

Extend your back-end integrations securely to partners and developers.

Extending access via APIs to your back-end integrations empower your partners and developer community to create new business value, technical value, and customer experiences for your products and offerings. Spur innovations where a number of technologies are combined to create something new, for example, extending the ability apply for loan pre-approvals that can be used within apps that search for cars or real estate. To do this, you must first create the back-end integrations, which combine data from existing core systems, disparate assets, or SaaS services with the ability to send critical data between systems reliably. Second, you need to provide APIs to your back-end integrations that secure access and apply rate limits.

In this tutorial, you learn how to deploy an app integration flow that takes data from one source and sends it to a message queue for reliable delivery. Then, you expose this integration as a rate-limited API secured by a key and secret. This integration flow is deployed **quickly and easily** as an independent, **auto-scalable microservice** running on **containers**. By using RedHat OpenShift as a foundation.

In this tutorial, you will explore the following key capabilities:

- Explore multiple integration capabilities within a single platform.
- Create a queue using MQ Console
- Create an integration flow between a public cloud service and on-prem message queue.
- Deploy the integration flow as a microservice using Helm.
- Provide access to the integration flow as a secure API.
- Explore Operations Dashboard for APIC, IBM MQ and App Connect Enterprise.

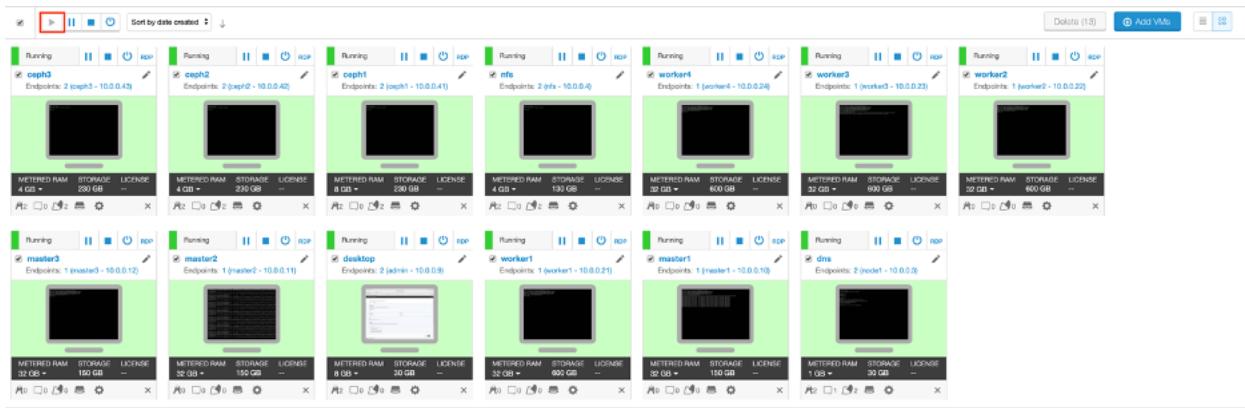
Task 1 - Start the Environment

As this is a new deployment of the Cloud Pak for Integration, you must execute some steps to prepare the environment. Initial setup steps

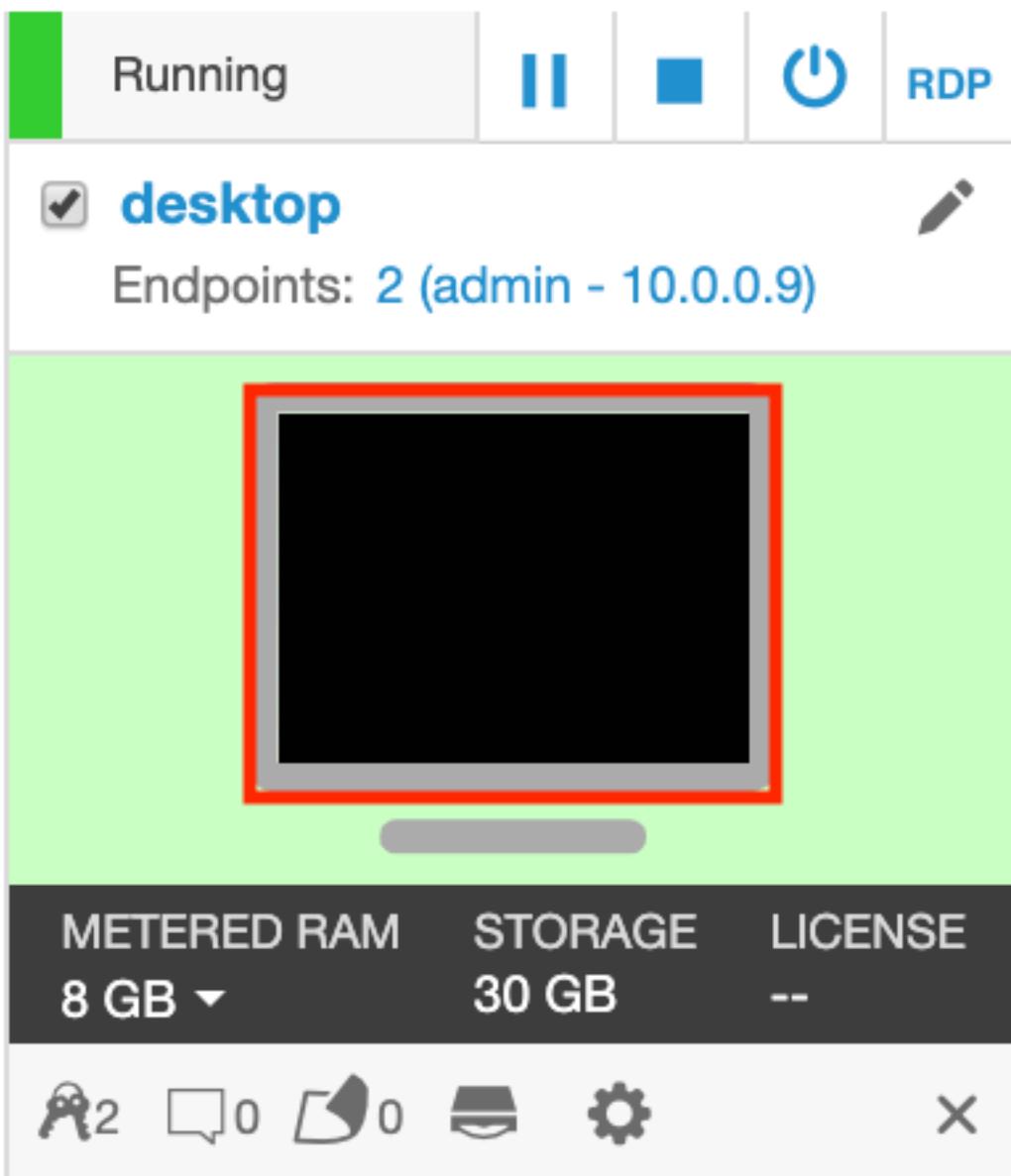
are only needed for a fresh installation of the platform. They do not need to be repeated.

All work for this lab is done on the Developer Machine. Open the **developer Machine** VM by clicking the tile.

1. If the environment is already up and running when you open your reservation link, skip to step 3. If it is not running, continue to step 2.
2. **Click** the Run VM(s) button as shown below to start the virtual machine environment that is used for this lab.



3. Once the virtual machine has started, click the **desktop** Machine tile to start your lab exercise.



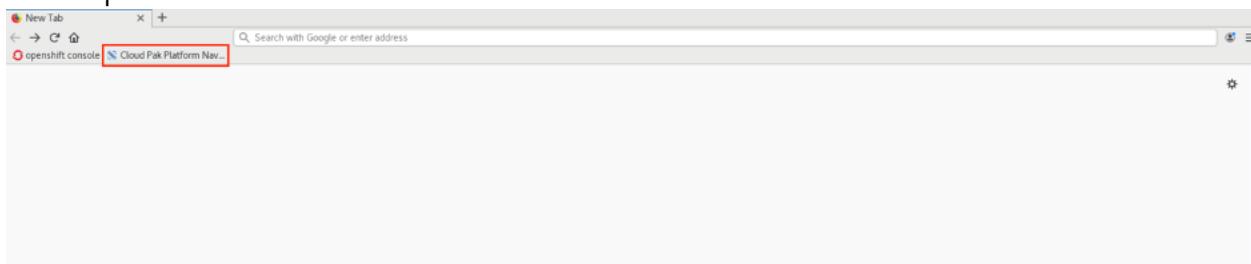
4. Log in as user **ibmuser**, password **engageibm**

Task 2 - Configure Message Queue (MQ) to Authorize and Accept Data. As this is a brand-new deployment of the Cloud Pak for Integration, all instances of integrations, message queues, and event streams are deployed as microservices. We need to authorize the Message Queue service to accept incoming data from the integration running on a separate server.

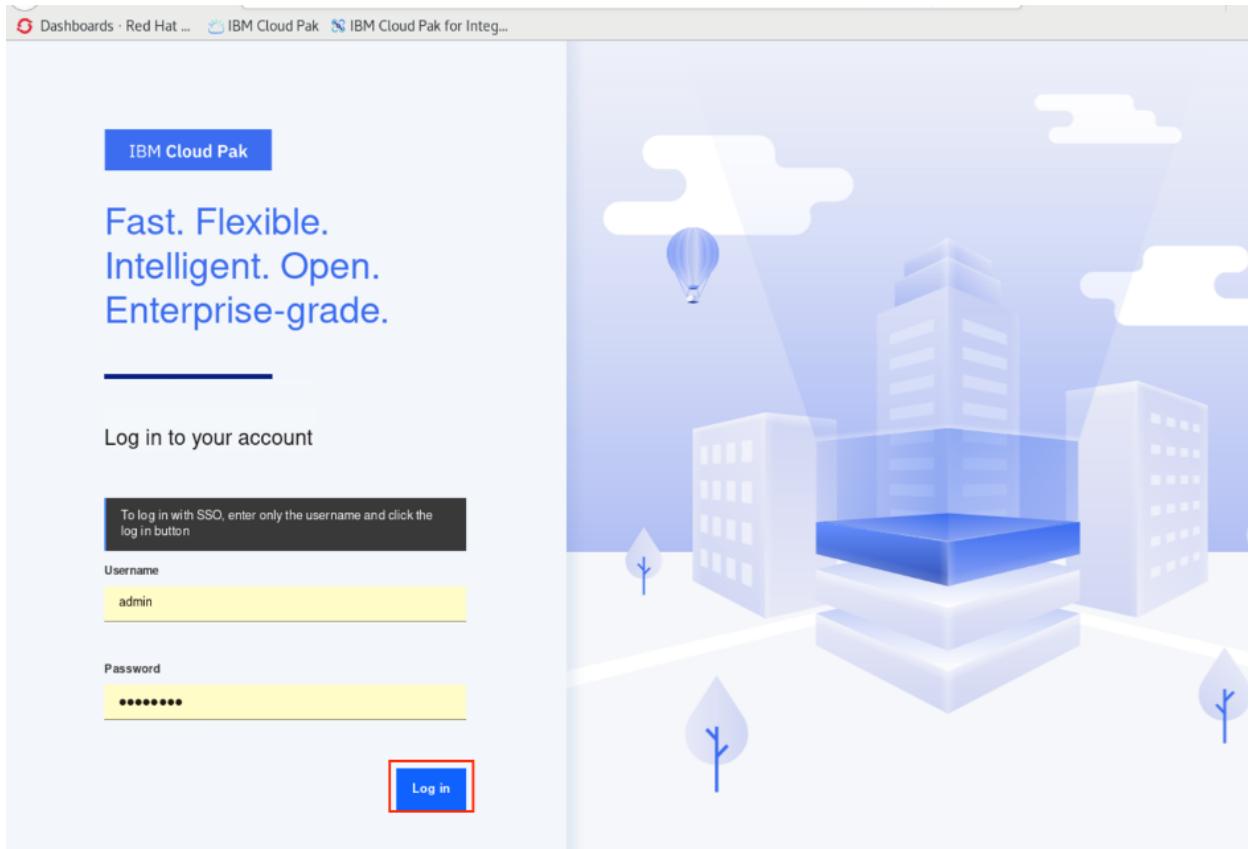
Cloud Pak for Integration provides a single solution for all of your enterprise integration needs. The platform provides a comprehensive set of industry-leading capabilities. Use any of them on their own or together through a single interface. Create, manage, and monitor all of your integrations across SaaS applications, messaging, streams, APIs, high-speed transfer, and more.

Unlock the power of your data and support the scale required for all of your integration and digital transformation initiatives.

1. Click the IBM Cloud Pak for Integration bookmark in the bookmarks bar at the top.



2. The login screen for IBM Cloud Pak might be displayed, **you don't need to enter** username: **admin** and Password: **passw0rd**. Click **log in**.



3. After logging in, the **Cloud Pak for Integration** home page is loaded. From here you are able to navigate to all the integration and development technology contained within the platform. Today, this technology includes: API Connect, App Connect integration, Aspera, DataPower, Event Streams, MQ, Tracing and Asset Repo . Click the **View Instances** link.

Welcome to IBM Cloud Pak for Integration
Let's get you going!

Show less

[Create instance](#) [View instances](#)

API Connect Create, manage and secure your APIs Create instance	App Connect Unlock the power of your data to drive new opportunities Create instance	Aspera Quickly and reliably transfer files and data sets of any size Create instance	DataPower Leverage the IBM DataPower Gateway directly deployed on its own, or embedded in API Connect. Create instance
Event Streams Apache Kafka for the Enterprise Create instance	MQ Proven, enterprise-grade messaging that moves data to where it is needed Create instance	Tracing Quick diagnosis of errors and performance issues across product instances by providing end-to-end data tracing Create instance	Asset Repo Accelerate building new workflows via design time collaboration and active re-use of existing integration assets and skills Create instance

Task 3 - Creating a queue in MQ

This task covers administering and creating a new queue in MQ. MQ for Cloud Pak for Integration has a Web GUI, which the Integration Developer, with security authorization, will be able to manage the different MQ objects (queues, channels, topics and so on)

1. Click in **View Instances** and then menu, **mq-1**, instance to open MQ console.

Welcome to IBM Cloud Pak for Integration
Let's get you going!

Show less

Create instance View instances

Capability type	Instance name	Namespace	Created	Versions	Status
Event Streams	es-1	eventstreams	2 hours ago	1	Running
MQ	mq-1	mq	2 days ago	1	Running
MQ	mq-2	mq	9 days ago	1	Running
Asset Registry	ar-registry-1	integration	12 days ago	1	Running
App Connect Designer	ac-design	ace	13 days ago	1	Running
App Connect Dashboard	ac-1	ace	13 days ago	1	Running
Event Streams	es-1	eventstreams	15 days ago	1	Running
Deployer	dp-1	deployer	15 days ago	1	Running
API Connect	apic-1	apic	15 days ago	1	Running
Tracing	tracing	tracing	15 days ago	1	Running

Items per page: 10 | 1-10 of 10 items

You might receive a warning message. Click **Advance**.

Dashboards · Red Hat ... IBM Cloud Pak IBM Cloud Pak for Integ...

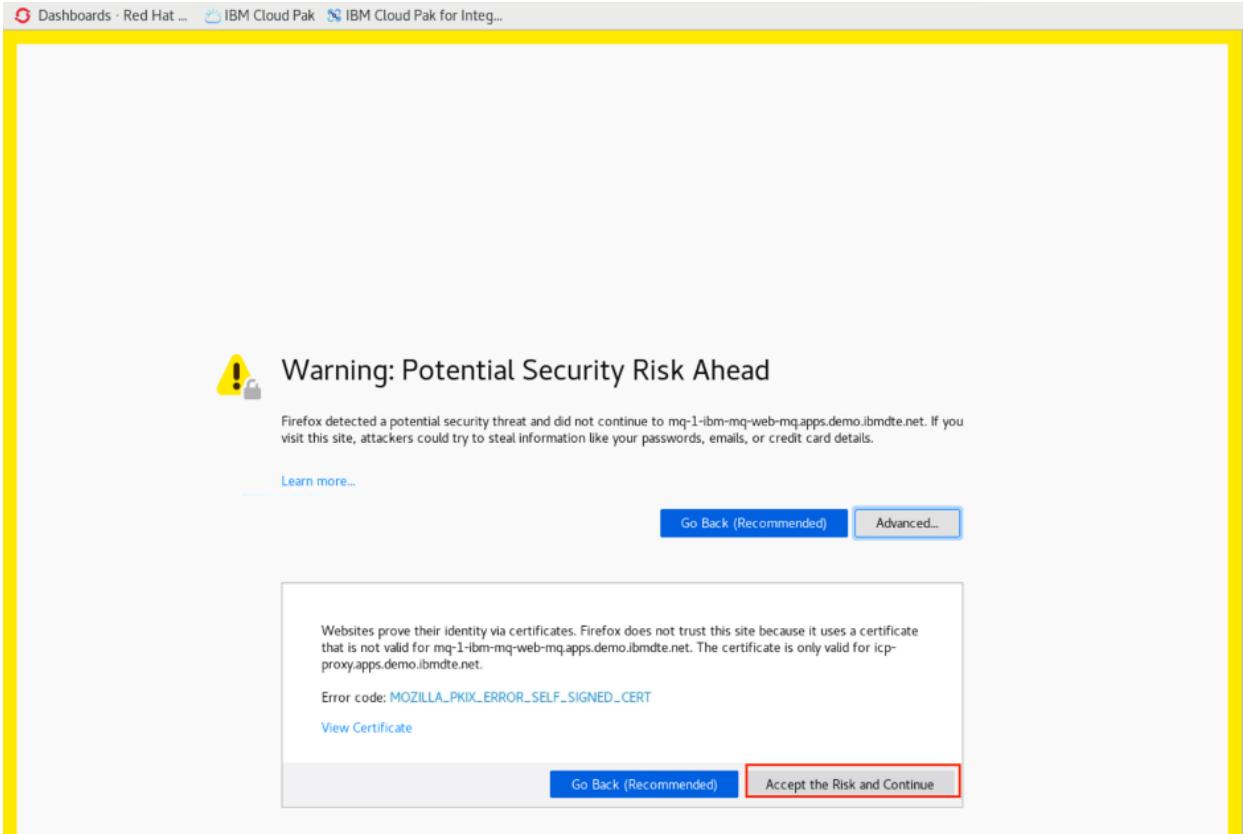
⚠️ Warning: Potential Security Risk Ahead

Firefox detected a potential security threat and did not continue to mq-1-lbm-mq-web-mq.apps.demo.libmte.net. If you visit this site, attackers could try to steal information like your passwords, emails, or credit card details.

[Learn more...](#)

[Go Back \(Recommended\)](#) [Advanced...](#)

Click **Accept the Risk and Continue** in the Firefox browser. Click the **Proceed (unsafe)** link in the Chrome browser.



2. You see the MQ Console and MQ server running as **mq**. You can add more widgets for more information on MQ objects, click **Add Widget** button.

The screenshot shows a dashboard titled "Local Queue Managers". At the top, there is a navigation bar with links to "Dashboards · Red Hat ...", "IBM Cloud Pak", "IBM Cloud Pak for Integ...", and "MQ mq | mq-1". Below the navigation, there are tabs labeled "Tab 1" and "+". In the top right corner, there are three icons: a gear, a question mark, and a refresh symbol. To the right of these icons is a vertical ellipsis (...). A red box highlights the "Add widget" button in the top right corner of the main content area.

Local Queue Managers

Search

Name	Status
mq	Running

Total: 1 Last updated: 8:01:10 PM

3. In the window, you can select the **Queues** widget.

Add a new widget

[Local Queue Managers](#) Manage local queue managers

[Chart](#) Monitor your MQ platform

Add a widget to display MQ object information for the specified queue manager
Queue manager:

mq

[Queues](#)

Configure destinations for messages

[Topics](#)

Administrative objects for assigning attributes to topics

[Listeners](#)

Configure processes to accept network requests

[Channels](#)

Queue manager communication paths

[Client-connection Channels](#)

Client connectivity details

[Authentication Information](#)

Configure authentication mechanisms

[Subscriptions](#)

Configure how subscriptions to topics are handled

[Channel Authentication Records](#)

Control access to channels

[Close](#)

4. You see the queues window. You see queue names, type and queue depth. Click **Create (+)** to create a queue.

Queues on mq		
Search		Create +
▲ Name	Queue type	Queue depth
AMQ.5E736320210B6405	Local	0
Total: 1		Last updated: 8:01:45 PM

5. Enter the queue name: **ORDERS (this is case-sensitive)**. Here you are able to create Local, Remote, Alias and Model queue. In our lab check **Local**.

Create a Queue

Queue name: * i

ORDERS|

1

Queue type:

Local Remote Alias Model

2

Cancel

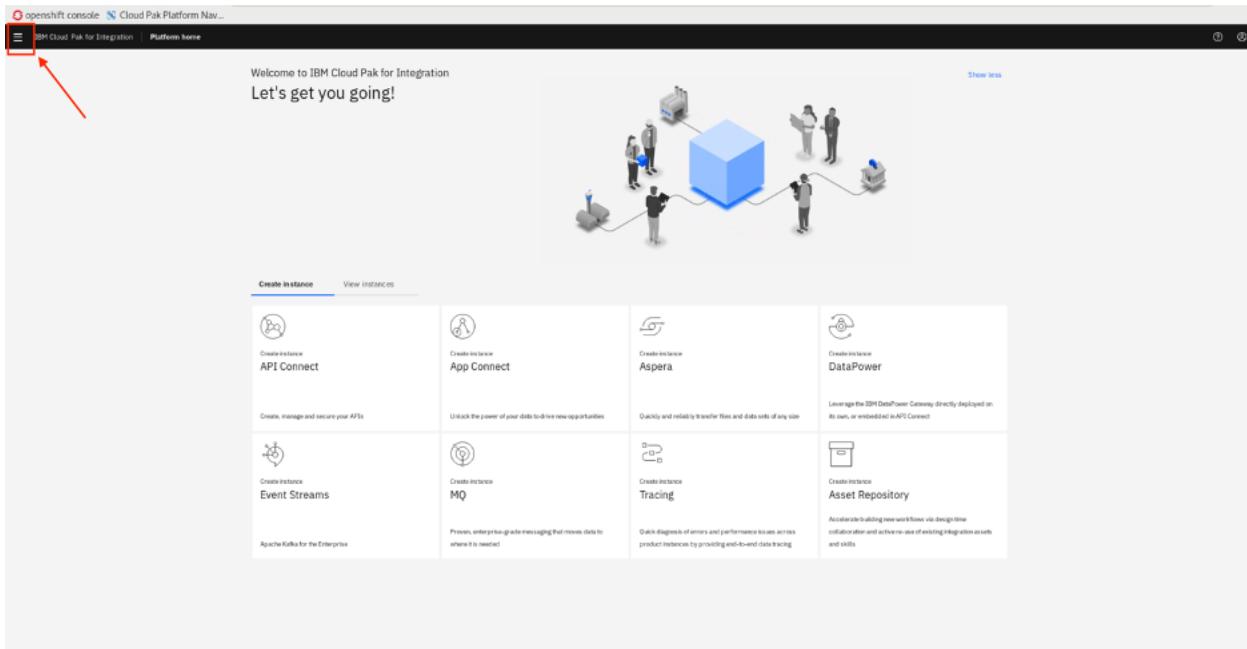
Create

6. Look at **Queues on mq** window and check the queue created (Local and Queue depth).

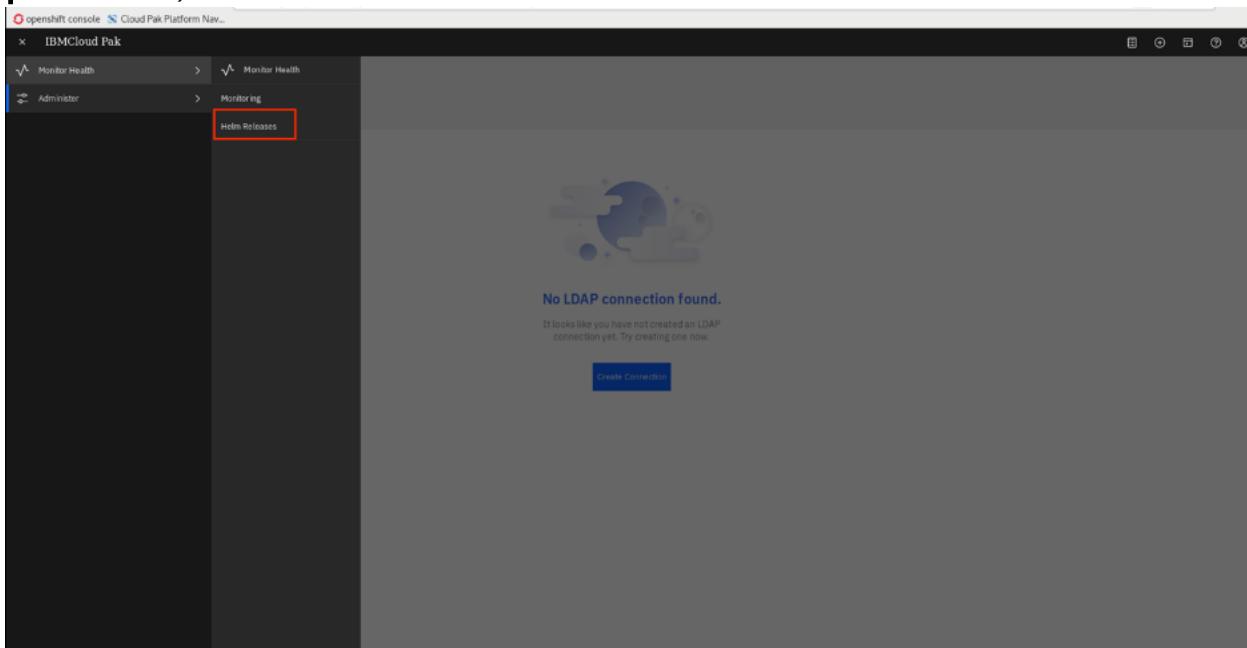
The screenshot shows the IBM Cloud Pak for Integration interface with the MQ mq | mq-1 tab selected. On the left, there is a 'Local Queue Managers' section with one entry: 'mq' status 'Running'. On the right, there is a 'Queues on mq' section with two entries: 'AMQ.5E736320210B6405' (Queue type: Local, Queue depth: 13) and 'ORDERS' (Queue type: Local, Queue depth: 0). The 'ORDERS' row is highlighted with a red border.

Name	Queue type	Queue depth
AMQ.5E736320210B6405	Local	13
ORDERS	Local	0

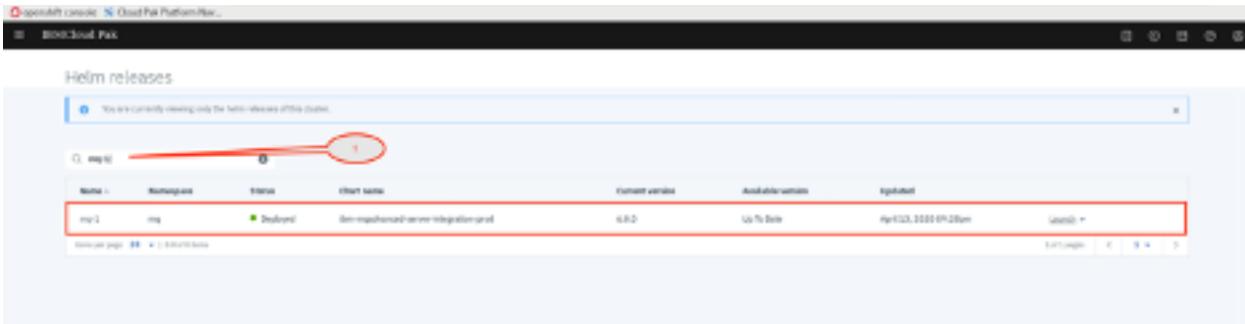
7. You need additional information about the MQ connection, click the “Hamburger” menu and then select **Cloud Pak Foundation**.
Note: You can navigate using the **Hamburger Menu** or **View Instances** in the Cloud Pak Navigator page



8. Open “Hamburger” menu again and select **Administer->Helm Repositories** .



9. On the **Helm Releases** page, type **mq** in the search box. The search finds mq helm releases. Click the line of **mq-1**.



10. Scroll down to the bottom of the page and check the MQ connection. You see the host address: **mq-1-ibm-mq.mq.svc** and the port **1414**. App Connect Enterprise will use a mq connection client for **mq-1**.

Name	ClusterIP	Port	Age
mq-1-ibm-mq-metrics	172.30.212.51	<none>	9157/TCP 17d
mq-1-ibm-mq	172.30.59.31	<none>	9443/TCP,1414/TCP 17d

Name	Secrets	Age
mq-1-ibm-mq	2	17d

Name	Desired	Current	Age
mq-1-ibm-mq	1	1	17d

Notes

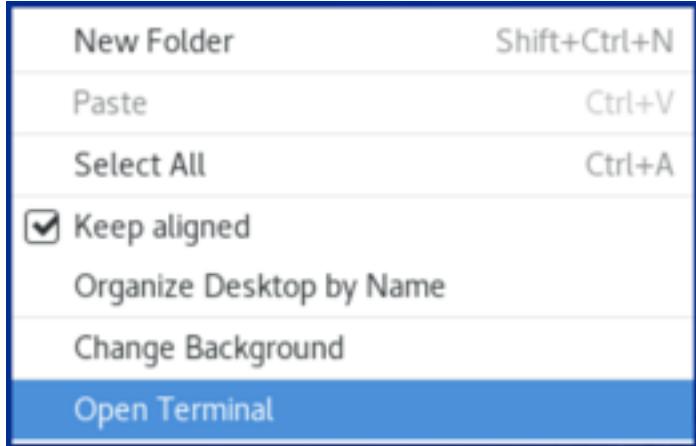
```
Get the MQ Console URL by running these commands:
export CONSOLE_ROUTE=$(kubectl get route mq-1-ibm-mq-web -n mq -o jsonpath=".spec.host")
echo https://$CONSOLE_ROUTE/ibmmq/console
```

The MQ connection information for clients inside the cluster is as follows:
mq-1-ibm-mq.mq.svc:1414

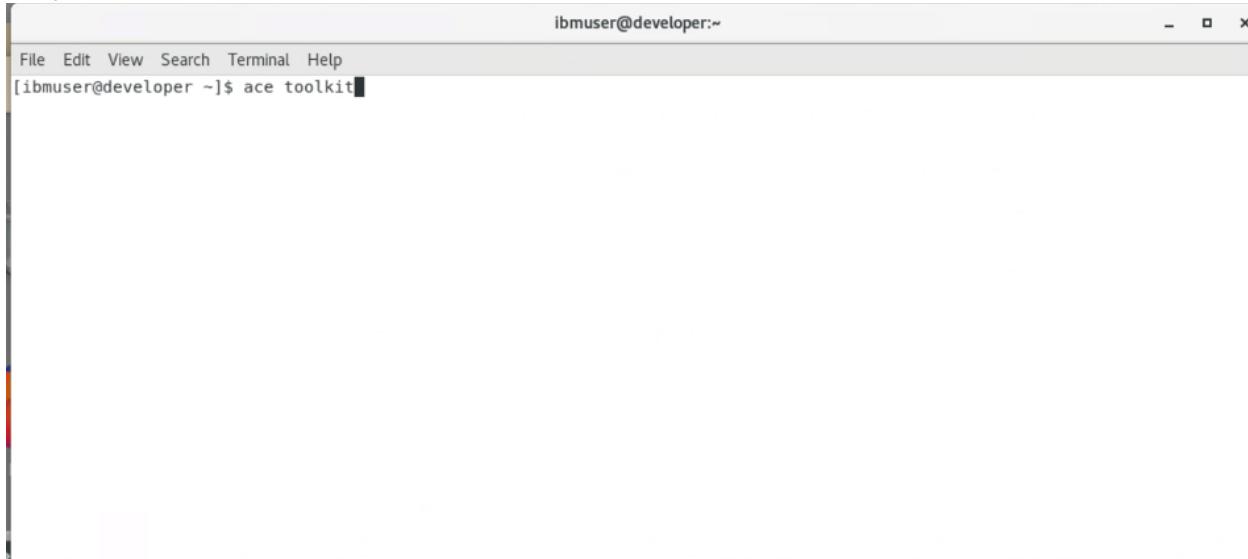
Task 4 - Configuring the app integration flow
This task covers opening and examining an application integration flow in the [IBM App Connect Enterprise Toolkit](#). With the Toolkit you can build powerful and complex integration applications, services, and

APIs quickly and easily using a visual designer. Your integration solutions can be directly deployed to the Cloud Pak for Integration on IBM Cloud Pak running on-premise, in any cloud, or combinations of both.

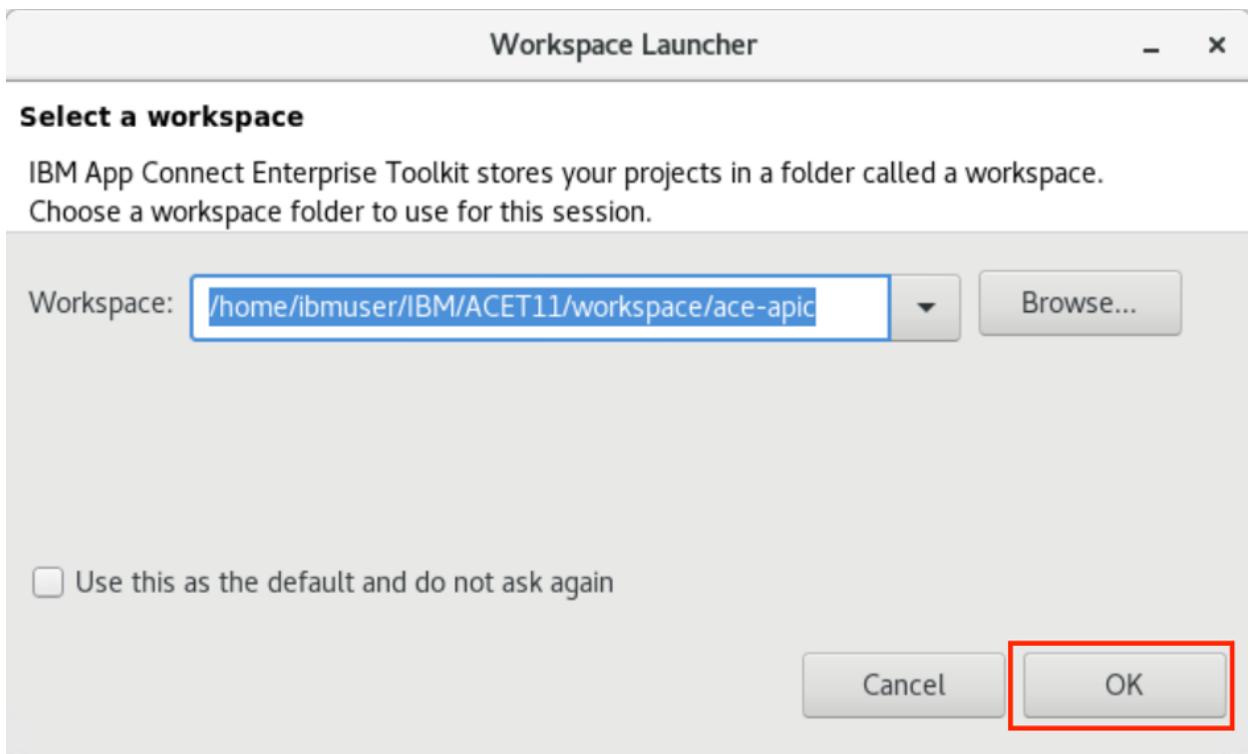
1. In the desktop click the right mouse to open a terminal window.



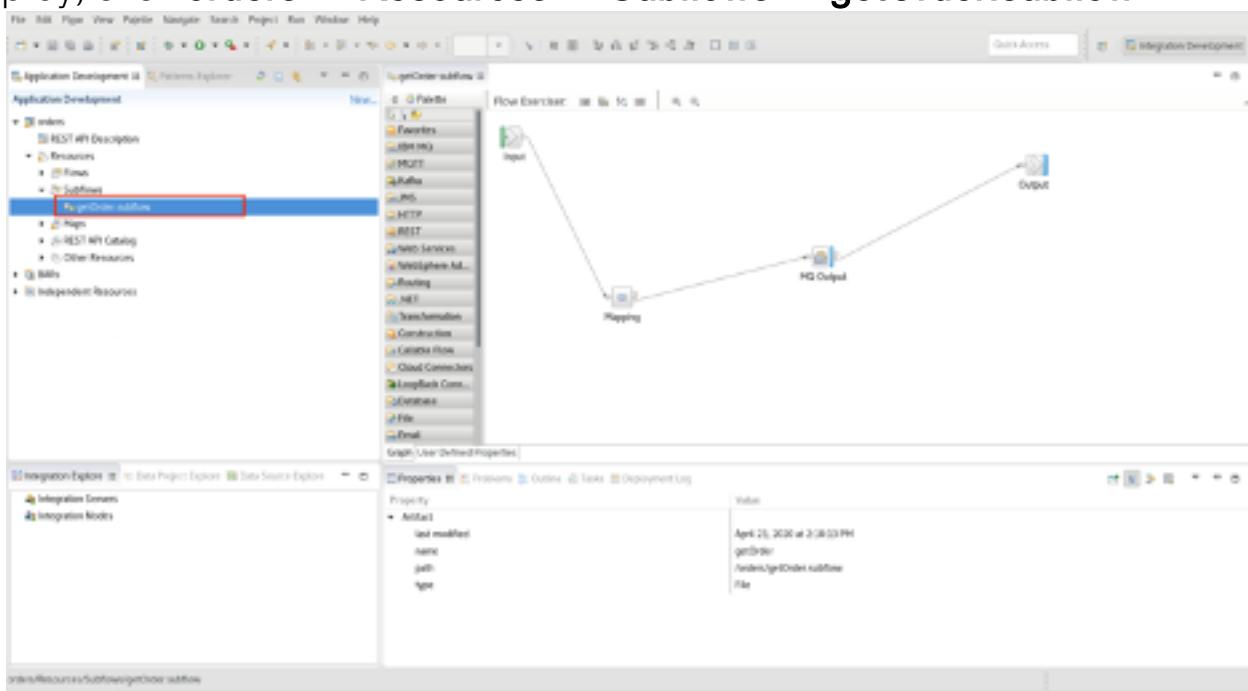
2. In the terminal window, type **ace toolkit** to open the App Connect Enterprise Toolkit.



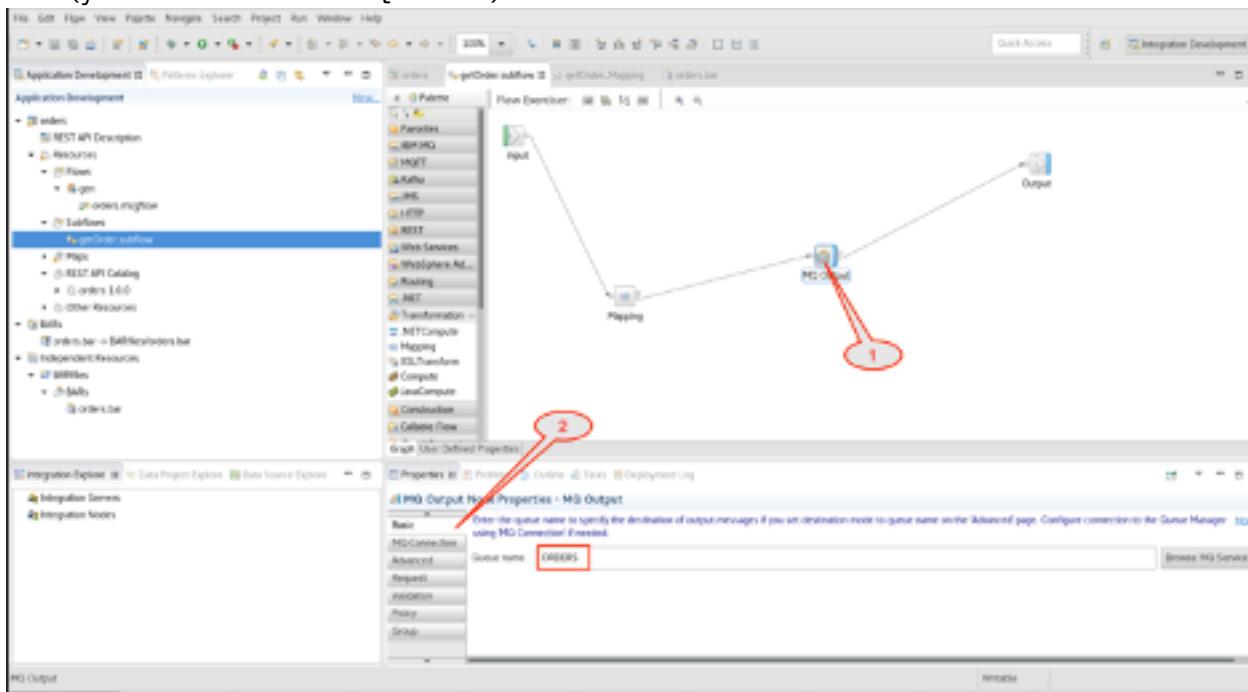
3. In the Workspace Launcher window, choose the workspace /home/student/IBM/ACET11/workspace/ace-apic. Verify the path routes to the folder: **ace-apic**. Click **OK**.



4. The toolkit opens the project. To view the integration flow that you deploy, click **orders** -> **Resources** -> **Subflows** -> **getOrder.subflow**



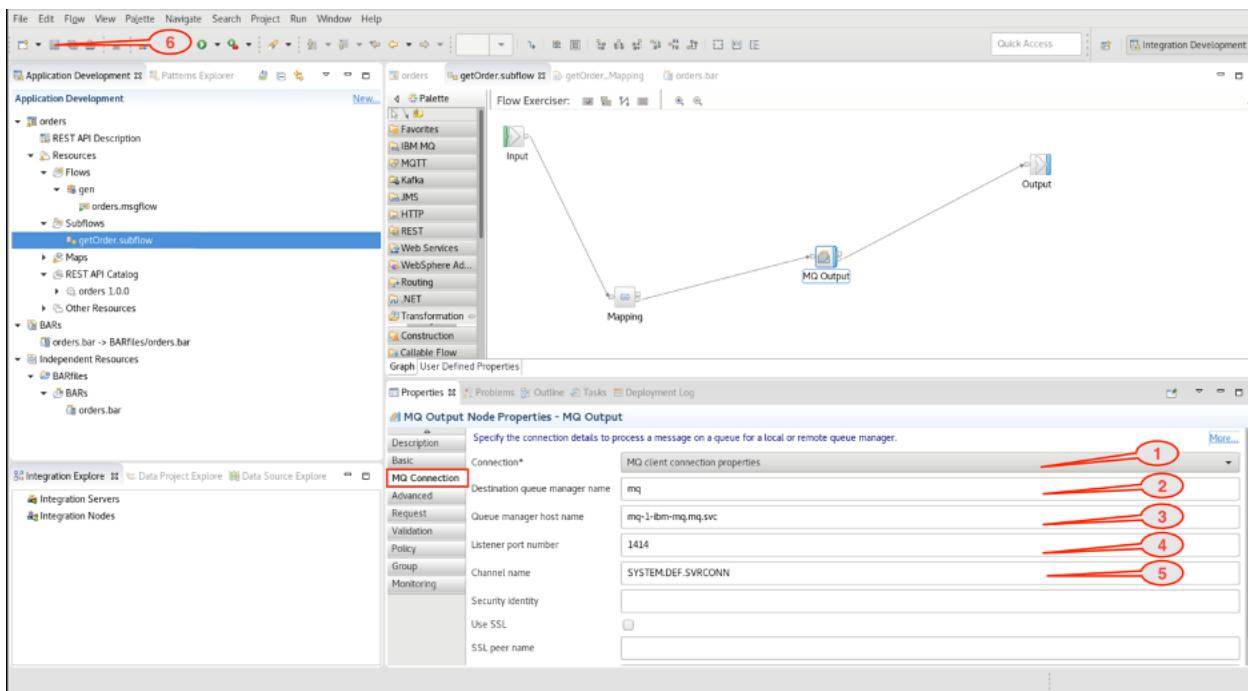
5. Check the MQ node by clicking the **MQ Output** Node. Click **Properties** and select **Basic**, If necessary type the queue name (you created in MQ Task) **ORDERS** .



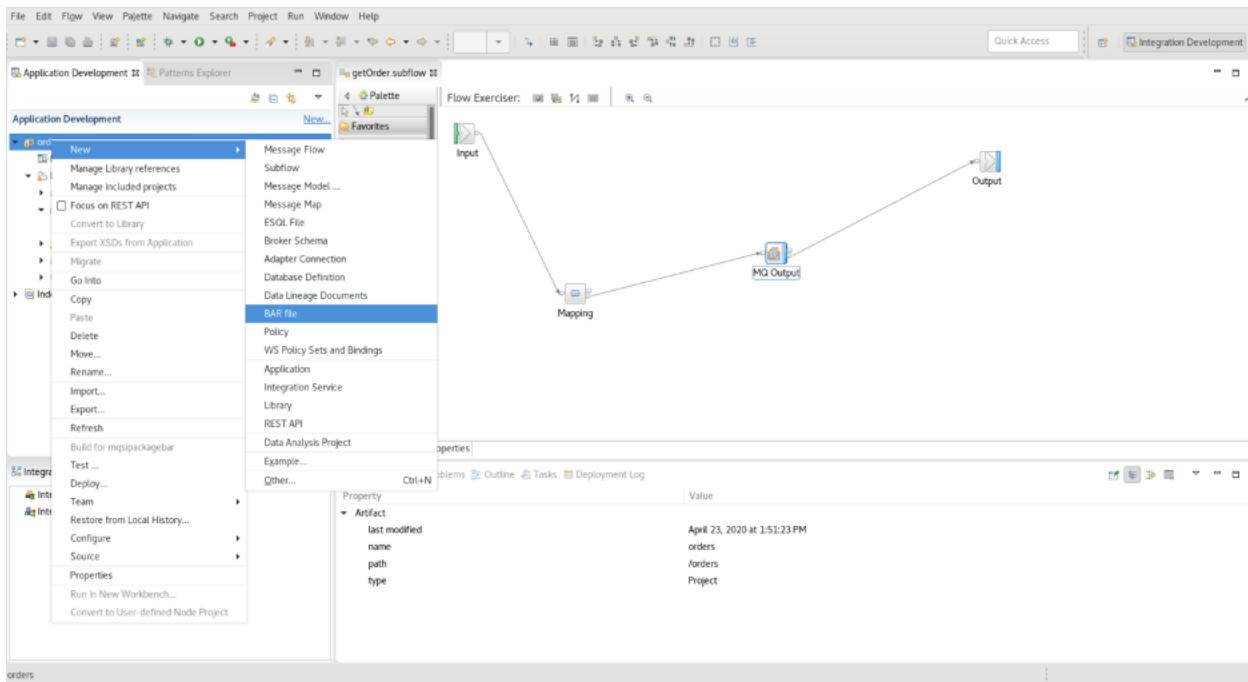
6. You have configured the queue name. You need to configure how the App Connect Enterprise server connects to the MQ server. Click **MQ Connection**.

Obs: All parameters are case-sensitive.

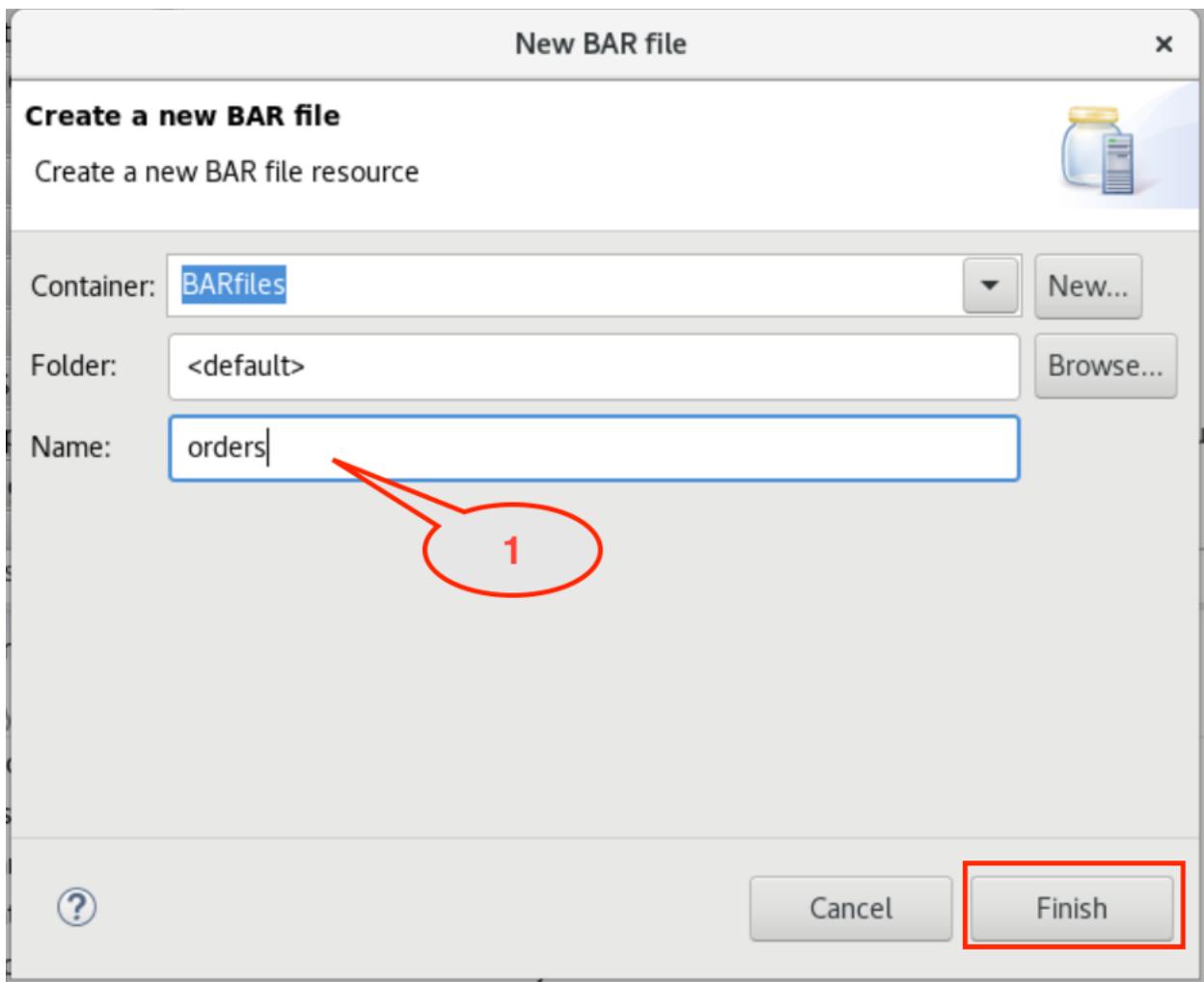
1. Select the **MQ** connection tab. App Connect Enterprise uses a mq client connection. Select **MQ client** connection properties
2. Type Destination queue manager name: **mq**
3. Type Queue manager host name: **mq-1-ibm-mq.mq.svc**
4. Type port number: **1414**
5. Type Channel name: **SYSTEM.DEF.SVRCCONN**
6. If necessary **Save the flow**.



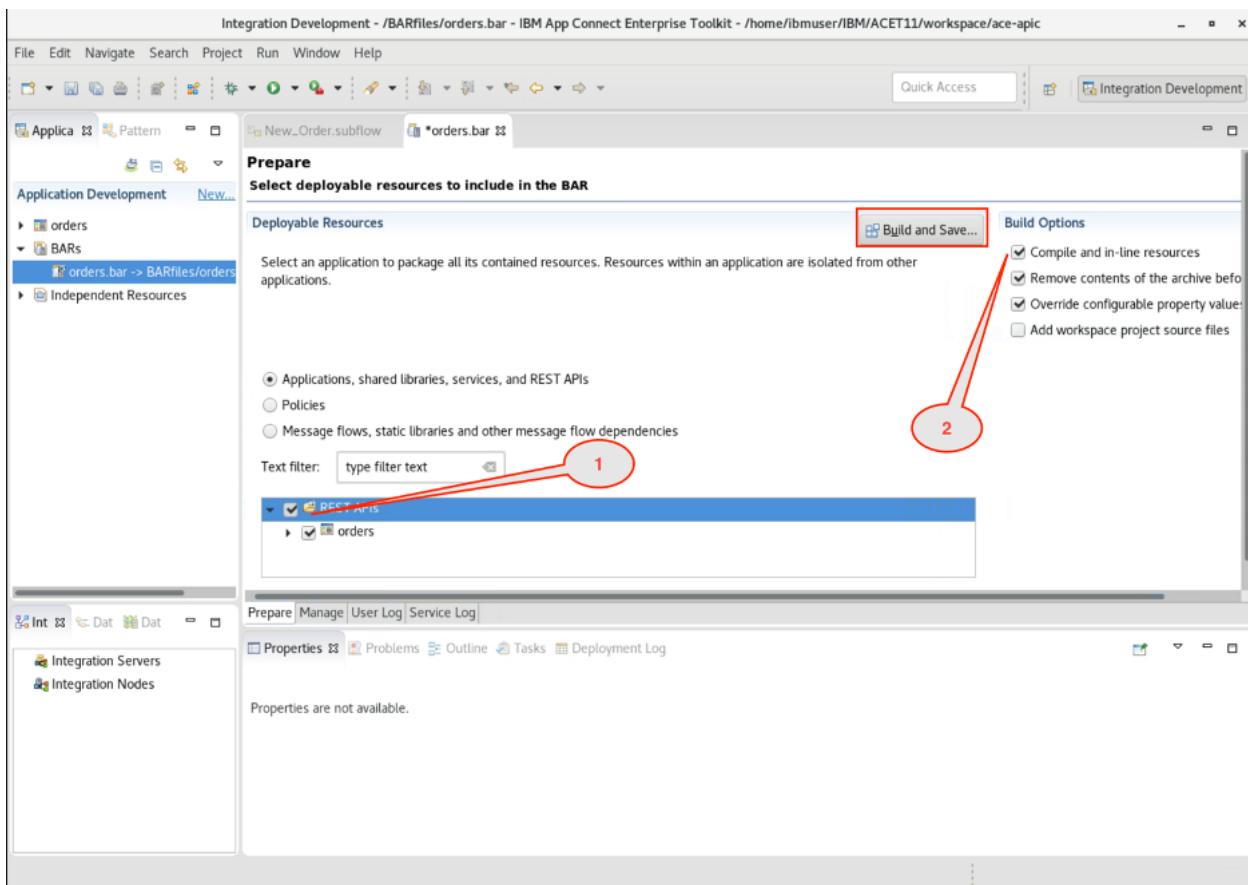
7. In the Application Development view on the left, on orders flow, right-click and then select **New -> BAR file**.



8. Type the name of BAR file: **orders** and click **Finish**. App Connect Enterprise is creating an empty BAR file



9. You need to configure which artifacts are compiled in the BAR file. Check **orders** and check **Compile and in-line resources**, then click the **Build and Save** button. A pop-up window displays “Operation completed successfully.” Click **OK**.



Task 5 - Deploy Integration BAR file as containers.

In this task, you deploy a BAR file in App Connect Enterprise Dashboard.

1. Click the bookmark bar **Cloud Pak Platform Navigator** in the browser.
Click **Skip Welcome**, Open **View instances**

openshift console Cloud Pak Platform Nav...

IBM Cloud Pak for Integration Platform Home

Welcome to IBM Cloud Pak for Integration
Let's get you going!

Create instance View instances

Create instance API Connect Create, manage and secure your APIs	Create instance App Connect Unlock the power of your data to drive new opportunities	Create instance Aspera Quickly and reliably transfer files and data sets of any size	Create instance DataPower Leverage the IBM DataPower Gateway directly deployed on Kubernetes, or embedded in API Connect
Create instance Event Streams Apache Kafka for the Enterprise	Create instance MQ Proven, enterprise-grade messaging that moves data to where it is needed	Create instance Tracing Quick diagnosis of errors and performance issues across product instances by providing end-to-end trace tracing	Create instance Asset Repository Accelerate building new workflows via design-time collaboration and active re-use of existing integration assets and assets

<https://ibm-icp4i-prod-integration.apps.demo.ibmdev.net/#>

2. Click ace-1 link to open App Connect Enterprise Dashboard.

Welcome to IBM Cloud Pak for Integration
Let's get you going!

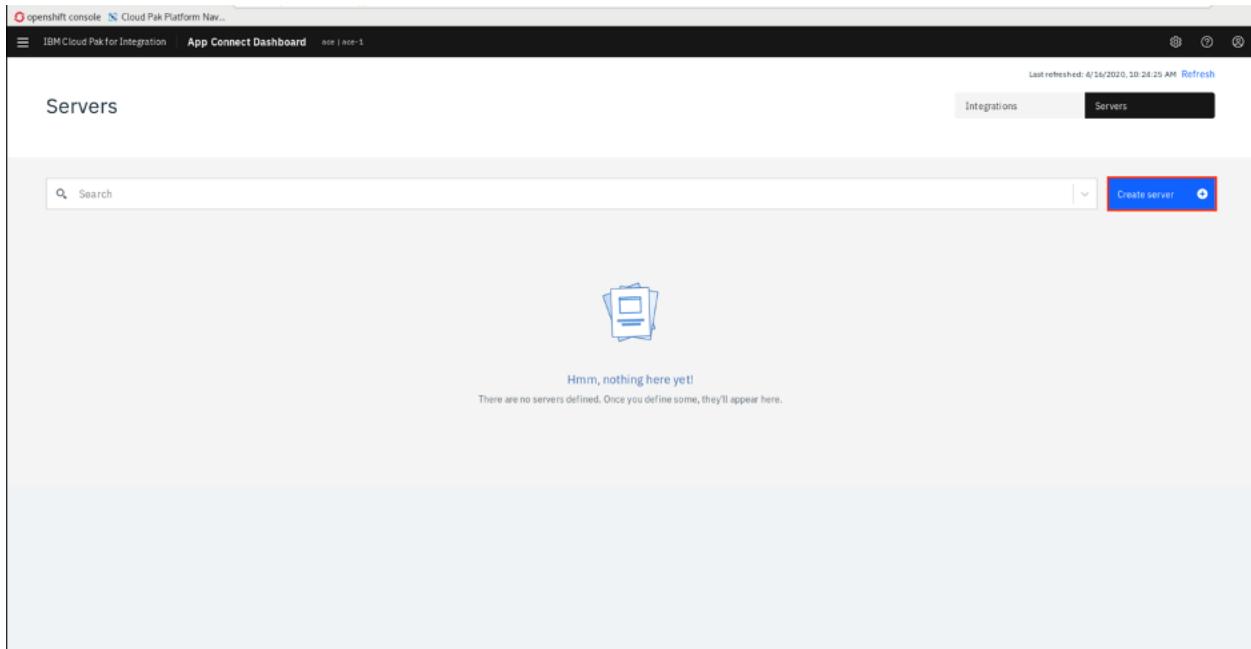
Create instance View instances

Capability type	Instance name	Namespace	Created	Version	Status
Aspera	aspera-1	aspera	a day ago		Running
MQ	mq-1	mq	10 days ago		Running
MQ	mq-2	mq	17 days ago		Running
Asset Repository	assetrepo-1	integration	20 days ago		Running
App Connect Designer	ace-design	ace	21 days ago		Running
App Connect Dashboard	ace-1	ace	21 days ago		Running
Event Streams	es-1	eventstreams	23 days ago		Running
DataPower	dp-1	datapower	23 days ago		Running
API Connect	apic-1	apic	23 days ago		Running
Tracing	tracing	tracing	23 days ago		Running

Items per page: 10 1–10 of 10 items 1 of 1 pages < >

<https://ibm-icp4i-prod-integration.apps.demo.ibmdev.net/#>

3. In the App Connect Enterprise Dashboard, you see the Integration Server deployed. To deploy the orders.bar file you saved and complied above click **Create server**.



4. Click the box **Add a BAR file**.

X

Add server

Provide a BAR file to deploy to the server

+
Add a BAR file

or use existing BAR file:

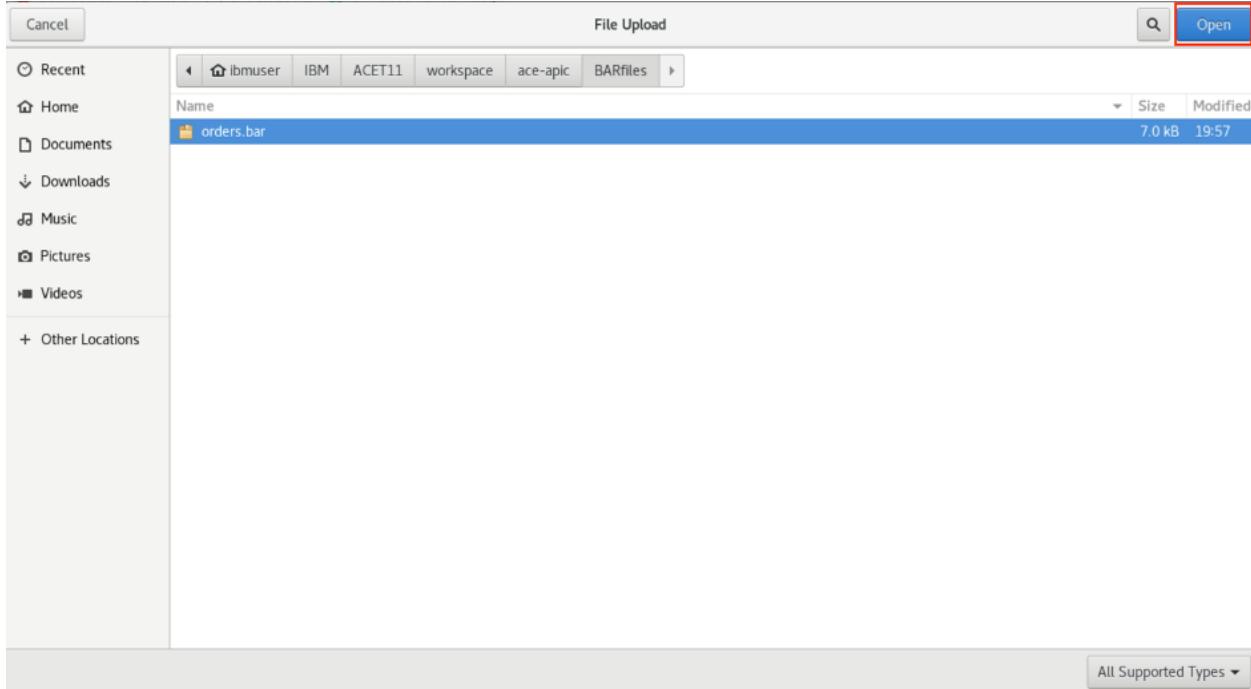
No existing BARS available

Cancel

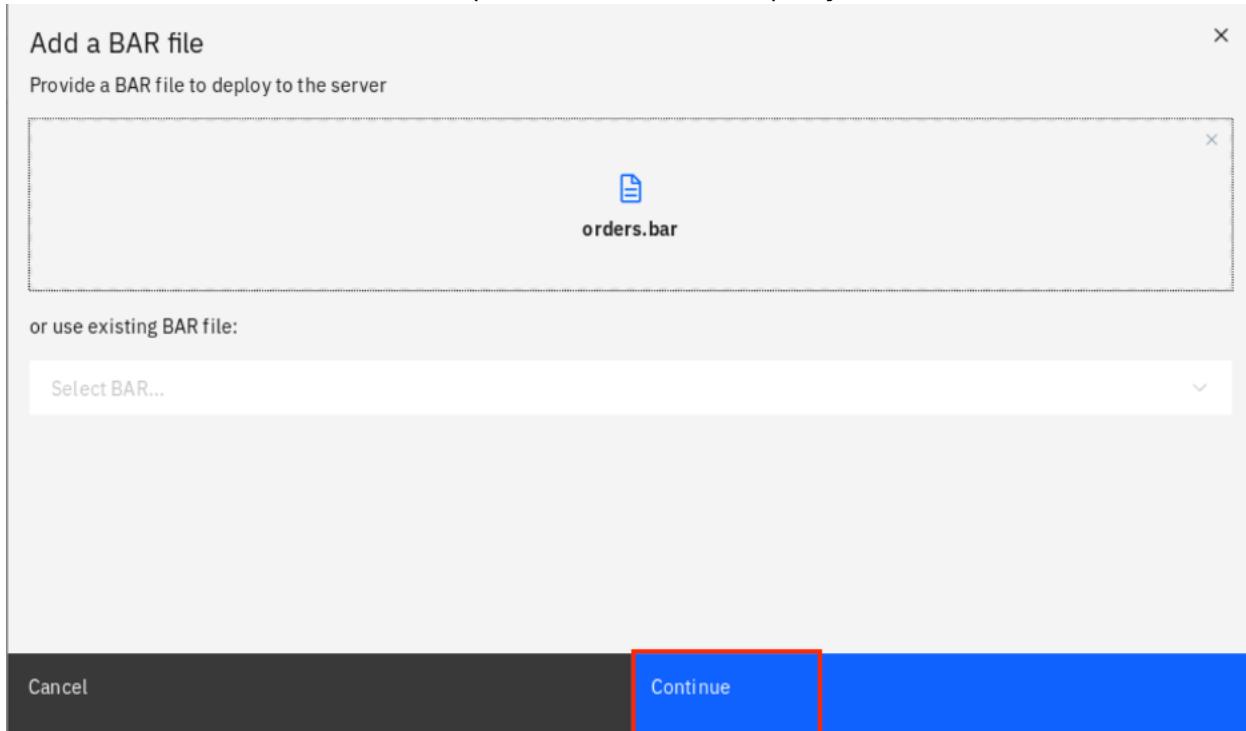
Continue

5. In the File Upload window. Open /home/ibmuser/IBM/ACET11/workspace/ace-apic/BARfiles and select

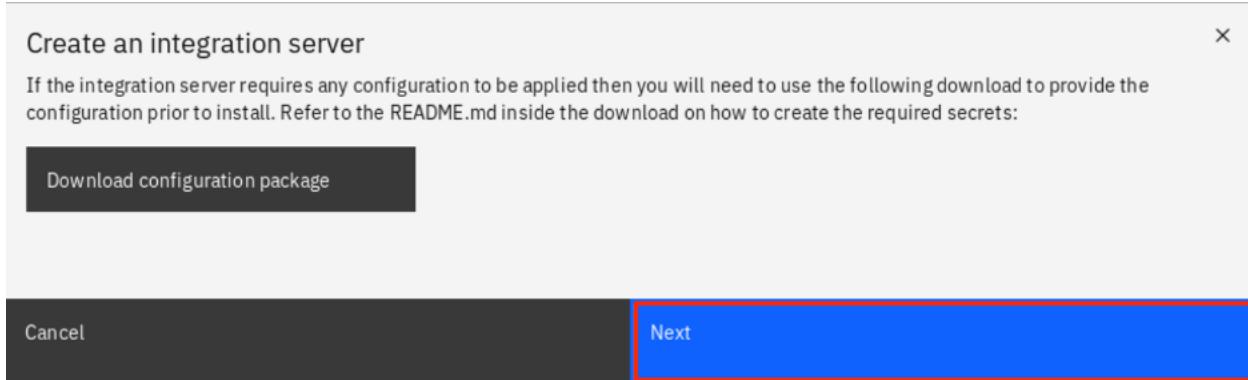
the file **orders.bar** and click **Open**. Verify the path routes to the **ace-apic** folder.



6. You see that orders.**bar** has uploaded to be deployed. Click **Continue**.



7. You don't need to download configuration package (Configuration package contains the files that you can use for App Connect Enterprise works with Databases, Event Streams, etc) click **Next** .

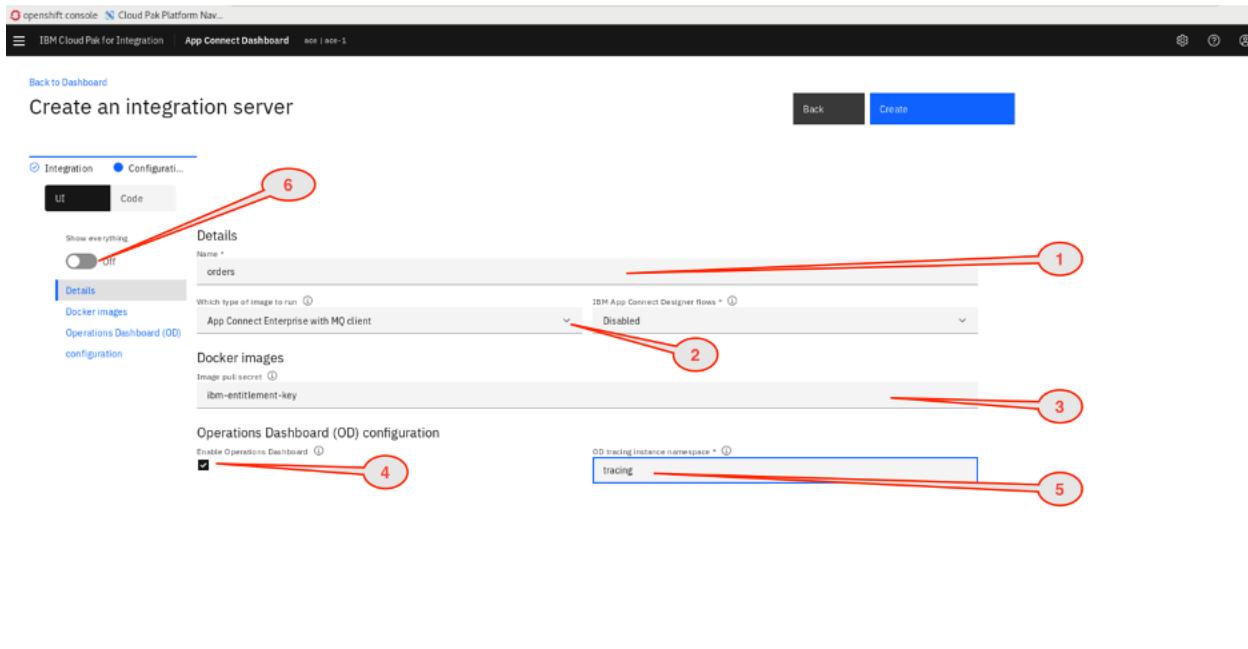


8. In the **Create an integration server** page. You have two option to deploy a BAR file. Deploy a BAR file from **App Connect Toolkit** or a BAR file from **App Connect Designer**. In this lab you deploy BAR file from App Connect Toolkit. Select **Toolkit** link and then click **NEXT** .

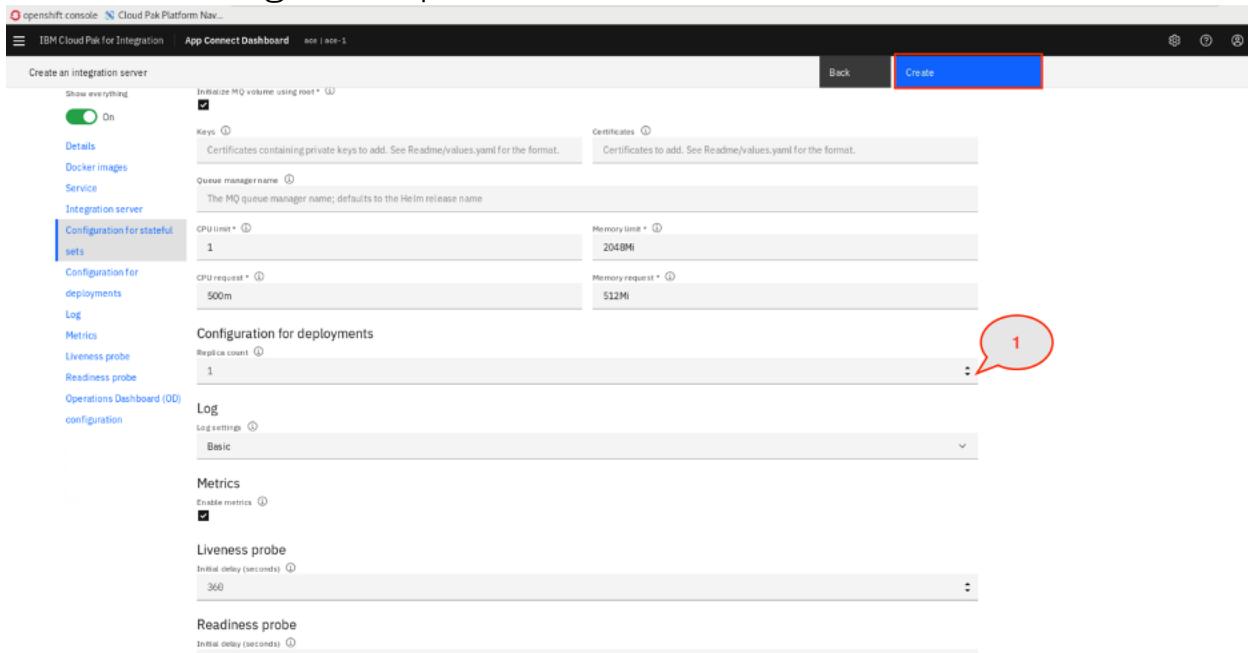


9. Enter the parameters:

1. Integration Server name: **orders**
2. Which type of image to run: **App Connect Enterprise with MQ Client**
3. Enter as Image pull secret: **ibm-entitlement-key**
4. **Check Enable Operations Dashboard**
5. Enter OO tracing instance namespace: **tracing**
6. Click **Show Everything to On** .

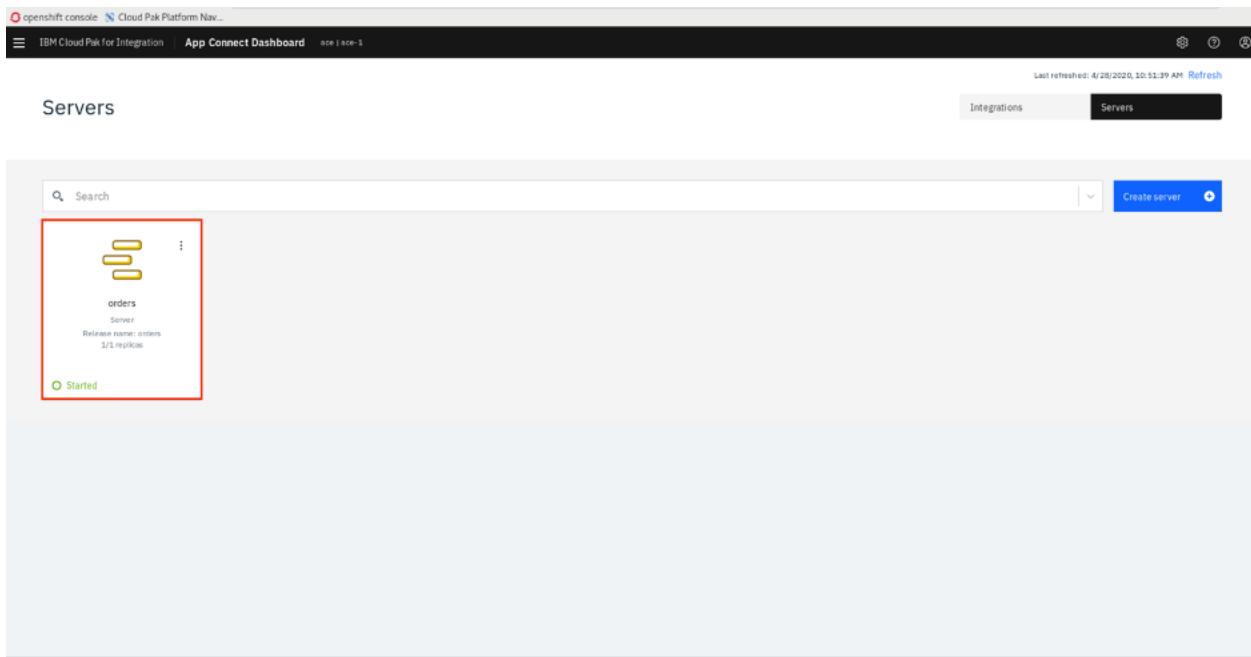


10. Scroll down and locate **Configuration for deployments** and click the **arrow** to change the replica count to **1** and then click **Create**.



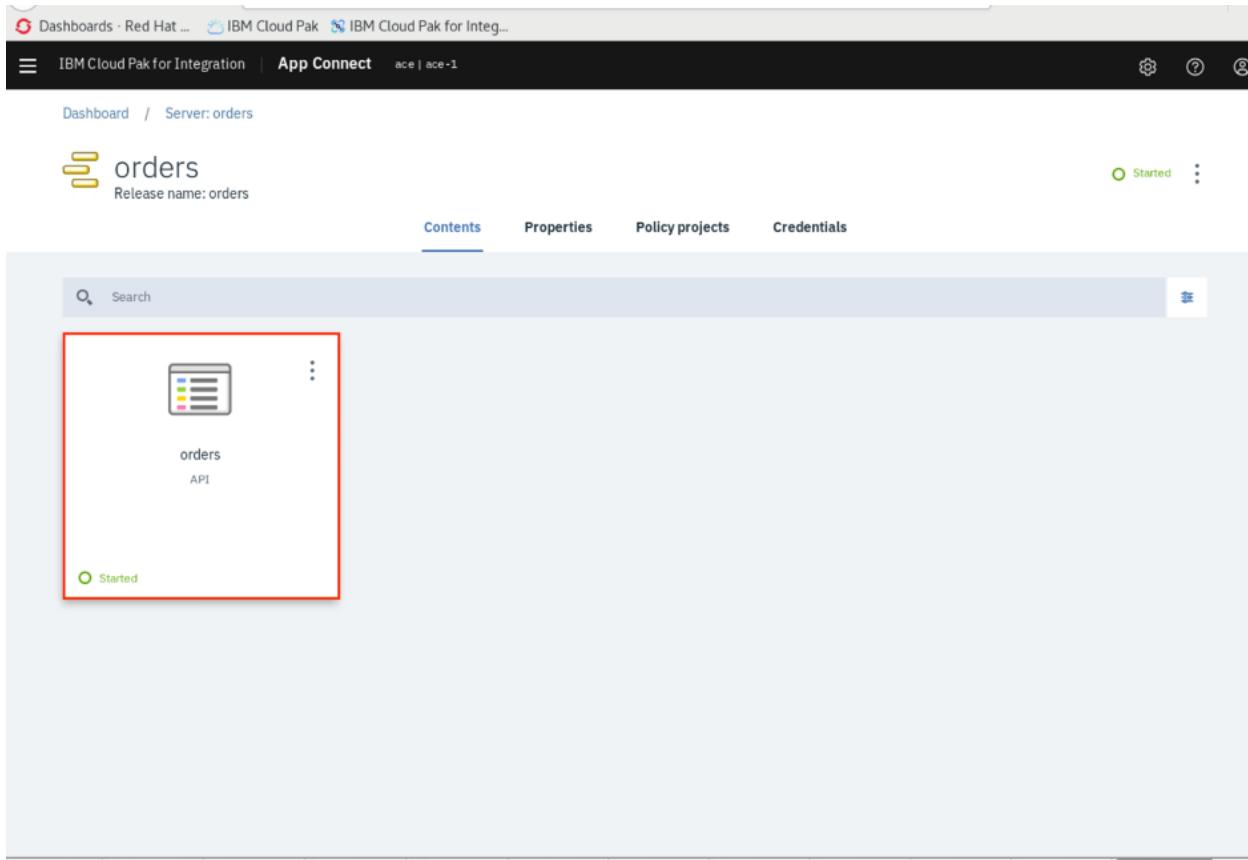
11. During the deployment process, App Connect Enterprise opens the servers page. You see the App Connect Enterprise Dashboard with the Integration Server **orders** deployed and started .Click the **orders** server icon.

Note: The deployment process takes 2-3 minutes, refresh the browser to see the BAR file .

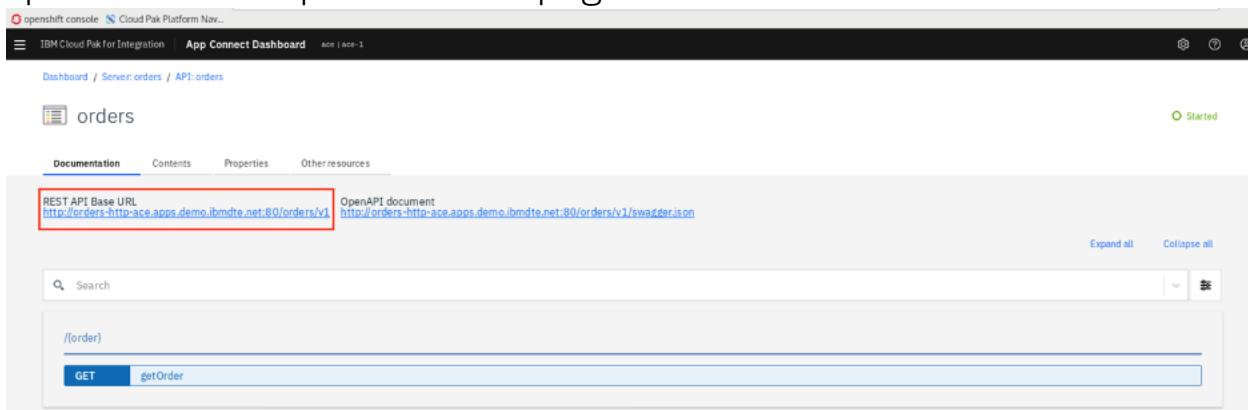


The screenshot shows the IBM Cloud Pak for Integration App Connect Dashboard. The top navigation bar includes 'IBM Cloud Pak for Integration' and 'App Connect Dashboard'. The main area is titled 'Servers' and contains a search bar and a 'Create server' button. A single server entry is displayed: 'orders' (Server), Release name: orders, 1/1 replicas, and a status of 'Started'. The 'orders' entry is highlighted with a red box.

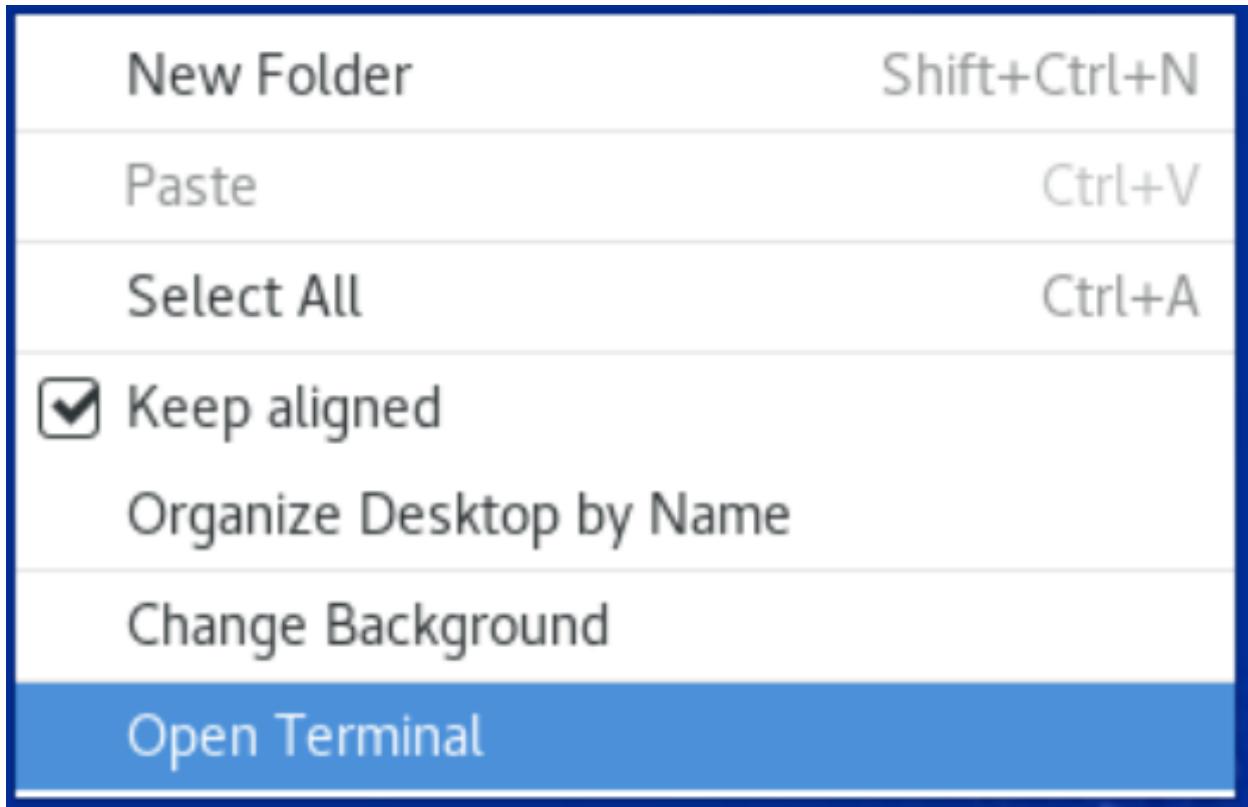
12. Click the **orders** server. Click **orders** API.



13. This page displays the REST API Base URL. you use the base URL (in the example below: <http://orders-http-ace.apps.demo.ibmdte.net:80/orders/v1>).
Keep the browser opened in this page.



14. Open a Terminal Window, right mouse on desktop workspace and select **Open Terminal**.



15. From the terminal window, execute the following curl command :

```
curl -k --request GET --url http://orders-http-  
ace.apps.demo.ibmdte.net/orders/v1/0000000
```

If the API call is successful, you see JSON reply with **{"accountid": "ABC-1234567890", "orderid": "0000000"}** .

16. You check the message arrived in queue **ORDERS** in MQ (**mq-1**).

You can check using MQ Console. Open a browser and click **Cloud Pak**

Platform Navigator bookmark bar and click “**Hamburger**” menu, and then select MQ application and click **mq-1**.

The screenshot shows the IBM Cloud Pak for Integration Platform home page. On the left, there is a sidebar with various application icons and names: Platform home, Cloud Pak Foundation, OpenShift Console, Logging, Monitoring, API Connect (1 instance), App Connect (2 instances), Aspera, DataPower, Event Streams, MQ (2 instances, highlighted with a red box), Tracing, and Asset repository (1 instance). The MQ section is expanded, showing two instances: mq-1 and mq-2. The main content area has a title "Cloud Pak for Integration" and a subtitle "you going!". It features a "View instances" button, a "Create instance" button for App Connect, and a "Create instance" button for MQ. Below the MQ section, there is a brief description: "Proven, enterprise-grade messaging that moves data to where it is needed". At the bottom left of the sidebar, it says "Platform version 3.1.0".

17. You see in **Queues on mq** window. The queue **ORDERS** has a message (look at **Queue Depth**)

The screenshot shows two tabs open in a browser window. The left tab, titled 'Local Queue Managers', lists one item: 'mq' with a status of 'Running'. The right tab, titled 'Queues on mq', lists three queues: 'AMQ_SEA1ABA7246FB05' (Local, depth 0), 'ORDERS' (Local, depth 1, highlighted with a red box), and 'Q1' (Local, depth 0). Both tabs show a total of 3 items and were last updated at 2:30:03 PM.

Task 6 – Configuring API Connect to test the integration

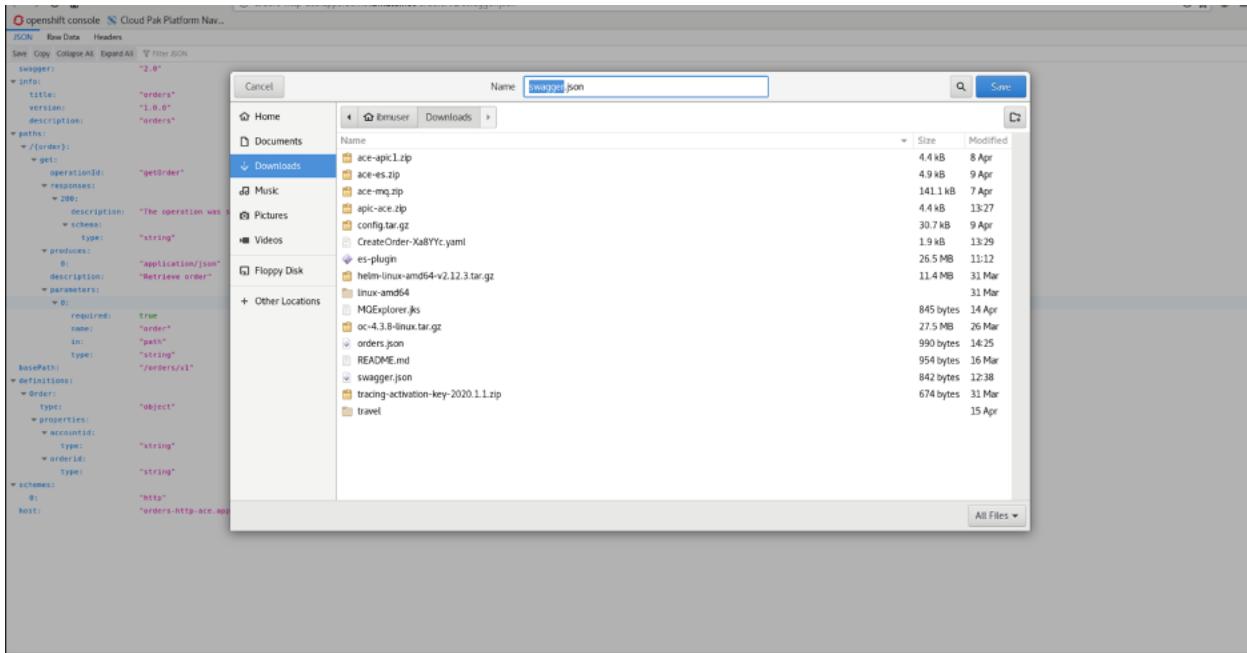
You've created an application integration flow and successfully called it via a REST API call! Now, to make it accessible to the rest of the world, it's important to add security around it—at least in the form of a client ID .This way, in addition to access control, you can get insights into which teams or customers are the least and most active. Adding security to an API is simply done via an OpenAPI configuration parameter. We can add rate limits to the API to increase the calls per second, minute, or hour to scale up as much as you need.

1. Open the browser window opened where you have the orders API window open. Click the URL under **OpenAPI document**.

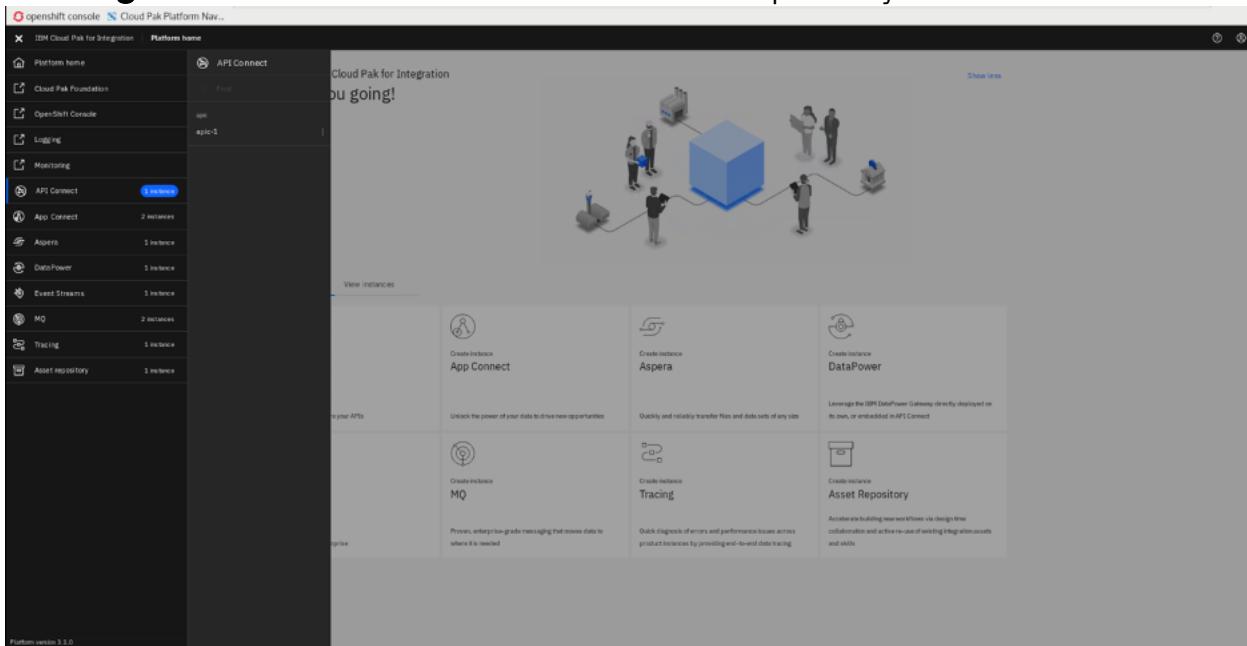
The screenshot shows the 'orders' API details page in the App Connect Dashboard. The 'Documentation' tab is selected. It displays the REST API Base URL: <http://orders-http-ace.apps.demo.ibmmtte.net:80/orders/v1>. Below this, there is a search bar and a code editor area containing a Swagger JSON snippet. The URL for the OpenAPI document is also present in the documentation section.

2. A new browser tab opens, containing the swagger file. Right-click anywhere in the body of the text and choose **Save as**. Click **Save** to save the swagger.json file.

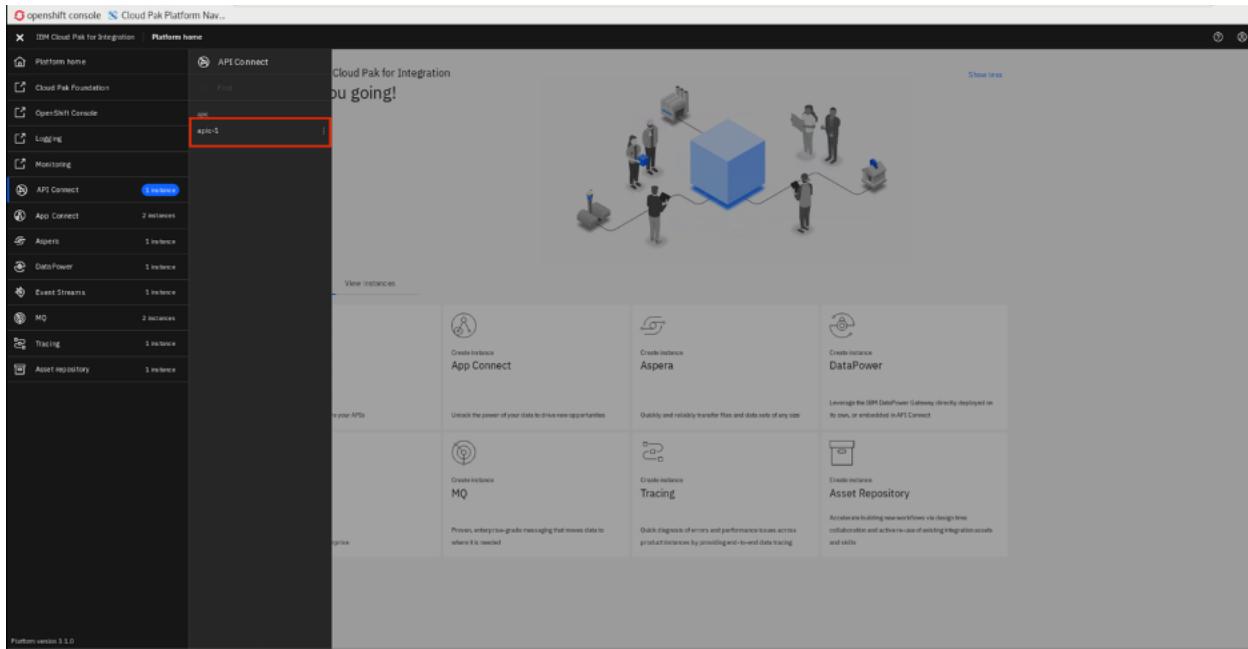
Note: You might need to confirm to replace.



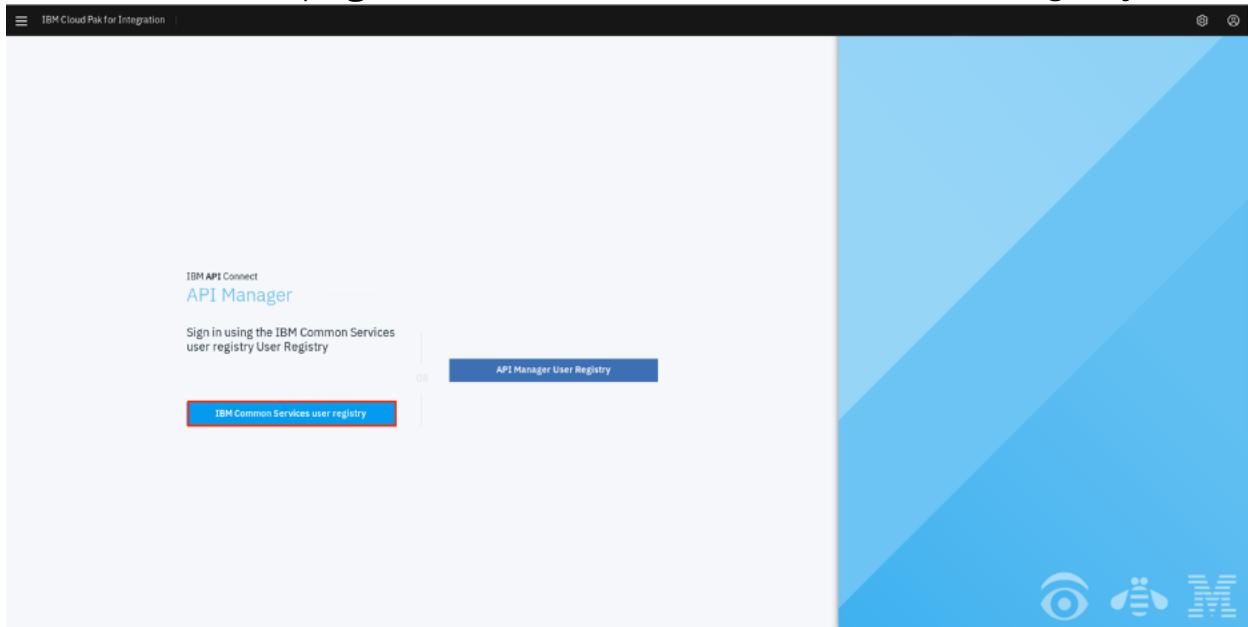
3. Cloud Pak Navigator has a bookmark that you access all services. Click “Hamburger” menu and select API Connect capability.



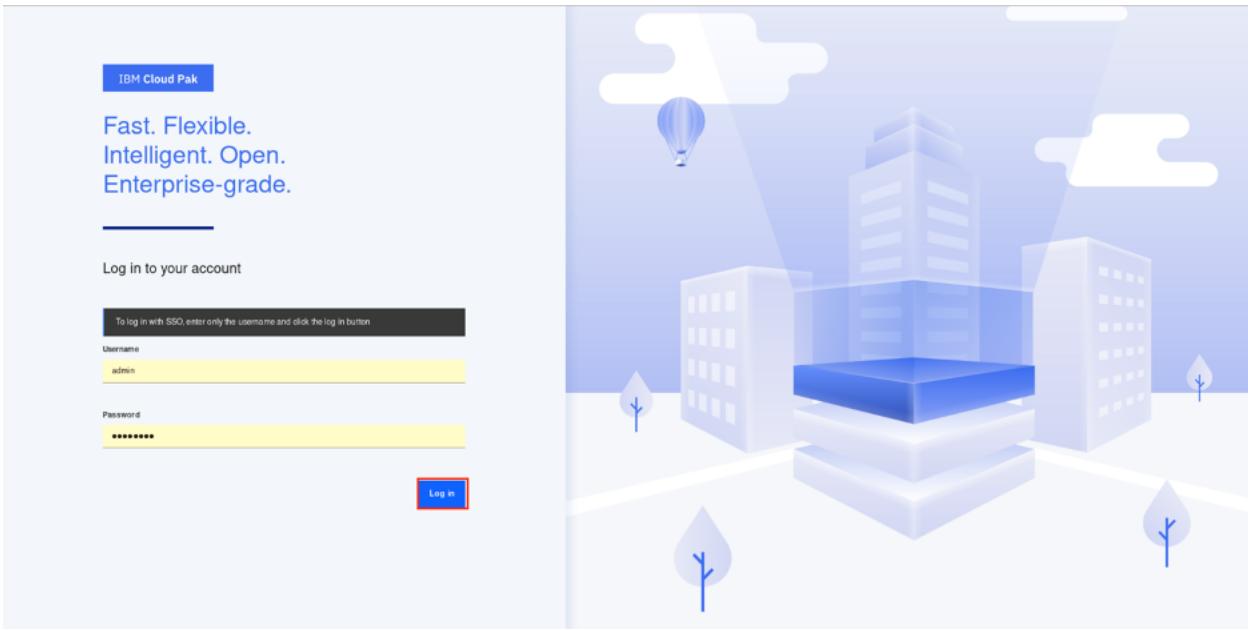
4. Click **apiic-1** instance.



5. In the API Connect page, click **IBM Common Services user registry**.



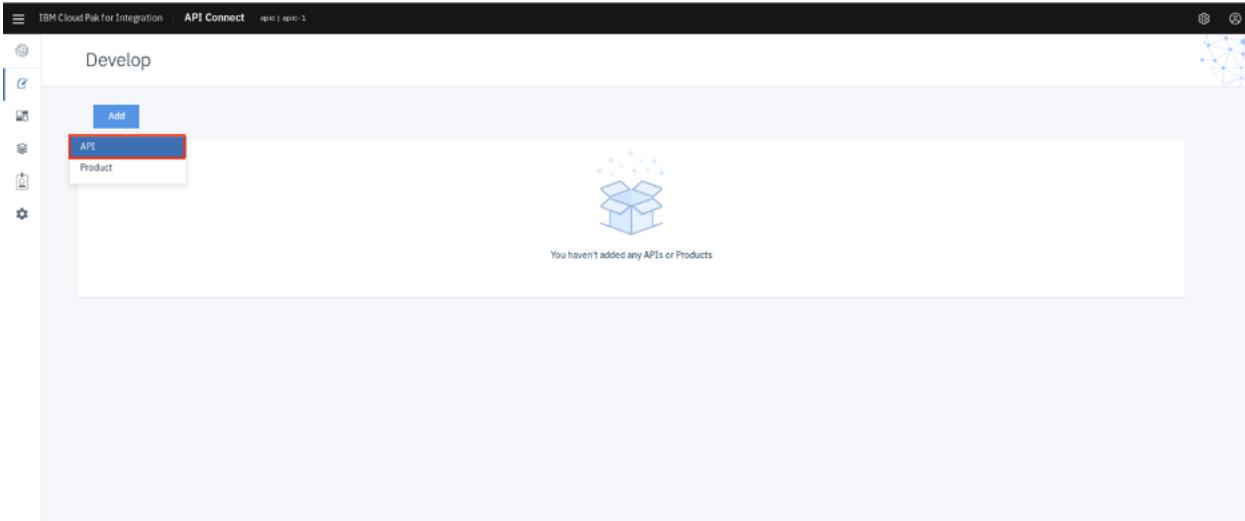
6. You might need to login to IBM Common Services. Juist click **Log in**



7. Click Develop APIs and Products .

The image shows the API Manager dashboard. At the top, there's a navigation bar with "IBM Cloud Pak for Integration" and "API Connect" and a search bar. Below the navigation is a "Welcome to the API Manager" message. The main area has several cards: "Develop APIs and Products" (highlighted with a red box), "Manage Catalogs", "Manage Resources", "Manage Settings", "Learn more", and "Connect". The "Develop APIs and Products" card contains the subtext: "Edit, assemble, secure and test APIs. Package APIs using products for publishing to consumers."

8. Click Add, then choose API from the drop-down menu.



9. Choose **From an existing OpenAPI service**, scroll down and click **Next**.

Add API

Create

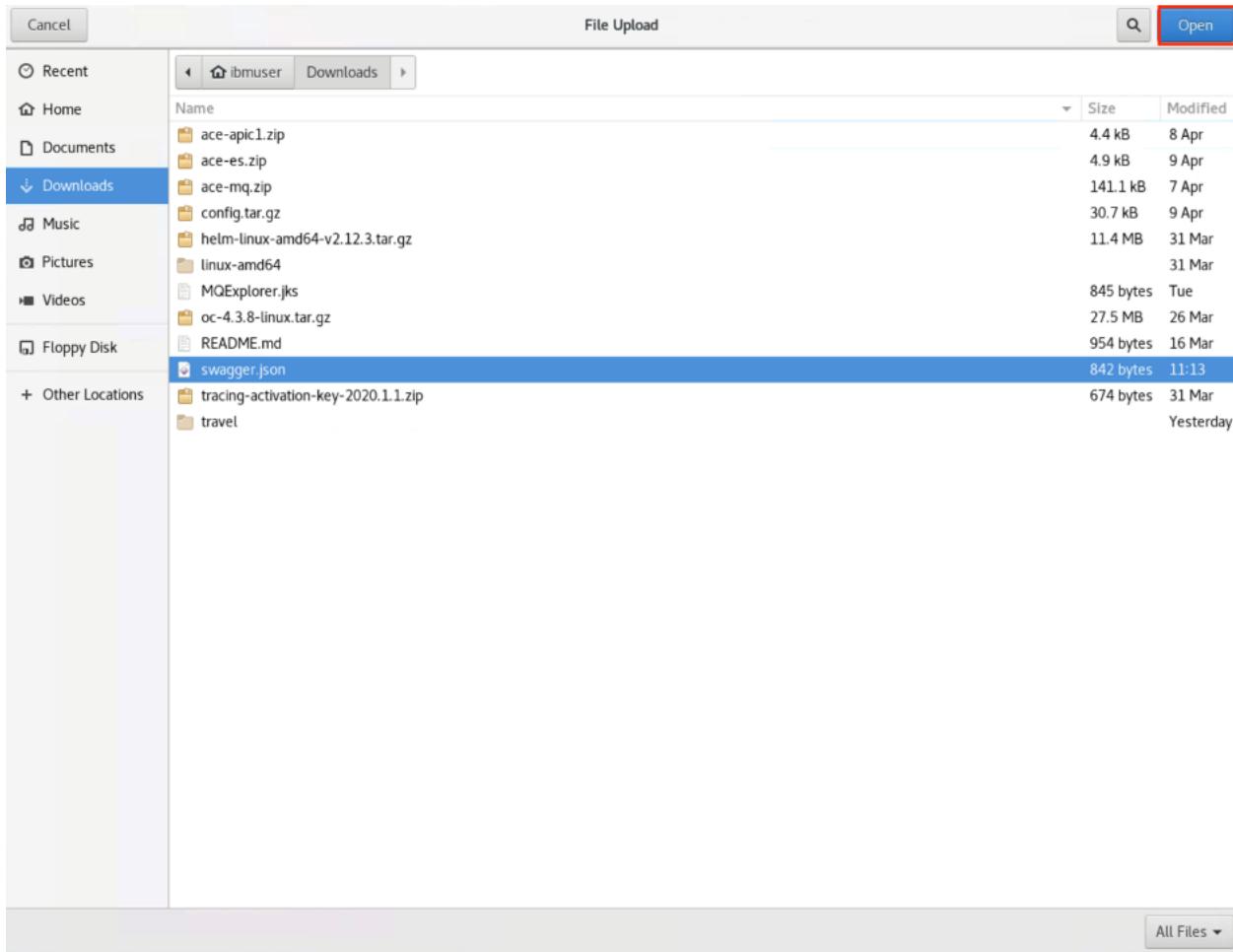
- From target service
- From existing OpenAPI service
- From existing WSDL service (SOAP proxy)
- From existing WSDL service (REST proxy)
- New OpenAPI

Import

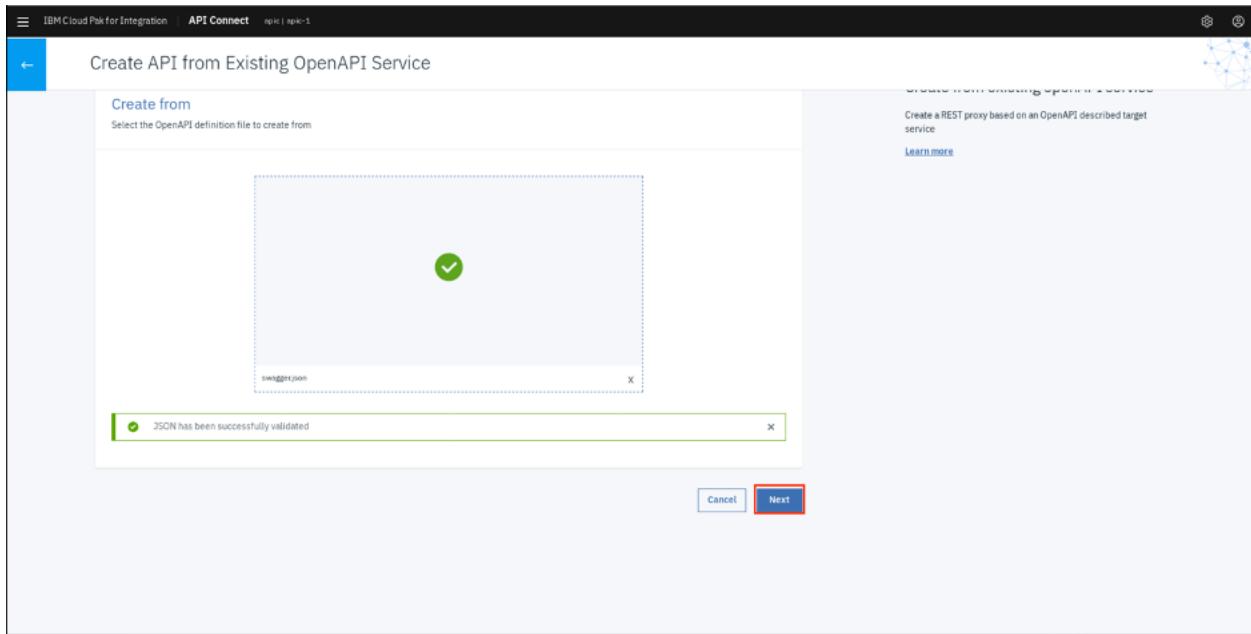
- Existing OpenAPI
- From asset repository

Diagram: App → REST → API Proxy (green circle) → REST → Target Endpoint

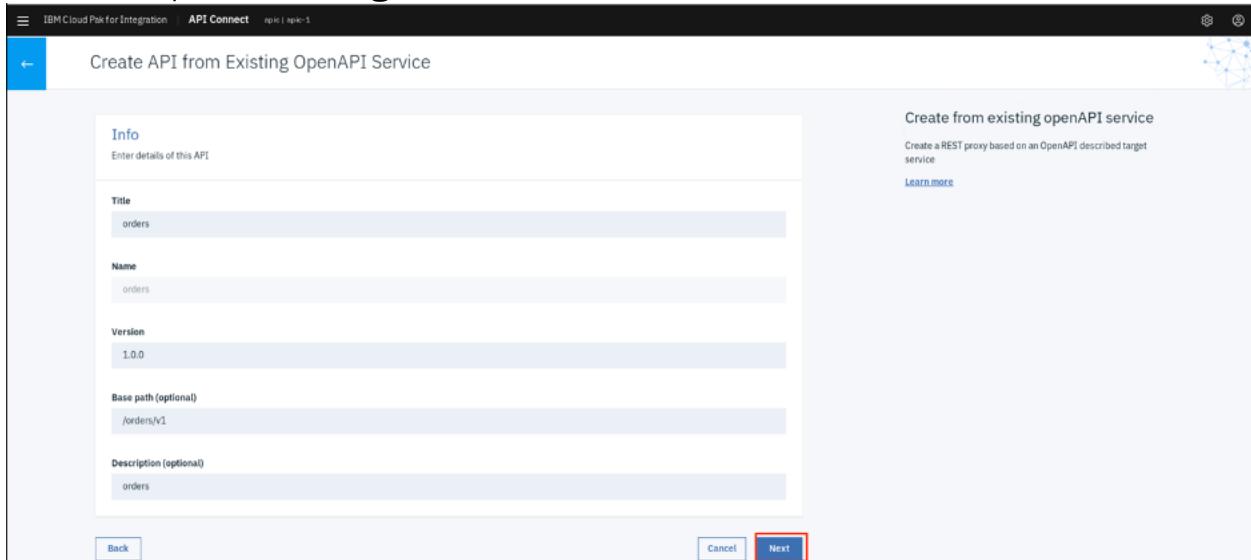
10. Click **Browse**, select the downloaded **swagger.json** file (The file is located under the /home/ibmuser/Downloads/swagger.json directory) and click **Open**.



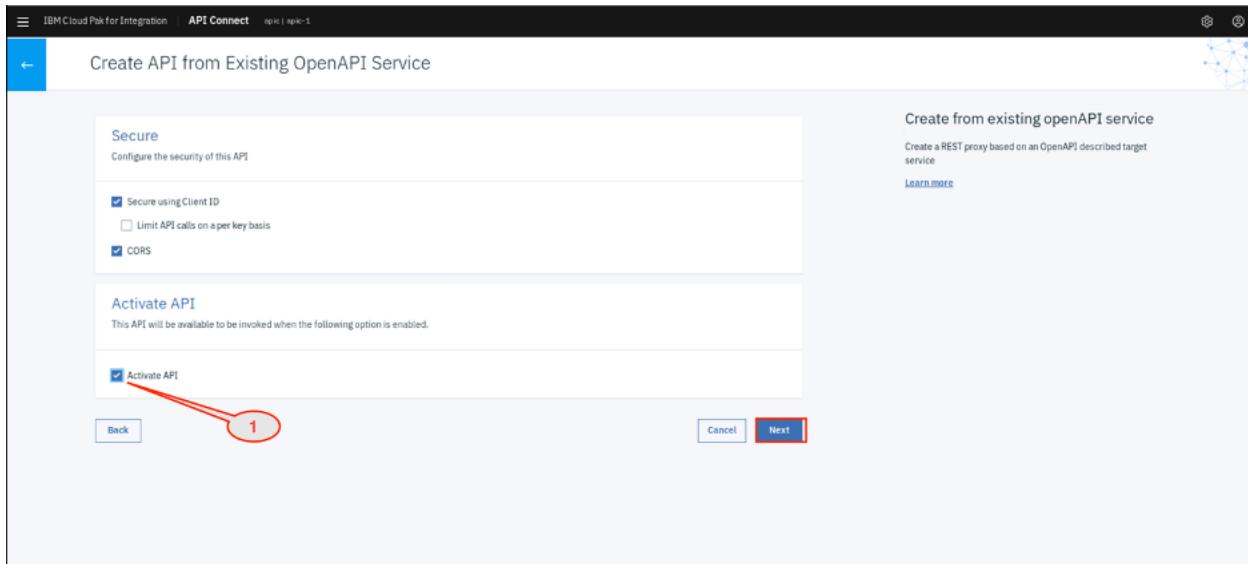
11. Make sure the JSON is successfully validated and then click **Next**.



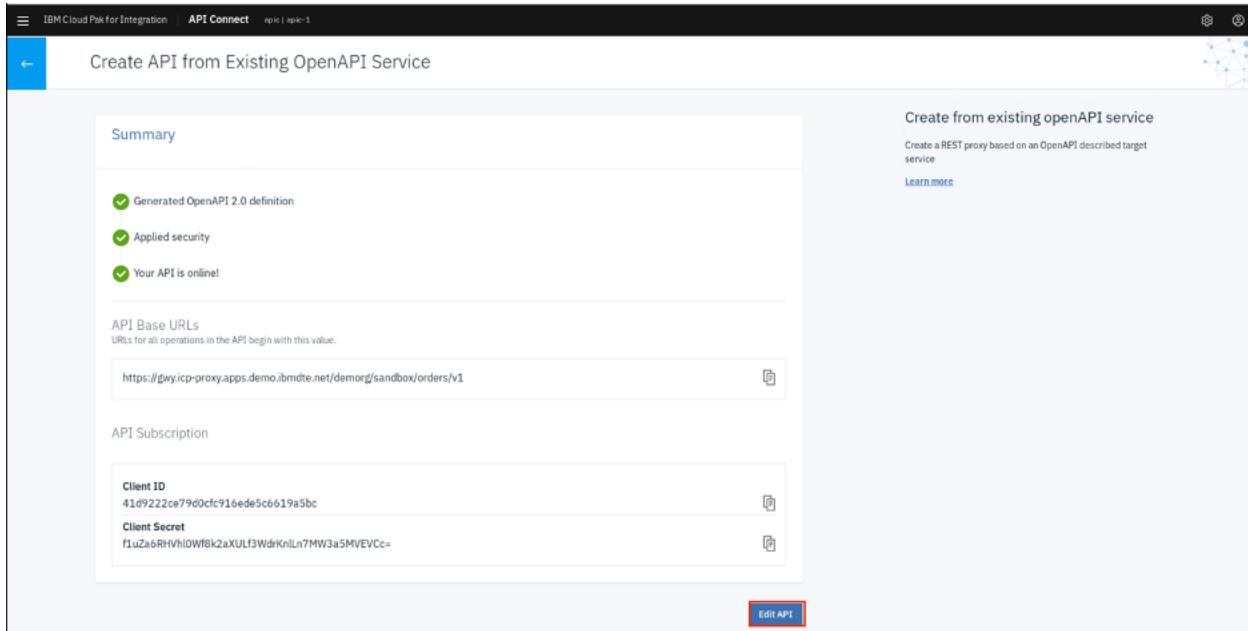
12. Keep the settings and click **Next**.



13. Under "Secure using Client ID", "CORS" and Check "Activate API". Click **Next**.



14. Your API with Client ID is created! Click **Edit API**.



15. In the API Setup page. You configure your API.

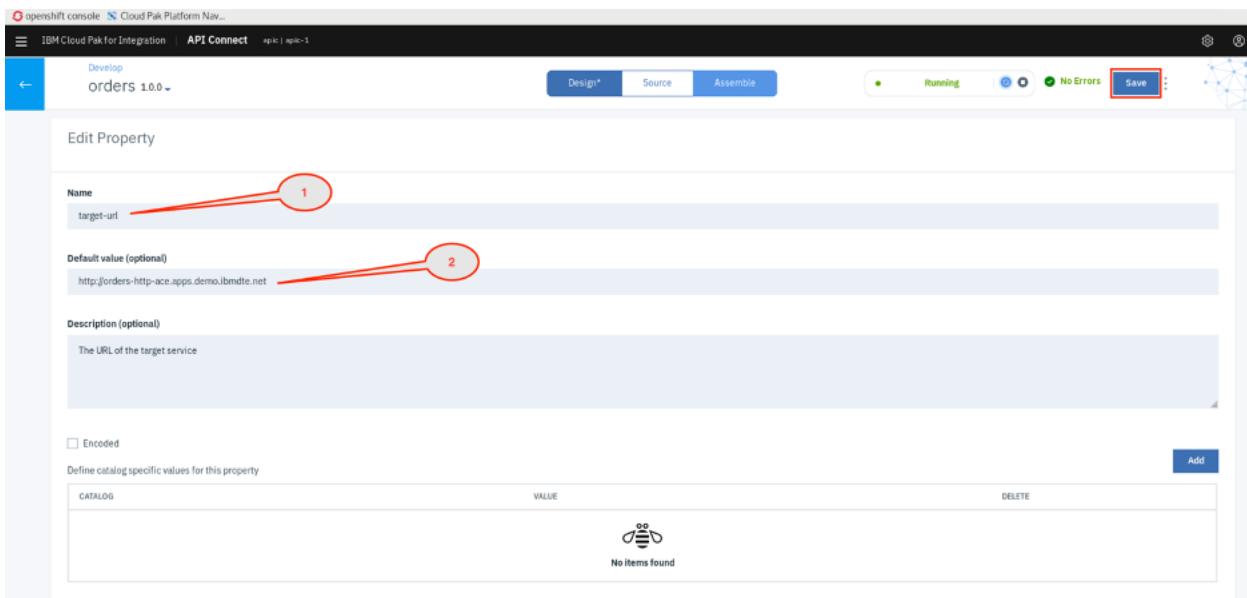
The screenshot shows the 'Info' section of the 'orders' API. The 'Title' is set to 'orders', 'Name' to 'orders', and 'Version' to '1.0.0'. The 'Description (optional)' field contains the word 'orders'. Under 'Schemes', the 'HTTPS' checkbox is selected. The left sidebar shows navigation options like 'API Setup', 'Properties', and 'Target Services'.

16. In the Design page, click **Properties** and the click **target-url** link, to enter the target gateway.

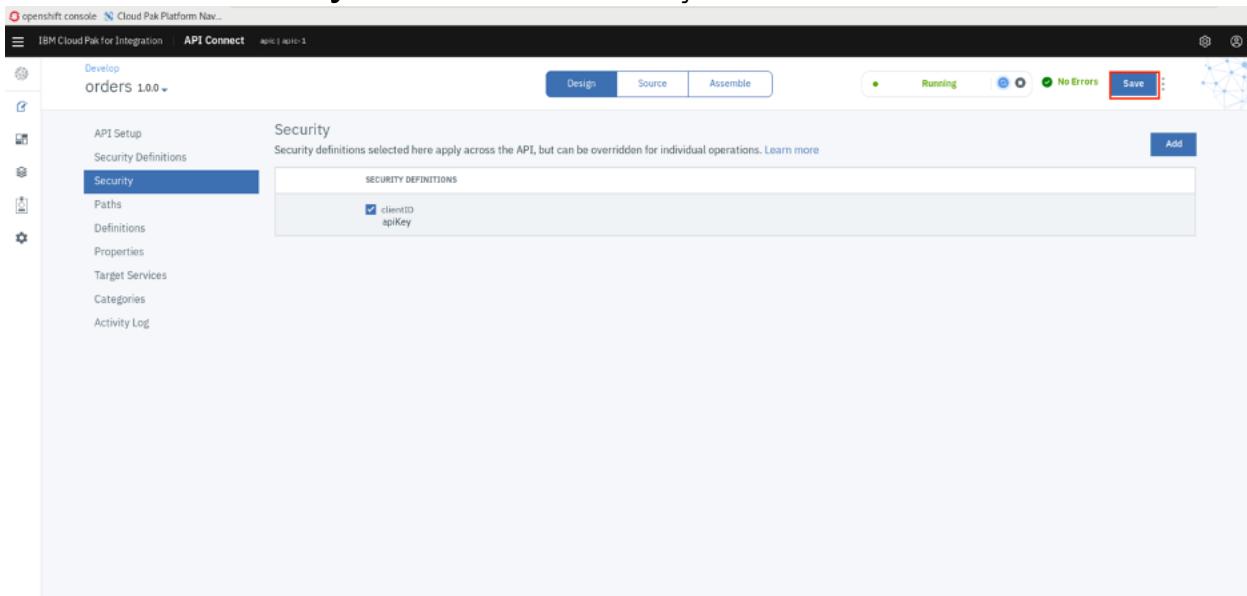
The screenshot shows the 'Properties' section of the 'orders' API. A red box highlights the 'target-url' entry in the table, which has a value of 'false'. A red circle with the number '1' points to the 'Properties' link in the left sidebar. The table columns are 'PROPERTY NAME', 'ENCODED', and 'DESCRIPTION'.

PROPERTY NAME	ENCODED	DESCRIPTION
target-url	false	The URL of the target service

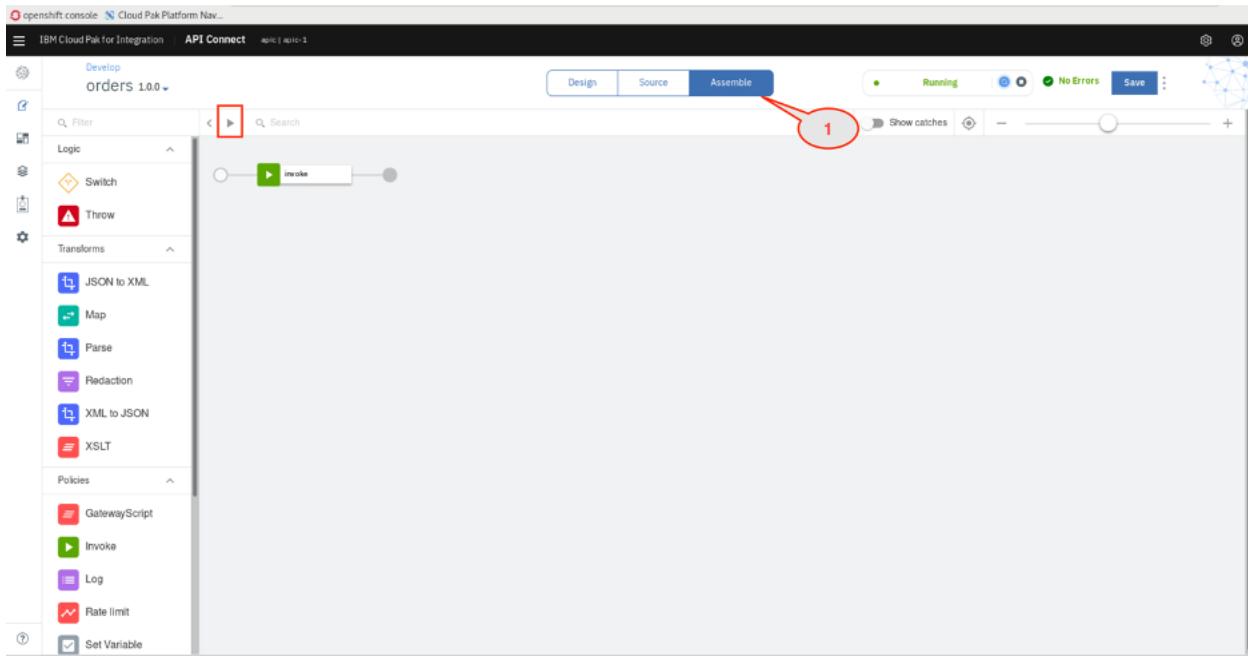
17. Enter as Name: **target-url** and Default Value (optional): <http://orders-http-ace.apps.demo.ibmdte.net>.



18. Go to **Security**. Check the Security Definitions click **Save** .



19. To test the API, Click **Assemble**, then click the **Test button** .



20. In the Test window, click the **operation** and choose **get(order)**. After the API is published, the Catalog, Product, Plan, and Application are displayed under setup. The default plan has a rate limit of 100 calls per hour.

openshift console Cloud Pak Platform Nav...

IBM Cloud Pak for Integration | API Connect apic | apic-1

Develop
orders 1.0.0 ▾

Test X

Setup

Catalog: sandbox

Product: orders-auto-product

Plan: Default Plan

Application: sandbox-test-app

[Republish product](#)

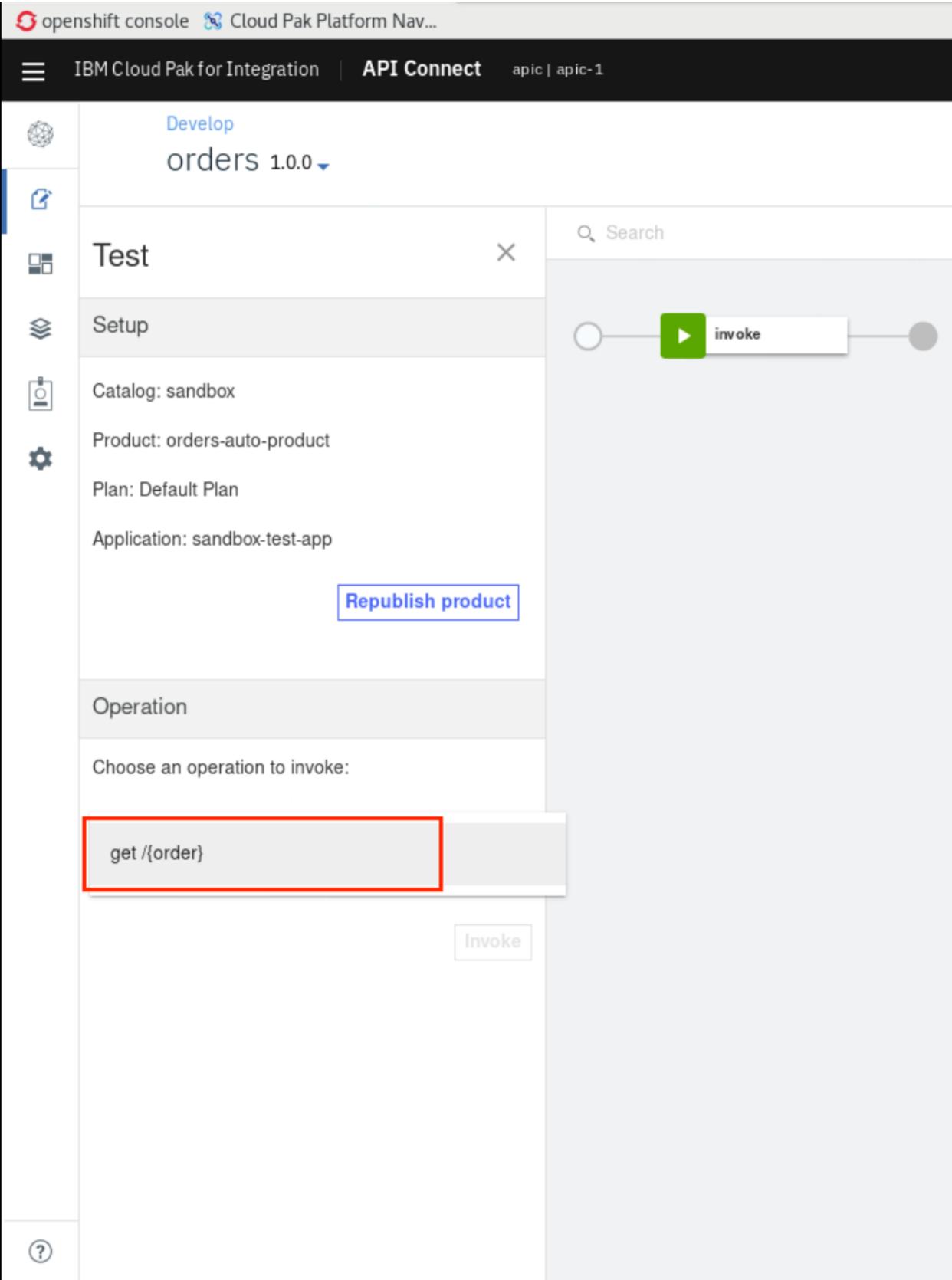
Operation

Choose an operation to invoke:

get /{order}

[Invoke](#)

(?)



The screenshot shows the 'Test' tab of the 'orders' product in the IBM Cloud Pak for Integration API Connect interface. The 'invoke' button is highlighted with a red border, indicating it is the active or selected action. The 'Operation' section displays the 'get /{order}' endpoint.

21. The clientId fields auto-populate with the test application values. Under parameters under **order**, enter a value **0000**.

The screenshot shows the IBM Cloud Pak for Integration API Connect interface. On the left, there's a sidebar with icons for Develop, Test, Configuration, and Help. The main area is titled "Develop orders 1.0.0". A "Test" dialog is open, showing a flow diagram with an "invoke" step. The "Test" dialog has the following sections:

- Choose an operation to invoke:** Operation dropdown set to "get /{order}".
- Identification:** Client ID input field containing "a366f86309c3d9d02ce3563c6cfba55f".
- parameters:** A red box highlights the "order" parameter input field, which contains "0000".
- Generate:** A "Repeat" checkbox is unchecked. Below it, "Stop after:" is set to "10" and "Stop on error" is checked.
- Buttons:** A "Invoke" button at the bottom right.

22. Click **Invoke**. Scroll down and see Body and check the results. You see a status code: 200 created with a response body containing the results details.

openshift console Cloud Pak Platform Nav...

IBM Cloud Pak for Integration | API Connect apic | apic-1

Develop
orders 1.0.0 ▾

Generate

Repeat

Repeat the API invocation a set number of times, or until the stop button is clicked

Stop after: 10 Stop on error

Invoke

Response

Status code:
200 OK

Response time:
178ms

Headers:

```
cache-control: private
content-type: application/json; charset=utf-8
x-ratelimit-limit: name=rate-limit,100;
x-ratelimit-remaining: name=rate-limit,99;
```

Body:

```
{ "accountid": "ABC-1234567890", "orderid": "0000" }
```

1

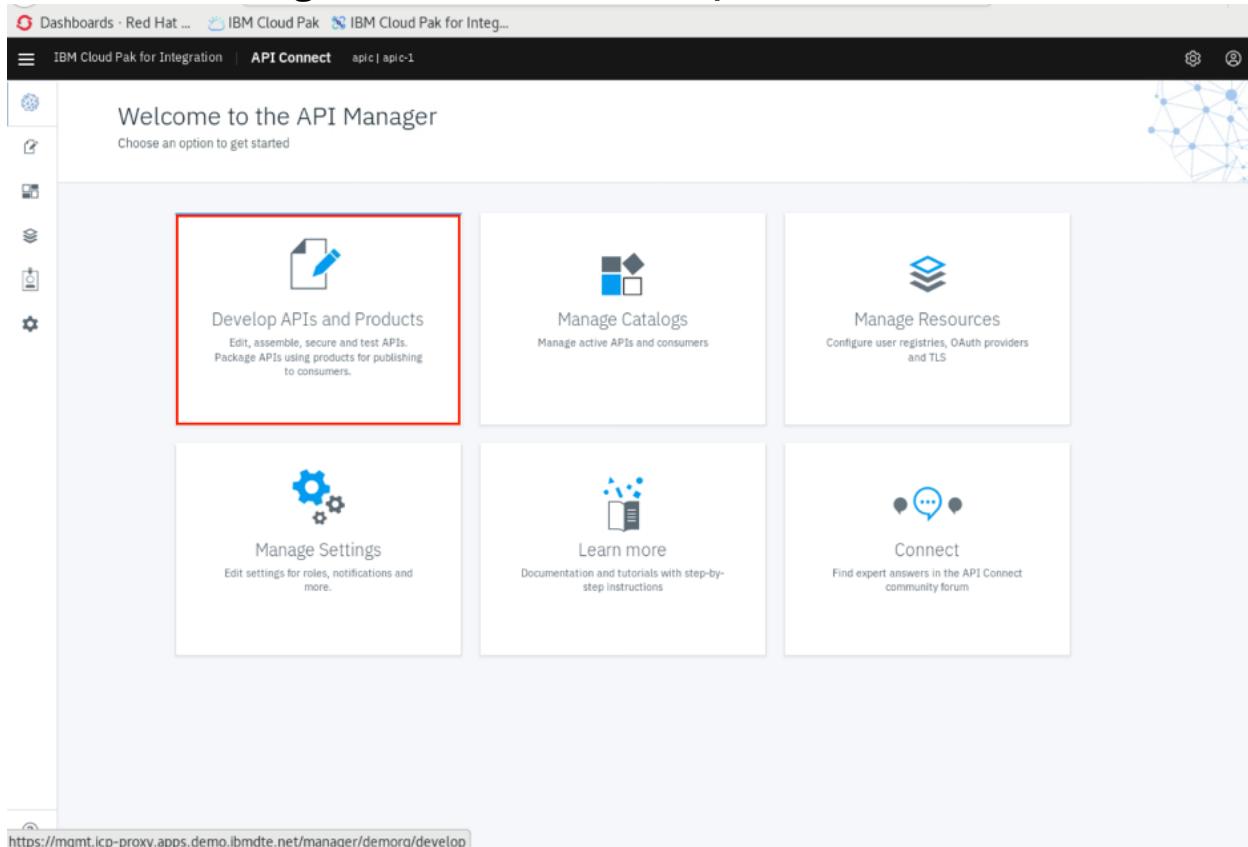
```
graph LR; Start(( )) --> Invoke[invoke]; Invoke --> End(( ));
```

Task 7 - Share the API

Now that you've built, secured, published, and tested your API, the last step is to add it to the Asset Repository. With the Asset Repository, your organization can store, manage, and share all of your integration assets in one central location. Sharing assets in this way increases

collaboration between teams, avoids unnecessary duplication and boosts productivity.

1. To push our new API to Asset Repository we must return to the API Manager. From the Cloud Pak for Integration home page, choose **apic-1**.
2. In the API Manager screen, click **Develop APIs and Products**.



The screenshot shows the 'Welcome to the API Manager' screen. It features a sidebar with icons for Dashboards, API Connect, and Settings. The main area has six cards:

- Develop APIs and Products** (highlighted with a red box): Edit, assemble, secure and test APIs. Package APIs using products for publishing to consumers.
- Manage Catalogs**: Manage active APIs and consumers.
- Manage Resources**: Configure user registries, OAuth providers and TLS.
- Manage Settings**: Edit settings for roles, notifications and more.
- Learn more**: Documentation and tutorials with step-by-step instructions.
- Connect**: Find expert answers in the API Connect community forum.

The URL in the address bar is <https://mqmt.icp-proxy.apps.demo.ibmdev.net/manager/develop>.

3. Returning to the Develop Screen, select the menu next to your API and from the drop-down, click **Push to asset repository**.

The screenshot shows the IBM Cloud Pak for Integration API Connect interface. The top navigation bar includes 'IBM Cloud Pak for Integration', 'API Connect', and 'apic | apic-1'. On the left, there's a sidebar with icons for Home, Catalog, Products, Services, and Settings. The main area is titled 'Develop' and contains a section for 'APIs and Products'. Two items are listed in the table:

TITLE	TYPE	LAST MODIFIED
orders-1.0.0	API (REST)	Today at 10:43 PM
orders auto product-1.0.0	Product	Today at 10:43 PM

Below the table are pagination controls: 'Items per page' set to 10, '1-2 of 2 items', '1 of 1 pages', and a download icon. A context menu is open over the first row ('orders-1.0.0'), with options: Publish, Stage, Save as a new version, Download, Push to asset repository (which is highlighted with a red box), and Delete.

4. Once the upload completes, you see a success dialogue at top-right.

IBM Cloud Pak for Integration API Connect apic | apic-1

Develop

APIs and Products

TITLE	TYPE	LAST MODIFIED
orders-1.0.0	API (REST)	Today at 10:43 PM
orders auto product-1.0.0	Product	Today at 10:43 PM

Items per page: 10 | 1-2 of 2 items

1 of 1 pages | < | 1 | >

5. To access the Asset Repository, click **assetrepo-1**.

openshift console | Cloud Pak Platform Nav...

IBM Cloud Pak for Integration | Platform home

Asset repository

Cloud Pak for Integration

You going!

Show less

assetrepo-1

View instances

Create instance App Connect

Create instance Aspera

Create instance DataPower

Unlock the power of your data to drive new opportunities

Quickly and reliably transfer files and data sets of any size

Leverage the IBM DataPower Gateway directly deployed on its own, or embedded in API Connect

Create instance MQ

Create instance Tracing

Create instance Asset Repository

Accelerate building new workflows via design-time collaborative planning and reuse re-use of existing integration assets and skills

<https://ibm-icp4-prod-integration.apps.demo.ibmmtie.net/#>

6. You see the orders asset you pushed from API Connect in the previous tasks.

The screenshot shows the Asset repository interface in IBM Cloud Pak for Integration. The top navigation bar includes 'IBM Cloud Pak for Integration', 'Asset repository', and 'integration | assetrepo'. On the right, there are icons for settings and help. The main title is 'Asset repository' with a 'Upload progress' link. Below it, tabs for 'Browse Assets', 'Remotes', and 'Access Control' are visible, with 'Browse Assets' being the active tab. A search bar says 'Search Assets' with placeholder text 'Search for assets, tags, types or owners...'. A modal window displays details for the 'orders' asset: 'Open API specification', 'Version: 1.0.0', 'Tags: apic openapi', 'Owner: admin', and 'Modified: a few seconds ago'. At the bottom left, it says 'Showing 1 of 1 assets'. A table lists the asset with columns: Name, Owner, Tags, Modified, and actions. The asset row shows 'orders', 'admin', 'apic openapi', 'a few seconds ago', and edit/delete icons. At the bottom, it says 'Items per page: 10' and '1-10 items'.

Name	Owner	Tags	Modified	
orders	admin	apic openapi	a few seconds ago	

7. Click the ellipsis and choose **Open** from the drop-down menu.to check orders API.

The screenshot shows the Asset repository interface. At the top, there are tabs for 'Browse Assets', 'Remotes', and 'Access Control'. A search bar is present. Below it, a card displays an asset named 'orders' with version 1.0.0. The 'Open' button in the card's context menu is highlighted with a red box. The card also shows tags: 'api' and 'openapi', owner: 'admin', and modified: '3 hours ago'. Below this, a table lists the asset 'orders' with columns: Name, Owner, Tags, Type, and Modified. The table shows 'orders' with owner 'admin', tags 'api' and 'openapi', type 'Open API specification', and modified '3 hours ago'. There are buttons for 'Add assets' and 'Upload progress' at the top right.

8. You check the API overview.

The screenshot shows the API overview page for the 'orders' asset. The left sidebar has 'Visualiser' selected. The main area has 'Overview' selected. It shows the asset name 'orders', download link for the Open API Document, REST type, endpoint 'https://\$(catalog.host)/orders/v1', and security information ('clientID' and 'apiKey located in header'). On the right, there is an 'Asset metadata' panel with fields: File name 'asset.json', Type 'Open API specification', Modified '3 hours ago', Owner 'admin', Description 'orders', and Tags 'api' and 'openapi'. Buttons for 'Delete asset' and 'Download' are at the top right.

9. Click **Get{/order}** link you see the API parameters.

The screenshot shows the Asset Repository interface for the 'orders' asset. On the left, there's a sidebar with 'Overview' and 'Definitions' tabs, where 'Definitions' is selected. The main area displays the 'getOrder' API endpoint. It includes a 'GET' method with the URL `https://$catalog.host/orders/v1/{order}`, a 'Security' section with a 'clientID' header (X-IBM-Client-Id), and a 'Parameters' section with an 'Accept' header (application/json) and a required 'order' parameter with an example value 'ukohatwov'. To the right, the 'Asset metadata' panel shows details like file name 'asset.json', type 'Open API specification', owner 'admin', and description 'orders'. Tags listed are 'apiic' and 'openapi'.

10. Click **Definitions** and then click the Arrow (Order). You see an example of results.

This screenshot shows the same Asset Repository interface, but the 'Definitions' tab is now active. A red box highlights the 'Definitions' tab in the sidebar. A circled '1' with an arrow points to the 'Order' link in the 'Definitions' section of the main content area, which is currently expanded. The content shows an example JSON object:

```
{ "accountid": "365989380262143", "orderid": "3718443327586304" }
```

. The rest of the interface remains the same, with the 'Asset metadata' panel on the right.

You've successfully added a review. Now your teammates know that this asset is reusable and reliable. Additional information about the asset is available in the sidebar including when the file was created, a description that explains the purpose and use, and any relevant tags

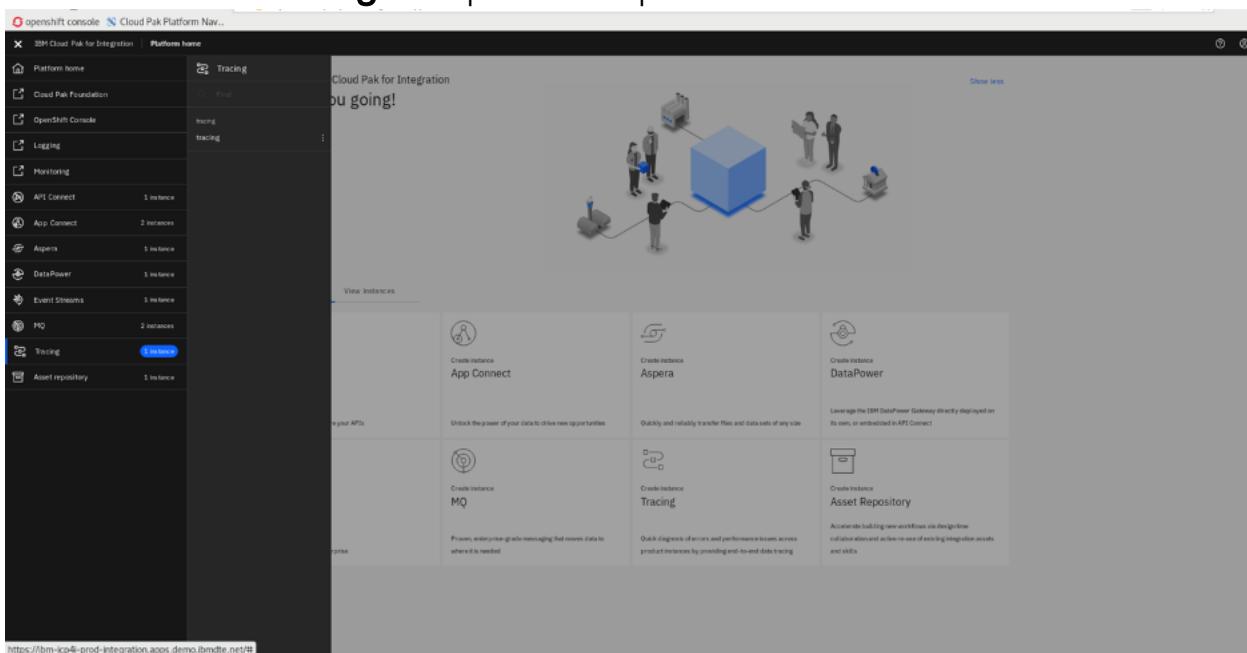
Task 8 - Using Operations Dashboard (tracing)

Cloud Pak for Integration - Operations Dashboard Add-on is based on Jaeger open source project and the OpenTracing standard to monitor and troubleshoot microservices-based distributed systems.

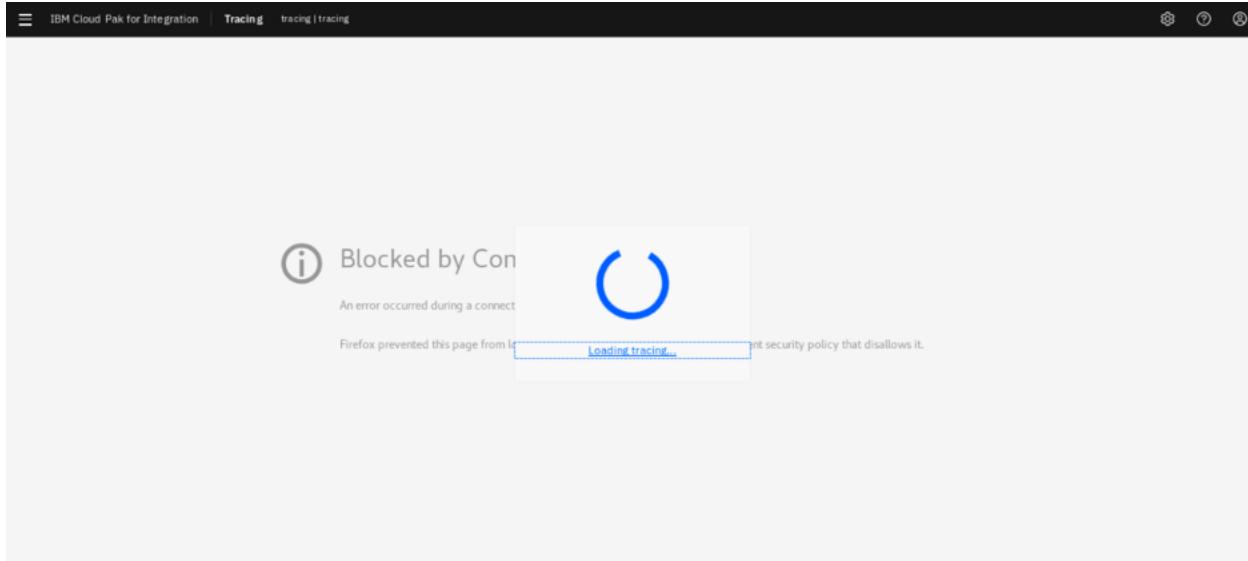
Operations Dashboard can distinguish call paths and latencies.

DevOps personnel, developers, and performance engineers now have one tool to visualize throughput and latency across integration components that run on Cloud Pak for Integration. Cloud Pak for Integration - Operations Dashboard Add-on is designed to help organizations that need to meet and ensure maximum service availability and react quickly to any variations in their systems.

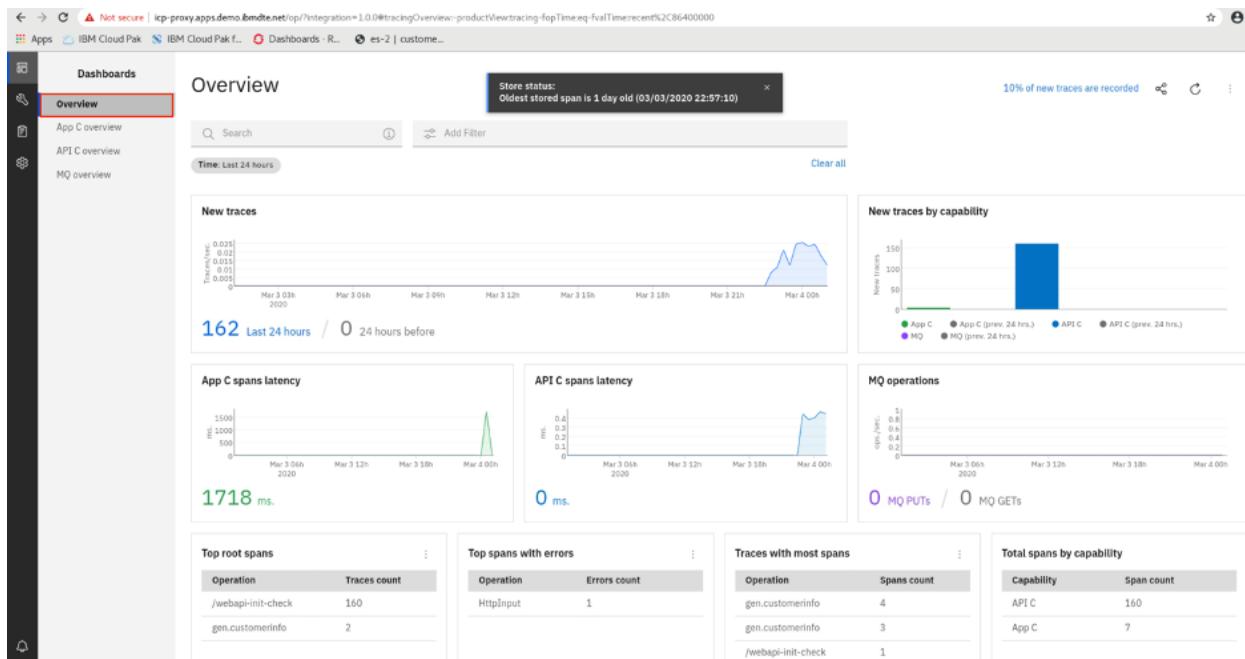
1. Go to the IBM Pak Cloud Integration main page select **View events** and click **tracing** to open the Operations Dashboard instance.



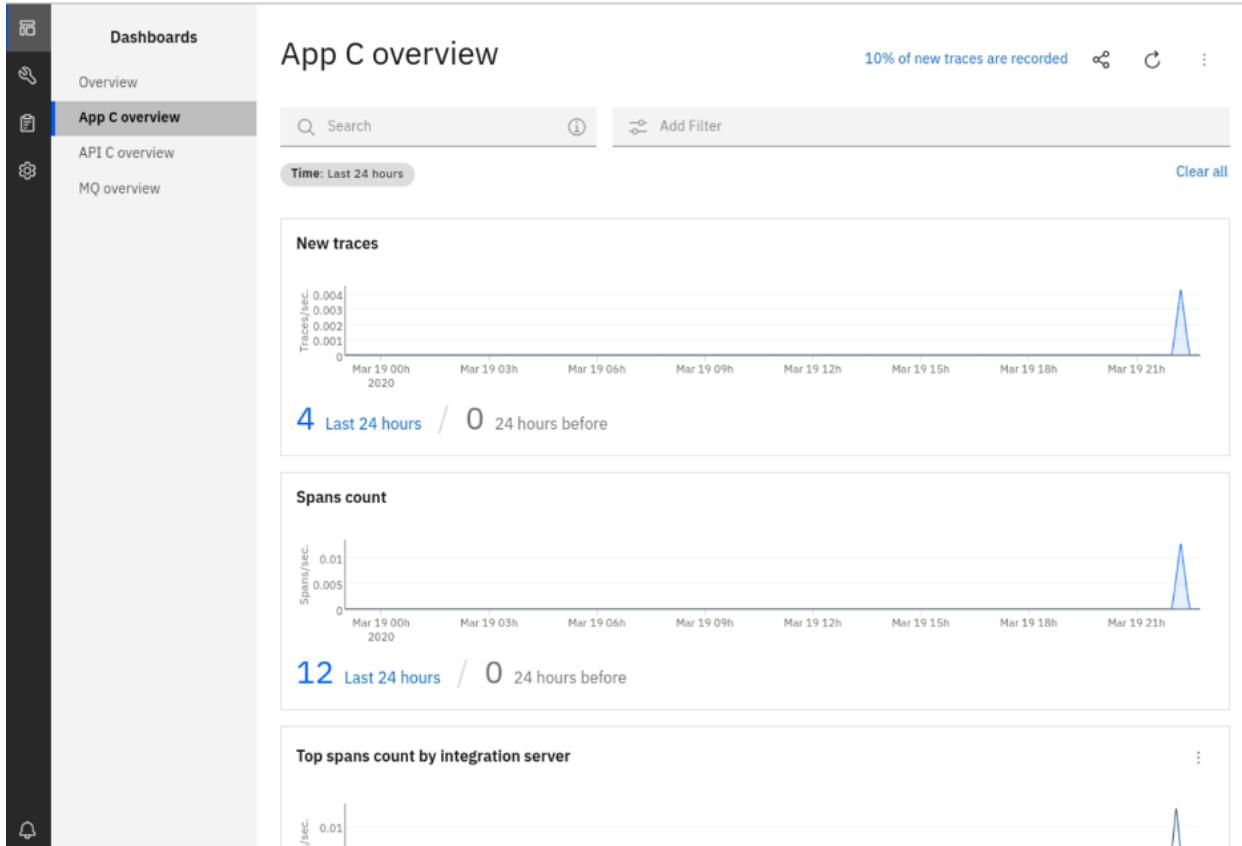
Note: You might see this page. Click the loading tracing to open tracing page.



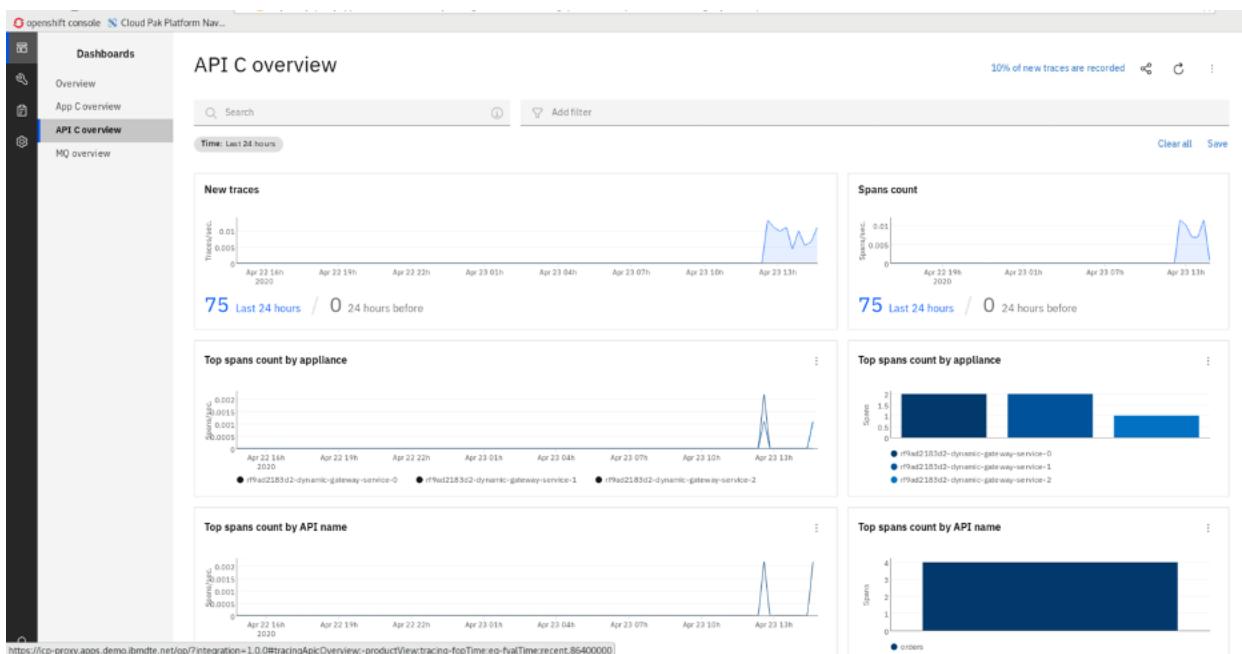
2. In the Tracing page, check the Overview page. You see all products that you can use this tool: APIC ,APP Connect and MQ. (more tracing products will add in the future releases).



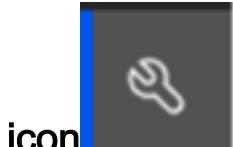
3. You can monitor each product separately. Click **App C overview**.



4. See API Connect overview.



5. Operations Dashboard generated a list of tracing. Select a line to analyze the trace of MQ App Connect Enterprise, and API Connect. select **traces**



[icon] the menu on the left. Select the line that you want to see the trace.)] Enter the name of App Connect server name: **ORDERS** and click the line (**gen.orders**) .

Trace ID	Start time	Duration (ms)	Root operation	Spans	Capabilities	Services
144f83c90fae30f3	04/27 13:10:14.046	374.12	gen.orders	6	App C, MQ	orders (5), mq (1)

6. You see the tracing chart .

Level	Root operation	Product	Duration	Span
^ 1	gen.orders	App C	4.34ms.	[green bar]
^ 2	msgFlowTransaction	App C	374.12ms.	[long green bar]
^ 3	HttpInput-HttpReply	App C	368.13ms.	[green bar]
^ 4	MqOutput	App C	148.10ms.	[green bar]
5	ORDERS	MQ	0.61ms.	[purple bar]
4	HttpReply	App C	2.91ms.	[green bar]

Summary

You have successfully completed this lab. In this lab you learned how to:

- Deploy a back-end integration to containers that are readily available as a scalable web service.

- Secure access to the back-end integration by creating a secure, governed API using the OpenAPI definition of the integration.
- Use Operations Dashboard to tracing MQ, APIC and APPC

Now that you've made your back-end integrations ready for external distribution, your developer community is able to access the APIs via a developer portal. The developer portal is included in the platform and provides a full-featured experience to onboard and nurture your API consumers. To try out more labs, go to [Cloud Pak for Integration Demos](#). For more information about Cloud Pak for Integration, go to <https://www.ibm.com/cloud/cloud-pak-for-integration>.