

OCI Setup MLFlow Tracking Server

Referenced Documentation: https://oci-mlflow.readthedocs.io/en/latest/tracking_server.html

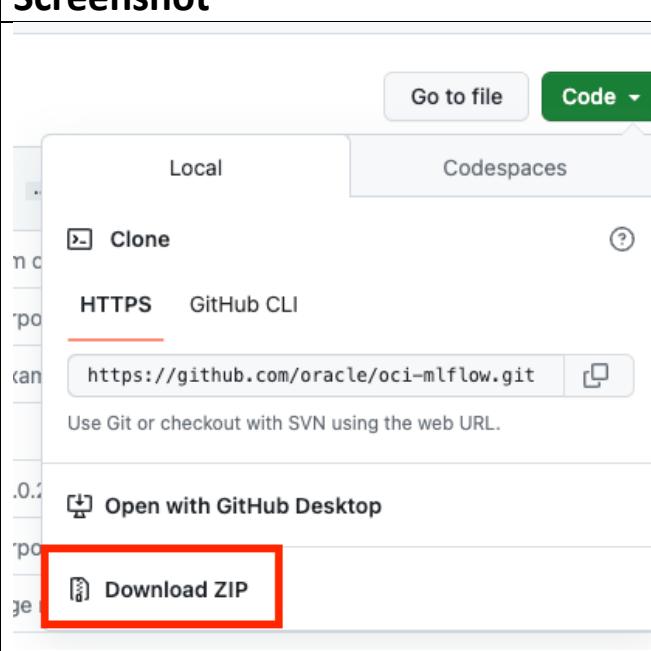
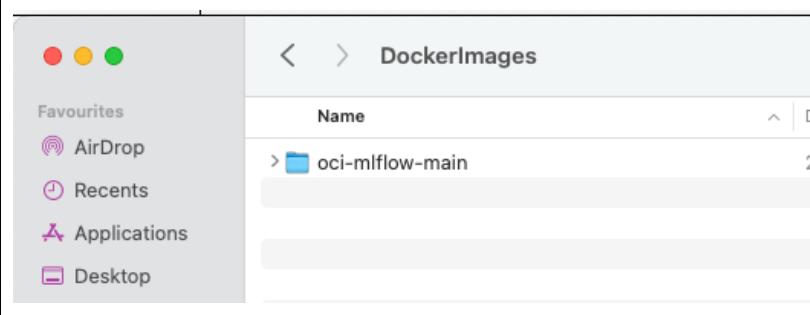
Pre-requisite: A running OCI Data Science Notebook Session within Private Network.

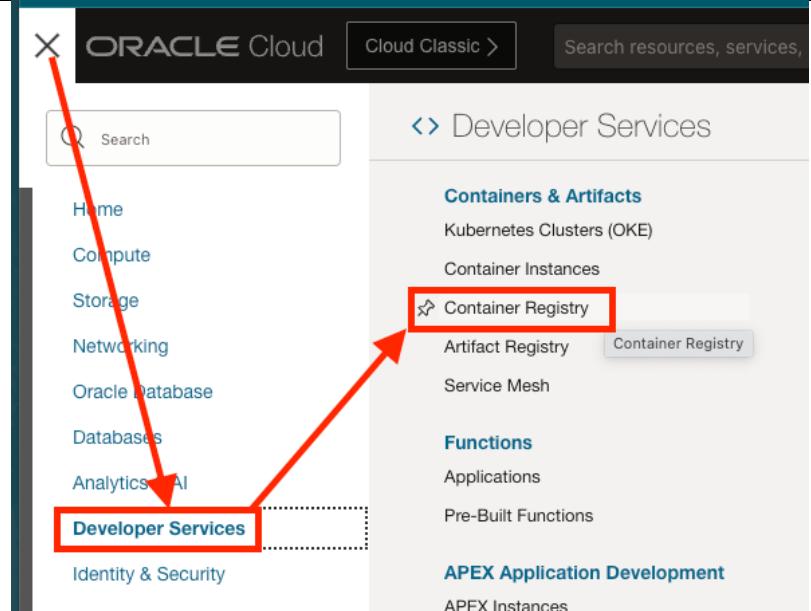
There are two ways to set up the tracker server within OCI:

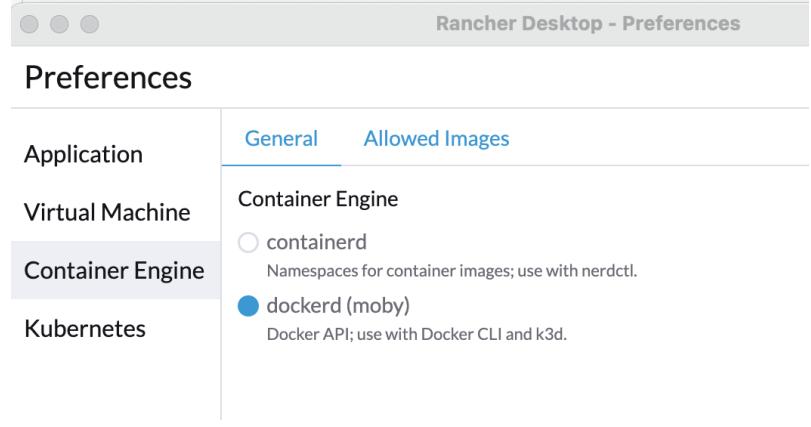
1. Running with the default options will assume you want to store the artifacts in your local file system and use SQLite for recording experiments data.
2. Configure setup with Object Storage to save artifacts and MySQL Database to save experiments data.

We will attempt both options using the **oci-mlflow** container image which we will deploy using the OCI Container Instances.

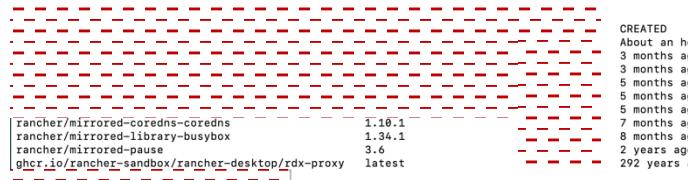
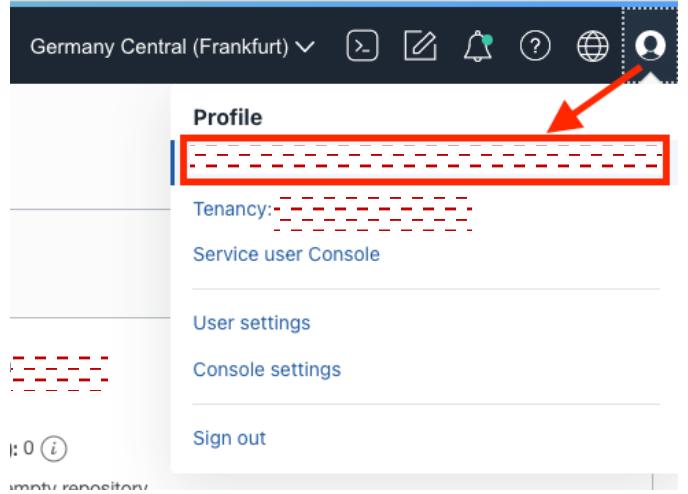
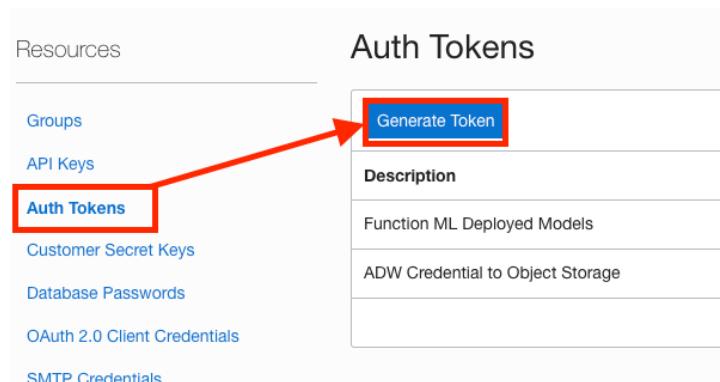
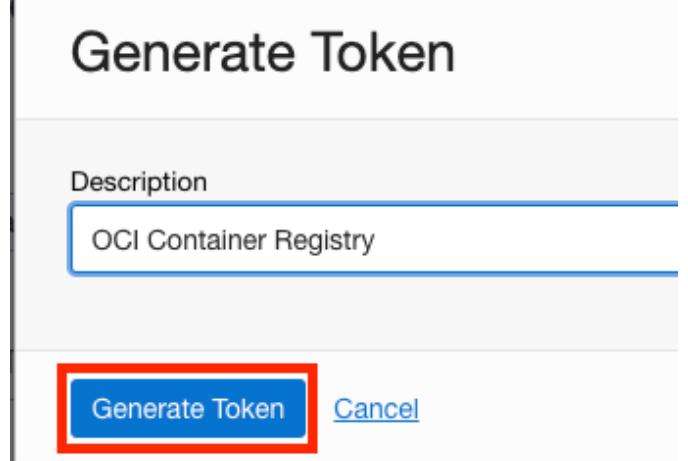
Approach 1: Store Artifacts Locally and Experiments in SQLite

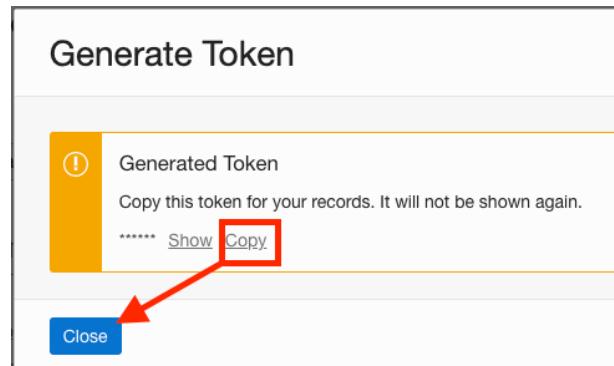
Step	Screenshot
Download the oci-mlflow docker file from the Oracle GitHub Repository. https://github.com/oracle/oci-mlflow/tree/main	
Unzip the folder into a Directory on your Laptop.	

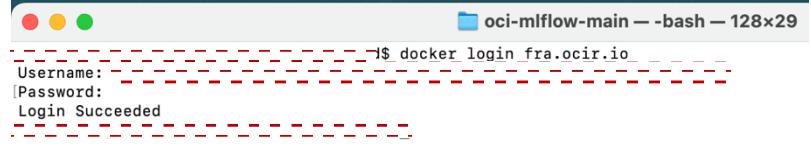
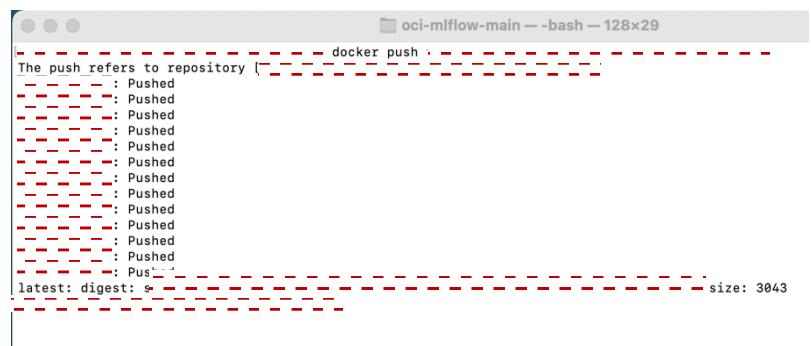
<p>Before we build our Docker Image, let's create a Container Registry to Store our MLFlow Image within Oracle Container Registry.</p> <p><i>Navigate from OCI Menu > Developer Services > Container Registry.</i></p>	 <p>The screenshot shows the Oracle Cloud Developer Services dashboard. On the left, there is a sidebar with various service links: Home, Compute, Storage, Networking, Oracle Database, Databases, Analytics, AI, Developer Services (which is highlighted with a red box), and Identity & Security. On the right, under the 'Containers & Artifacts' heading, there is a link to 'Container Registry' which is also highlighted with a red box. A red arrow points from the 'Developer Services' link in the sidebar to the 'Container Registry' link in the main content area.</p>
<p>Click Create Repository.</p>	<p>Container Registry in █████████████████████ compartment</p> <p>Create repository</p>
<p>Select your Compartment.</p> <p>Select access as Private.</p> <p>Give the Repo a Name.</p> <p>Make note of the repo name as we will need this later.</p> <p>Click Create.</p>	<p>Create repository</p> <p>Create in compartment We ██████████ █████████████████████</p> <p>Access <input checked="" type="radio"/> Private <input type="radio"/> Public</p> <p>Repository name is-oci-mlflow</p> <p>Tags Add tags to organize your resources. What can I do with tagging?</p> <p>Tag namespace None (add a free-form tag)</p> <p>Tag key</p>

<p>To build our MLFlow Docker Image I will be using Rancher Desktop. Download and install Rancher Desktop. https://rancherdesktop.io/</p> <p>When you open the application select the container engine as <i>dockerd (moby)</i>.</p>	
<p>Once Rancher Desktop is up and running, we can build our Docker Image while in the unzipped directory we downloaded from GitHub.</p> <p>We will build the image using the following format:</p> <pre>docker build -t {region}.ocir.io/{tenancy-namespace}/{repo-name}:{tag} --network host -f container-image/Dockerfile .</pre> <p>See parameter definitions to the right.</p>	<p>Parameter Definitions:</p> <ul style="list-style-type: none"> ○ <region-key> is the key for the Container Registry region you're using. For example, <i>fra</i>. See Availability by Region. ○ <i>ocir.io</i> is the Container Registry name. ○ <tenancy-namespace> is the auto-generated Object Storage namespace string of the tenancy that owns the repository to which you want to push the image (as shown on the Tenancy Information page). ○ <repo-name> is the name of the target repository to which you want to push the image (for example, <i>project01/acme-web-app</i>). ○ <tag> is an image tag you want to give the image in Container Registry (for example, <i>version2.0.test</i>). <pre>\$ docker build -t r-image/Dockerfile . [+] Building 2.8s (17/17) FINISHED => [internal] load build definition from Dockerfile => => transferring dockerfile: 1.73kB => [internal] load .dockerignore => => transferring context: 2B => [internal] load metadata for ghcr.io/oracle/oraclelinux8-instantclient:21 => [1/12] FROM r-ocir.ocir.oracle/oraclelinux8-instantclient => [internal] load build context => => transferring context: 465B => CACHED [2/12] RUN rm -rf /var/cache/yum/* && yum clean all && yum install -y gcc make patch vim iproute net-tools git => CACHED [3/12] RUN curl -L https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh >> miniconda.sh => CACHED [4/12] RUN bash ./miniconda.sh -b -p /miniconda; rm ./miniconda.sh; => CACHED [5/12] COPY container-image/environment.yaml /opt/env.yaml => CACHED [6/12] RUN conda install conda-forge::mamba && mamba env create -f /opt/env.yaml --name oci-mlflow && conda c => CACHED [7/12] RUN conda init bash && source ~/.bashrc && conda activate oci-mlflow => CACHED [8/12] RUN cd /mlflow => CACHED [9/12] COPY container-image/run/* /etc/mlflow/ => CACHED [10/12] RUN chmod a+x /etc/mlflow/launch_mlflow.sh => CACHED [11/12] RUN if [-f /etc/mlflow/oci_mlflow.whl]; then local_whl=\$(find /etc/mlflow -name "*.whl" -ex => CACHED [12/12] RUN echo "conda activate oci-mlflow">>/root/.bashrc => exporting to image => => . => => . => => . \</pre>



<p>Once the Image is built, we can list our images using:</p> <p><i>docker image ls</i></p>	 <table border="1"> <thead> <tr> <th></th> <th></th> <th>CREATED</th> <th>SIZE</th> </tr> </thead> <tbody> <tr> <td>rancher/mirrored-coredns-coredns</td> <td>1.10.1</td> <td>About an hour ago</td> <td>3.16GB</td> </tr> <tr> <td>rancher/mirrored-library-busybox</td> <td>1.34.1</td> <td>3 months ago</td> <td>254MB</td> </tr> <tr> <td>rancher/mirrored-pause</td> <td>3.6</td> <td>5 months ago</td> <td>133MB</td> </tr> <tr> <td>ghcr.io/rancher-sandbox/rancher-desktop/rdx-proxy</td> <td>latest</td> <td>5 months ago</td> <td>48.3MB</td> </tr> <tr> <td></td> <td></td> <td>5 months ago</td> <td>66.9MB</td> </tr> <tr> <td></td> <td></td> <td>7 months ago</td> <td>51.4MB</td> </tr> <tr> <td></td> <td></td> <td>8 months ago</td> <td>3.73MB</td> </tr> <tr> <td></td> <td></td> <td>2 years ago</td> <td>484KB</td> </tr> <tr> <td></td> <td></td> <td>292 years ago</td> <td>4.78MB</td> </tr> </tbody> </table>			CREATED	SIZE	rancher/mirrored-coredns-coredns	1.10.1	About an hour ago	3.16GB	rancher/mirrored-library-busybox	1.34.1	3 months ago	254MB	rancher/mirrored-pause	3.6	5 months ago	133MB	ghcr.io/rancher-sandbox/rancher-desktop/rdx-proxy	latest	5 months ago	48.3MB			5 months ago	66.9MB			7 months ago	51.4MB			8 months ago	3.73MB			2 years ago	484KB			292 years ago	4.78MB
		CREATED	SIZE																																						
rancher/mirrored-coredns-coredns	1.10.1	About an hour ago	3.16GB																																						
rancher/mirrored-library-busybox	1.34.1	3 months ago	254MB																																						
rancher/mirrored-pause	3.6	5 months ago	133MB																																						
ghcr.io/rancher-sandbox/rancher-desktop/rdx-proxy	latest	5 months ago	48.3MB																																						
		5 months ago	66.9MB																																						
		7 months ago	51.4MB																																						
		8 months ago	3.73MB																																						
		2 years ago	484KB																																						
		292 years ago	4.78MB																																						
<p>Now we can push our image to the OCI Registry.</p> <p>We then need to Generate an Auth Token for Authentication from our Local Docker to Container Registry.</p> <p>Click on your Profile Icon then your Username.</p>																																									
<p>Navigate to Auth Tokens and Click on Generate Token.</p>																																									
<p>Give it a Description and Click on Generate Token.</p>																																									

<p>Copy the Token as it will be needed later to authenticate.</p> <p>You can't view this again.</p>	
<p>Ensure your local Docker installation is running, before doing the following tasks.</p> <p>In a terminal window on the client machine that is running Docker, log into your Container Registry by entering: docker login <region-key>.ocir.io</p> <p>Where <region-key> corresponds to the key for the Container Registry Region you're using. For example, Frankfurt has the key fra.</p>	
<p>When prompted for a username, enter your username in the format <tenancy-namespace>/<username></p> <p>If your tenancy is federated with Oracle IDCS, use the format <tenancy-namespace>/oracleidentitycloudservice/<username></p>	

<p>When prompted for a password, enter the Auth Token you generated and copied earlier.</p>	 <pre>oci-mlflow-main -- bash -- 128x29 \$ docker login fra.ocir.io Username: Password: Login Succeeded</pre>
<p>Push the local Docker Image to the OCI Container Registry.</p> <p><code>docker push <repository-name>:<tag></code></p> <p>If you need a reminder of the above details, you can always run <code>docker image ls</code></p>	<p>Format of <repository-name> must be <region-key>.ocir.io/<tenancy-namespace>/<repo-name> where:</p> <ul style="list-style-type: none"> ○ <region-key> is the key for the Container Registry region you're using. For example, fra. See Availability by Region. ○ ocir.io is the Container Registry name. ○ <tenancy-namespace> is the auto-generated Object Storage namespace string of the tenancy that owns the repository to which you want to push the image (as shown on the Tenancy Information page). ○ <repo-name> is the name of the target repository to which you want to push the image (for example, project01/acme-web-app). ○ <tag> is an image tag you want to give the image in Container Registry (for example, version2.0.test).  <pre>oci-mlflow-main -- bash -- 128x29 \$ docker push 'fra.ocir.io/mlflow-project01/acme-web-app:latest' The push refers to repository [REDACTED] : Pushed : Pushed latest: digest: [REDACTED] size: 3043</pre>

We can then view our Image in the OCI Container Registry.

Click on **OCI Menu > Developer Services > Container Registry**

The screenshot shows the Oracle Cloud Developer Services interface. On the left, there's a sidebar with various services like Home, Compute, Storage, Networking, Oracle Database, Databases, Analytics, AI, Developer Services (which is highlighted with a red box), and Identity & Security. To the right, under 'Containers & Artifacts', there are links for Kubernetes Clusters (OKE), Container Instances, Container Registry (which is also highlighted with a red box), Artifact Registry, Service Mesh, Functions (Applications and Pre-Built Functions), and APEX Application Development (APEX Instances).

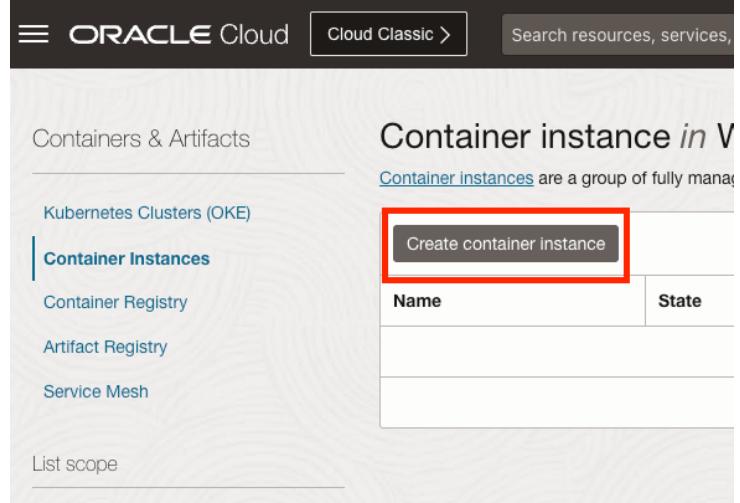
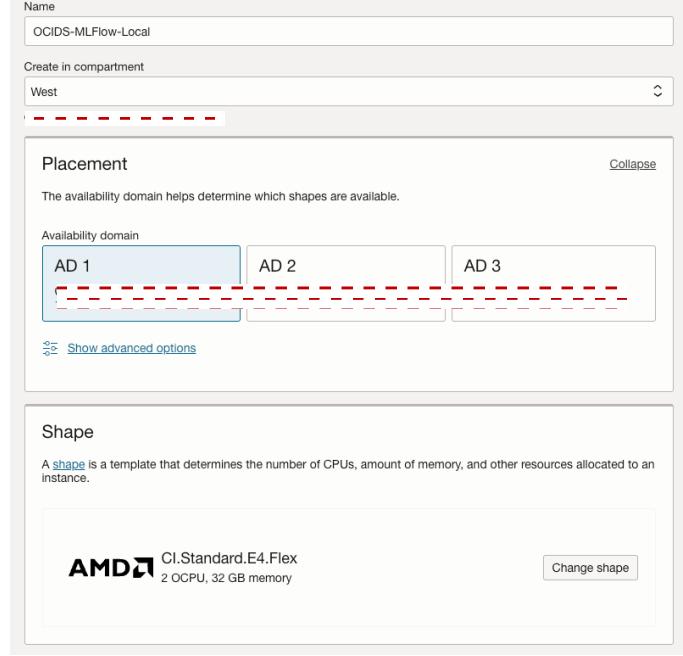
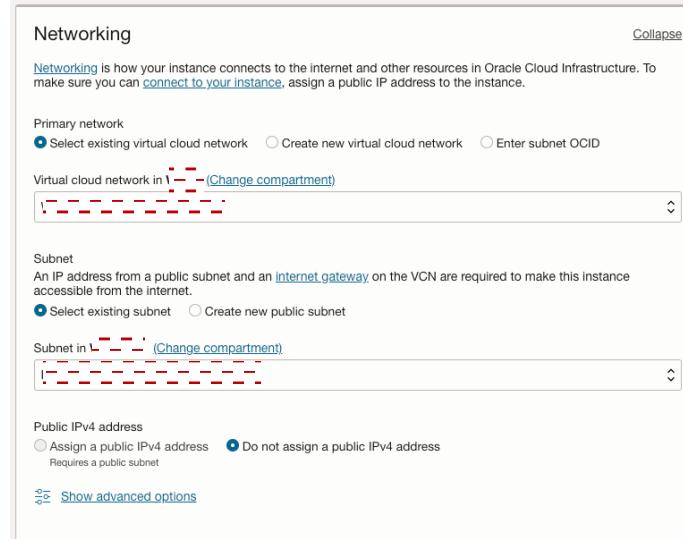
We can then select our Repository and see our uploaded Docker Image.

This screenshot shows the details of a Docker image named 'latest'. It displays various metadata fields such as Full path, Published by, OCID, Digest, and Repository. To the right, it shows Name, Date created, Size (GB), Total pulls, and Last pull. Below this, there are tabs for Signatures and Scan results, and a table for Description and Verification response with a note 'No items found.'

Next, we need to deploy and run this container.

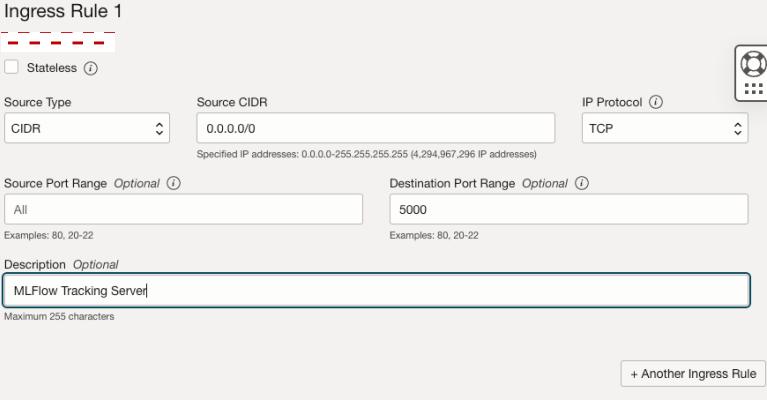
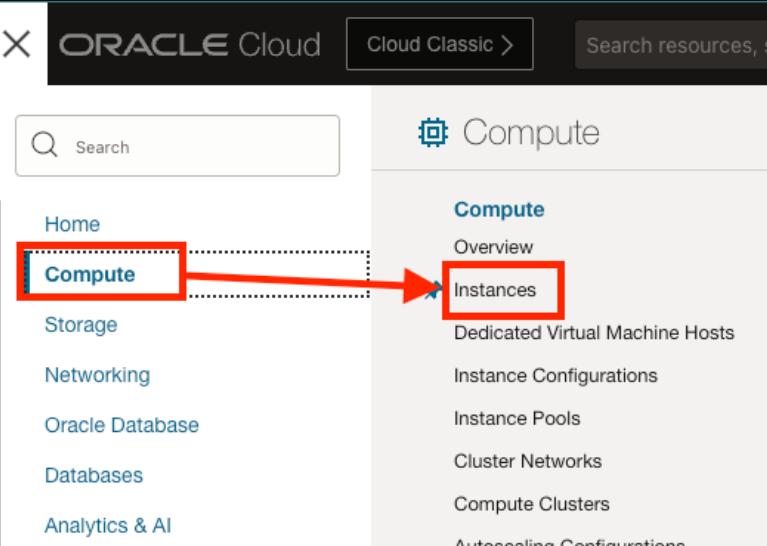
Navigate under the **Containers & Artifacts Menu** to **Container Instances**.

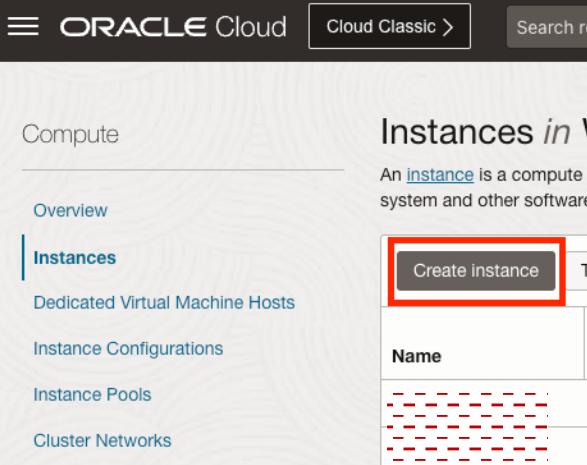
The screenshot shows the 'Containers & Artifacts' menu. It includes links for Kubernetes Clusters (OKE), Container Instances (which is highlighted with a red box), Container Registry, Artifact Registry, Service Mesh, List scope, and Compartment. To the right, there's a 'Container' section with a 'Create repo' button and a 'Repositories' section showing a count of 4.

<p>Click <i>Create Container Instance</i>.</p>	
<p>Give the Container:</p> <ul style="list-style-type: none"> • Name • Compartment • Availability Domain • Compute Shape 	
<p>Select the <i>VCN And Private Subnet</i> of where your OCI Data Science Notebook Session is located.</p> <p>Click <i>Next</i>.</p>	

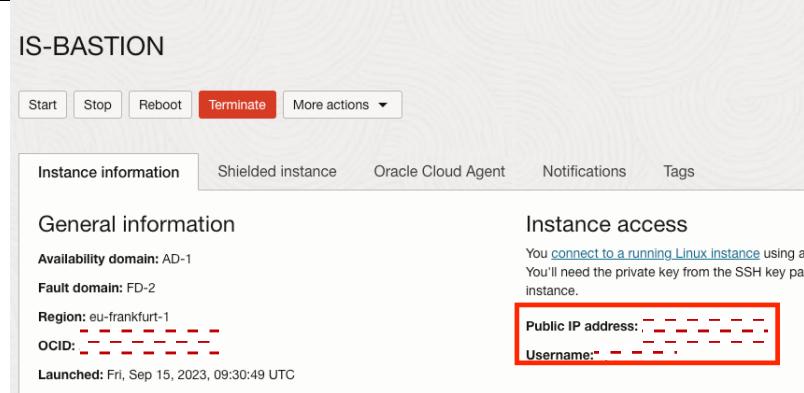
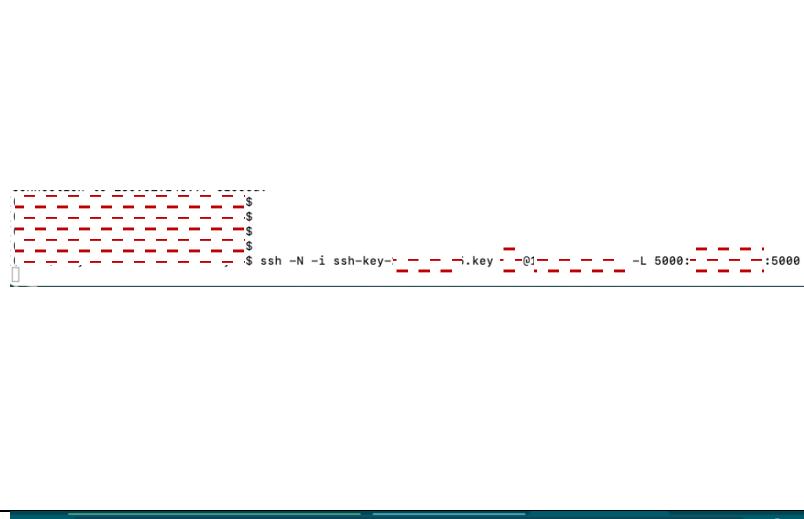
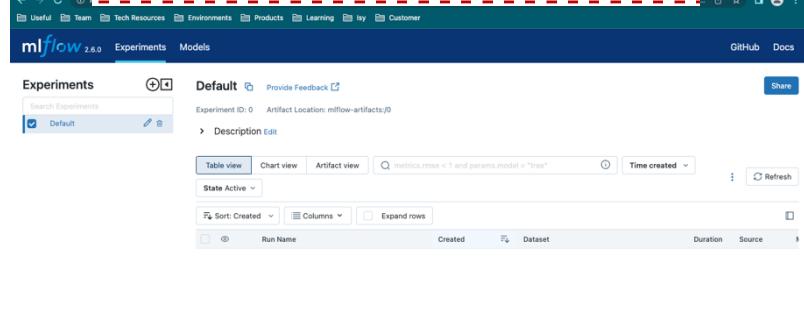
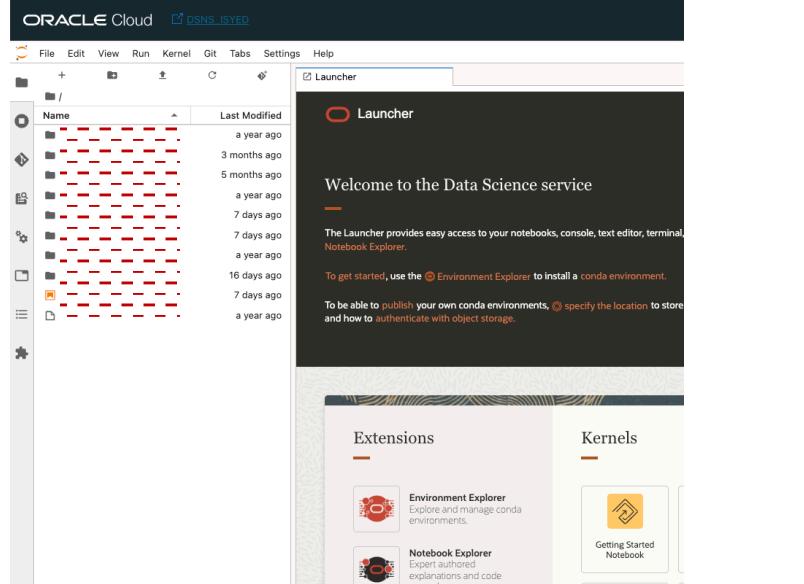
<p>Give the Container a Name.</p> <p>Select our Container Image from the Registry</p>	
<p>Enter the following Environment Variables:</p> <p>MLFLOW_HOST = 0.0.0.0</p> <p>MLFLOW_GUNICORN_OPTS = --log-level debug</p> <p>OCIFS_IAM_TYPE=resource_principal</p> <p>Click Next and Create.</p>	
<p>While the container instance is creating, we will open the correct ports on our Private Subnet Security List to allow us to connect.</p> <p>From OCI Menu > Networking > Virtual Cloud Network.</p>	
<p>Select your VCN.</p>	

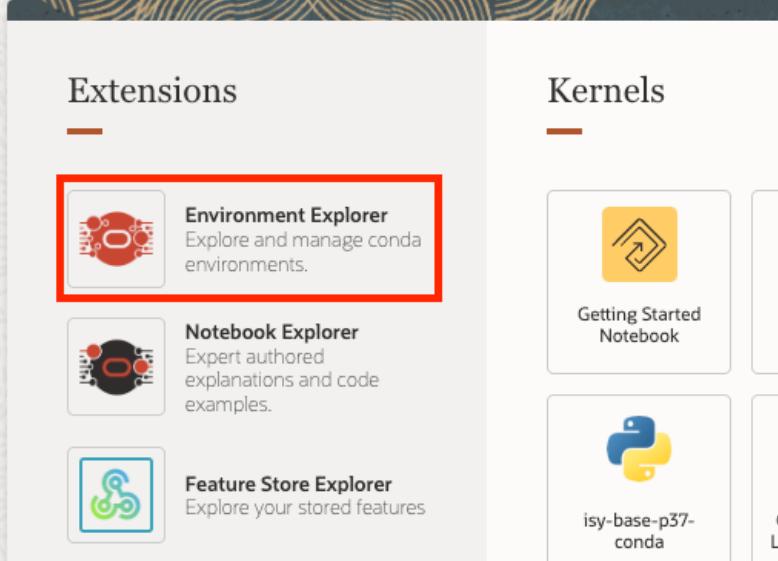
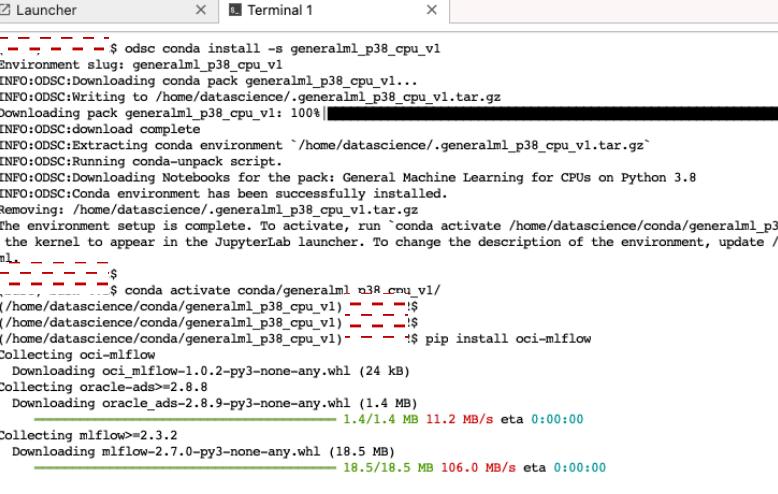
<p>Select your Private Subnet that has the Container Instance deployed in.</p>	<h3>Subnets in West Compartment</h3> <table border="1"><thead><tr><th colspan="3">Create Subnet</th></tr><tr><th>Name</th><th>State</th><th>IPv4 CIDR Block</th></tr></thead><tbody><tr><td>Public Subnet</td><td>Available</td><td>[REDACTED]</td></tr><tr><td>Private Subnet</td><td>Available</td><td>[REDACTED]</td></tr></tbody></table>	Create Subnet			Name	State	IPv4 CIDR Block	Public Subnet	Available	[REDACTED]	Private Subnet	Available	[REDACTED]
Create Subnet													
Name	State	IPv4 CIDR Block											
Public Subnet	Available	[REDACTED]											
Private Subnet	Available	[REDACTED]											
<p>Select the Security List.</p>	<h3>Security Lists</h3> <table border="1"><thead><tr><th colspan="2">Add Security List</th></tr><tr><th>Name</th><th>State</th></tr></thead><tbody><tr><td>Security List for VCN</td><td>Available</td></tr></tbody></table>	Add Security List		Name	State	Security List for VCN	Available						
Add Security List													
Name	State												
Security List for VCN	Available												
<p>Add a new Ingress Rule.</p>	<h3>Ingress Rules</h3> <table border="1"><thead><tr><th colspan="2">Add Ingress Rules</th><th>Edit</th><th>Remove</th></tr><tr><th></th><th>Stateless ▾</th><th>Source</th><th>IP Protocol</th></tr></thead><tbody><tr><td><input type="checkbox"/></td><td>No</td><td>[REDACTED]</td><td>TCP</td></tr></tbody></table>	Add Ingress Rules		Edit	Remove		Stateless ▾	Source	IP Protocol	<input type="checkbox"/>	No	[REDACTED]	TCP
Add Ingress Rules		Edit	Remove										
	Stateless ▾	Source	IP Protocol										
<input type="checkbox"/>	No	[REDACTED]	TCP										

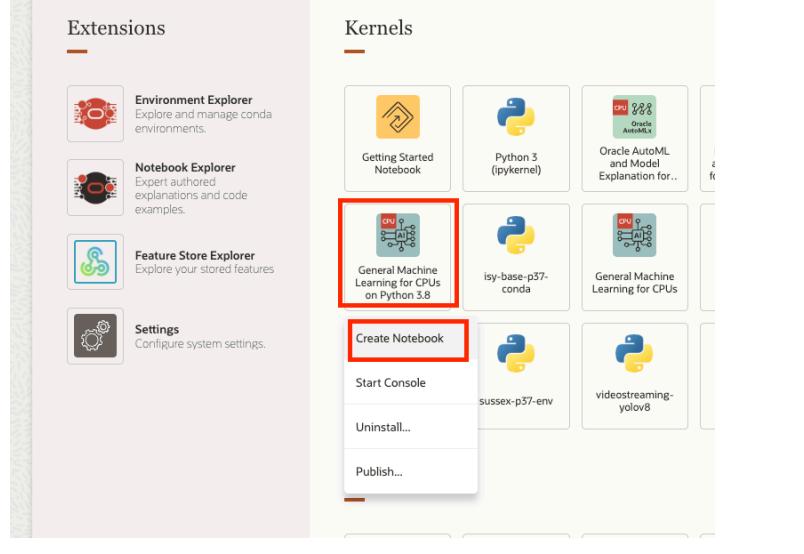
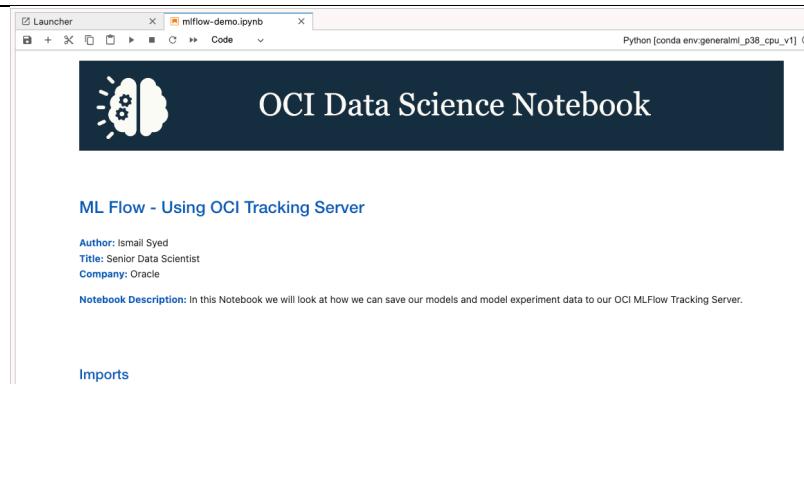
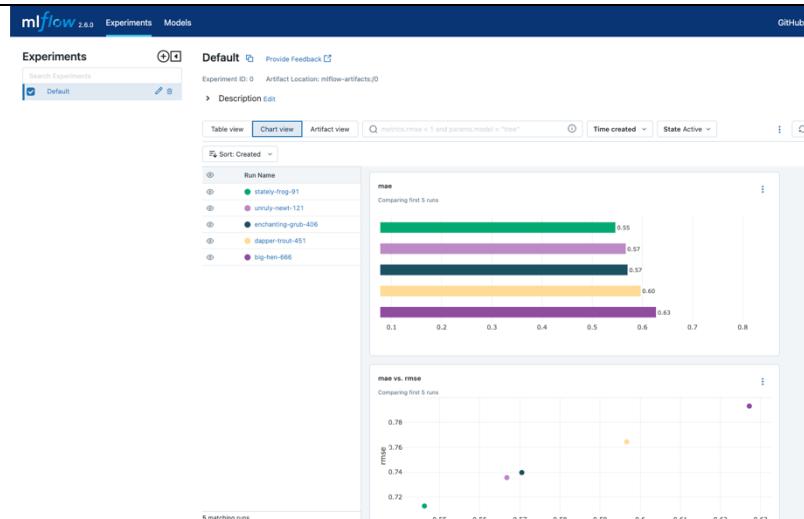
<p>Set:</p> <p>Source: 0.0.0.0/0 (<i>Access from anywhere</i>) IP Protocol: TCP Source Port: All Destination Port: 5000</p> <p>Click Create.</p> <p>You can change this according to your network configurations.</p>	
<p>We can then go check our Container Instance is up and running.</p>	
<p>We will now create a Jump Server so that we can connect to our Private MLFlow Tracking Server.</p> <p>OCI Menu > Compute > Instances.</p>	

<p>Create Instance.</p>	
<p>Select:</p> <ul style="list-style-type: none"> • Name • Compartment • Availability Domain 	<p>Name [REDACTED]</p> <p>Create in compartment West</p> <p>Placement The availability domain helps determine which shapes are available.</p> <p>Availability domain AD 1 [REDACTED] AD 2</p> <p>Show advanced options</p>
<p>Select Image and Shape.</p> <p>I have gone for Oracle Linux 8 and a small AMD Shape of 1 OCPU and 16GB RAM.</p>	<p>Image and shape</p> <p>A shape is a template that determines the number of CPUs, amount of memory, and other resources image is the operating system that runs on top of the shape.</p> <p>Image</p> <p>Oracle Linux 8 ORACLE Linux Image build: 2023.08.31-0 [REDACTED]</p> <p>Shape</p> <p>VM.Standard.E4.Flex AMD [REDACTED] Virtual machine, 1 core OCPU, 16 GB memory, 1 Gbps network bandwidth</p>

<p>Select the Public Subnet within the same VCN as our Tracking Server is Deployed.</p>	<p>Primary network <input checked="" type="radio"/> Select existing virtual cloud network <input type="radio"/> Create new virtual cloud network</p> <p>VCN in [REDACTED] (Change compartment) [REDACTED]</p> <p>Subnet An IP address from a public subnet and an internet gateway on the VCN are required. <input checked="" type="radio"/> Select existing subnet <input type="radio"/> Create new public subnet</p> <p>Subnet in [REDACTED] (Change compartment) Public Subnet: [REDACTED]</p>
<p>Ensure you have 'Automatically assign public IPv4 address' selected.</p>	<p>Primary VNIC IP addresses</p> <p>Private IPv4 address <input checked="" type="radio"/> Automatically assign private IPv4 address <input type="radio"/> Manually assign private IPv4 address</p> <p>Public IPv4 address <input checked="" type="checkbox"/> Automatically assign public IPv4 address If you're not sure whether you need a public IP address, you can always assign one later.</p> <p>IPv6 addresses <input type="checkbox"/> Assign IPv6 addresses from subnet prefixes You can only assign one IPv6 address per subnet prefix at first instance creation. Subnets can have more than one IPv6 prefix.</p> <p>i The selected VCN and subnet combination does not support IPv6 addresses. You must change the VCN and subnet before you can assign IPv6 addresses to this instance.</p>
<p>Generate and save a new pair of SSH Keys.</p> <p>Click Create.</p>	<p>Add SSH keys</p> <p>Generate an SSH key pair to connect to the instance using a Secure Shell (SSH) connection, or upload an existing key.</p> <p><input checked="" type="radio"/> Generate a key pair for me <input type="radio"/> Upload public key files (.pub) <input type="radio"/> Paste public keys <input type="radio"/> None</p> <p>i Download the private key so that you can connect to the instance using SSH. It will be deleted after you download it.</p> <p>Save private key Save public key</p>

<p>Once the instance is created make note of the Public IP address and Username.</p>	
<p>We can now open up an SSH Tunnel and Port Forward from our Private Tracking Server to Local Laptop via the Jump Server.</p> <pre>ssh -N -i <private-key> <user>@<bastion-ip> -L 5000:<tracking-server-ip>:5000</pre>	
<p>We can now open your local browser and visit localhost:5000. This will open our MLFlow Tracking Server UI.</p>	
<p>We will now open our OCI Data Science Environment and push some experiment data to our tracking server.</p>	

<p>We will install the General Machine Learning for CPU on Python 3.8 Conda Environment.</p> <p>Click on Environment Explorer.</p>	 <p>Extensions</p> <ul style="list-style-type: none"> Environment Explorer Explore and manage conda environments. Notebook Explorer Expert authored explanations and code examples. Feature Store Explorer Explore your stored features <p>Kernels</p> <ul style="list-style-type: none"> Getting Started Notebook isy-base-p37-conda 																							
<p>Find the General Machine Learning for CPU on Python 3.8 Conda Environment, select the 3 dots and click Install.</p>	 <table border="1"> <thead> <tr> <th>Environment</th> <th>Version</th> <th>Type</th> <th>Last Updated</th> <th>Size</th> <th>Action</th> </tr> </thead> <tbody> <tr> <td>General Machine Learning for CPUs on Python 3.8 Python 3.8 CPU</td> <td>1.0</td> <td>Data Science</td> <td>10 months ago</td> <td>1.14 GB</td> <td>...</td> </tr> <tr> <td>TensorFlow 2.8 for Python on Python 3.8 Python 3.8 GPU</td> <td>1.0</td> <td>Data Science</td> <td>10 months ago</td> <td>Install...</td> </tr> <tr> <td>TensorFlow 2.8 for CPU on Python 3.8 Python 3.8 CPU</td> <td>1.0</td> <td>Data Science</td> <td>10 months ago</td> <td>1.21 GB</td> <td>...</td> </tr> </tbody> </table>	Environment	Version	Type	Last Updated	Size	Action	General Machine Learning for CPUs on Python 3.8 Python 3.8 CPU	1.0	Data Science	10 months ago	1.14 GB	...	TensorFlow 2.8 for Python on Python 3.8 Python 3.8 GPU	1.0	Data Science	10 months ago	Install...	TensorFlow 2.8 for CPU on Python 3.8 Python 3.8 CPU	1.0	Data Science	10 months ago	1.21 GB	...
Environment	Version	Type	Last Updated	Size	Action																			
General Machine Learning for CPUs on Python 3.8 Python 3.8 CPU	1.0	Data Science	10 months ago	1.14 GB	...																			
TensorFlow 2.8 for Python on Python 3.8 Python 3.8 GPU	1.0	Data Science	10 months ago	Install...																				
TensorFlow 2.8 for CPU on Python 3.8 Python 3.8 CPU	1.0	Data Science	10 months ago	1.21 GB	...																			
<p>Once the Conda Environment is finished installing, let's activate it.</p> <p>conda activate conda/generalml_p38_cpu_v1/</p> <p>You can then install the oci-mflow library.</p> <p>pip install oci-mflow</p>	 <pre> \$ odsc conda install -s generalml_p38_cpu_v1 Environment slug: generalml_p38_cpu_v1... INFO:ODSC:Downloading conda pack generalml_p38_cpu_v1... INFO:ODSC:Writing to /home/dataservice/.generalml_p38_cpu_v1.tar.gz Downloading pack generalml_p38_cpu_v1: 100% [██████████] INFO:ODSC:download complete INFO:ODSC:Extracting conda environment `~/home/dataservice/.generalml_p38_cpu_v1.tar.gz` INFO:ODSC:Running conda-unpack script. INFO:ODSC:Downloading Notebooks for the pack: General Machine Learning for CPUs on Python 3.8 INFO:ODSC:Conda environment has been successfully installed. Removing: /home/dataservice/.generalml_p38_cpu_v1.tar.gz The environment setup is complete. To activate, run `conda activate /home/dataservice/conda/generalml_p38_cpu_v1`. \$ conda activate conda/generalml_p38_cpu_v1/ (/home/dataservice/conda/generalml_p38_cpu_v1) \$ (/home/dataservice/conda/generalml_p38_cpu_v1) \$ (/home/dataservice/conda/generalml_p38_cpu_v1) \$ pip install oci-mflow Collecting oci-mflow Downloading oci_mflow-1.0.2-py3-none-any.whl (24 kB) Collecting oracle-ads>=2.8.8 Downloading oracle_ads-2.8.9-py3-none-any.whl (1.4 MB) Collecting mlflow>=2.3.2 Downloading mlflow-2.7.0-py3-none-any.whl (18.5 MB) </pre>																							

<p>Once the <i>oci-mlflow</i> library is installed, head to the Launcher and create a new Notebook from our new Conda Environment.</p>	
<p>Execute the <i>mlflow-demo.ipynb</i> Notebook. Notebook should be in the same directory as this guide. The goal of this Notebook is to have all the experiments written to the Tracking Server.</p>	
<p>We can then visit our MLFlow Tracking Server at localhost:5000 to see all our experiments done.</p>	

We can also see the different Versions of the Elastic Net Model we saved.

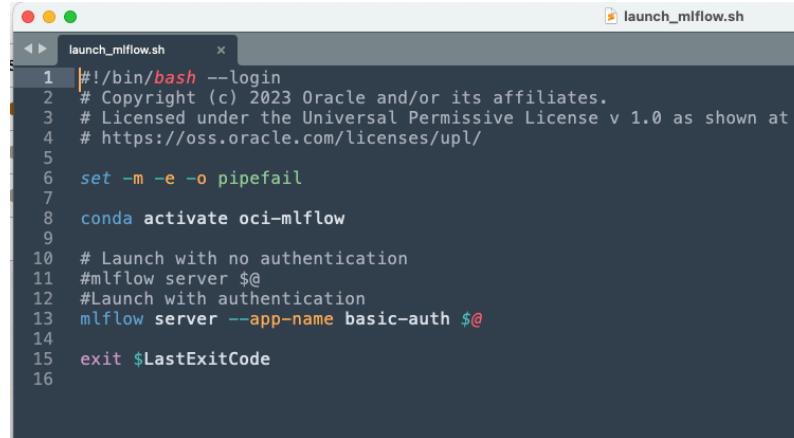
Version	Registered at	Created by
Version 5	2023-09-15 14:37:00	
Version 4	2023-09-15 14:36:57	
Version 3	2023-09-15 14:36:55	
Version 2	2023-09-15 14:36:52	
Version 1	2023-09-15 14:36:23	

Optional: Setting up MLFlow Authentication

We will now look at how to set up authentication to the tracking server. **This step will need to be done before building and pushing our MLFlow Image to the Oracle Container Registry.**

We will need to edit our Docker Image. Open the following file: ***oci-mlflow-main/container-image/run/launch_mlflow.sh***

We can then replace the line ***mlflow server \$@*** with ***mlflow server --app-name basic-auth \$@***



```

#!/bin/bash --login
# Copyright (c) 2023 Oracle and/or its affiliates.
# Licensed under the Universal Permissive License v 1.0 as shown at
# https://oss.oracle.com/licenses/upl/
set -m -e -o pipefail
conda activate oci-mlflow
# Launch with no authentication
#mlflow server $@
#Launch with authentication
mlflow server --app-name basic-auth $@
exit $LastExitCode

```

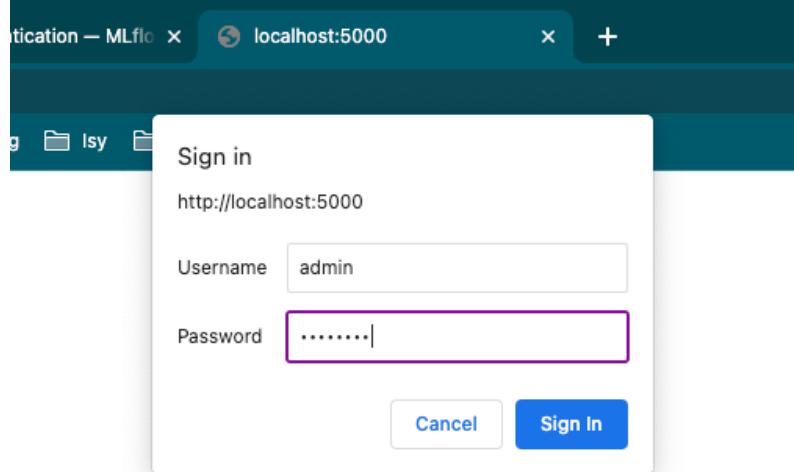
You will then have to follow the same steps above to Build, Push, and Create the Container Instance.

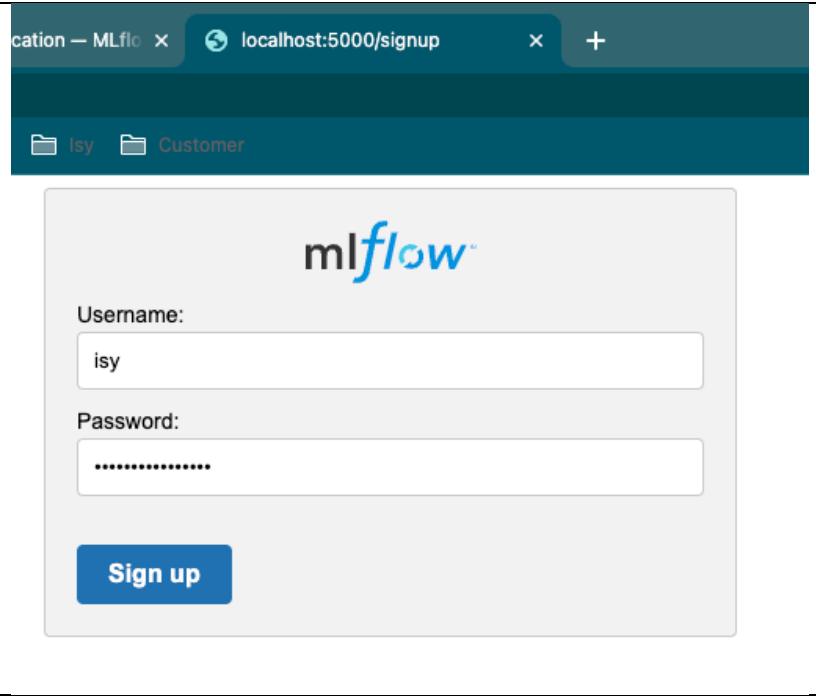
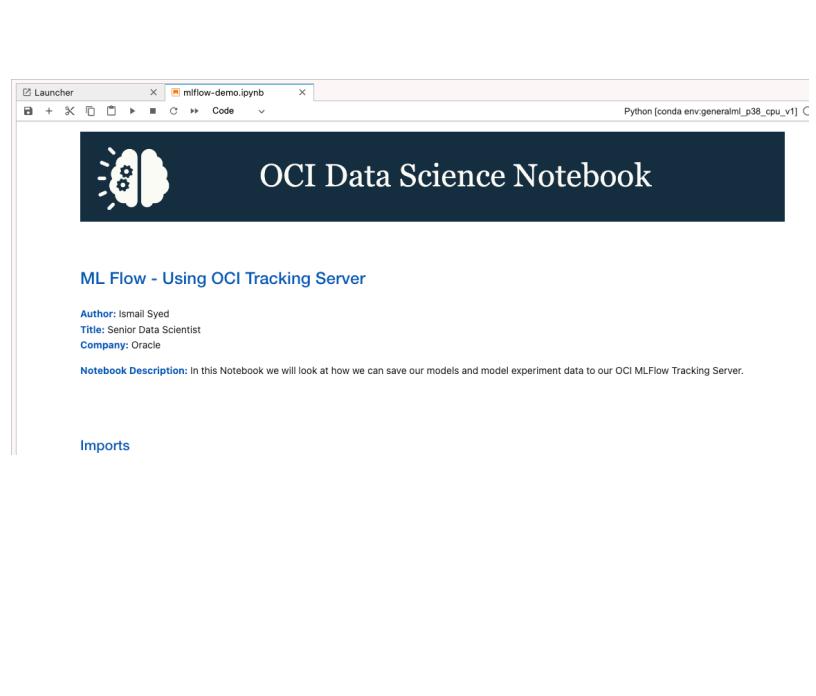
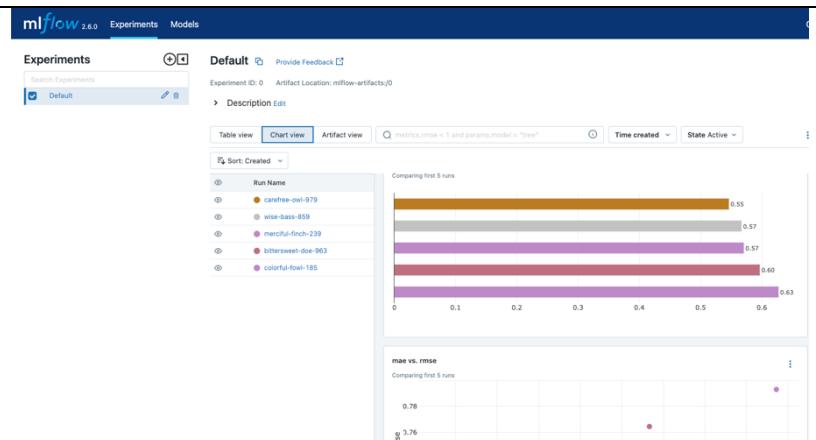
You will then have to follow the same steps above to Build, Push, and Create the Container Instance.

This time when the Container Instance is up and running and you visit the Tracker Server you will be prompted with a username and password.

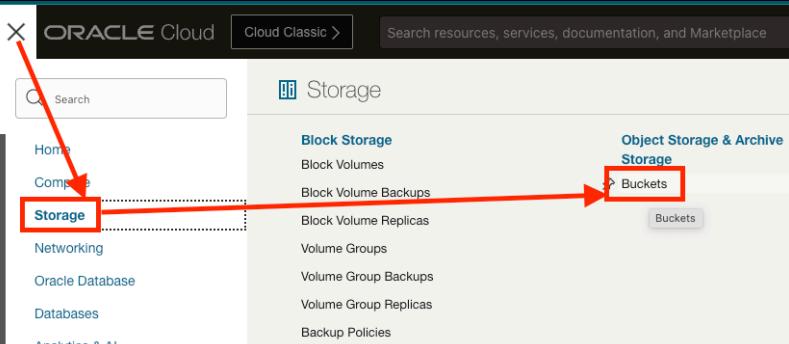
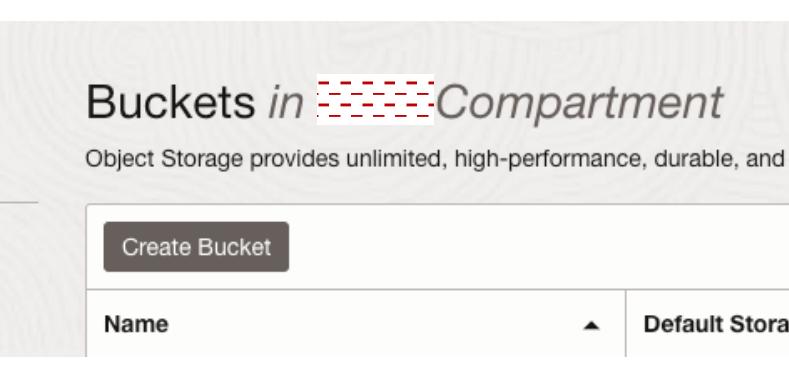
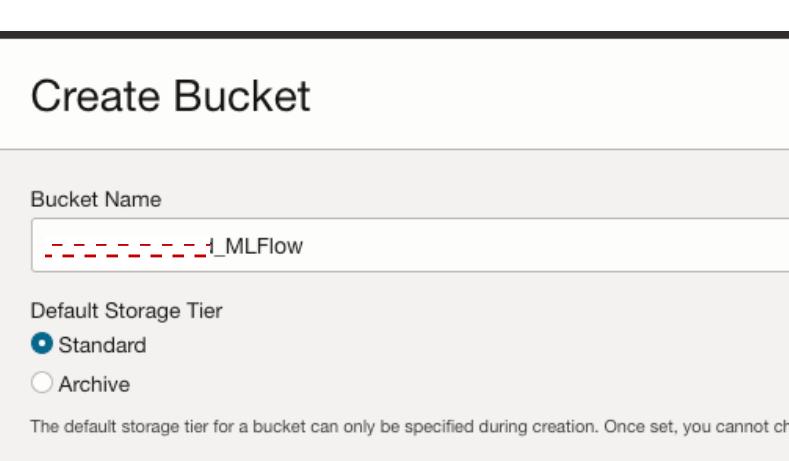
Default username is:
admin

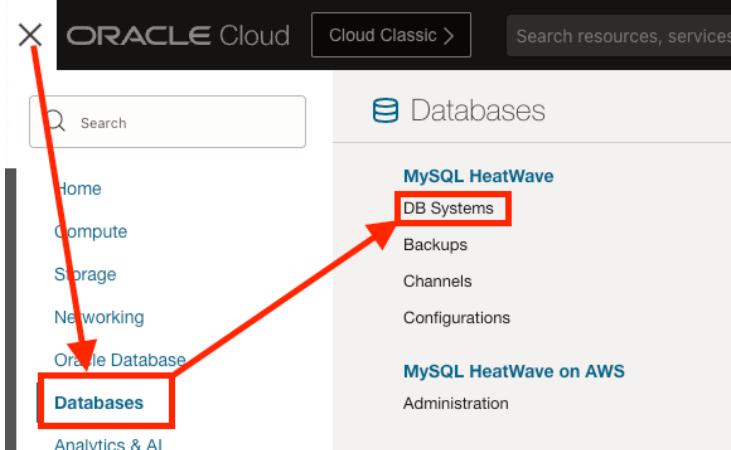
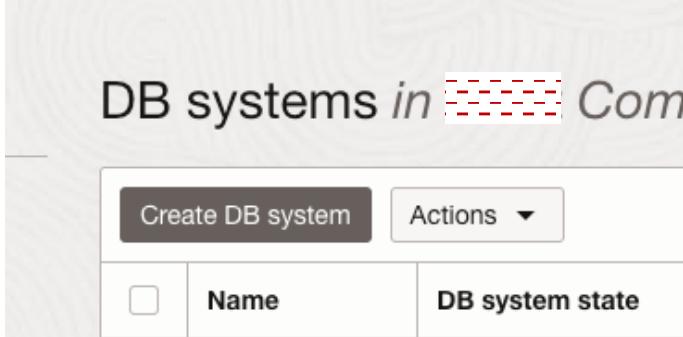
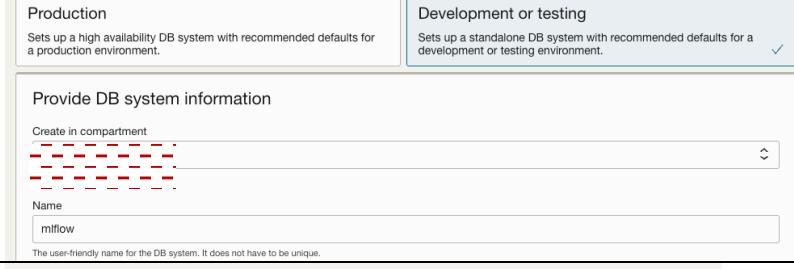
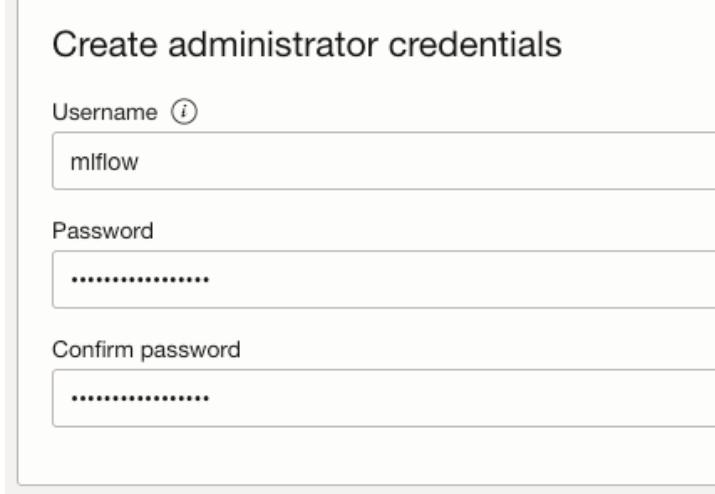
Default password is:
password



<p>We can create a new user by visiting the URL <i>localhost:5000/signup</i></p> <p>Let's create a new user called <i>isy</i> with password <i>*****</i>.</p> <p>By default, this user only has Read Access. To grant more permissions you can do this using the Python API. Visit the MLFlow Docs.</p>	
<p>Execute the <i>mlflow-auth-demo.ipynb</i> Notebook which has a few modifications to deal with authentication.</p> <p>Notebook should be in the same directory as this guide.</p> <p>The goal of this Notebook is to have all the experiments written to the Tracking Server which requires Authentication.</p>	
<p>We can then visit our MLFlow Tracking Server at <i>localhost:5000</i> to see all our experiments done.</p> <p>Remember to login using our username and password as defined in our Notebook. – In the notebook we will use the admin user.</p>	

Approach 2: Store Artifacts in Object Storage and Experiments in MySQL

Step	Screenshot
<p>Majority of the Steps are the same for utilising Object Storage and MySQL as storage options. In this Approach we will focus on setting up Object Storage, MySQL and Configuring a new Container Instance to utilise them.</p>	<p>First let's create an Object Storage Bucket.</p> <p>Navigate to OCI Menu > Storage > Buckets.</p> 
<p>Click Create Bucket.</p>	
<p>Give your Bucket a Name and Click Create.</p>	

We will now create a MySQL Instance. Navigate OCI Menu > Databases > DB Systems.	 A screenshot of the Oracle Cloud interface. At the top left is the Oracle Cloud logo. To its right are links for 'Cloud Classic' and 'Search resources, services'. Below the logo is a search bar with a magnifying glass icon. On the left side, there's a vertical sidebar with links: Home, Compute, Storage, Networking, Oracle Database, Databases (which is highlighted with a red box), and Analytics & AI. To the right of the sidebar is a main content area titled 'Databases'. Under 'Databases', there are sections for 'MySQL HeatWave' (with 'DB Systems' highlighted with a red box) and 'MySQL HeatWave on AWS' (with 'Administration' listed). Other options like Backups, Channels, and Configurations are also visible.
Click Create DB System.	 A screenshot of the 'Create DB system' page. At the top is a header with 'DB systems in [REDACTED] Com'. Below it is a 'Create DB system' button and an 'Actions' dropdown. The main form has three fields: a checkbox, a 'Name' field containing 'mlflow', and a 'DB system state' dropdown set to 'Development or testing' (which is highlighted with a blue box).
Select Type as Development or Testing. Give a name of mlflow .	 A screenshot of the 'Provide DB system information' section. It includes a 'Create in compartment' dropdown (which is redacted with a dashed line), a 'Name' field containing 'mlflow', and a note below stating 'The user-friendly name for the DB system. It does not have to be unique.'
Set Username and Password . mlflow *****	 A screenshot of the 'Create administrator credentials' section. It has three fields: 'Username' (containing 'mlflow'), 'Password' (containing '*****'), and 'Confirm password' (containing '*****').

<p>Select the VCN and Private Subnet of where our MLFlow Tracking Server will be located.</p>	<p>Configure networking</p> <p>The VCN and subnet where the DB system endpoint will be attached. accessible from the internet. How do I connect to a DB system? If you</p> <p>Virtual cloud network in West (Change compartment)</p> <p>VCN: <input type="text" value="dashed"/></p> <p>Subnet in West (Change compartment)</p> <p>Private Subnet: <input type="text" value="dashed"/></p>
<p>Choose Hardware requirements.</p> <p>I have gone for a small shape with MySQL.VM.Standard.E4.1 .8GB Compute 100GB Storage.</p> <p>Leave all the other settings as default.</p> <p>Click Create.</p>	<p>Configure hardware</p> <p>Select a shape</p> <p>MySQL.VM.Standard.E4.1.8GB</p> <p>CPU core count: 1</p> <p>Memory size: 8 GB</p> <p>Max network bandwidth: 1Gbps</p> <p>A shape determines the number of OCPUs, memory, and other resources allocated to a MySQL instance. A shape has associated configurations, which you can select in the Configuration tab under the instance.</p> <p>Data storage size (GB)</p> <p>100</p> <p>Storage allocated for data and log files. Storage size impacts IOPS and throughput. Data storage is required for the MySQL instance.</p> <p>Total IOPS: 7500</p> <p>Total throughput: 59 MB</p>

<p>Once created, make note of the Username, Password, Private IP Address, Port and db_name.</p>	
<p>We will now login to our MYSQL DB and create a new Database to host our experiments.</p> <p>SSH into our BASTION server.</p> <p>ssh -i <private-key> opc@<host></p>	
<p>Download the MySQL RPM Package. Make sure you use the correct version according to your environment – View Versions here:</p> <p>https://dev.mysql.com/downloads/repo/yum/</p> <p>curl -sSLO</p> <p>https://dev.mysql.com/get/mysql80-community-release-el8-8.noarch.rpm</p>	

We will now install the yum repository.

sudo rpm -ivh mysql80-community-release-el8-8.noarch.rpm

```
$
$ sudo rpm -ivh mysql80-community-release-el8-8.noarch.rpm
warning: mysql80-community-release-el8-8.noarch.rpm: Header V4 RSA/SHA256 Signature, key ID [REDACTED]
Verifying... ##### [100%]
Preparing... ##### [100%]
Updating / installing...
  1:mysql80-community-release-el8-8 ##### [100%]
    Warning: native mysql package from platform vendor seems to be enabled.
      Please consider to disable this before installing packages from repo.mysql.com.
Run: yum module -- disable mysql
$
```

Now *install* the **mysql server** using:

sudo yum install mysql-server

Package	Arch	Version	Repository	Size
Installing:				
mysql-server	x86_64	8.0.32-1.module+el8.8.0+21055+76bd398b	ol8_appstream	32 M
Installing dependencies:				
mariadb-connector-c-config		noarch 3.1.11-2.el8_3	ol8_appstream	15 k
mecab	x86_64	0.996-2.module+el8.8.0+21055+76bd398b	ol8_appstream	394 k
mysql	x86_64	8.0.32-1.module+el8.8.0+21055+76bd398b	ol8_appstream	15 M
mysql-common	x86_64	8.0.32-1.module+el8.8.0+21055+76bd398b	ol8_appstream	137 k
mysql-errmsg	x86_64	8.0.32-1.module+el8.8.0+21055+76bd398b	ol8_appstream	629 k
protobuf-lite	x86_64	3.5.0-15.el8	ol8_appstream	149 k
Enabling module streams:				
mysql		8.0		
Transaction Summary				
Install 7 Packages				
Total download size: 48 M				
Installed size: 245 M				
Is this ok [y/N]: y				

We can then login to our remote MySQL Server using:

mysql -u <username> -p<password> -h <hostname> -P <port>

```
$ mysql -u mlflow -[REDACTED]
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 10505
Server version: 8.0.34-u3-cloud MySQL Enterprise - Cloud

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> 
```



We can then list our Databases.

show databases;

```
[mysql]>
[mysql]> show databases;
+-----+
| Database      |
+-----+
| information_schema |
| mysql          |
| mysql_audit    |
| performance_schema |
| sys            |
+-----+
5 rows in set (0.00 sec)

mysql>
```

Let's create a new database called **mlflow**.

create database mlflow;

Once created, we can exit our MySQL env and Bastion host.

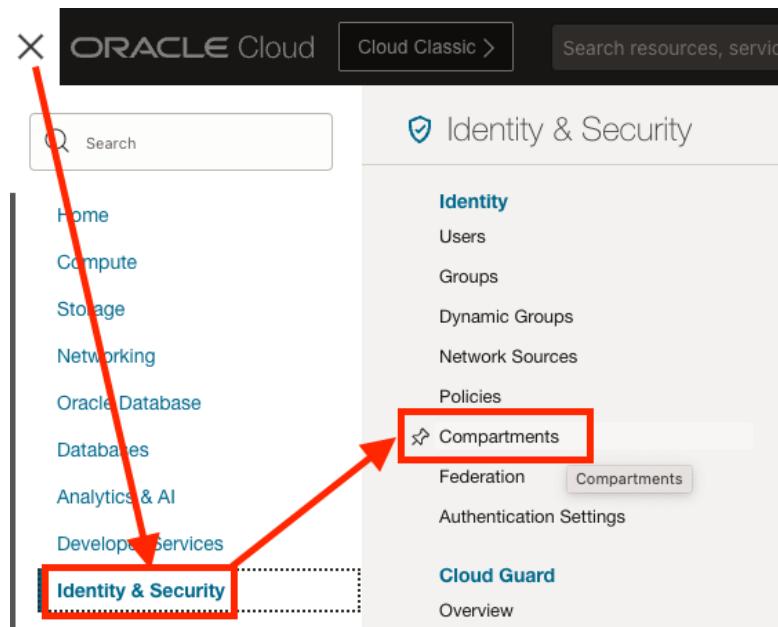
```
[mysql]> create database mlflow;
Query OK, 1 row affected (0.01 sec)

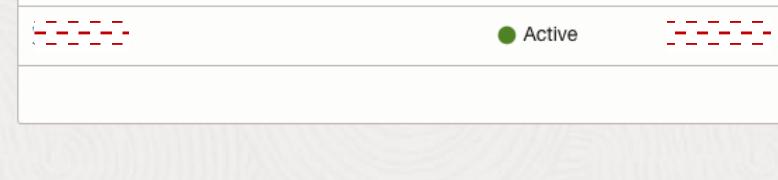
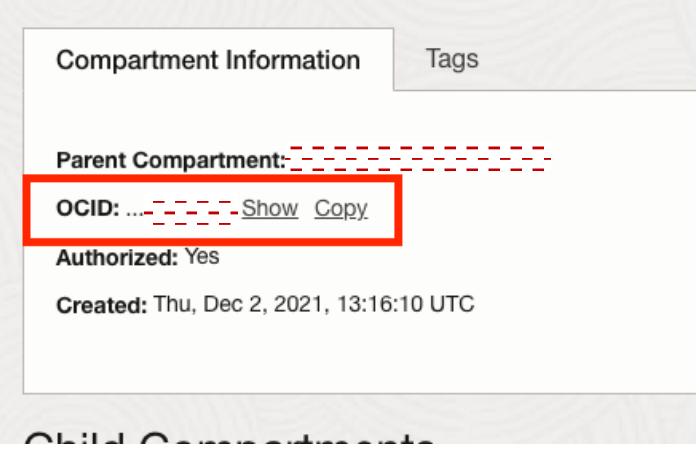
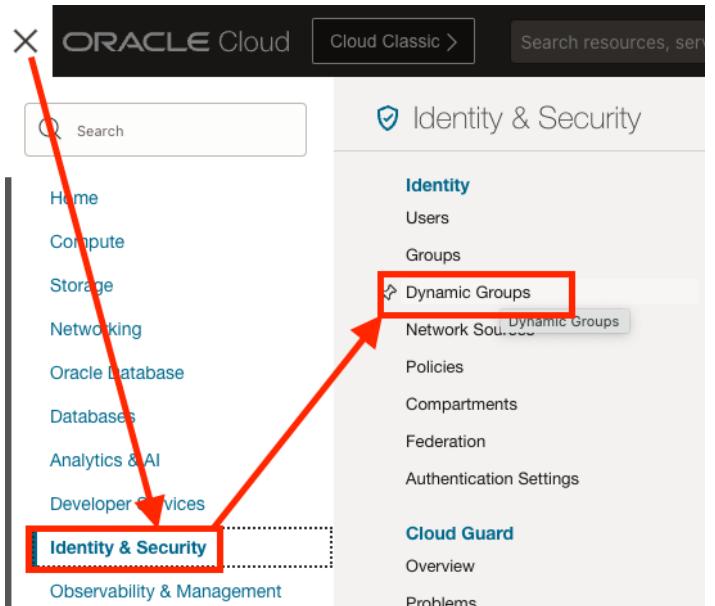
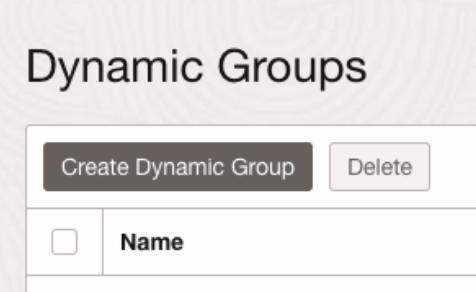
[mysql]> show databases;
+-----+
| Database      |
+-----+
| information_schema |
| mlflow          |
| mysql          |
| mysql_audit    |
| performance_schema |
| sys            |
+-----+
6 rows in set (0.00 sec)

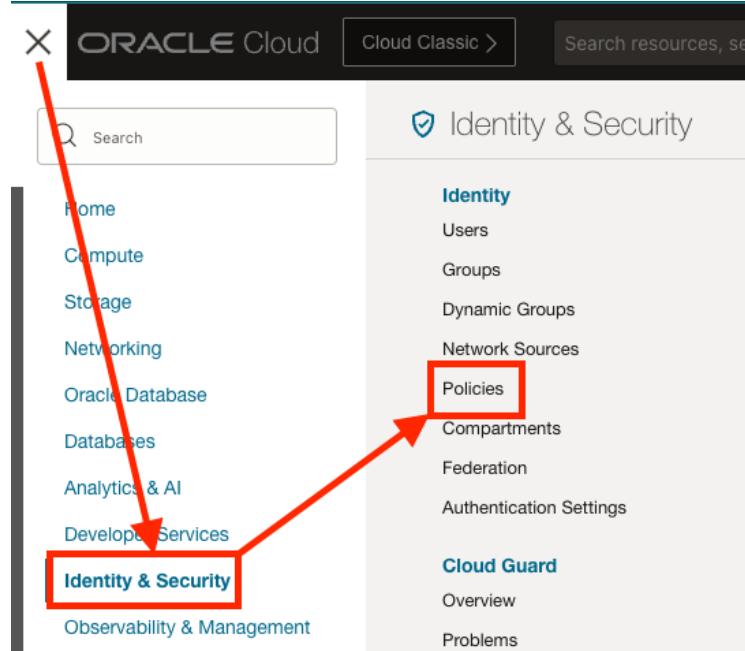
mysql>
```

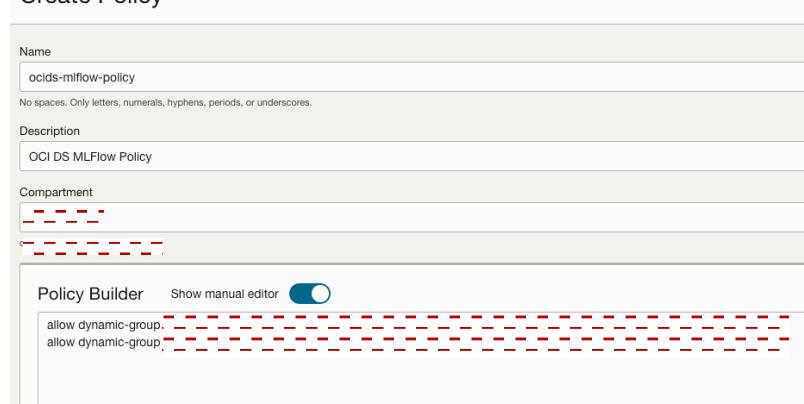
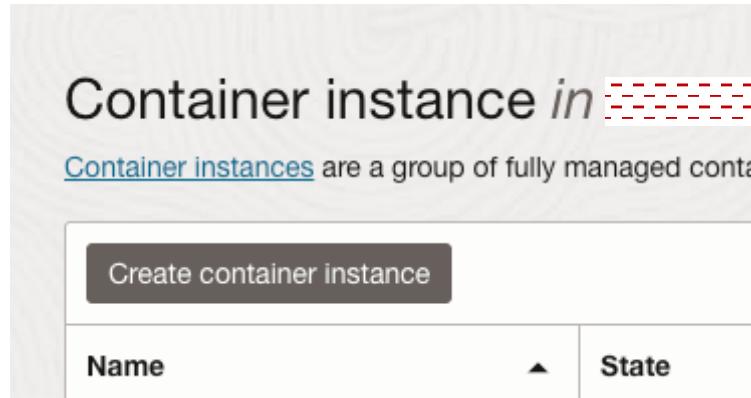
Now let's create a dynamic group to identify **computecontainerinstances** within our compartment.

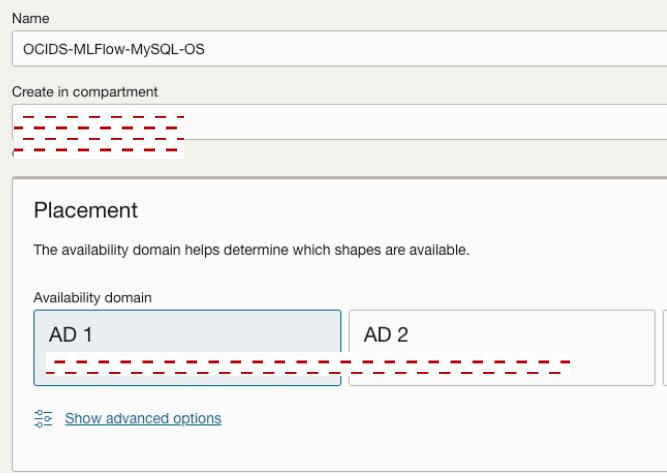
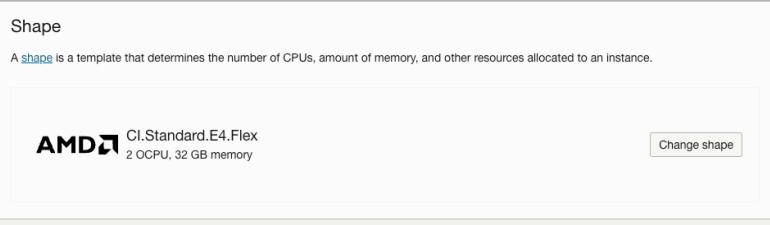
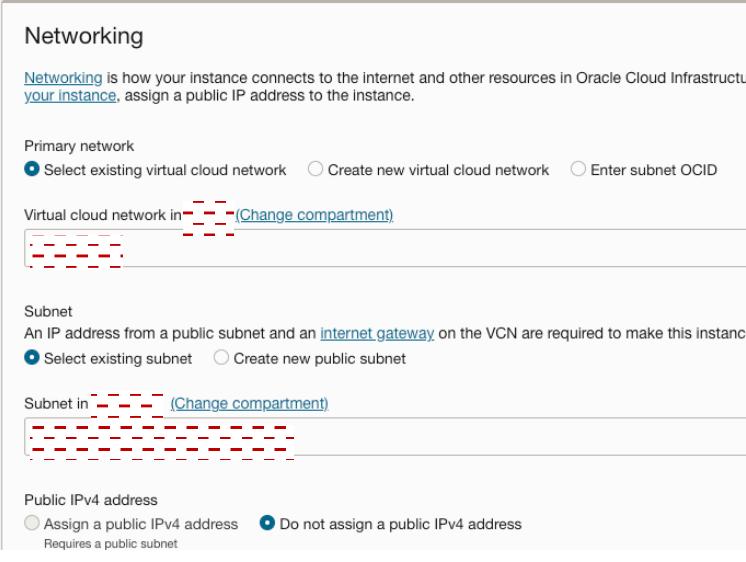
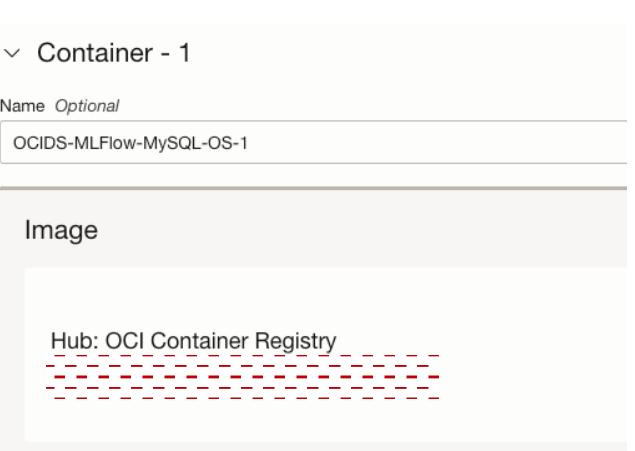
Navigate to **OCI Menu > Identity and Security > Compartment**.



Select your compartment hosting your resources.	
Make note of your Compartment OCID .	
Now we will navigate to OCI Menu > Identity & Security > Dynamic Groups.	
Click Create Dynamic Group.	

<p>Give the Dynamic Group a Name, Description and Enter the Following Rule.</p> <pre>all {resource.type='computecontainerinstance',resource.compartment.id='<compartment-ocid>'}</pre>	<p>Create Dynamic Group</p> <p>Name: dg-ocids-mflow No spaces. Only letters, numerals, hyphens, periods, or underscores. Description: OCI DS MLFlow Dynamic Group</p> <p>Matching Rules</p> <p>Rules define what resources are members of this dynamic group. All instances that meet the criteria are added automatically.</p> <p>Example: Any (instance.id = 'ocid1.instance.oc1.iad.exampleuniqueid1', instance.compartment.id = 'ocid1.compartment.oc1..exampleuniqueid2')</p> <p>Match any rules defined below Match all rules defined below</p> <p>Rule 1</p> <pre>all {resource.type='computecontainerinstance',resource.compartment.id='<compartment-ocid>'}</pre>		
<p>We will now add some policies to grant the dynamic group access to other OCI resources.</p> <p>Navigate OCI Menu > Identity & Security > Policies.</p>			
<p>Click Create Policy.</p>	<p>Policies in Compartment</p> <p>Create Policy Delete</p> <table border="1"> <tr> <td><input type="checkbox"/></td> <td>Name</td> </tr> </table>	<input type="checkbox"/>	Name
<input type="checkbox"/>	Name		

<p>Give the Policy a Name, Description and Enter the following rules and Click Create.</p> <p>allow dynamic-group <your dynamic group> to manage objects in compartment <your_compartment_name> where a II {target.bucket.name=<your_bucket_name>}</p> <p>allow dynamic-group container-instance-group to use secret-family in compartment <your_compartment_name></p>	
<p>We will now create a new Tracking Server Container Instance which is configured to use the Object Storage and MySQL DB to store Artifacts and Experiments.</p> <p>Navigate to OCI Menu > Developer Services > Container Instances.</p>	
<p>Click Create Container Instance.</p>	

<p>Give the container instance a Name and select the Availability Domain.</p>	 <p>Placement The availability domain helps determine which shapes are available.</p> <p>Availability domain</p> <p>AD 1 AD 2</p> <p>Show advanced options</p>
<p>Select a Compute Shape.</p> <p>I have gone for 2 OCPU and 32 GB Memory.</p>	 <p>Shape A shape is a template that determines the number of CPUs, amount of memory, and other resources allocated to an instance.</p> <p>AMD Ci.Standard.E4.Flex 2 OCPU, 32 GB memory Change shape</p>
<p>Select the VCN and Private Subnet we will be deploying our Tracking Server into.</p> <p>Best to select the same VCN and Private Subnet as our MySQL DB and OCI Data Science.</p> <p>Click Next.</p>	 <p>Networking Networking is how your instance connects to the internet and other resources in Oracle Cloud Infrastructure. your instance, assign a public IP address to the instance.</p> <p>Primary network</p> <p><input checked="" type="radio"/> Select existing virtual cloud network <input type="radio"/> Create new virtual cloud network <input type="radio"/> Enter subnet OCID</p> <p>Virtual cloud network in (Change compartment)</p> <p>Subnet</p> <p>An IP address from a public subnet and an internet gateway on the VCN are required to make this instance accessible. your instance, assign a public IP address to the instance.</p> <p><input checked="" type="radio"/> Select existing subnet <input type="radio"/> Create new public subnet</p> <p>Subnet in (Change compartment)</p> <p>Public IPv4 address</p> <p><input type="radio"/> Assign a public IPv4 address <input checked="" type="radio"/> Do not assign a public IPv4 address Requires a public subnet</p>
<p>Give our Container Instance a Name and Select our Container Image from the Registry.</p> <p>I have selected my image with Authentication Enabled.</p>	 <p>✓ Container - 1</p> <p>Name <i>Optional</i></p> <p>OCIDS-MLFlow-MySQL-OS-1</p> <p>Image</p> <p>Hub: OCI Container Registry</p>

<p>Next, we will enter the following Environment Variables. This is where we will tell our Tracking Server the Location of our Object Storage and MySQL DB.</p> <p>Click Next and then Create.</p>	<p>Environmental variables</p> <table border="1"> <tbody> <tr> <td>Key</td> <td>Value</td> </tr> <tr> <td>MLFLOW_HOST</td> <td>0.0.0.0</td> </tr> <tr> <td>MLFLOW_GUNICORN_OPTS</td> <td>--log-level debug</td> </tr> <tr> <td>OCIFS_IAM_TYPE</td> <td>resource_principal</td> </tr> <tr> <td>MLFLOW_DEFAULT_ARTIFACT_ROOT</td> <td>[REDACTED]</td> </tr> <tr> <td>MLFLOW_ARTIFACTS_DESTINATION</td> <td>[REDACTED]</td> </tr> <tr> <td>BACKEND_PROVIDER</td> <td>mysql</td> </tr> <tr> <td>MLFLOW_BACKEND_STORE_URI</td> <td>[REDACTED]</td> </tr> </tbody> </table> <p>+ Another variable</p> <p>MLFLOW_HOST=0.0.0.0</p> <p>MLFLOW_GUNICORN_OPTS=--log-level debug</p> <p>OCIFS_IAM_TYPE=resource_principal</p> <p>MLFLOW_DEFAULT_ARTIFACT_ROOT=oci://<bucket_name>@<namespace>/ARTIFACTS/</p> <p>MLFLOW_ARTIFACTS_DESTINATION=oci://<bucket_name>@<namespace>/ARTIFACTS/</p> <p>BACKEND_PROVIDER=mysql</p> <p>MLFLOW_BACKEND_STORE_URI=mysql+mysqlconnector://{{mysql-user}}:{mysql-password}@{mysql-host}:3306/mlflow</p>	Key	Value	MLFLOW_HOST	0.0.0.0	MLFLOW_GUNICORN_OPTS	--log-level debug	OCIFS_IAM_TYPE	resource_principal	MLFLOW_DEFAULT_ARTIFACT_ROOT	[REDACTED]	MLFLOW_ARTIFACTS_DESTINATION	[REDACTED]	BACKEND_PROVIDER	mysql	MLFLOW_BACKEND_STORE_URI	[REDACTED]
Key	Value																
MLFLOW_HOST	0.0.0.0																
MLFLOW_GUNICORN_OPTS	--log-level debug																
OCIFS_IAM_TYPE	resource_principal																
MLFLOW_DEFAULT_ARTIFACT_ROOT	[REDACTED]																
MLFLOW_ARTIFACTS_DESTINATION	[REDACTED]																
BACKEND_PROVIDER	mysql																
MLFLOW_BACKEND_STORE_URI	[REDACTED]																
<p>Once the instance is created make note of the Private IP address.</p>	<h3>Primary VNIC</h3> <p>Name: [REDACTED]</p> <p>OCID: ...[REDACTED] Show Copy</p> <p>Public IP address: -</p> <p>Private IP address: [REDACTED] Copy</p> <p>-----</p>																

We can now open up an SSH Tunnel and Port Forward from our Private Tracking Server to Local Laptop via the Jump Server we created earlier in this documentation.

```
ssh -N -i <bastion-private-key>
opc@<bastion-host> -L 5000:<tracking-server-host>:5000
```

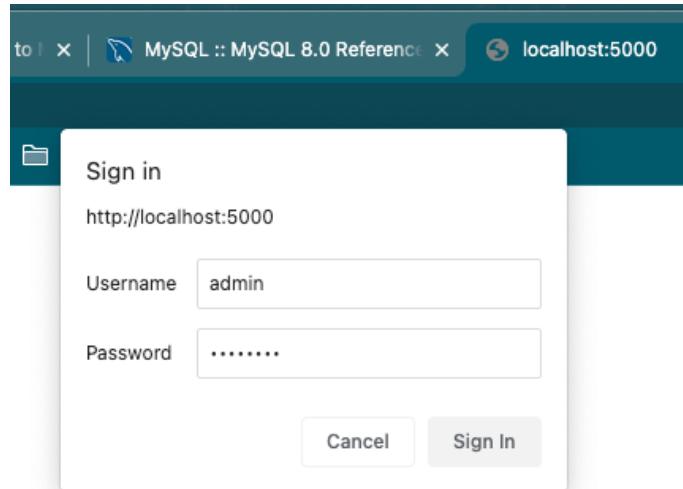
```
Last login: Fri Sep 15 16:35:25 on ttys000
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
$ cd .ssh
$ $ ssh ...
```

Using a local browser, we can visit the tracking server at **localhost:5000**.

The Tracker Server will prompt you with a username and password.

Default username is: **admin**

Default password is: **password**



Execute the **mlflow-auth-mysql-demo.ipynb** Notebook.

Notebook should be in the same directory as this guide.

The goal of this Notebook is to have all the experiments written to the Tracking Server which requires Authentication.

ML Flow - Using OCI Tracking Server

Author: Ismail Syed
Title: Senior Data Scientist
Company: Oracle

Notebook Description: In this Notebook we will look at how we can save our models and model experiment data to our OCI MLFlow Tracking Server.
The data set used in this example is from <http://archive.ics.uci.edu/ml/datasets/Wine+Quality> P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553, 2009.

Once executed, we can return to our MLflow Tracking Server and view the experiments that have been logged.

The screenshot shows the MLflow UI with the title 'mlflow 2.6.0 Experiments Models'. Under 'Experiments', it says 'Default' and 'Experiment ID: 0'. Below is a table with columns: Run Name, Created, and Dataset. The table lists several runs, each with a color-coded icon (red, green, black) and a name like 'wise-hare-811', 'traveling-skunk-413', etc. All runs were created 1 minute ago.

Run Name	Created	Dataset
wise-hare-811	1 minute ago	-
traveling-skunk-413	1 minute ago	-
invisible-doe-429	1 minute ago	-
invisible-hawk-67	1 minute ago	-
mysterious-panda-889	1 minute ago	-

We can also view the Models stored in the tracking server.

The screenshot shows the MLflow UI with the title 'mlflow 2.6.0 Experiments Models'. Under 'Registered Models', it shows 'ElasticnetWineModel'. It displays 'Created Time: 2023-09-25 12:10:44' and 'Last Modified: 2023-09-25 12:11:07'. Below is a table of model versions with columns: Version, Registered at, and Created by. Versions 1 through 5 are listed, all registered at 2023-09-25 12:10:47 and created by the same user.

Version	Registered at	Created by
Version 5	2023-09-25 12:11:07	
Version 4	2023-09-25 12:11:01	
Version 3	2023-09-25 12:10:56	
Version 2	2023-09-25 12:10:50	
Version 1	2023-09-25 12:10:45	

Finally, let's check our Object storage Bucket and MySQL DB (mlflow) where we should see some stored entries and artifacts.

The screenshot shows the 'Objects' section of the MLflow UI. It lists a directory structure under 'ARTIFACTS': 'model/artifacts'. Inside 'model' are files: 'MLmodel', 'conda.yaml', 'model.pkl', 'python_env.yaml', and 'requirements.txt', all modified on Sep 25, 2023, at 11:10:49 UTC.

```

mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 10554
Server version: 8.0.34-0ubuntu0.22.04.1-log MySQL Enterprise - Cloud
Copyright (c) 2000, 2023, Oracle and/or its affiliates.
Name: Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql> connect mlflow
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with '-A'
Connection id: 10555
Current database: mlflow

mysql> select * from experiments;
+-----+-----+-----+-----+
| experiment_id | name | artifact_location | lifecycle_stage | creation_time | last_update_time |
+-----+-----+-----+-----+
| 0 | Default | | active | 1695639494991 | 1695639494991 |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql>

```