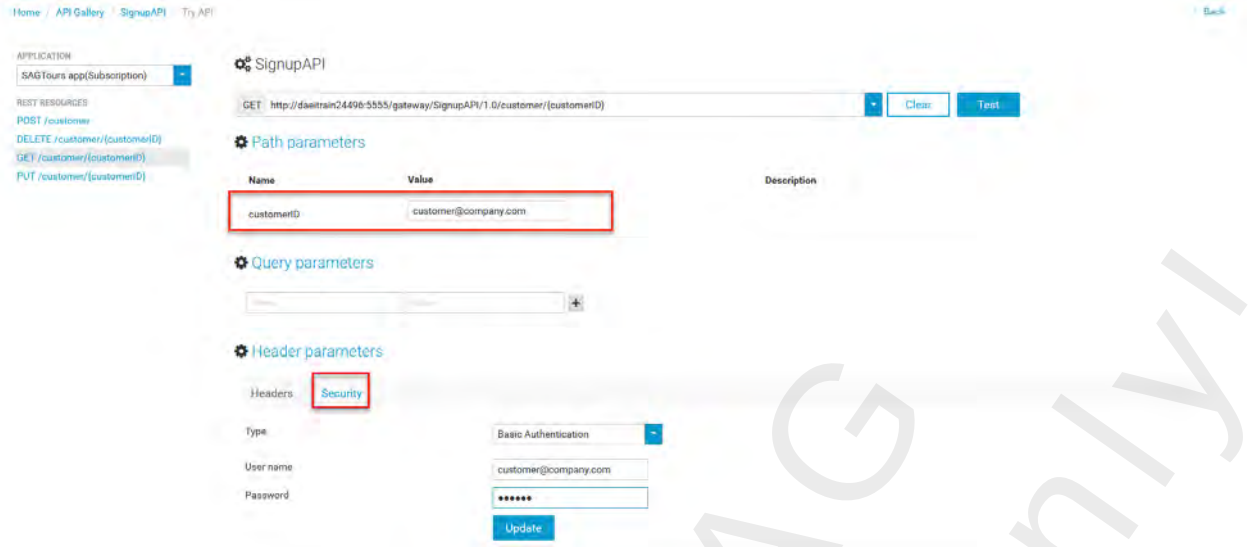
b. Scroll down to the **Request payload** section. Within the payload provide the following properties
   - **i.** Firstname: Carl
   - **ii.** Surname: Customer
   - **iii.** Title: Mr.
   - **iv.** emailAddress: customer@company.com




Click **Test.**

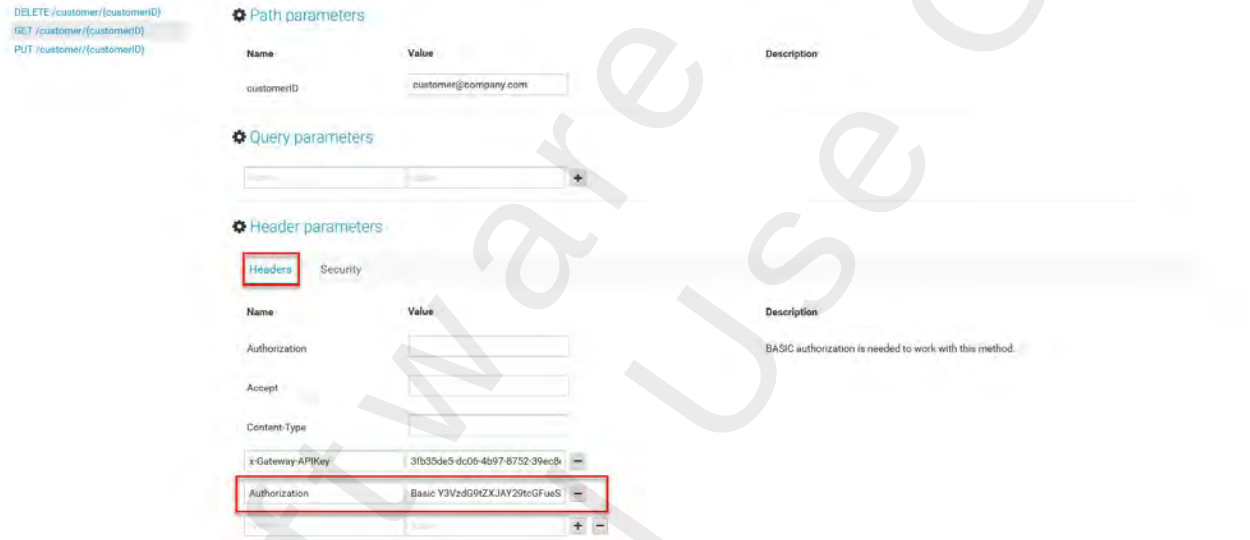
The response should show response code 200**.**


22. We want to review the detailed information for this account. From the list or resources select the resource **GET /customer/{customerID}.**
    a. Provide the **Path parameter**
       i. **customerID:** customer@company.com
    b. Remember accessing an account needs Basic authentication. Within the section **Header Parameter**, select the **Security** tab.Provide the following properties:
       i. **Type:** Basic Authentication
       ii. **User name:** customer@company.com
       iii. **Password:** manage

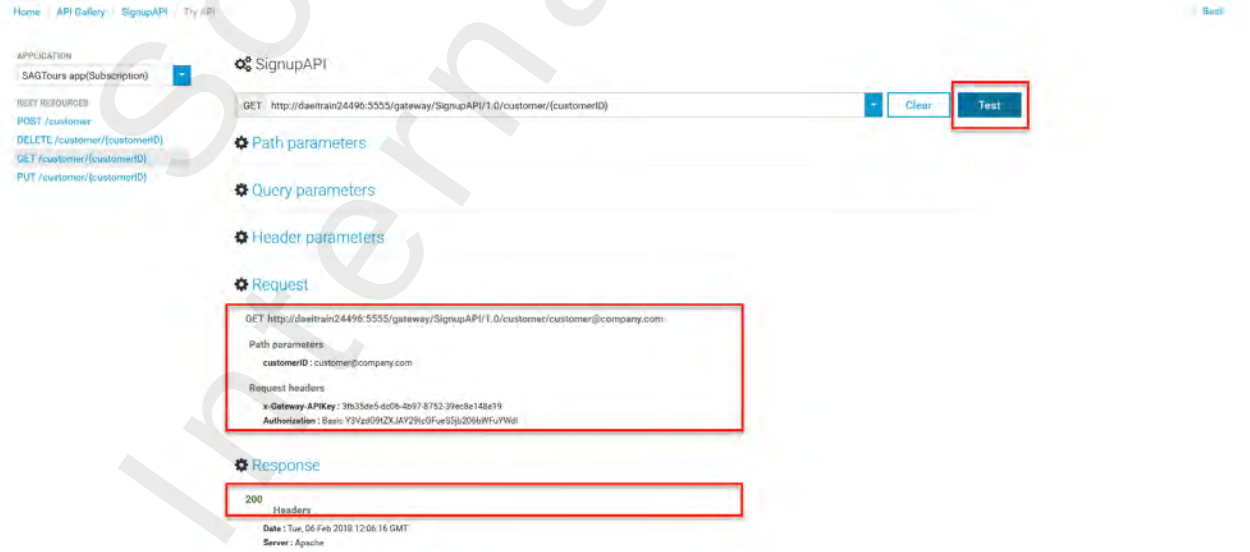



Click **Update**.

23. Within **Header parameter** change back to **Headers**. A new parameter **Authorization** has been added




24. Click **Test** to run the request. The response code is **200** and the payload shows the new created account.



This page intentionally left blank

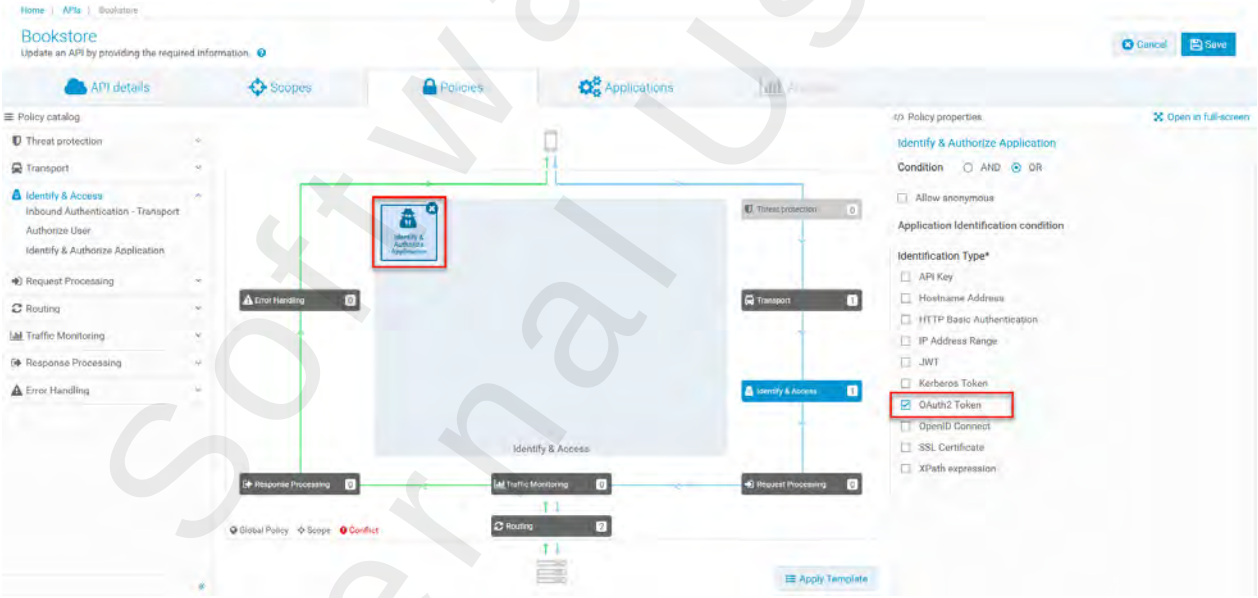
# Exercise 21:
# Try out API secured with OAuth2 token

## Objectives

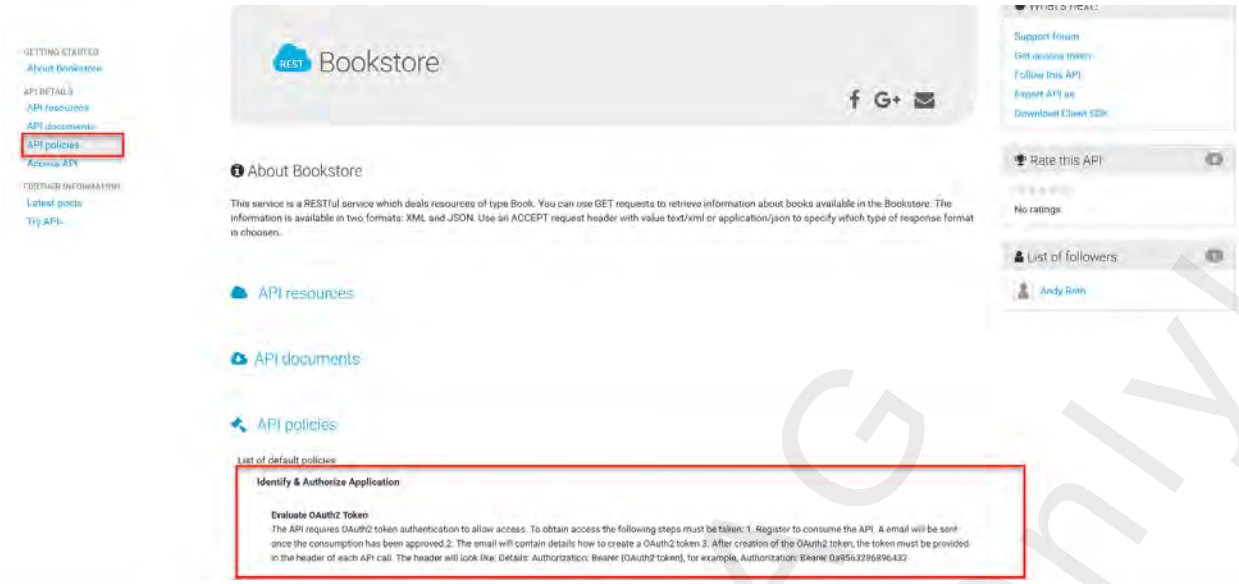
In this exercise, you will use API Portal Tryout function to access an **OAuth2** protected APIs published from API Gateway. We will enforce API Gateway to create a corresponding bearer token

## Steps

1. Open Windows Services UI to double check that the following services, needed API Gateway, are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Event Data store 10.1 – should be started together with IS**
   c. **Software AG API Portal 10.1**
2. In order to have API Gateway validating an **OAuth2 token**, we will need to enforce this in a corresponding **Identify & Access** policy. We will use the **Bookstore** API and enhance the API with adding enforcing a valid OAuth2 token. Logon to **API Gateway** as user **Andy/manage**. Navigate to **APIs**. From the list of APIs select **Bookstore**. Unpublish and deactivate the API and switch to **Edit** mode.
3. Navigate to **Policies > Identify & Access**. Select **Identify & Authorize Application** to be added to the API. Provide the following properties:
   a. **Identification Type**: OAuth2 Token



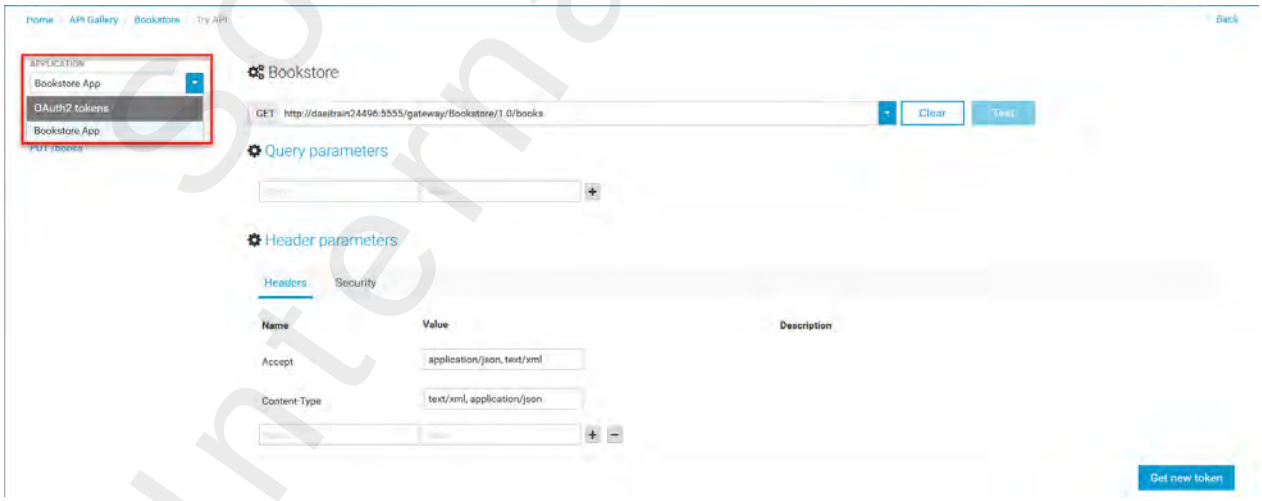

   Click **Save.**
4. Activate and republish **Bookstore** API to **API Portal**.
5. Login to **API Portal** as consumer **customer@company.com** with password **manage**. Use the Search functionality to get the detail view of the **Bookstore** API. Within in the left hand part of the page navigate to **API Details > API policies**. This shows that the API requires **OAuth2 token** authentication and explains how to obtain access to **Bookstore** API.



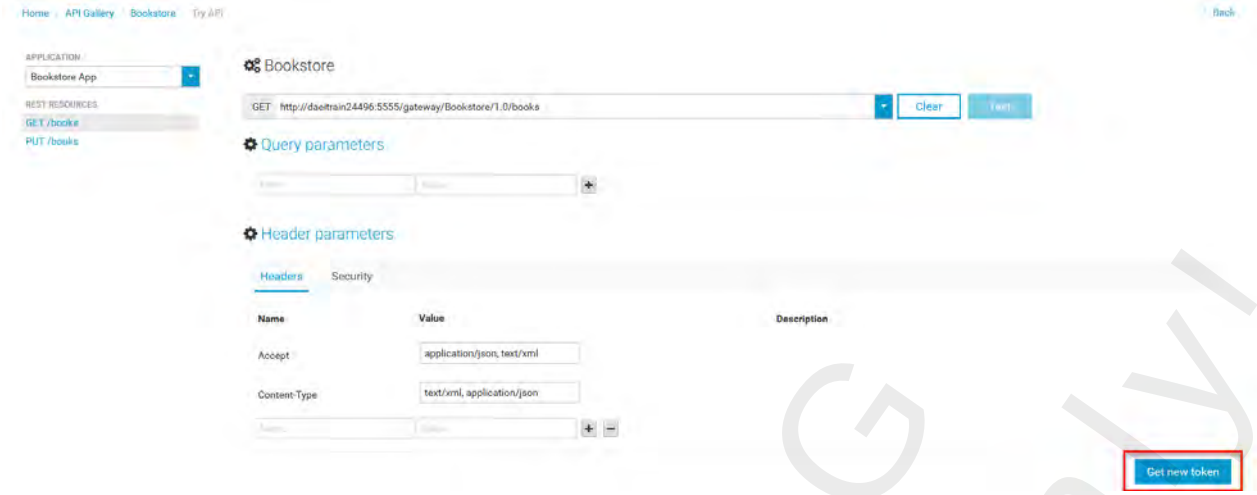

6. The properties you need in order to generate the **OAuth2 token** are part of an application. Click **What's next > Get access token** to request for an application.


7. To request for an API access token provide the following properties
   a. **Application name**: Bookstore App
   b. **Application description:** Try Out application to test evaluation of OAuth2 access token
   c. **Application redirect URI:** - leave defaults –

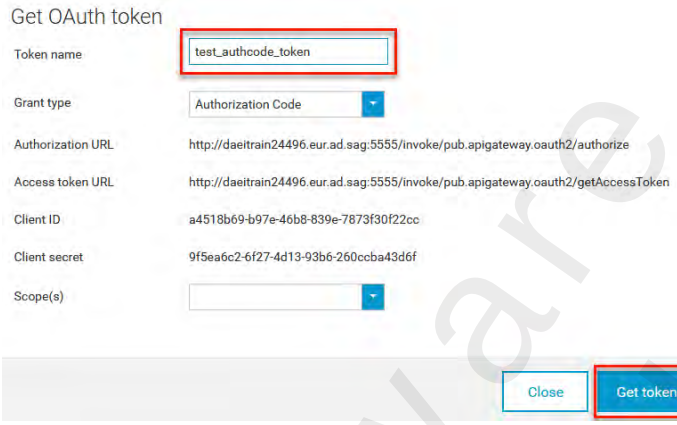
   Click **Request token.**
8. Navigate to **Further information > Try API>>.** Click **BookstoreApp** within **Application.**



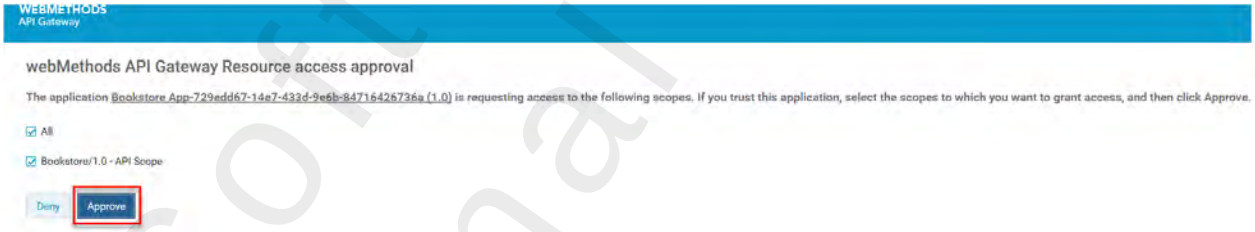

9. Click **Get new token** in the right hand bottom corner on the page.



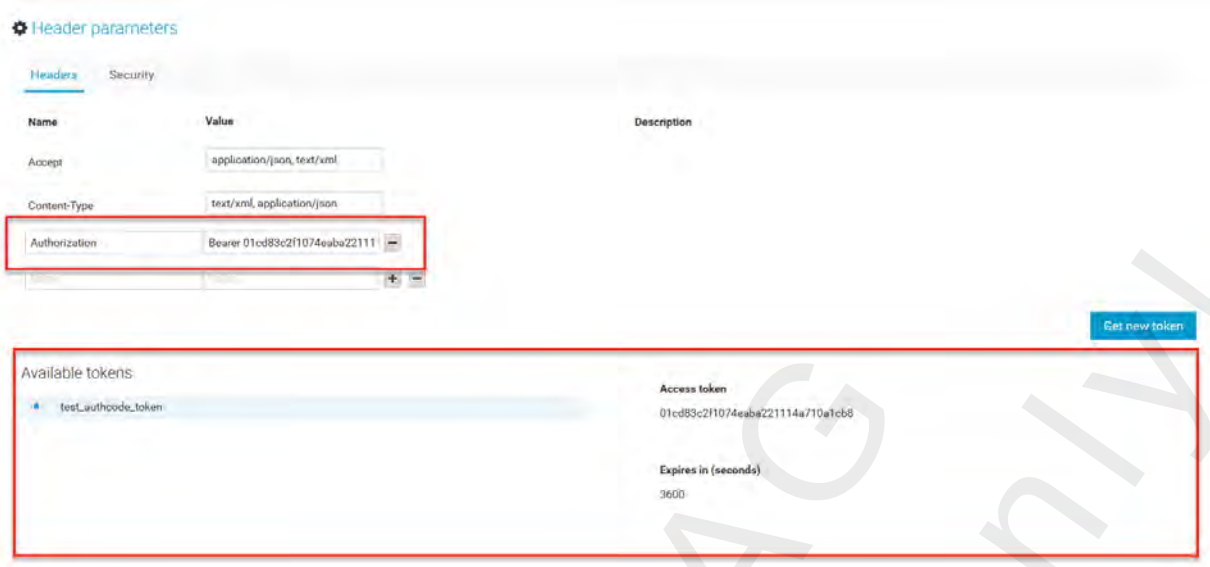

10. To request for an OAuth token provide the following properties
    a. Token name: test_authcode_token



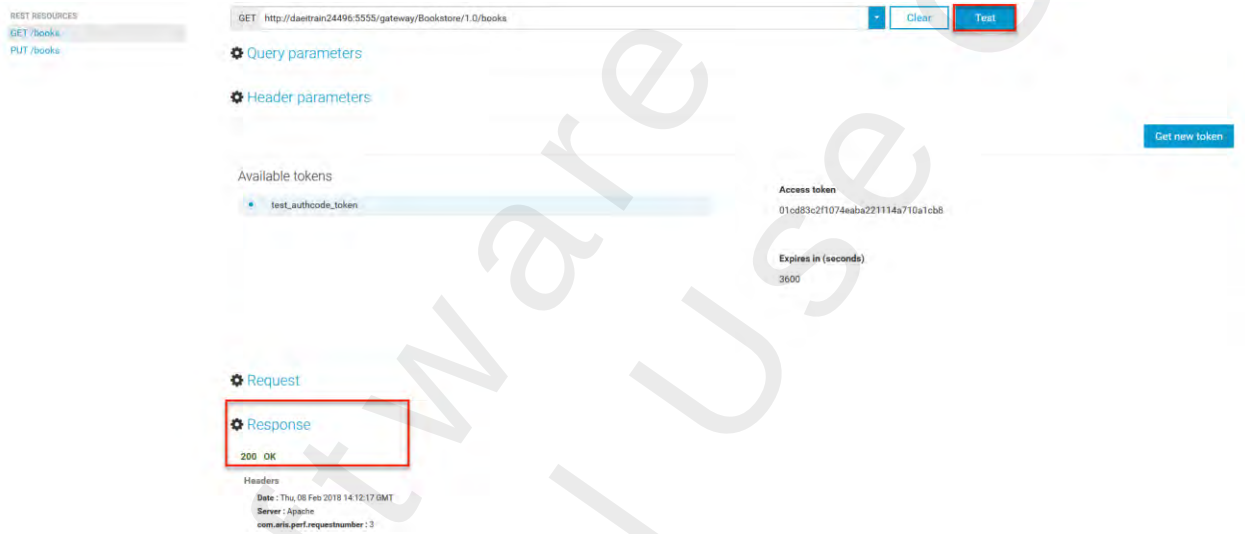

    Click **Get token.**
11. A new Browser window will be opened, asking for **webMeethods API Gateway Resource access approval.**




    Click **Approve.**
    In case Integration Server asks for authentication, provide **Administrator/manage.**
12. On the next screens, if needed, click appropriate button to allow an unsecure connection.
13. The generated token will be given back from **API Gateway** to **API Portal**. It is listed on the **Try API** gape of the **Boostore** API. A corresponding Header variable **Authorization** is added to the section of Header parameters.




14. Click Test to run the request with the OAuth token. The response with response code 200 shows the booklist.



# Exercise 22:
# Editing APIs in API Portal

## Objectives

In this exercise, you will customize/edit an API in API-Portal.

## Steps

1. Open **Windows Services UI** to double check that the following services, needed for **API Gateway** and **API Portal** are up and running. If service is not running, start the service.
   a. **Software AG Integration Server 10.1**
   b. **Software AG Event Data store 10.1 – should be started together with IS**
   c. **Software AG Runtime 10.1**
   d. **Software AG API Portal 10.1**
   e. **hMail Server**
2. Verify that all runnables of **API Portal** are up and running using the **Start Menu -> Software AG -> Administration -> Start API Portal Cloud Controller**
3. Login to **API-Portal** in a private Firefox Brouwser window using the URL **http://sagbase:18101/#default/home** or use the shortcut defined when starting **Mozilla Firefox**.
4. Login as **API Provider** with credentials **andy/manage.** Navigate to the **API Gallery** menu and select **SearchCruise** API.
5. Navigate to **API Details** > **API resources**. Open the resource **/cruises/{cruiseID}**.
6. We would like to update the description of the resource. Hover your mouse over the description of the resource. An edit icon will pop up.

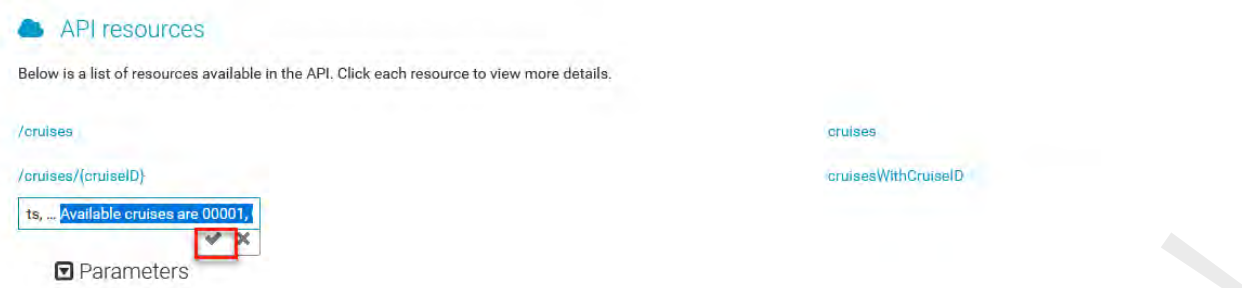

   a. Only the cruiseIDs 00001 and 00002 are available cruiseIDs. Switch to edit mode of the description and add the following text to the existing description:

      Available cruises are 00001, 00002

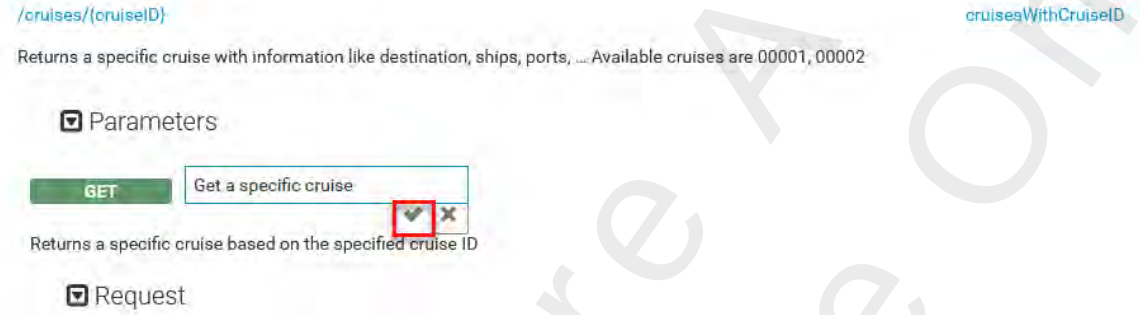
   b. Save your changes and click the corresponding icon



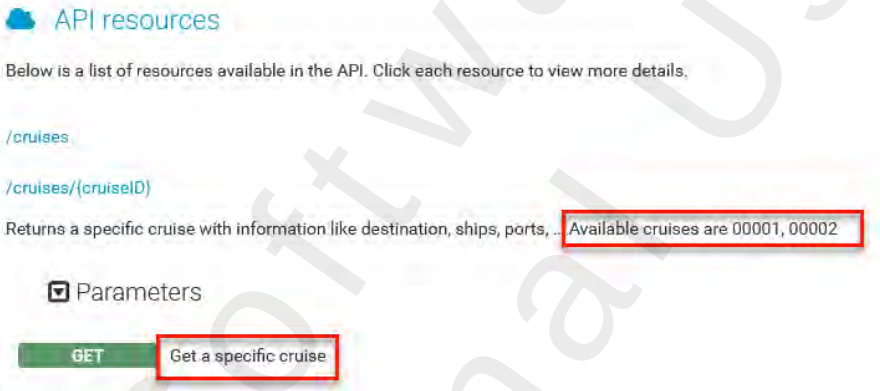
 



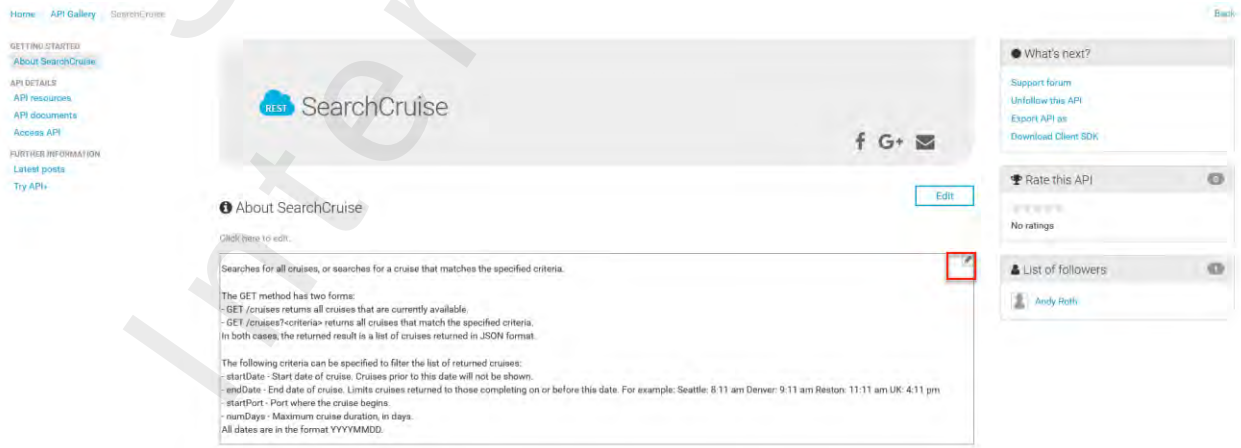

7. We would like to update the description of the **GET** method. Click **Click here to edit**.
   a. Provide the following content:
      Get a specific cruise
   b. Save your changes and click the corresponding icon




8. Verify your changes.

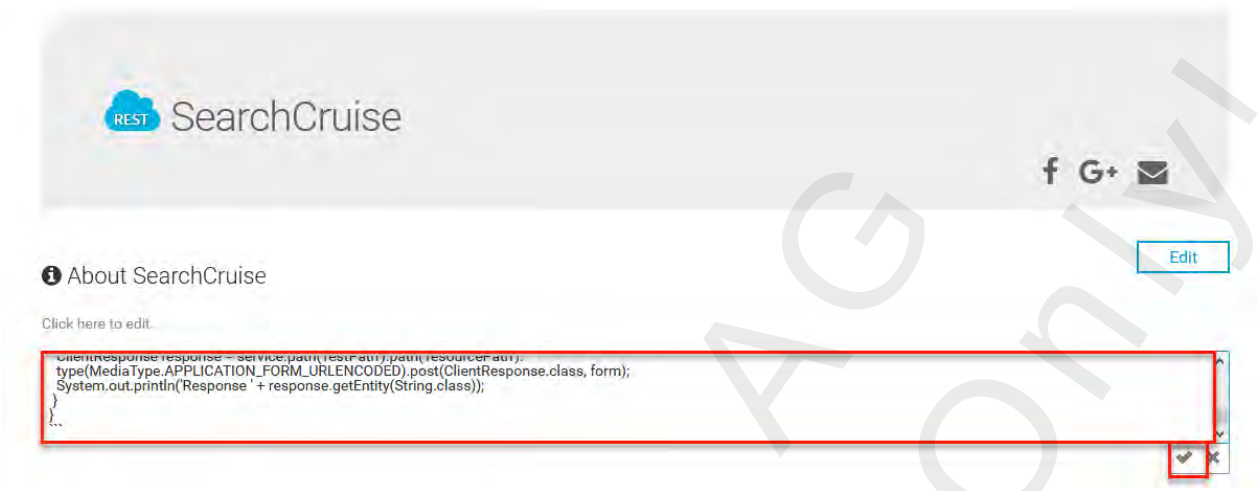



9. We would like to do more changes on the description of the **SearchCruise API.** Hover over the Description of the API and click the Edit icon.



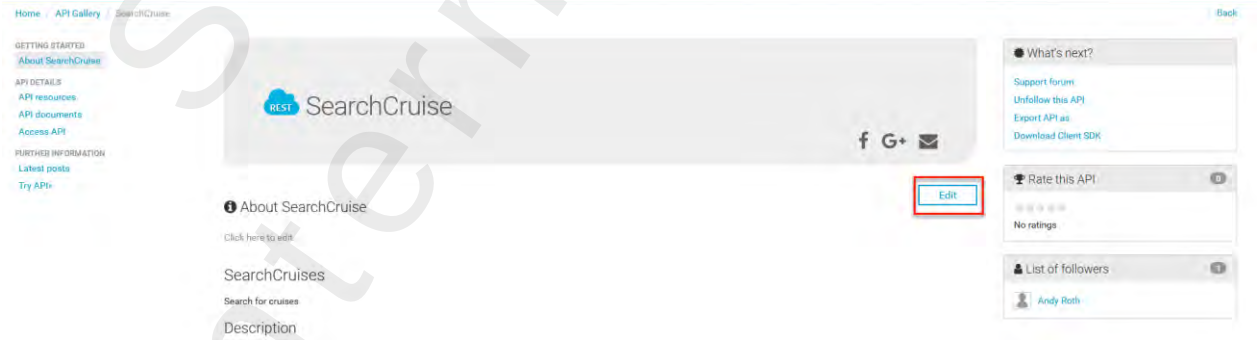
a. Open the file **SearchCruise.md** available in folder **C:\Training\456-71E\Exercise22** with **Notepad+.**

b. Copy the content of this file and add the content at the end of the current description in **SearchCruise**. Save your changes and click the corresponding icon.

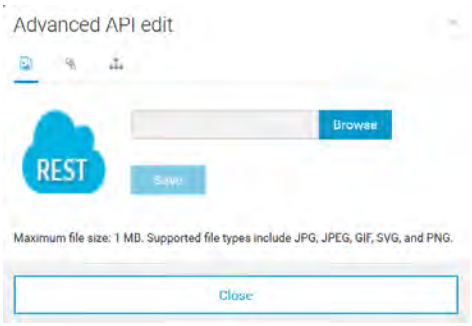



10. Verify the changes.

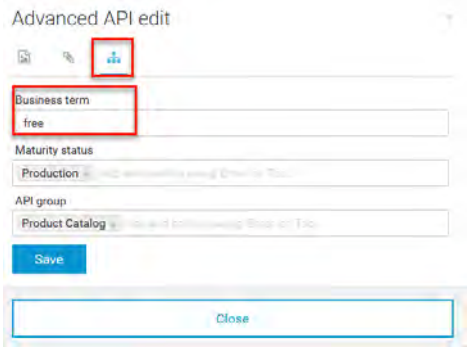

11. We would like to add some additional attributes to **SearchCruises**. Switch over to Edit mode by clicking the **Edit**.




a. Click **What's next > Advanced edit**.

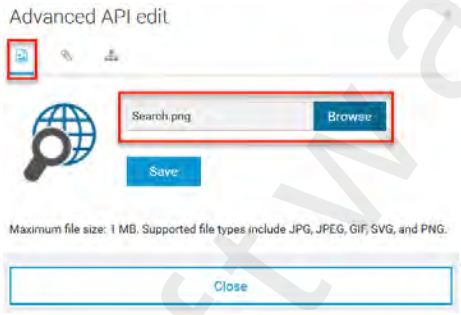



b. Within the header select the icon to add a classification attribute. Provide the following value
   i. **Business term**: Free



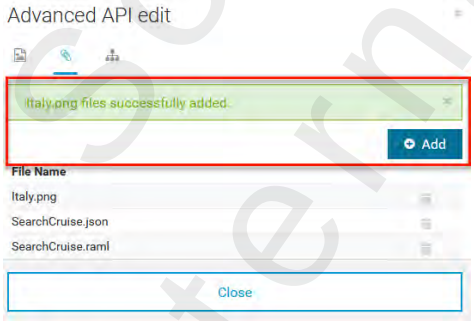

Confirm using **Enter**. Click **Save**.

c. Within the header select the icon to add an icon. Browse for **Search.png** available in folder **C:\Training\456-71E\Exercise22.**




Click **Save**.

d. Within the header select the icon to add a document. Click **Add**. Browse for **Italy.png** available in folder **C:\Training\456-71E\Exercise22.**




Click **Close.**

6. Use the **Search** functionality again and search for the **BookingsAPI**.
7. Use the Edit API functionality to add the following properties to the API. The icon is available in folder **C:\Training\456-71E\Exercise22.**
   e. **API categories**:
      i. **Business term**: Free
   f. **Icon:** Booking.png

   Click **Save**. Click **Close.**

8. Use the **Search** functionality again and search for the **SignupAPI**.
9. Use the Edit API functionality to add the following properties to the API. The icon is available in folder **C:\Training\456-71E\Exercise22**
   g. **API categories**:
      i. **Business term**: Free
   h. **Icon:** Customers.png
   Click **Close**.