**Functions: Get Started using Cloud Shell**

In this tutorial, you use an Oracle Cloud Infrastructure account to set up Oracle Functions development using Cloud Shell. Then, we create a function application and a function. Key tasks include how to:

Set up an authentication token.

Gather required information.

Set up a VCN.

Log in to OCI Registry (OCIR).

Configure Cloud Shell to deploy functions.

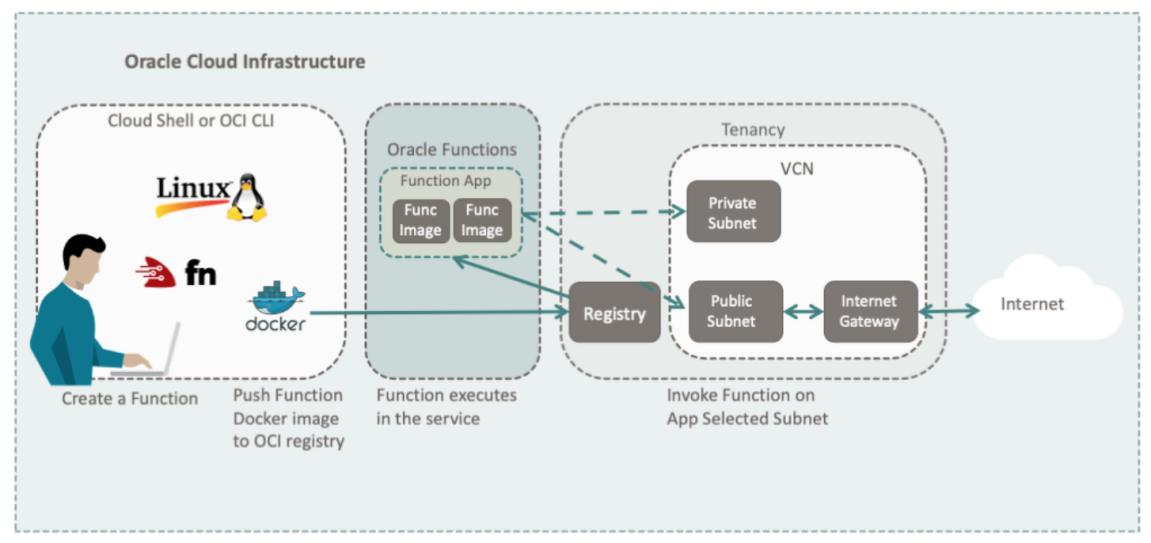
Configure your Fn context.

Create an app for your Oracle function.

Create a function.

Deploy your function.

Test your function.



For additional information, please see:

Start for Free : <https://www.oracle.com/cloud/free/>

Launch your first Linux VM: <https://docs.oracle.com/iaas/Content/GSG/Reference/overviewworkflow.htm>

Cloud Shell: <https://docs.oracle.com/iaas/Content/API/Concepts/cloudshellintro.htm>

Oracle Functions: <https://docs.oracle.com/iaas/Content/Functions/Concepts/functionsoverview.htm>

OCI Registry: <https://docs.oracle.com/iaas/Content/Registry/Concepts/registryoverview.htm>

**Before You Begin**

To successfully perform this tutorial, we must have the following:

A paid Oracle Cloud Infrastructure account. See Signing Up for Oracle Cloud Infrastructure.

Link- <https://docs.oracle.com/iaas/Content/GSG/Tasks/signingup.htm>

Your OCI account configured to support Oracle Functions development. See Create Policy for Oracle Functions.

<https://www.oracle.com/webfolder/technetwork/tutorials/infographics/oci_functions_cloudshell_quickview/functions_quickview_top/functions_quickview/index.html#>

OCI Cloud Shell which is included with your account and includes:

OCI CLI

Docker

Python 3.6+

Java 1.8+

Node.js 10+

1. **Gather Required Information**

Collect all the information needed to complete the tutorial.

Gather Region and Registry Information

Prepare the information we need from the Oracle Cloud Infrastructure Console.

Find your region identifier and region key from Regions and Availability Domains.

Example: us-ashburn-1 and iad for Ashburn.

<https://docs.oracle.com/iaas/Content/General/Concepts/regions.htm>

Create a registry project name to store our function images in OCI Registry (OCIR).

When we publish a function, a Docker image is created in OCIR. Your OCIR project name is prepended to our function images to make them easy to find. For example, given:

Registry project name: my-func-prj

Function name: node-func

Your function image would be stored on OCIR under: my-func-prj/node-func

**Get Compartment, Auth Token, and Collect your Data**

Create or Select a Compartment

<https://docs.oracle.com/iaas/Content/Identity/Tasks/managingcompartments.htm#To>

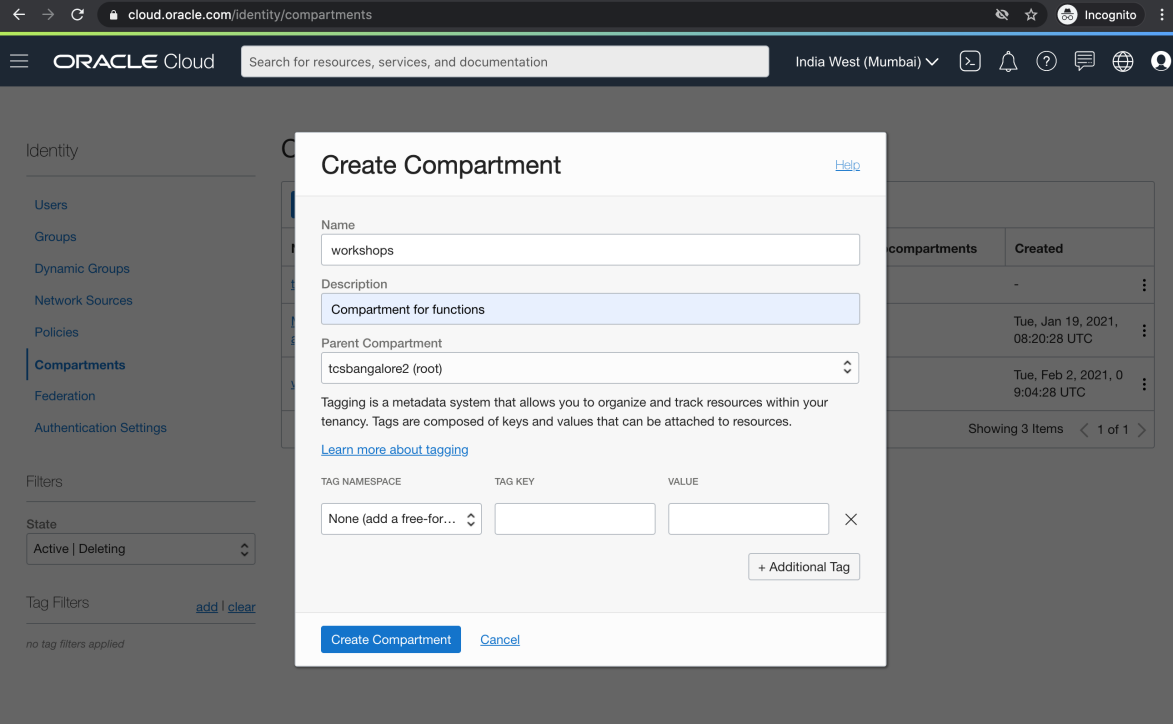
To create a compartment see Create a compartment. After our compartment is created, save the compartment OCID.

To get the compartment OCID from an existing compartment:

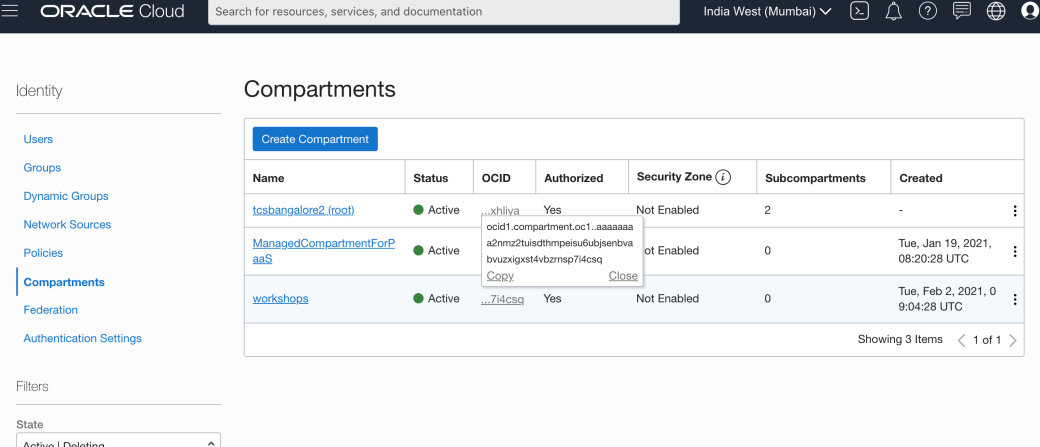
Go to Identity then Compartments.

Select your compartment.

Click the Copy link for the OCID field.



Compartment is Created:



**Create an Authorization Token**

We create an authorization token to log in to the OCI Registry. To create an authorization token:

In the top navigation bar, click your user avatar.

Select your username.

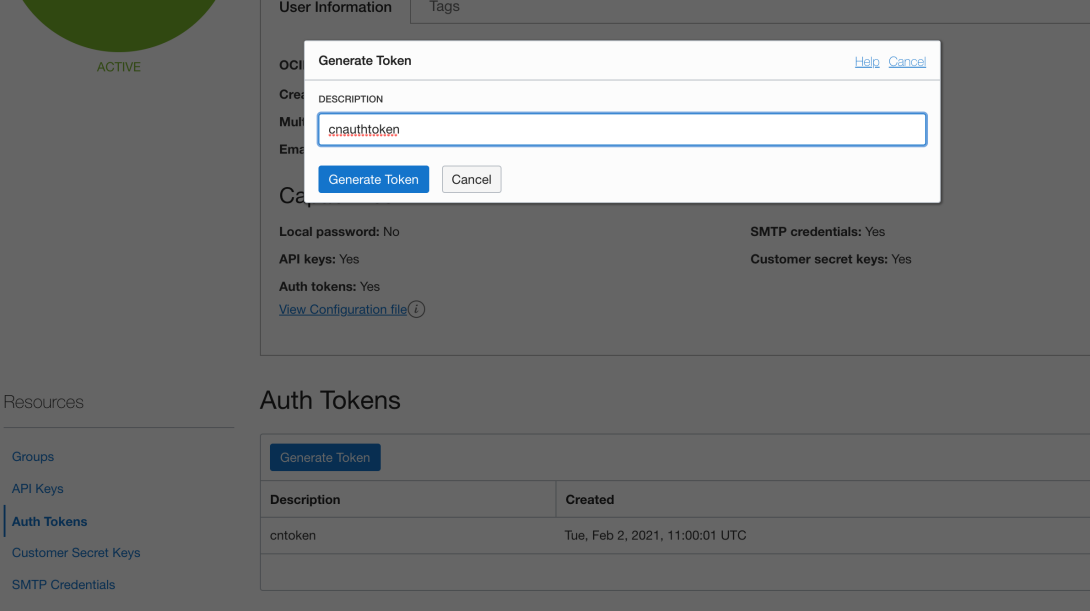
Click Auth Tokens.

Click Generate Token.

Give it a description.

Click Generate Token.

Copy the token and save it.



**Collect our Information**

Collect all the information needed to complete the tutorial. Copy the following information into your notepad.

Region: <region-identifier>

Example: ap-mumbai-1.

Region Key: <region-key>

Example: bom.

Registry Project Name: <your-project-name>

Example: my-func-prj.

Compartment ID: <compartment-id>

Example: ocid1.compartment.oc1..aaaaaaaa2nmz2tuisdthmpeisu6ubjsenbvabvuzxigxst4vbzrnsp7i4csq

Auth Token: <auth-token>

Example: 3:0y1BF2Jj+;s\_<sn]fE

Tenancy name: <tenancy-name>

From your user avatar, example: tcsbangalore2

Tenancy OCID: <tenancy-ocid>

From your user avatar, go to Tenancy: <your-tenancy> and copy OCID, example: ocid1.tenancy.oc1..aaaaaaaaonu3r4bfkxpkccywutzd4ekmzt3ys4krlrojg25ixuwd3oxhliya

Username: <user-name>

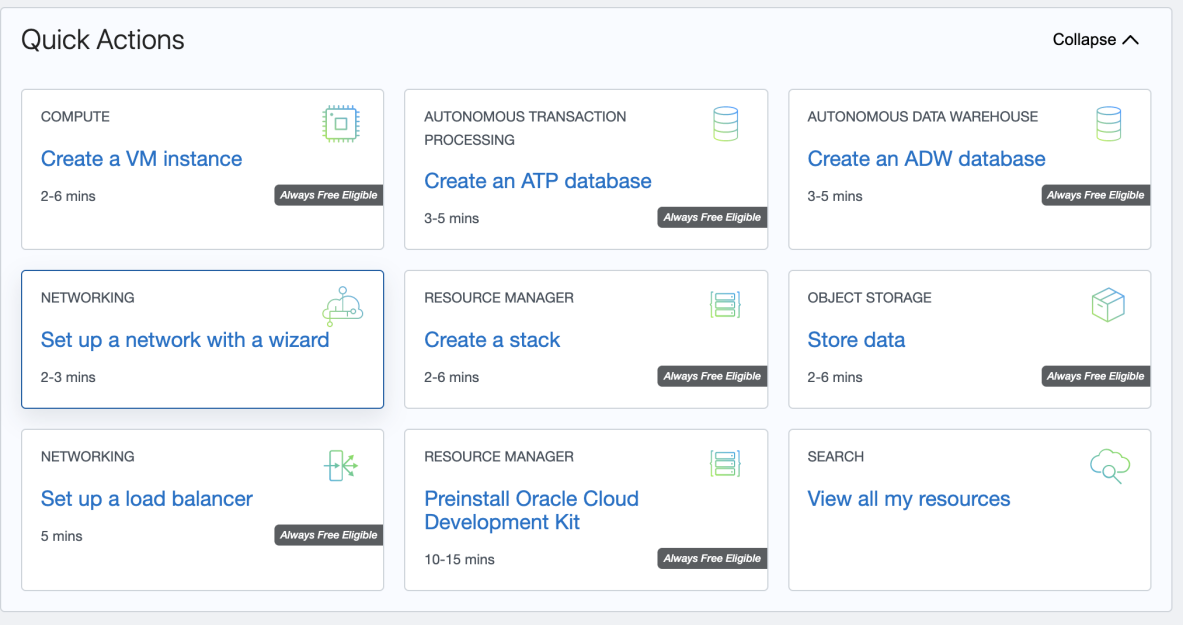
From your user avatar.

Eg: oracleidentitycloudservice/Saikat

**Note**: For a user with admin access the username would be like: saikatdey

1. **Create your Virtual Cloud Network (VCN)**

1. From the main landing page, select Set up a network with a wizard.



2. In the Start VCN Wizard workflow, select VCN with Internet Connectivity and then click Start VCN Wizard .

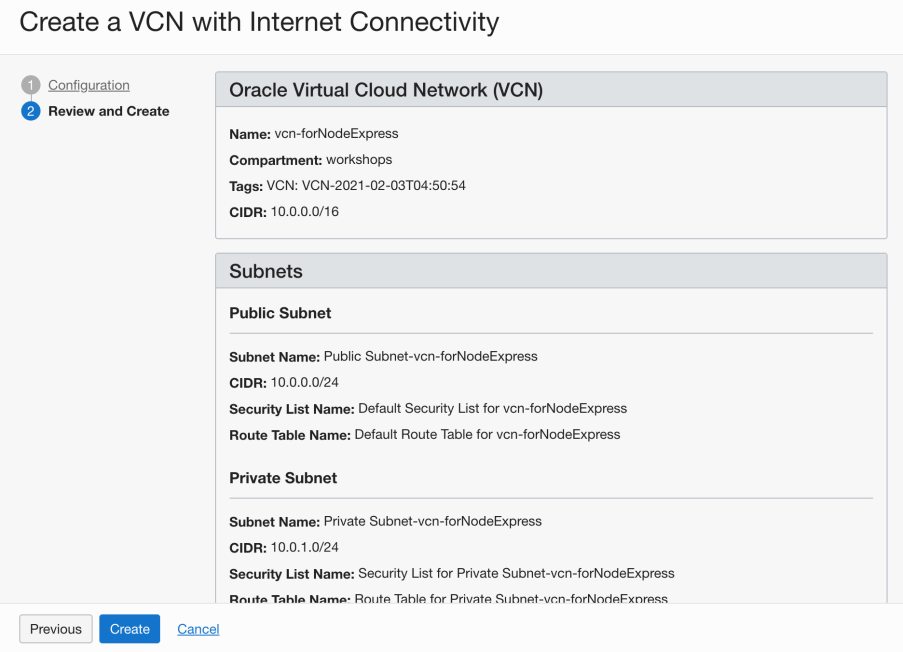
3. In the configuration dialog, fill in the VCN Name for our VCN. Our Compartment is already set to its default value of <our-tenancy> (root).

4. In the Configure VCN and Subnets section, keep the default values for the CIDR blocks:

VCN CIDR BLOCK: 10.0.0.0/16

PUBLIC SUBNET CIDR BLOCK: 10.0.0.0/24

PRIVATE SUBNET CIDR BLOCK: 10.0.1.0/24



5. For DNS RESOLUTION uncheck USE DNS HOSTNAMES IN THIS VCN.Picture shows the USE DNS HOSTNAMES IN THIS VCN option is unchecked.

Click Next.

6. The Create a VCN with Internet Connectivity configuration dialog is displayed (not shown here) confirming all the values our just entered.

7. Click Create to create our VCN.

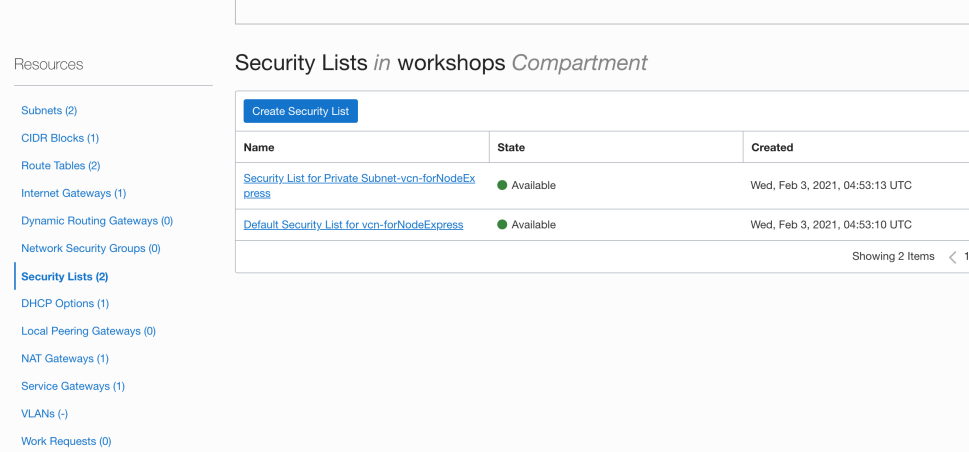
The Creating Resources dialog is displayed (not shown here) showing all VCN components being created.

8. Click View Virtual Cloud Network to view our new VCN.

Our new VCN is displayed. Now we need to add a security rule to allow HTTP connections on port 3000, the default port for our application.

9. With our new VCN displayed, click our Public subnet link.

The public subnet information is displayed with the Security Lists at the bottom of the page. There should be a link to the Default Security List for our VCN.



10. Click on the Default Security List link.

The default Ingress Rules for our VCN are displayed.

11. Click Add Ingress Rules.

An Add Ingress Rules dialog is displayed.

12. Fill in the ingress rule with the following information. Once all the data is entered, click Add Ingress Rules.

Fill in the ingress rule as follows:

Stateless: Checked

Source Type: CIDR

Source CIDR: 0.0.0.0/0

IP Protocol: TCP

Source port range: (leave-blank)

Destination Port Range: 3000

Description: VCN for applications

Once we click Add Ingress Rule, HTTP connections are allowed to our public subnet.

1. **Log into the OCI Registry**

Next, you log Docker into the OCI Registry (OCIR).

Log your Docker into OCIR

Get the information you gathered earlier.

Open a terminal window.

Log in to OCIR:

docker login <region-key>.ocir.io

You are prompted for your login name and password.

Username: <tenancy-name>/<user-name>

Password: <auth-token>

**In Cloud Shell:**

sudo docker login ap-mumbai-1.ocir.io

bmdrgwy1wsjh/oracleidentitycloudservice/Saikat

3:0y1BF2Jj+;s\_<sn]fE

You have logged your instance into OCIR.

**4. Configure Functions**

To use Oracle Functions, we must configure the Fn application context. The context stores the values needed to connect to the Oracle Functions service. Fn client commands are used to add the required configuration data.

Configure the Fn Context for the Cloud Shell

We need the information you gathered from earlier on. Use Fn client commands to configure Fn.

Open your Cloud Shell instance.

Get a list of Fn contexts.

fn list context

We see contexts for default and <your-region-identifier>.

Select the context named with <your-region-identifier>.

For example: fn use context us-phoenix-1

List the Fn contexts to ensure <our-region-identifier> is selected. (Has a star next to it.)

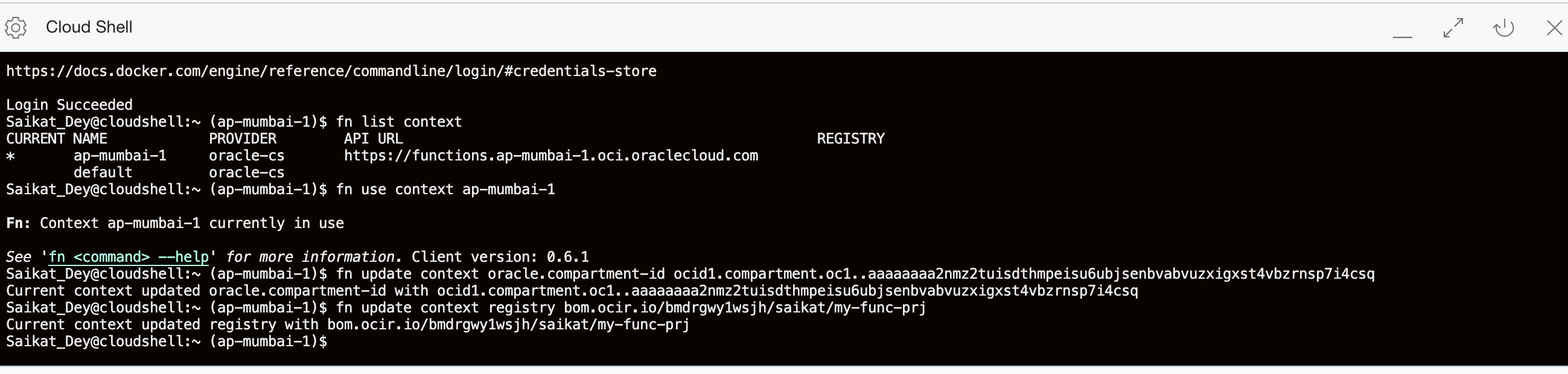
Set the compartment for Oracle Functions.

Example: fn update context oracle.compartment-id ocid1.compartment.oc1..aaaaaaaarvdfa72n...

Set the URL for our Registry repository.

Sample command: fn update context registry <region-key>.ocir.io/<tenancy-namespace>/<registry-project-name>

Example: fn update context registry phx.ocir.io/my-tenancy/my-func-prj



**Note**: View/Edit your Context

Our Fn context files are located in the ~/.fn/contexts directory. Each context is stored in a .yaml file. For example, our us-phoenix-1.yaml file might look similar to:

api-url: https://functions.us-phoenix-1.oci.oraclecloud.com

oracle.compartment-id: ocid1.compartment.oc1..aaaaaaaarvdfa72n...

provider: oraclecs

registry: phx.ocir.io/my-tenancy/my-func-prj

We can edit the file directly with an editor if necessary.

For a detailed explanation of each step, see:

<https://www.oracle.com/webfolder/technetwork/tutorials/infographics/oci_functions_cloudshell_quickview/functions_quickview_top/functions_quickview/index.html#>

You have now setup the Fn context for your instance.

1. **Create and Deploy a Function**

With your configuration complete, it is time to create and deploy a function.

**Create an Application**

An Application is the main storage container for functions. Each function must have an application for deployment. To create application, follow these steps.

Open Developer Services and then Functions which are part of the Solutions and Platform grouping.

Click Create Application.

Fill in the form data.

Name: <your-app-name>

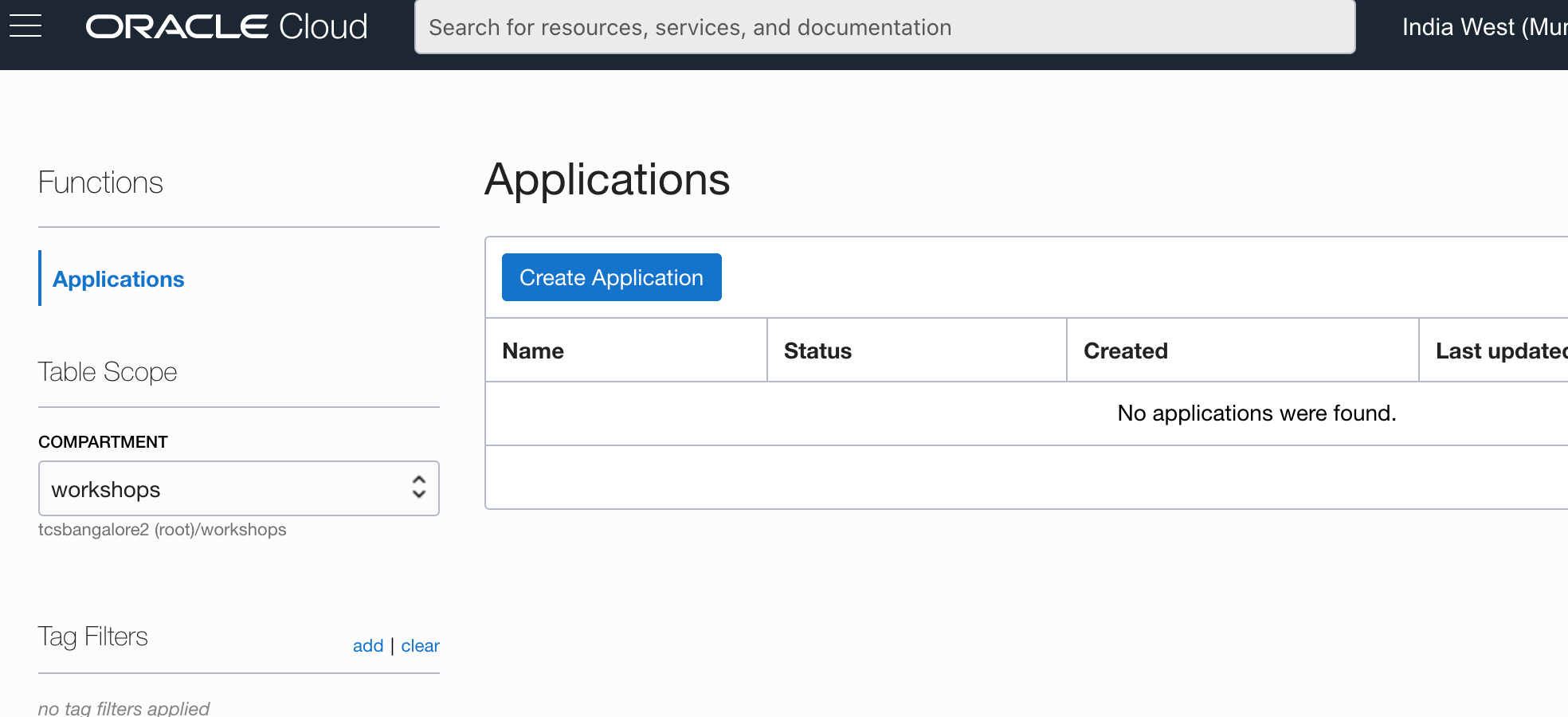
VCN: <your-VCN>

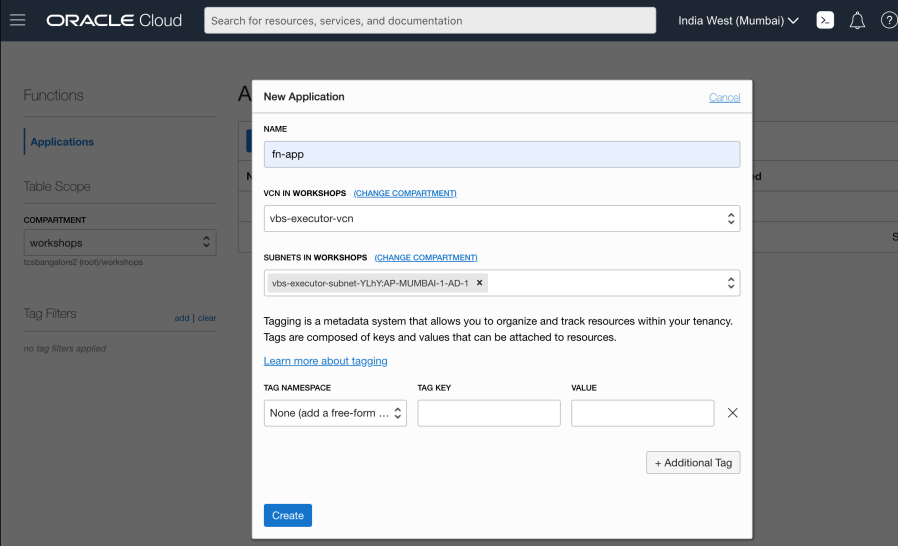
Subnets: <your-public-subnet> or <your-private-subnet>

**Note**: A public or private subnet may be used, select one.

Click Create.

Our app is created.





**Choose a Language to Create and Deploy a Function**

Select one of the following languages to create and deploy a function. If you want, you can do all three.

Select one of the following languages to create and deploy a function. If you want, you can do all three.

Create and Deploy a Java Function

With your application created, deploy a Java function. Follow these steps to create a Java "Hello World" function.

**Note**: Ensure Java 8+ is installed to perform these steps.

Open Cloud Shell.

Create a directory to store your functions and change into that directory.

mkdir my-dir-name

cd my-dir-name

Create a Java "Hello World" function with Fn.

fn init --runtime java my-func-name

This command creates a directory named my-func-name with several files in it.

func.yaml - Function configuration file.

pom.xml - Maven build file.

src/main/java/com/example/fn/HelloFunction.java - The actual function file.

Change into the directory.

Deploy the function.

fn -v deploy --app your-app-name

Various messages are displayed as the docker images are built, pushed to OCIR, and eventually deployed to Oracle Functions.

Invoke the function.

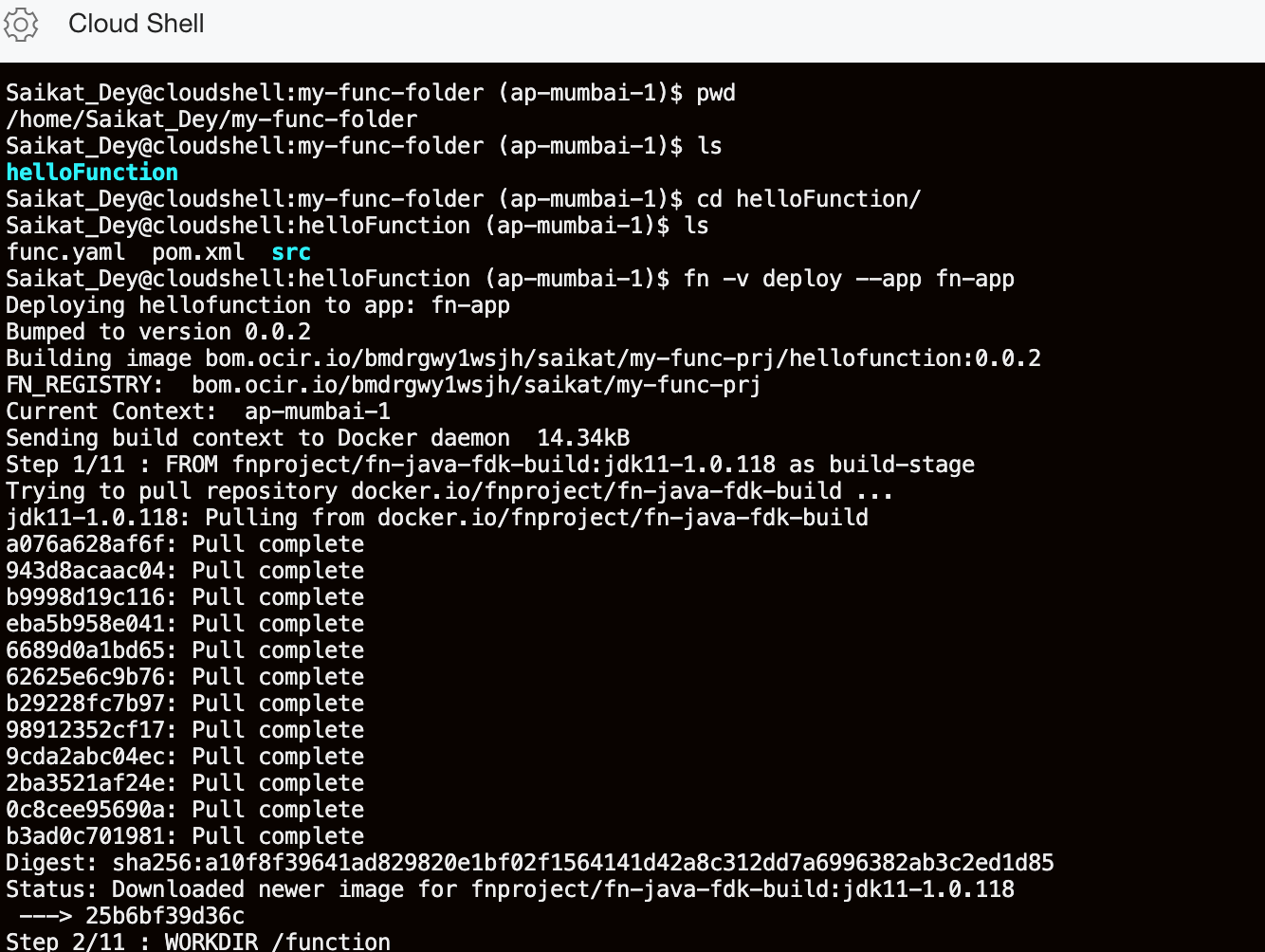
fn invoke your-app-name my-func-name

Returns: Hello, world!

Invoke the function with a parameter.

echo -n "Bob" | fn invoke your-app-name my-func-name

Returns: Hello, Bob!





If you want to connect to your function from the net, you need to get the function's invoke endpoint. To find our invoke endpoint use the inspect command.

fn inspect function your-app-name my-func-name

Examine the results of the inspect command. Notice the invoke endpoint URL is included in the annotatins section of the returned JSON data.

{

"annotations": {

"fnproject.io/fn/invokeEndpoint": "https://aaaaaaaaa.us-ashburn-1.functions.oci.oraclecloud.com/1111111/functions/ocid1.fnfunc.oc1.iad.aaaaaaaaa.../actions/invoke",

"oracle.com/oci/compartmentId": "ocid1.compartment.oc1..aaaaaaaa...",

"\_\_comment":"Remaining output left out for brevity",

Use the URL returned from inspect to invoke the function. Because functions require requests to be digitally signed, the oci raw-request command is used for this example.

oci raw-request --http-method POST --request-body "" --target-uri https://https://aaaaaaaaa.us-ashburn-1.functions.oci.oraclecloud.com/1111111/functions/ocid1.fnfunc.oc1.iad.aaaaaaaaa.../actions/invoke

The command returns:

{

"data": "Hello, world!",

"headers": {

"Content-Length": "13",

"Content-Type": "text/plain",

"Date": "Tue, 20 Oct 2020 00:53:08 GMT",

"Fn-Call-Id": "11111111111",

"Fn-Fdk-Version": "fdk-java/1.0.111 (jvm=OpenJDK 64-Bit Server VM, jvmv=11.0.8)",

"Opc-Request-Id": "1111111/11111"

},

"status": "200 OK"

}

**Note:** We can connect to a Functions endpoint using tools like curl. However, because of security considerations, the script is complex. For details and an example, see the oci-curl section on the Invoking Functions page.



We have successfully deployed and tested a Java function.

1. **Review Function Information**

**View Function Images in OCIR**

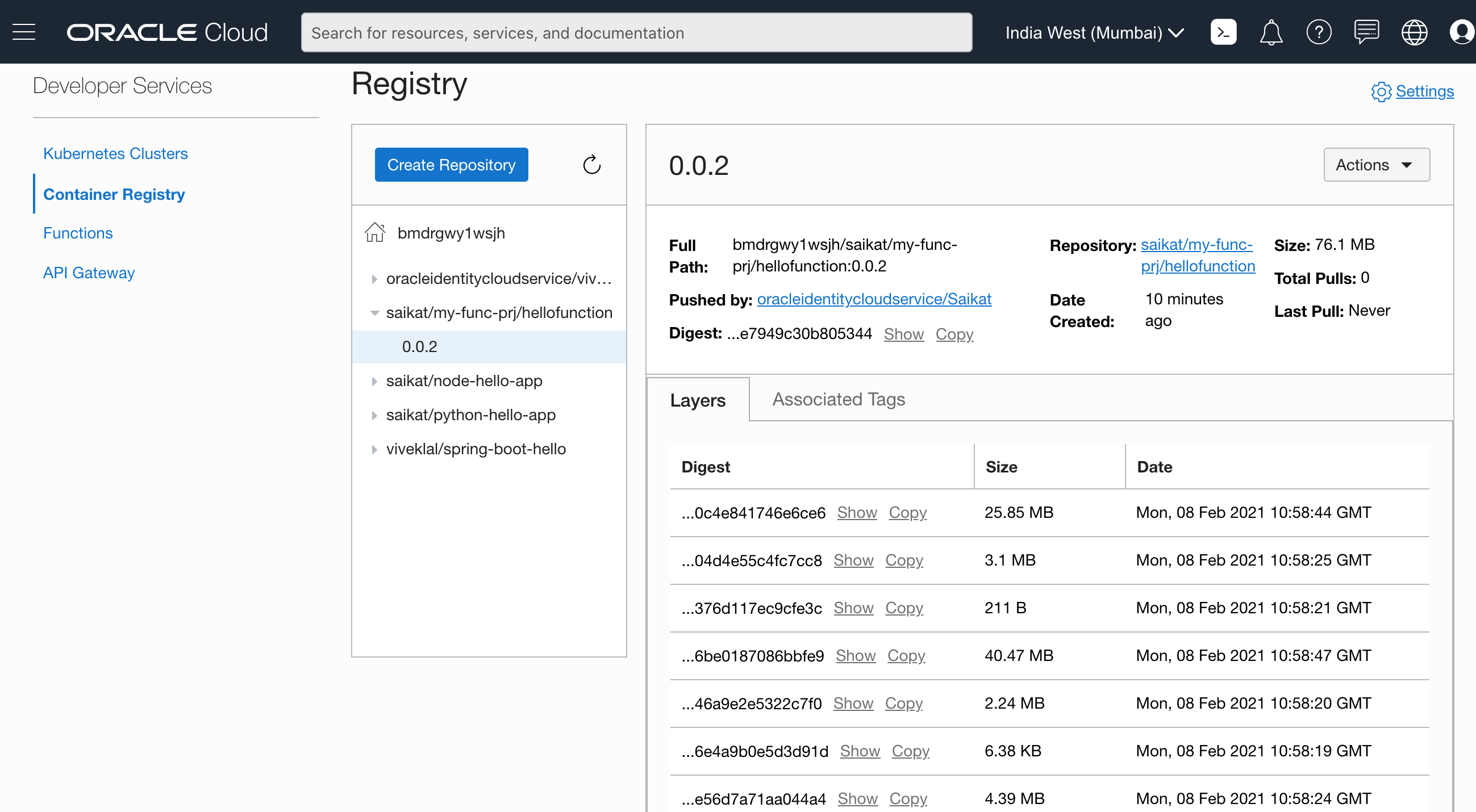
When you deploy a function, it is uploaded and stored in OCIR. We can navigate to OCIR and examine the function images.

From the main menu, select Developer Services then Container Registry which is part of the Solutions and Platform grouping.

Search for the <your-repository-project-name>.

Under our project name, you should see an entry for each function we deployed.

Click the link of each image you want to see information about.



**View Function Execution Information**

After you run a function, you can display metrics for that function.

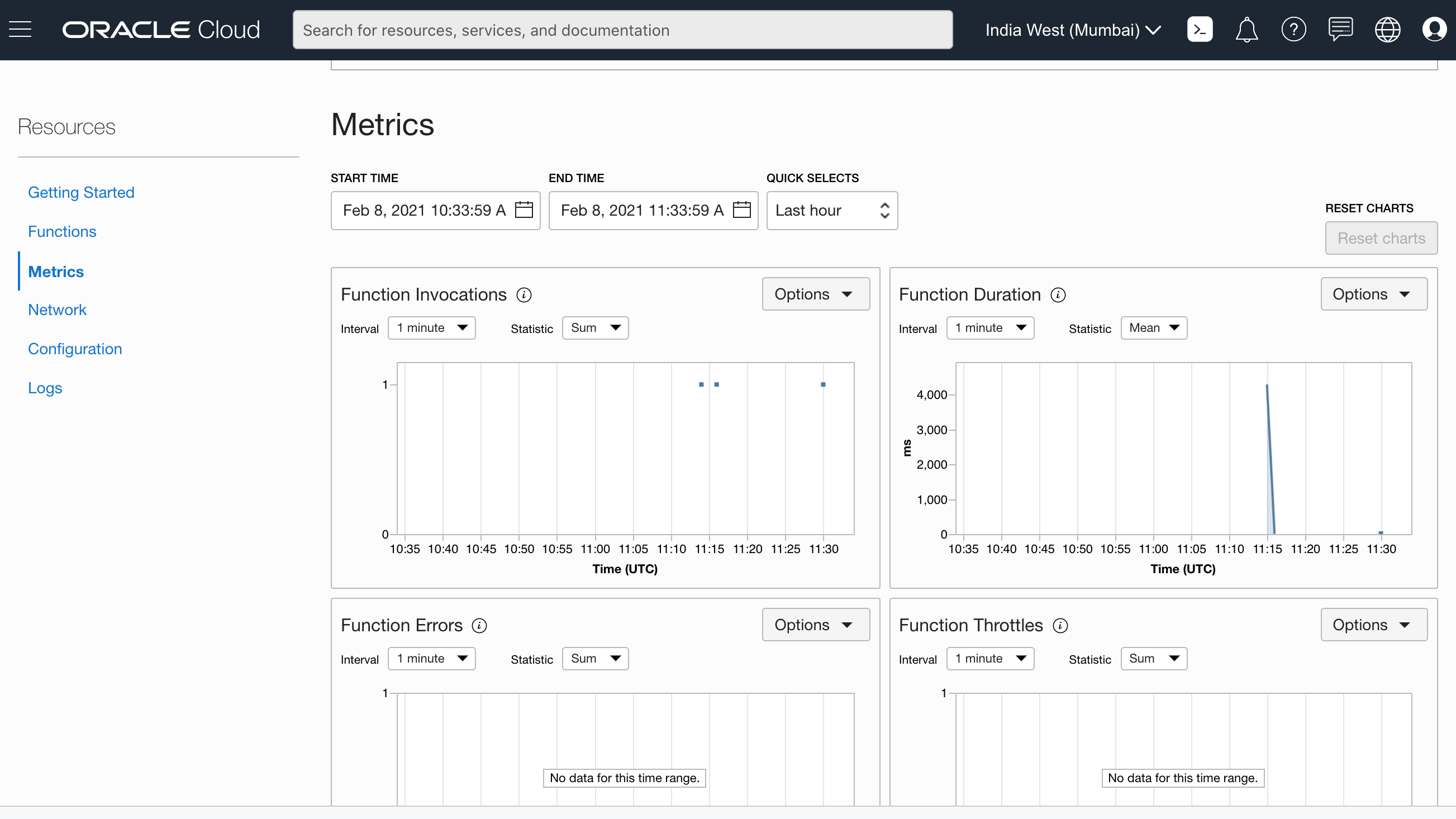
From the main menu, select Developer Services and then Functions which are part of the Solutions and Platform grouping.

Our applications are listed on the page.

Click the link to the application you created.

Click the link to the function you want to examine.

Metric information about your function is displayed.



**Enable and View Logging Information**

To enable, logging for an application, follow these steps.

From the main menu, select Developer Services and then Functions which are part of the Solutions and Platform grouping.

Our applications are listed on the page.

Click the link to the application you created.

On the left side of the application page, click the Logs link.

Click Disabled to enable logging for your application.

The Enable Log dialog is displayed. Fill in the following information:

Compartment: <your-compartment-name>

Log Group: Take the default value Auto-Create a Default Log Group

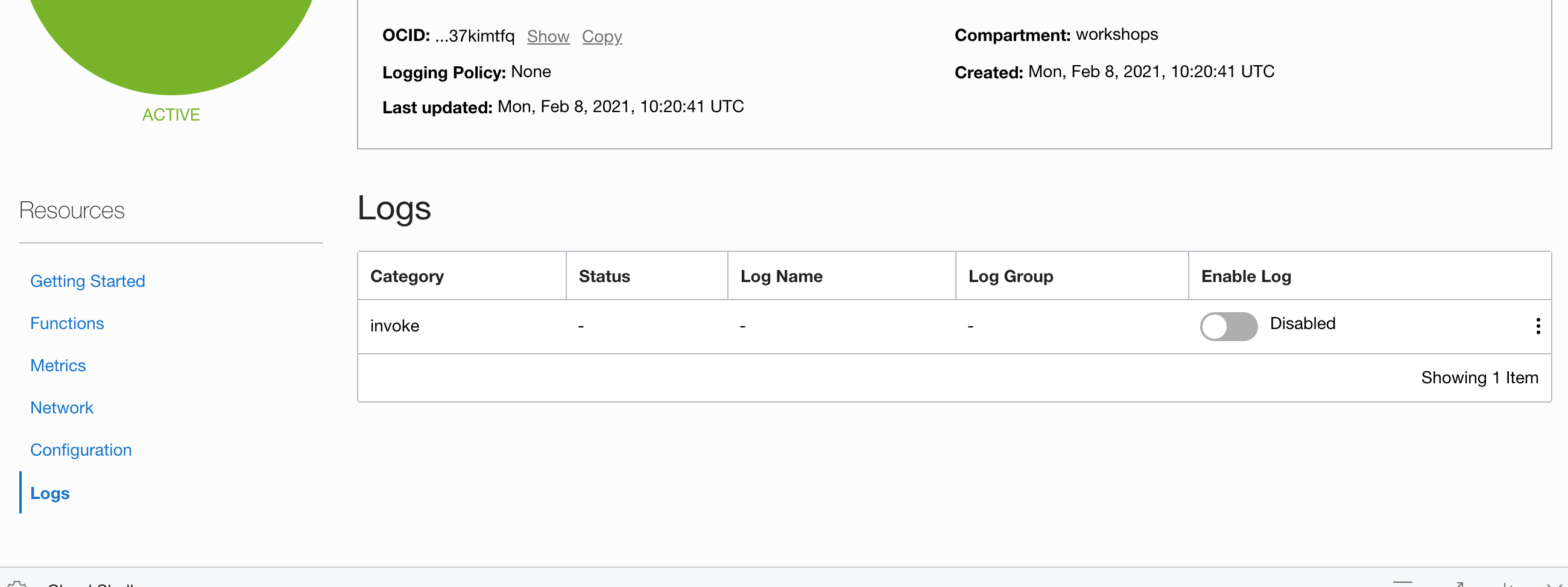
Log name: <take-default>

Log Retention: <take-default>

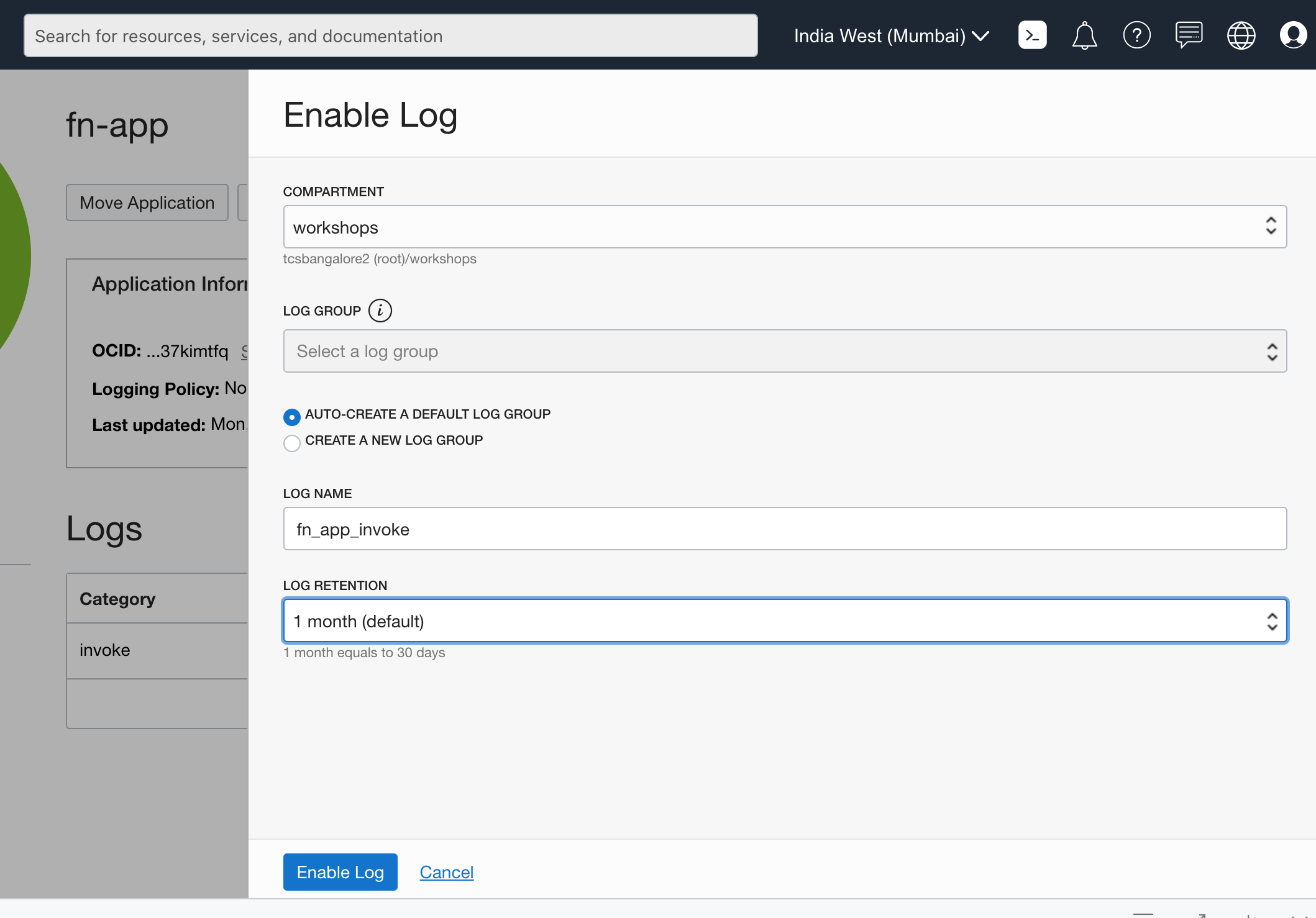
Click Enable Log

Wait a moment for your log to be created.

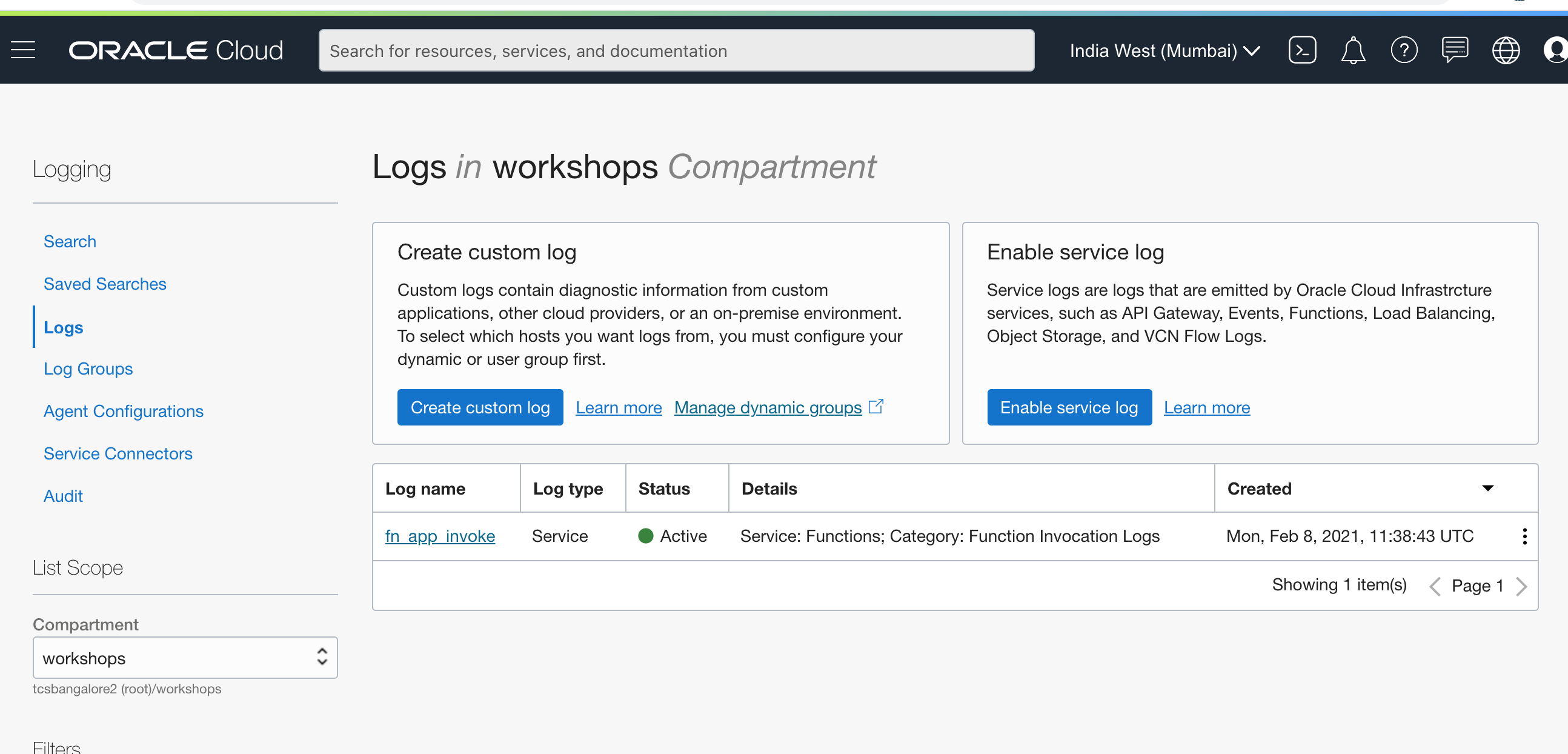
To view your log, click the log name link created by the preceding steps.



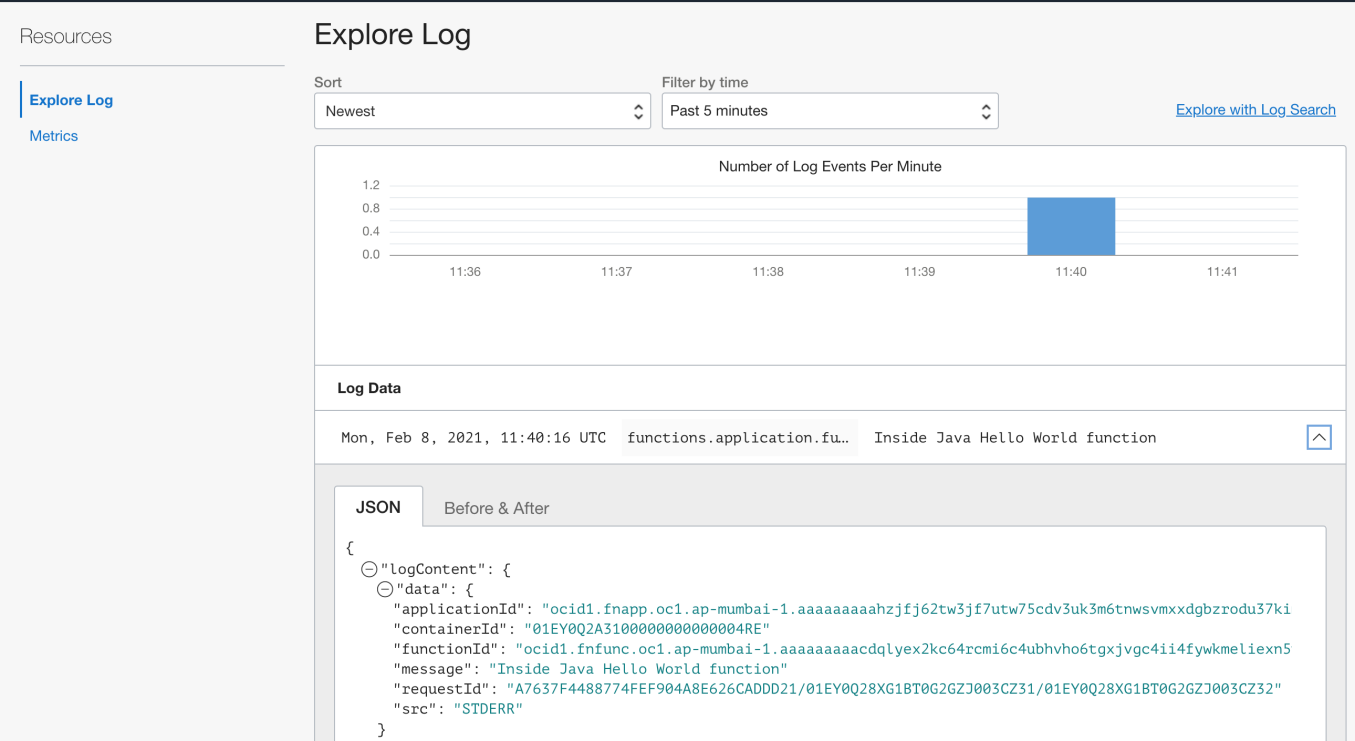
Enabling Logs:



Logs in the compartment:



Logs:



We have successfully created a function and deployed it to Oracle Functions.

To explore more information about development with Oracle products, check out these sites:

Oracle Developers Portal- <https://developer.oracle.com/>

Oracle Cloud Infrastructure- <https://www.oracle.com/cloud/>

Oracle Functions- <https://docs.oracle.com/iaas/Content/Functions/Concepts/functionsoverview.htm>

---------