

From Zero to Production in Kubernetes - Workshop – Lab Guide

Table of Content

INTRODUCTION	2
GET STARTED	3
LAB 1 - DEPLOYING MUSHOP LOCALLY USING DOCKER.....	13
LAB 2 – DEPLOY OKE CLUSTER WITH TERRAFORM	30
LAB 3 - OCI VAULT INTEGRATION WITH OKE.....	33
LAB 4 - INSTALL MUSHOP USING HELM	46
LAB 5 - INSTALL INGRESS CONTROLLER	51
LAB 6 - INSTALL CERTMANAGER.....	52
LAB 7 – DEPLOYMENT AUTOSCALING	57
LAB 8 – MONITORING THE DEPLOYMENT.....	64
LAB – 9 - CREATE ALARMS TO DETECT UNSCHEDULED PODS	71
LAB 10 - USE TAINTS AND TOLERATIONS FOR RESOURCE ALLOCATION (OPTIONAL)	77

From Zero to Production in Kubernetes - Workshop – Lab Guide

Introduction

Build and Deploy the MuShop App

Learn how to deploy, Maintain and Troubleshoot the MuShop App into a cloud-native microservice-based development framework leveraging the Oracle Cloud Native solutions. You will deploy an eCommerce application with polyglot microservices for each component of a highly scalable architecture leveraging Kubernetes and Oracle Cloud Infrastructure components.

Objectives

In this lab, you will:

- Create Resources and setup a Developer Environment based on OKE
- Deploy an eCommerce microservices-based solution to Kubernetes (MuShop App)
- Working with infrastructure as a code tools
- Troubleshooting challenges
- Explore your App
- Expose your app publicly
- Integrate the application with Logging, monitoring and 3rd party services

Prerequisites

1. Proficient knowledge and experience with Kubernetes
2. Familiarity with YAML, commands, manifests, and Kubernetes architecture
3. Experienced with Helm charts
4. An Oracle Free Tier (Trial) Account

Get Started

Introduction

Before you get started, you will need an Oracle Cloud account. This lab walks you through the steps of getting an Oracle Cloud Free Tier account and signing in.

Existing Cloud Accounts

If you already have access to an Oracle Cloud account, skip to **Task 2** to sign in to your cloud tenancy.

Objectives

- Create an Oracle Cloud Free Tier account
- Sign in to your account

Prerequisites

- A valid email address
- Ability to receive SMS text verification (only if your email isn't recognized)

Note: Interfaces in the following screenshots may look different from the interfaces you will see.

Task 1: Create Your Free Trial Account

If you already have a cloud account, skip to **Task 2**.

1. Validate that you received email in the following format from Oracle's marketing and click the link for registration or copy it to your browser.

בהתשך להרשמתך לקורס ההוראות עם ענן אורקל שמותקיים מחר הסדנא כולל תרגול מעשי בסביבת הענן של אורקל. לקרואת הסדנא אנחנו מעניקים עבור הנרשמים **400 אירו קודיש בסביבת ענן חדשה** כהטבה לנרשמי הסדנא.

*יש לפתח את חשבון הענן לפני הגעתכם לסדנא!

ניתנה לך הרשאה התקפה ל-30 יום יש צורך להזין פרטי כרטיס אשראי עבור זיהוי המשתמש אך הוא לא יחויב והוא ישמר באמצעות חיבור.

מצורפות הנחיות לפיתוח החשבון:

קישור לפתיחת החשבון:

[signup.oraclecloud.com] Oracle Cloud Free Tier Signup

חשיבות: בהרשמה השתמשו באימייל אותו נרשמתם לקורס.

במידה ובעבר נפתח עבורכם חשבון ענן באורקל עם האימייל שנרשמתם, יש צורך לשלוח לנו אימייל אחר שאיתו תוכלו לקבל גישה.

בנוסף מצורף לינק לסרטון הדרכה קצר איך להציג את החשבון:

[[youtube.com](https://www.youtube.com/watch?v=Yr1Cz-U-Kok)] <https://www.youtube.com/watch?v=Yr1Cz-U-Kok>

2. An alternative option in case of no marketing email available is to Open up a web browser to access the Oracle Cloud account registration form at oracle.com/cloud/free.
3. In both cases above you will be presented with a registration page.

Oracle Cloud Free Tier

Get started with...

Always-Free access to essential services including:

- Autonomous Database
- Object storage

Plus, \$500 of credits for 30 days to use on even more services:

- Container Engine for Kubernetes
- Analytics Cloud
- Data Integration

Account Information

Country/Territory

First Name Last Name

Email

I am human

Verify my email

Terms of Use

By clicking on the button, you understand and agree that the use of Oracle's web site is subject to the [Oracle.com Terms of Use](#). Additional details regarding Oracle's collection and use of your personal information, including information about access, retention, rectification, deletion, security, cross-border transfers and other topics, is available in the [Oracle Privacy Policy](#).

4. Enter the following information to create your Oracle Cloud Free Tier account.
 - Choose your **Country**
 - Enter your **Name** and **Email**
 - Use hCaptcha to verify your identity
5. Once you have entered a valid email address, select the **Verify my email** button. The screen will appear as follows and you should click **Select Offer**

Special Oracle Offer

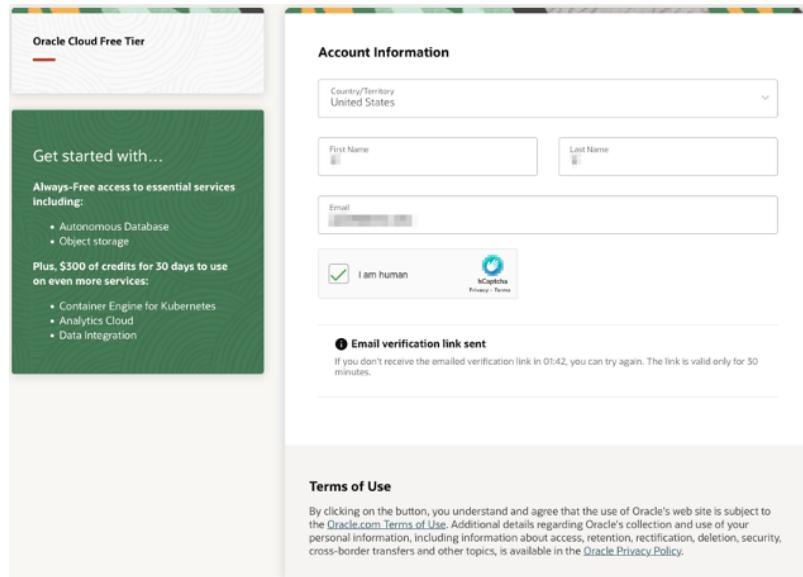
Welcome

You have been invited to sign up for Oracle Cloud and have the following offers available:

For Oracle Cloud Trial with €400 credits for 30 days

Select Offer

Wait for Email verification link sent notification



6. Go to your email. You will see an account validation email from Oracle in your inbox. The email will be similar to the following:

An email from Oracle Cloud. The subject is 'Verify your email to create your Oracle Cloud account'. The sender is Oracle Cloud <noreply@verify...> and the recipient is [REDACTED]. The date is Thu 12/8/2022 12:37 AM. The email body starts with 'Hello [REDACTED]' and says: 'Thanks for your interest in creating an Oracle Cloud account. To create your account, please verify your email address by clicking below.' It features a large orange button labeled 'Verify email'. At the bottom, it says 'Thanks,' and 'The Oracle Cloud Team'.

7. Click **Verify email**.
8. Enter the following information to create your Oracle Cloud Free Tier account.
 - Choose a **Password**
 - Customer Type : **Change to Individual**
 - Your **Cloud Account Name** will generate automatically based on your inputs. You can change that name by entering a new value. **Remember what you wrote. You'll need this name later to sign in.**
 - Choose a **Home Region : Frankfurt** ! Your Home Region cannot be changed once you sign-up.
 - Checkbox the acknowledge message
 - Click **Continue**

Oracle Cloud Free Tier

Get started with...

Always-Free access to essential services including:

- Autonomous Database
- Object storage

Plus, \$300 of credits for 30 days to use on even more services:

- Container Engine for Kubernetes
- Analytics Cloud
- Data Integration

Account Information

Country/Territory
United States

First Name _____ Last Name _____

Email _____

Password _____

>Password must contain a minimum of 8 characters, 1 lowercase, 1 uppercase, 1 numeric, and 1 special character.
Password cannot exceed 40 characters, contain the users first name, last name, email address, spaces, or ~ < > \ characters.

Confirm Password _____

Customer type

Corporate Individual

Cloud Account Name
itvictormartindev

This will be assigned to your company's or organization's environment when signing into the Console. You can always [rename](#) it later from the Console.

Home Region

NORTH AMERICA

9. Enter your address information. Choose your country and enter a phone number.
 Click **Continue**.

Address Information

Address Line 1

Address Line 2 Optional

City

State Zip/Postal Code

Phone Number

Trunk prefix/codes are not used when entering your mobile number (only use 123... instead of *0*123... or *1*123...). Enter numbers without spaces and special characters included.

Please provide a valid phone number. Oracle does not accept text only mobile numbers as we may need to speak to you if there are questions about your account.

Continue

10. Click the **Add payment verification method** button.

Address Information

Payment Verification

You won't be charged unless you elect to upgrade the account.

You may see a small, temporary charge on your payment method. This is a verification hold that will be removed automatically. See the [FAQ](#) for more information.

Add payment verification method

Oracle uses third-party payment processor CyberSource for Oracle Store payment processing. CyberSource will request and collect certain information as part of the payment processing. Please refer to CyberSource's privacy statement at: <http://www.cybersource.com/privacy> for the terms applicable to the data collected..

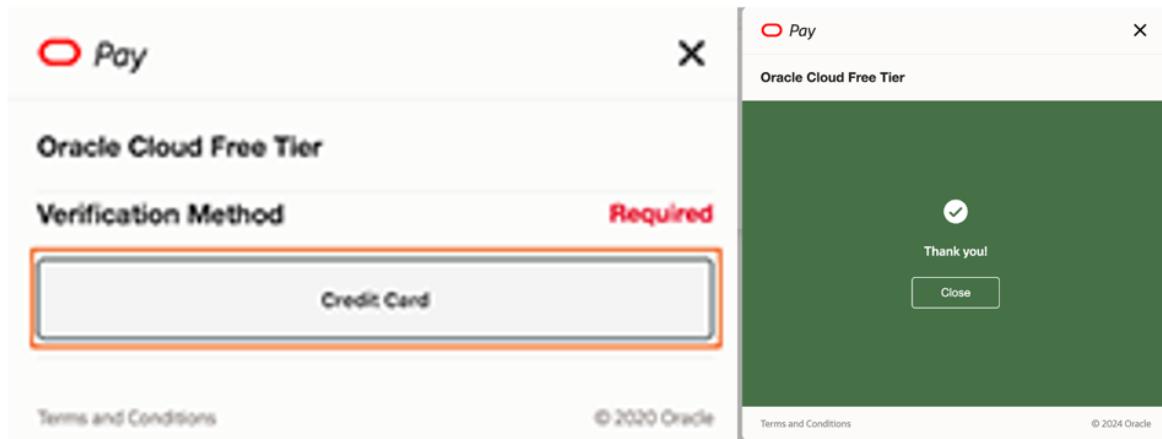
Agreement

By clicking [Start my free trial](#), I agree to the terms and conditions of the [Oracle Cloud Services Agreement for Oracle America, Inc.](#) (also available [here](#)) and this order including Service Description for Free Oracle Cloud Promotion Universal Credits - Part Number B88385

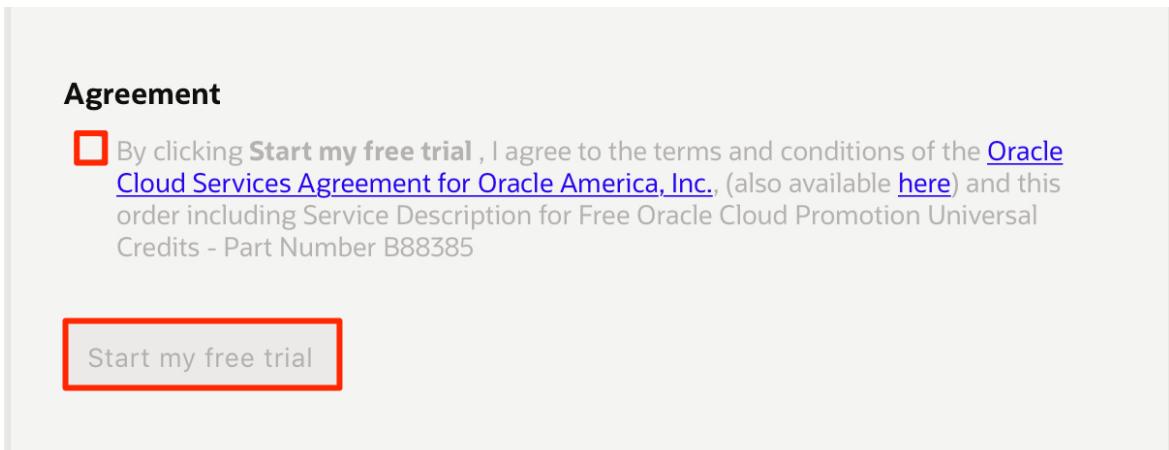
[Start my free trial](#)

11. Choose the verification method. In this case, click the **Credit Card** button. Enter your information and payment details.

Note: This is a free credit promotion account. You will not be charged unless you elect to upgrade the account.



12. Once your payment verification is complete, review and accept the agreement by clicking the check box. Click the Start my free trial button.

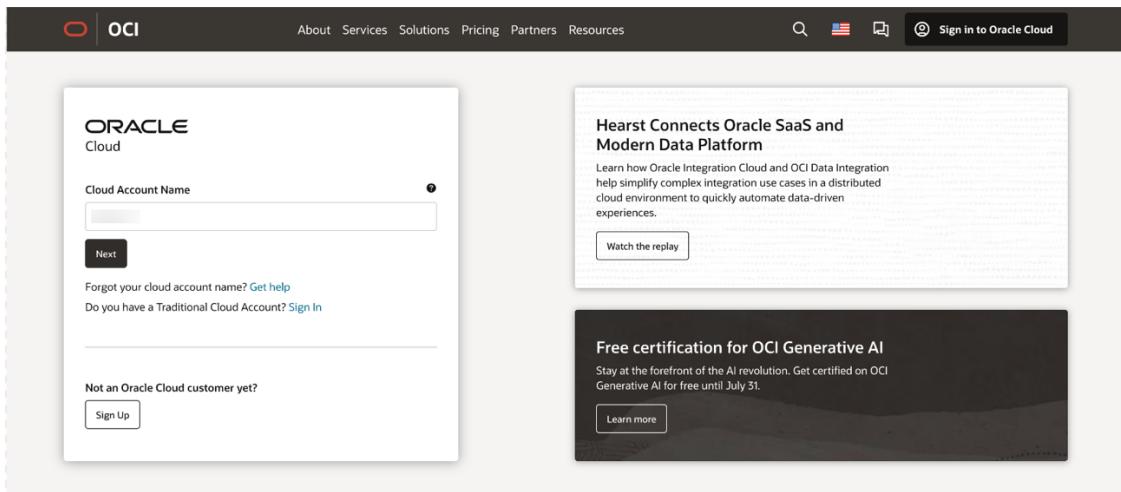


13. Your account is provisioning and should be available soon! You might want to log out as you wait for your account to be provisioned. You'll receive an email from Oracle notifying you that provisioning is complete, with your cloud account and username.

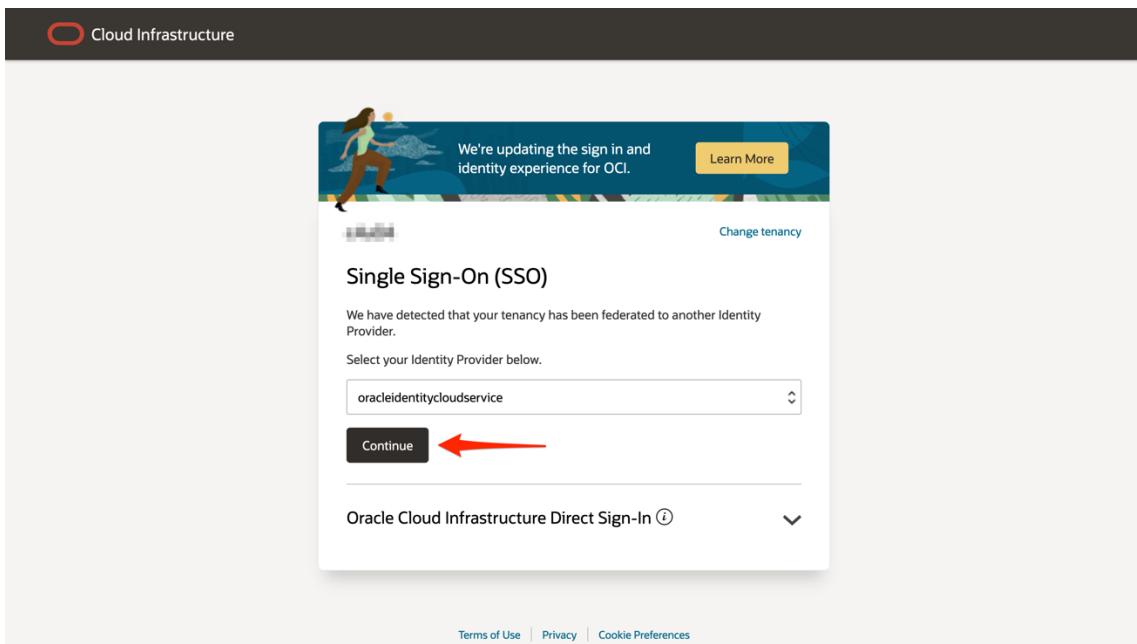
Task 2: Sign in to Your Account

Please note that while your tenancy is initially created, you will only see a direct login. Once your tenancy is fully provisioned, you will see the screens as described below.

1. Go to cloud.oracle.com. Enter your Cloud Account Name and click Next. This is the name you chose while creating your account in **Task 1/Section 8**. It's NOT your email address. If you've forgotten the name, see the confirmation email.

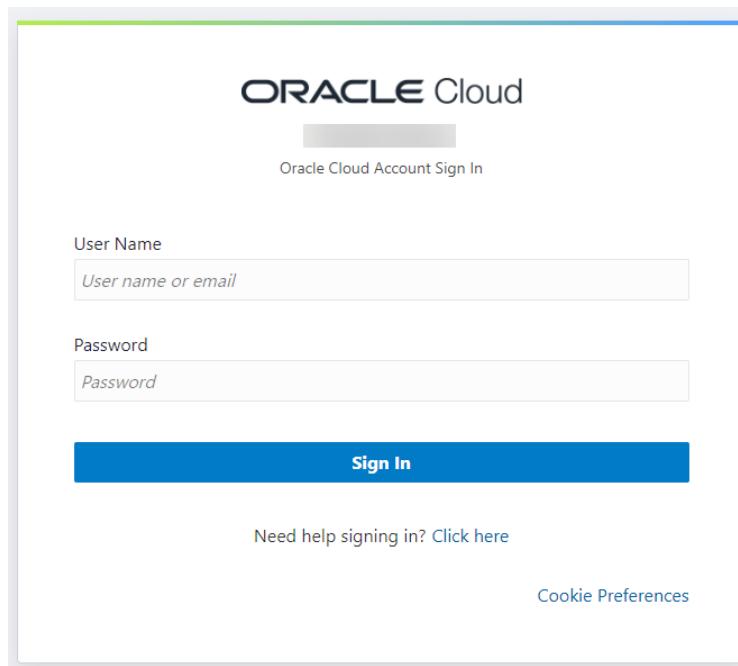


2. Click **Continue** to sign in using the "*oraclecloudidentityservice*".

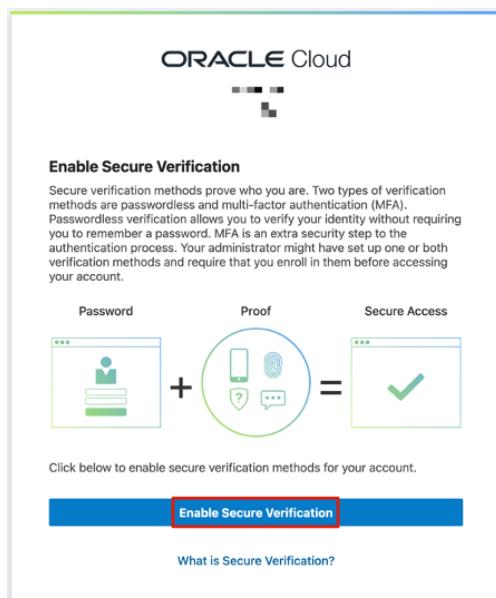


When you sign up for an Oracle Cloud account, a user is created for you in Oracle Identity Cloud Service with the username and password you selected. You can use this single sign-on option to sign in to Oracle Cloud Infrastructure and then navigate to other Oracle Cloud services without re-authenticating. This user has administrator privileges for all the Oracle Cloud services included with your account.

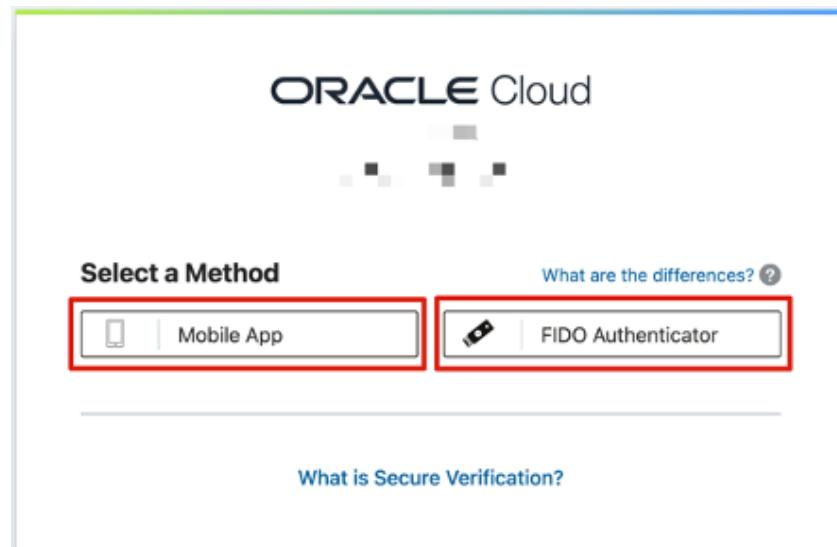
3. Enter your Cloud Account credentials and click **Sign In**. Your username is your email address. The password is what you chose when you signed up for an account.



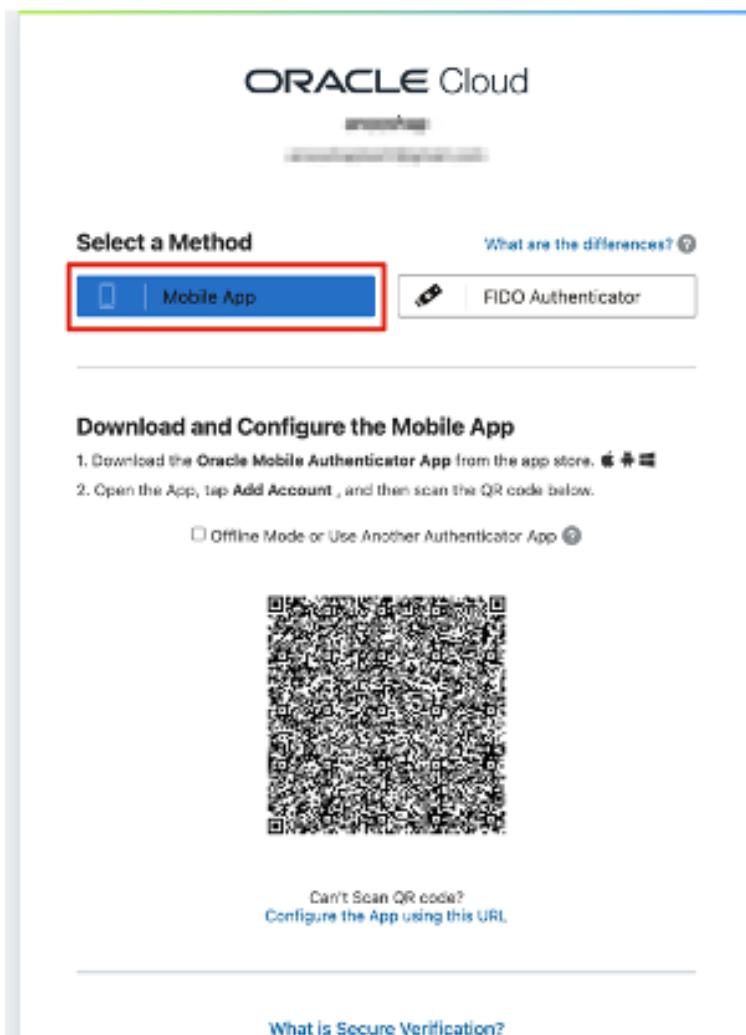
4. You will be prompted to enable secure verification. Click **Enable Secure Verification**. For more details, refer the [Managing Multifactor Authentication documentation](#)



5. Select a method - **Mobile App** or **FIDO Authenticator** to enable secure verification.



6. Choose **Mobile App** – Follow the steps as shown in the screenshot to setup authentication.



- Once you have verified authentication, you will now be signed in to Oracle Cloud!

The screenshot shows the Oracle Cloud Dashboard. At the top, there's a search bar and navigation links for 'Get started' and 'Dashboard'. Below the search bar, there are sections for 'Service links' (Pinned: Instances Compute, Virtual cloud networks Networking; Recently Visited: Autonomous Data Warehouse, Autonomous Database, Bastion Identity & Security, Data Safe Oracle Database, Tenancy Details Account Management, Stack Resource Manager; Recommended: Users Identity, Tenancies Organization Management, Buckets Object Storage & Archive Storage, Autonomous Database Autonomous Database, Policies Identity, Logging Logging), 'Cost savings opportunities' (Estimated savings: 0, View recommendations (10)), and 'OCI mobile app' (Review alarms, access billing and usage data, and manage resources on the go, Install now). There's also a 'Get early access to OCI features' section and a 'What's new' section. At the bottom, there are links for 'View my deployments', 'View all resources', 'Terms of Use and Privacy', and 'Cookie Preferences'.

- In case you received the following view go to Profile and Click OCI Console

The screenshot shows the 'My Apps' page. It has a blue header with the 'ORACLE' logo and the title 'My Apps'. Below the header, there are buttons for 'Favorites' and 'Add'. A search bar is on the right with a 'Search' button. A dropdown menu is open, showing options: Catalog (selected), My Profile, OCI Console, My Access Tokens, and Sign Out. The main content area says 'You don't have access to any apps.' with a small cloud icon above it.

LAB 1 - Deploying MuShop Locally Using Docker

Introduction

Task 1: Log into OCI Tenancy

Log in to your OCI dashboard and retrieve information required to create resources.

- Once you log in you will see a page similar to the one below. Click on "Infrastructure Dashboard."

The screenshot shows the Oracle Cloud Infrastructure (OCI) dashboard. In the top right corner, there is a green box indicating "All systems operational" and a link to "View health dashboard". Below this, the "Account Center" section includes links for "User Management" (Add a user to your tenancy), "Billing" (Analyze costs, Upgrade your account), and "What's New". The "What's New" section lists several updates: "Performance Hub time zone selector" (Jun 3, 2020), "Compliance Documents is now available" (Jun 2, 2020), "Experience zero downtime when you migrate apps using Application Migration" (Jun 2, 2020), "Performance Hub Blocking Sessions" (Jun 1, 2020), and "Online resizing for block volumes and".

Quick Actions

- COMPUTE: Create a VM instance (2-6 mins)
- AUTONOMOUS TRANSACTION PROCESSING: Create an ATP database (3-5 mins)
- AUTONOMOUS DATA WAREHOUSE: Create an ADW database (3-5 mins)
- NETWORKING: Set up a network with a wizard (2-3 mins)
- OBJECT STORAGE: Store data (2-6 mins)
- RESOURCE MANAGER: Create a stack (2-6 mins)

Start Exploring

- Get Started: Deploy Websites & Apps, Explore Developer Tools, Manage Rife
- Key Concepts and Terminology: DOCUMENTATION (To get started with Oracle Cloud Infrastructure, familiarize yourself with some key concepts and terminology.)
- Introduction to APEX: BLOG (Always Free Eligible) (Oracle Application Express (APEX) is a low-code development framework that enables you to rapidly build modern, data-driven apps right from your browser.)

Task 2: Create Virtual Cloud Network (VCN)

- Navigate to **Networking > Virtual Cloud Networks** > Click **Start VCN Wizard** > Select **VCN with Internet Connectivity** > Start VCN Wizard

The screenshot shows the "Start VCN Wizard" dialog box. It displays a diagram of the network architecture for a VCN with Internet Connectivity. The diagram shows a VCN (Virtual Cloud Network) containing a public subnet and a private subnet. The public subnet is connected to the Internet via an Internet Gateway (IG). The private subnet is connected to the Internet through a NAT gateway and a service gateway. The Oracle Services Network is also shown. A note at the bottom states: "Creates a VCN with a public subnet that can be reached from the internet. Also creates a private subnet that can connect to the internet through a NAT gateway, and also privately connect to the Oracle Services Network." A checkbox indicates that the "Includes: VCN, public subnet, private subnet, internet gateway (IG), NAT gateway (NAT), service gateway (SG)." option is selected.

2. Enter the following:

- o VCN Name : okeworkshop
- o Compartment : choose the root compartment

Basic information

VCN name [i](#)

OKE-Workshop

Compartment [i](#)

OKE-Workshop

oraseemeal (root)/mroter/OKE-Workshop

- o CIDR Block: 192.168.0.0/16
- o Public CIDR Block: 192.168.1.0/24
- o Private CIDR Block: 192.168.2.0/24

Configure VCN

VCN IPv4 CIDR block [i](#)

192.168.0.0/16

If you plan to peer this VCN with another VCN, the VCNs must not have overlapping CIDR blocks. [Learn more.](#)

IPv6 prefixes *Optional*

Enable IPv6 in this VCN

DNS resolution

Use DNS hostnames in this VCN

Required for instance hostname assignment if you plan to use VCN DNS or a third-party DNS. This choice cannot be changed after the VCN is created. [Learn more.](#)

Configure public subnet

IP address type

IPv4 CIDR block

192.168.1.0/24

Example: 172.16.0.0/16.

(Maximum number of items added)

Configure private subnet

IP address type

IPv4 CIDR block

192.168.2.0/24

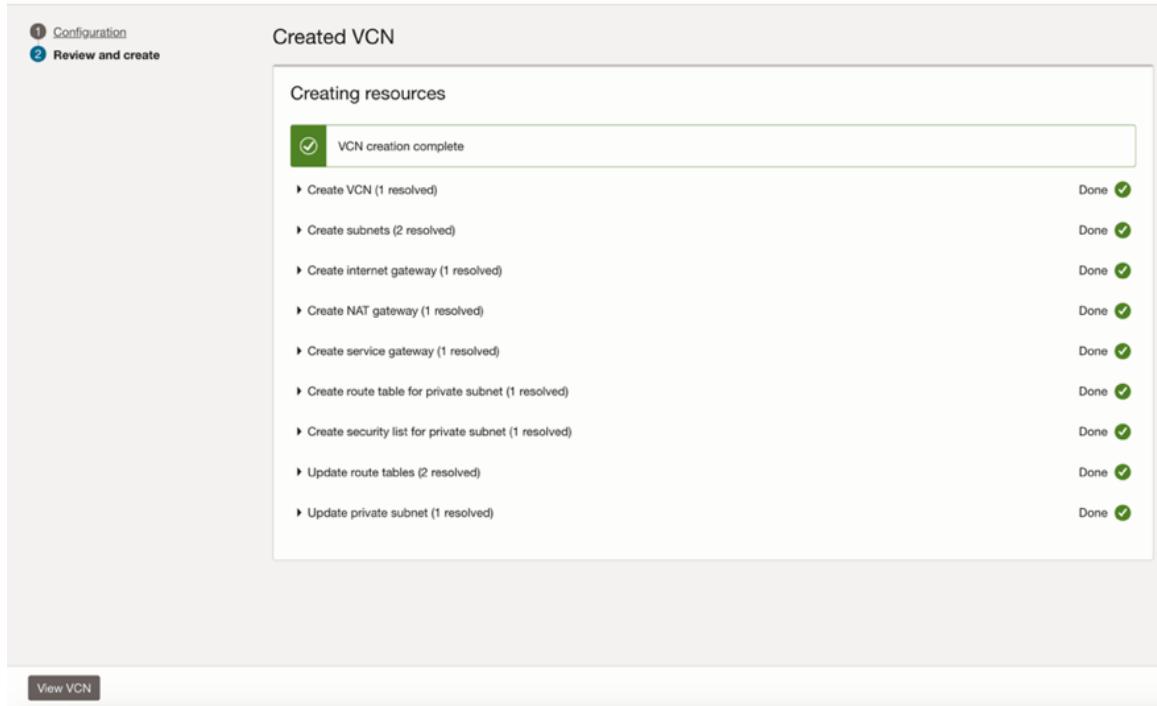
Example: 172.16.0.0/16.

- o Click **Next**
- o Validate all the information completed and Click **Create**

3. Click View VCN on the bottom left

Note: This process creates a Virtual Network with the following Components: VCN, Public subnet, Private subnet, Internet gateway (IG), NAT gateway (NAT), Service gateway (SG)

Create a VCN with internet connectivity



4. The following view shows the configured VCN in details

Note: if no information available go to List scope > Compartment > Validate the proper compartment is in place



Task 3: Configure Security List for Public Access

1. In the VCN view navigate to **Resources > Security Lists > Click Default Security list for < VCN Name>**

The screenshot shows the OCI Resources page with the 'Security Lists' section selected. On the left sidebar, under 'Security Lists (2)', the 'Default Security List for OKE-Workshop' is highlighted. The main panel displays a 'Create Security List' button and a 'Name' field containing 'security list for private subnet-OKE-Workshop'. Below it is another row with the name 'Default Security List for OKE-Workshop'.

2. Click **Ingress Rules > Add Ingress Rules**

The screenshot shows the OCI Resources page with the 'Ingress Rules' section selected. Under 'Ingress Rules (3)', there is an 'Add Ingress Rules' button and 'Edit' and 'Remove' buttons.

3. Enter the following :

- Source CIDR: 0.0.0.0/0
- Destination Port Range: All (default)

The screenshot shows the 'Ingress Rule 1' configuration dialog. It includes fields for 'Source Type' (CIDR), 'Source CIDR' (0.0.0.0/0), 'IP Protocol' (TCP), 'Source Port Range' (All), and 'Destination Port Range' (All). A note at the top states 'Allows TCP traffic for ports: all'.

- Click **Add Ingress Rules** and Validate rule is in the list
- Egress Rule : 0.0.0.0/0 (default)

Task 4: Import custom Image

1. Navigate to **Menu > Compute > Custom Images** in OCI Console.
2. Select **Import Image** and enter the following :
 - o Compartment: Select the appropriate compartment.
 - o Name: Assign a descriptive name. Avoid confidential information
 - o Operating System: Choose Ubuntu.
 - o Check the option “Import from an Object Storage URL”
 - o Object Storage URL: Paste the URL : <https://objectstorage.uk-london-1.oraclecloud.com/p/2o164n0KlkY--neEr78QVLUySUIELPxPB4cPhKTaApNEDzQzhWm86En4aFGaaR/n/oraseemail/b/bucket-oke-workshop/o/exported-image-20241117-1641>
 - o Image Type: VMDK (default)
 - o Launch Mode: Paravirtualized (default).
3. Click **Import Image** to complete.
 - o In case of “invalid source URL “ error message please check if you copy the URL without spaces in the beginning and in the end

The screenshot shows the 'Import image' dialog box. It has the following fields and settings:

- Create in compartment:** oraseemail (root)
- Name:** OKE Workshop-Student-VM
- Operating system:** Ubuntu
- Import type:** Import from an Object Storage URL (selected)
- Object Storage URL:** <https://objectstorage.eu-frankfurt-1.oraclecloud.com/p/Bgn0x1XGbmHcWo8AN7Gph9gFkVpyDTBjNV1rzsbqf0HZZ80WBSDuohPejs/n/troggl9fr/b/bucket-workshop/o/exported-image-20241117-1641>
- Image type:** VMDK (selected)
- Launch mode:** Paravirtualized mode (selected)
- Firmware:** BIOS
- Boot volume type:** PV
- NIC attachment type:** PV NIC
- Remote data volume:** PV

Now Wait until the image uploaded before moving to Task 5

Task 5: Create a Compute Instance

1. Navigate to **Compute > Instances > Create Instance.**
2. Enter the following to setup the Instance:

- o Instance Name and Compartment: Input relevant details.
- o Placement : Frankfurt-AD1 (default)
- o Security: Default
- o Image Source: Select **Image and shape > Change image**

The screenshot shows the 'Image and shape' configuration screen. At the top, there's a heading 'Image and shape' and a 'Collapse' link. Below it, a note states: 'A **shape** is a template that determines the number of CPUs, amount of memory, and other resources allocated to an instance. The image is the operating system that runs on top of the shape.' A preview section shows 'Oracle Linux 8' with an 'Image build: 2024.09.30-0'. To the right of the preview is a red arrow pointing to a 'Change image' button.

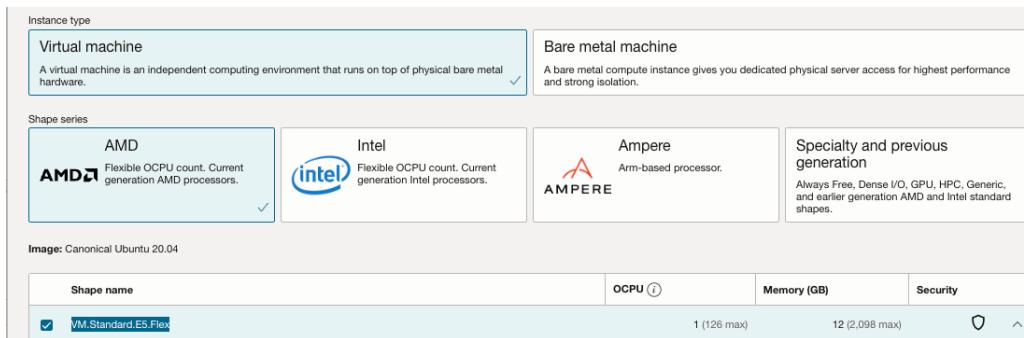
- o Select **My images > Custom images > Uploaded in Task 1 > click select image**

The screenshot shows the 'My images' page. It displays a grid of images: Oracle Linux, Ubuntu, Red Hat, CentOS, Windows, AlmaLinux, Rocky Linux, and Marketplace. Below the grid, a section for 'Custom images' is shown, with a sub-section for 'My images' which has a checked checkbox. At the bottom, there are filters for 'Custom images', 'Boot volumes', and 'Image OCID', and a note about 'Custom images' including software, configuration, and customizations. A compartment dropdown is set to 'OKE-Workshop'. The table at the bottom lists a single item: 'OKE Workshop-Student-VM' with a checked checkbox, security status, and creation date 'Mon, Nov 11, 2024, 09:56:03 UTC'.

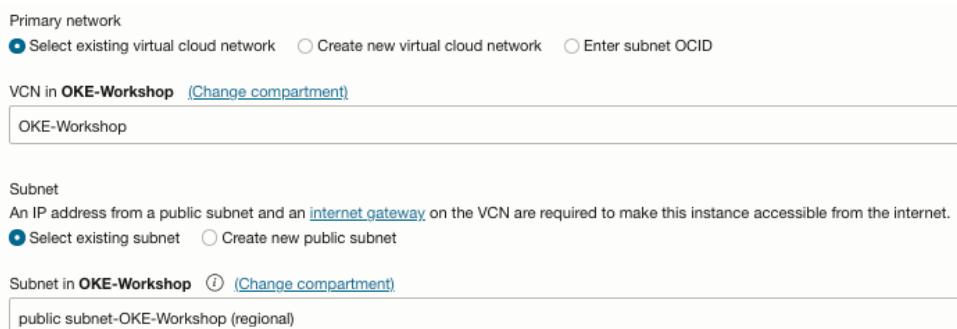
- o **VM Shape :** Select **Image and shape > Change shape**

The screenshot shows the 'Shape' configuration screen. It displays the 'VM.Standard.E5.Flex' shape, which is described as a 'Virtual machine, 1 core OCPU, 12 GB memory, 1 Gbps network bandwidth'. To the right of the description is a 'Change shape' button.

- **Select the following Parameters:** **Instance type:** Virtual machine , **Shape series:** AMD and **Shape name:** VM.Standard.E5.Flex



- **Network Settings** (Primary VNIC information):
 - **Primary network** - Select existing virtual cloud network > Choose the VCN created in Task 2.
 - **Subnet** - Select existing subnet > Choose the Public Subnet created in Task 2.



- **Note :** Validate the Compartment name if the VCN/Subnet is not in the drop-down list
- **Primary VNIC IP addresses:**
 - **Private Ipv4 address** > Checkbox **Automatically assign private Ipv4 address** (default)
 - **Public Ipv4 address** > Checkbox **Automatically assign public Ipv4 address** (default).

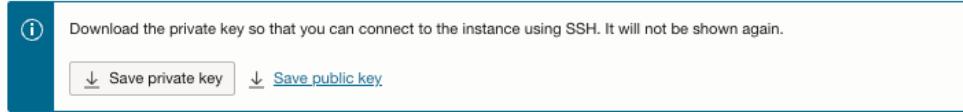


- **Add SSH Keys:**
 - Choose **Generate a key pair for me** > Save both **private and public keys**

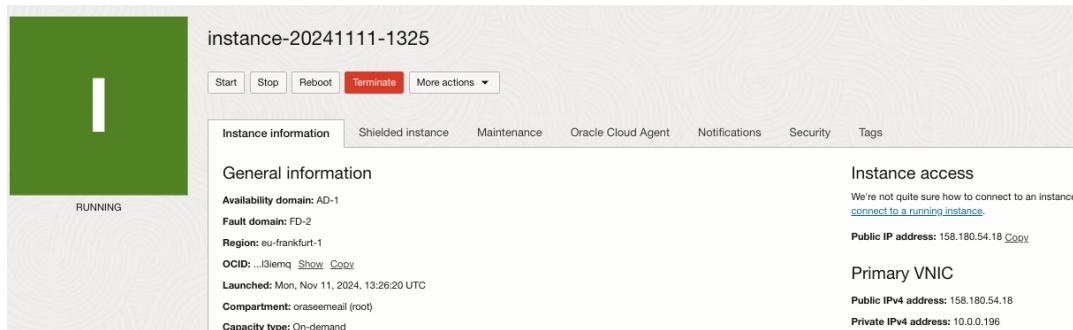
Add SSH keys

Generate an [SSH key pair](#) to connect to the instance using a Secure Shell (SSH) connection, or upload a public key that you already have.

Generate a key pair for me Upload public key files (.pub) Paste public keys No SSH keys



- **Boot Volume:** Unchecked values (default)
- **Block Volume:** No volumes (default)
- **Live migration:** Enabled (default)
- Click **Create** and validate instance is in Running state
- **Make a note of public ip address**



Task 6: Connect the VM via SSH

1. From terminal (On local computer – MacOS, Windows, Linux) enter the following command to change permission for the downloaded Private Key before SSH to the VM

```
chmod 700 ssh_private_key.key
```

```
% chmod 700 ssh-key-2024-11-17.key
%
```

2. From the terminal SSH to the VM by executing the following command:

```
ssh -i <path/to/private key/ssh_private_keyname> ubuntu@<PUBLIC_IP_OF_COMPUTE>
```

Note : Replace the <PUBLIC_IP_OF_COMPUTE> with the instance assigned public IP

Task 7: Configure OCI CLI on VM

1. Execute the following command on the VM for OCI CLI setup :

```
oci setup config
```

```
PS D:\nadeem> oci setup config
```

This command provides a walkthrough of creating a valid CLI config file.

The following links explain where to find the information required by this script:

User OCID and Tenancy OCID:

<https://docs.us-phoenix-1.oraclecloud.com/Content/API/Concepts/apisigningkey.htm#Other>

Region:

<https://docs.us-phoenix-1.oraclecloud.com/Content/General/Concepts/regions.htm>

General config documentation:

<https://docs.us-phoenix-1.oraclecloud.com/Content/API/Concepts/sdkconfig.htm>

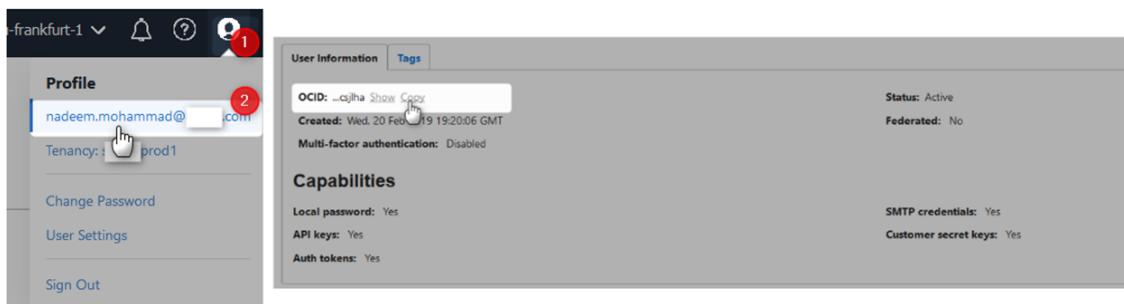
2. Choose default location > Click Enter

```
Enter a location for your config [C:\Users\nadeem\.oci\config]:
```

3. Enter User OCID when prompted:

```
ENTER A USER OCID: ocid1.us-phoenix-1.ocid1.user.oc1..xxxxxxxxxxxxxxpl6esjk0mwa7sdbae25sejrlallllllll
```

4. Obtain the User OCID as follow :



5. Enter **Tenancy OCID** when prompted:

```
Enter a tenancy OCID: ocid1.tenancy.oc1..xxxxxxxxxxxxxx4bp2xinnnndy25ps6csc7yyyyyyyyyyyy
```

6. Obtain the **Tenancy OCID** as follow :



7. Enter **Region** when prompted: 27 (eu-frankfurt-1)

8. **Generate API Keys** as prompted:

Do you want to generate a new RSA key pair? (If you decline you will be asked to supply the path to an existing key.) [Y/n]: Y

Enter a directory for your keys to be created [\Users\nadeem\.oci]:

Enter a name for your key [oci_api_key]:

Public key written to: \.oci\oci_api_key_public.pem

Enter a passphrase for your private key (empty for no passphrase): N/A

Private key written to: \Users\nadeem\.oci\oci_api_key.pem

Fingerprint: b2:04:c3:ee:22:d0:85:83:b6:fa:24:9e:93:2f:c5:27

Config written to \Users\nadeem\.oci\config

9. Copy public key

- Execute the following command:

```
ubuntu@instance-vm-oke:~$ cd .oci/  
ubuntu@instance-vm-oke:~/oci$ ls  
config oci_api_key.pem  oci_api_key_public.pem  
ubuntu@instance-vm-oke:~/oci$ vi oci_api_key_public.pem
```

```
ubuntu@instance-vm-oke:~$ cd .oci/  
ubuntu@instance-vm-oke:~/oci$ ls  
config  oci_api_key.pem  oci_api_key_public.pem  
ubuntu@instance-vm-oke:~/oci$ vi oci_api_key_public.pem
```

- Copy the file

```
-----BEGIN PUBLIC KEY-----  
MIIBIjANBqkqhkiG9w0BAQEFAAOCAQ8AMIIIBCgKCAQEAj2g0vo3mN9QRtfE11FQU  
WT4yACMoB0DbSmqFyvZKJJY06NZZ2B8DEj7p0H3tJ1C9juKiFDZyU/JI1pPP2XAj  
YGd+A79+tDGsITODC32xu2Kkw1Xb/KT2oVaWXks2bzz4ZQnjYl7+/h8mnHyZqQ1M  
GzGErNwjkFafCf9ezoT/0C8ZEZ/vWDe4vqaVI9r/aDnZc93ifoekK7TDJjv1BfA  
0wNQ2fgQl4T1HH4mq2pZ6hLNECSxWKS/makp/i0jZ15y0LmMqmik+9AFdq97V2Jt  
jw2BwBtYcb3myKZE6ZAX02i0JSv6w/2t5MSPPo4oFhVmW3/cp08MQW6CycZ4nari  
HQIDAQAB  
-----END PUBLIC KEY-----
```

- Exit the file

Press ESC and type :q

```
~  
:q
```

10. Upload Public key

- In the OCI Console **Navigate to User Setting > Click API keys > Add API key**

The screenshot shows two side-by-side pages from the OCI console. On the left, the 'User settings' page is displayed, showing profile information like 'Identity domain' and 'Tenancy'. On the right, the 'API keys' page is shown, featuring a 'Resources' sidebar and a main area with tabs for 'My groups', 'Integrated applications', and 'API keys'. A blue box highlights the 'API keys' tab.

- Choose **Paste a public key > Paste the public key** copied earlier

The screenshot shows the 'Add API key' dialog box. It has a note about what an API key is and three options: 'Generate API key pair', 'Choose public key file', and 'Paste a public key'. The third option is selected. Below the input field, there is a large text area containing a long RSA public key in PEM format, with a red circle highlighting the 'Add' button at the bottom.

- Validate** public key added

The screenshot shows the 'API keys' list page. It displays a table with columns for 'Fingerprint' and 'Created'. One row is visible, showing a redacted fingerprint and the creation date 'Thu, May 30, 2024, 07:05:02 UTC'.

- Validate** the OCI config file with the following command:

```
oci os ns get
```

```
PS C:\WINDOWS\system32> oci os ns get
{
  "data": "prod1"
}
```

Note : Wait few minutes for the Key to be updated before moving to Task 8

Task 8: Set Up Policy to manage OCI Resources

1. Navigate to **Identity & Security > Domains > Click Default > Groups > Click Create Group and Add User to the Group**

The screenshot shows the OCI console navigation bar on the left with various services like Home, Compute, Storage, Networking, Oracle Database, Databases, Analytics & AI, Developer Services, and Identity & Security. The Identity & Security service is selected. The main content area has a sidebar titled 'Identity & Security' with options: Overview, Domains (selected), Network Domains, Policies, Compartments, Federation, Access Governance, Cloud Guard, and Overview. The main panel is titled 'Identity' with sections: Overview, Domains (selected), Network Sources, Policies, and Compartments. A right-hand sidebar titled 'Domains in oras' shows a 'Create domain' button, a 'Name' field with placeholder 'Domain Name', and a 'Default' link.

2. Create Group

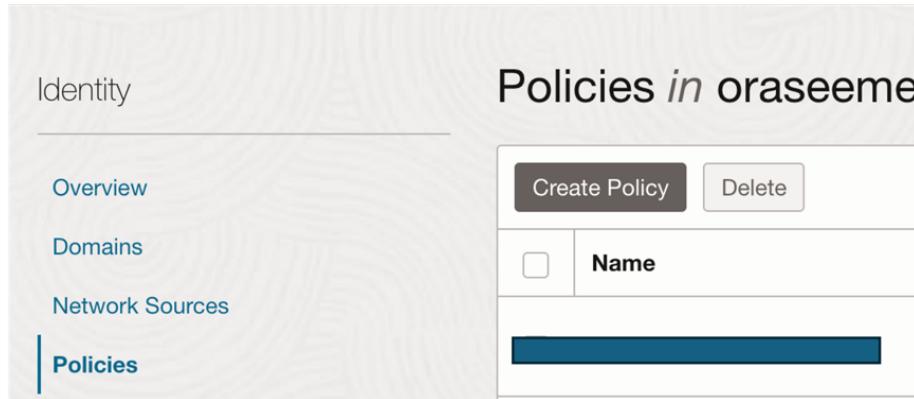
The screenshot shows the 'Groups in Default Domain' page. The left sidebar has 'Identity domain' and 'Groups' selected. The main area has a search bar 'Search by group name or description.' and a 'Create group' button. To the right, a 'Create group' form is open with fields: 'Name' (containing 'OKE-WORKSHOP'), 'Description' (empty), and a checkbox 'User can request access' (unchecked). The 'Name' field has a blue circular icon with a white arrow pointing up.

- Add the User to the Group via the Create group tab: Search **Username** > checkbox **the username**

The screenshot shows the 'Users' section of the 'Create group' form. It includes a checkbox 'User can request access' (unchecked). Below it is a 'Users' section with the note 'Optional' and 'Select users to assign this group.' A search bar 'Search by user name, first name, last name, or email address' is present. Below the search bar is a table with columns: First name, Last name, and Email. The 'First name' column contains a checkbox and a placeholder 'First name'. The 'Last name' column contains a checkbox and a placeholder 'Last name'. The 'Email' column contains a checkbox and a placeholder 'Email'.

3. Create Policy

- Navigate to **Identity & Security > Domains > Policies > Create Policy**



- **Add Policy :**

Allow group <GroupName> to manage all-resources in tenancy <Tenancy Name>

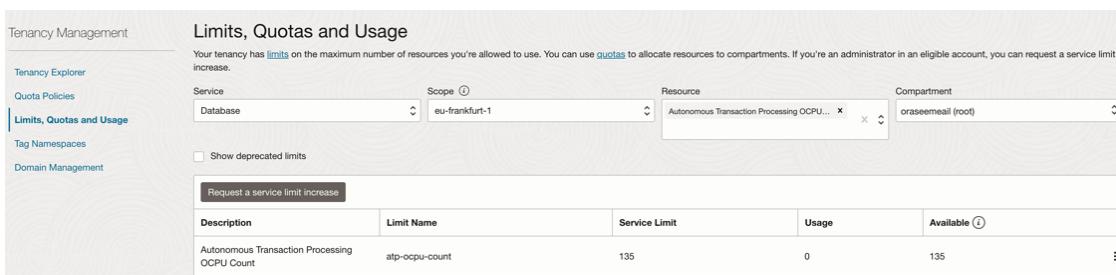
Change the view to manual builder

Notes: Replace <GroupName> with the newly created group and <CompartmentName> with the root compartment name for the purpose of this lab.

- This policy is broad for simplicity; consider refining it for real-world usage.

Task 9: View the Tenant Service Limits

1. Ensure OCI account has adequate limits, particularly for Autonomous Database configurations.
2. Navigate to **Governance & Administration > Limits, Quotas and Usage > Service Database > Scope eu-frankfurt-1 > Resource Autonomous Transaction Processing OCPU > Count > 1**



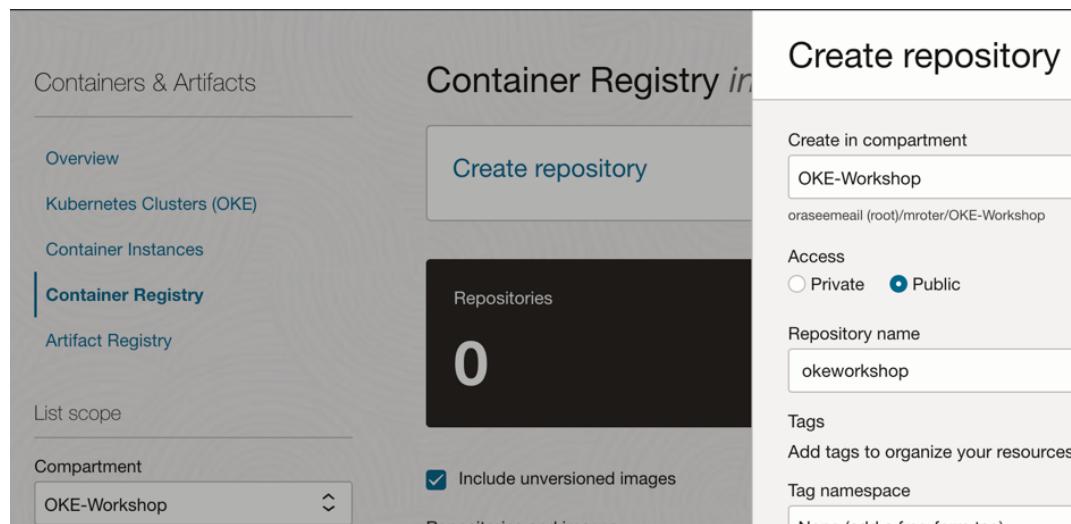
Task 10: Running Containers Locally with Docker

Introduction

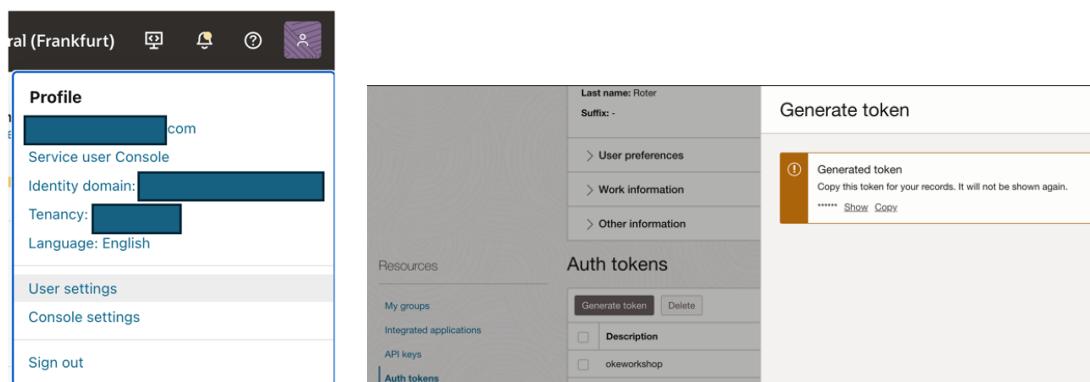
This Task demonstrate how to use, build the code and microservices on the created VM and push them to OCI Container Registry and run them using Docker Compose.

1. Create Container Registry:

- o Navigate to **Developer Services > Container Registry > Click Create Repository**
 > Select **root compartment** > **Access Type Public** > **Specify Repository name**
- o Click **Create**



2. Generate Auth Token: Click Profile on the top-right corner > User Settings > Auth Tokens, generate a token and copy it for future use.



3. Staging Docker Images Locally:

- o Login to the Repository by Using the following command

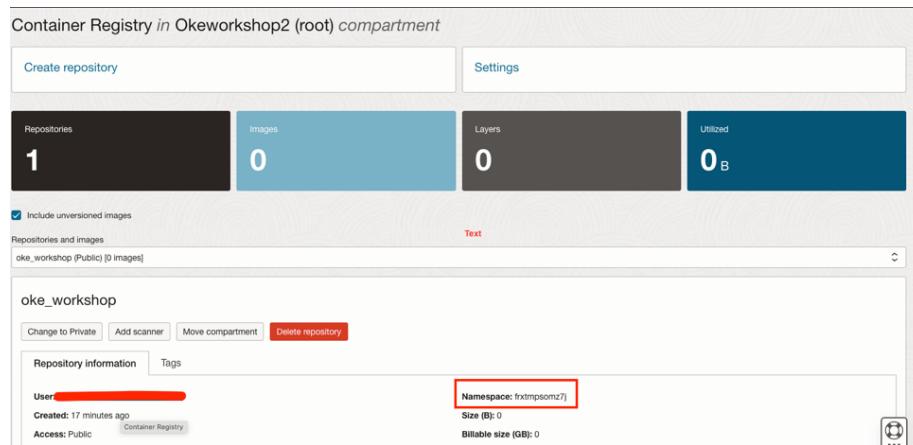
```
docker login <region_code>.ocir.io
```

Example for Frankfurt region - docker login fra.ocir.io

- User Name

<Registry-namespace>/default/<username>

Example - froqjg8h9ftr/default/user@domain.com



- Password: Provide the auth token when prompted

Task 11: Build and Deploy Using Docker

1. Create folder “oci_workshop”

```
mkdir oci_workshop  
cd oci_workshop
```

2. Clone Git_Hub Folder with the following command:

```
git clone https://github.com/oci-oke-workshop-il/oci-cloudnative-ext.git
```

3. Go to SRC folder and validate the following microservices exists:

```
ke:~/oci-cloudnative-internal/src$ ls -l
ubuntu 5927 Nov  3 12:01 README.md
ubuntu 4096 Nov  3 12:01 api
ubuntu 4096 Nov  3 12:01 assets
ubuntu 4096 Nov  3 12:01 carts
ubuntu 4096 Nov  3 12:01 catalogue
ubuntu 4096 Nov  3 12:01 dbtools
ubuntu 4096 Nov  3 12:01 docs
ubuntu 4096 Nov  3 12:01 edge-router
ubuntu 4096 Nov  3 12:01 events
ubuntu 4096 Nov  3 12:01 fulfillment
ubuntu 4096 Nov  3 12:01 functions
ubuntu 4096 Nov  3 12:01 load
ubuntu 4096 Nov  3 12:01 orders
ubuntu 4096 Nov  3 12:01 payment
ubuntu 4096 Nov  3 12:01 storefront
ubuntu 4096 Nov  3 12:01 user
```

4. Build Docker images as follow :

- **Build docker for “api” microservice using:**

```
docker build -t give_name_of_docker_image:version .
```

Example: docker build -t mushop_api:v1 .

- **Tag the newly created images as follow:**

```
docker tag give_name_of_docker_image:version
```

```
oci_region/registry_namespace/registry_name/name_of_docker_image:version
```

Example:

```
docker tag mushop_api:v1
fra.ocir.io/froqjg8h9ftr/oke_workshop/mushop_assets:v1
```

- **Push Docker images to the OCI container registry as follow:**

```
docker push oci_region/registry-
namespace/registry_name/name_of_docker_image:version
```

Example:

```
docker push fra.ocir.io/froqjg8h9ftr/oke_workshop/mushop_api:v1
```

5. Repeat tasks 4-6 for each image to ensure all microservices are uploaded

6. Validate all images are built by executing the following command:

```
docker images -a
```

LAB 2 – Deploy OKE Cluster with Terraform

Introduction

You will take on the persona of an Operations Engineer. You will initiate the Oracle cloud environment that will be used to create and deploy your microservices applications. This environment will be contained within a cloud Compartment, and communication within the Compartment will be via a Virtual Cloud Network (VCN). The Compartment and VCN will isolate and secure the overall environment. You will deploy the Oracle Cloud Infrastructure Container Engine for Kubernetes(OKE).

Task 1: Go to Terraform Directory

1. Change directory to the '**Terraform-templates**' folder within the cloned repository:

```
cd oci-cloudnative-ext/Terraform-templates
```

Explanation: The 'cd' command navigates to the 'Terraform-templates' directory where you will configure the deployment templates.

Task 2: Edit '**terraform.tfvars**'

1. Open '**terraform.tfvars**' using a text editor:

```
vi terraform.tfvars
```

Explanation: This file contains the variables required for your Terraform deployment. You need to modify it to specify details about your OCI environment.

2. Update **the following variables:**

compartment_id: This is the OCID of the compartment where you want to deploy your cluster.

availability_domain: Specify the availability domain (e.g., 'AD-1') where OKE cluster will be deployed

tenancy_ocid : This is the OCID of your tenancy.

Example of edited 'terraform.tfvars**:**

```
compartment_id = ocid1.compartment.oc1..exampleuniqueID
availability_domain = XXXX:EU-FRANKFURT-1-AD-1
tenancy_ocid = ocid1.tenancy.oc1..exampleuniqueID
```

Explanation: Replacing these variables with the correct OCIDs ensures that Terraform knows where to deploy your OKE cluster within your OCI environment.

3. Obtain the `compartment_id` (OCID)

- Navigate to **Identity & Security > compartments**

The screenshot shows a table with two columns: 'Name' and 'Status'. A context menu is open over the first row, showing the OCID: 'ocid1.tenancy.oc1..aaaaaaaaaaaaq5n3yixynnkwpwlkvehz4rd4d6qogt2q5vcdj3kkkek46pyc5tuq'. The menu also includes 'Copy' and 'Close' options.

Name	Status
okeworkshop1	● Active

4. Obtain the `availability_domain`

- Navigate to **Compute > Instances > Create Instance > Locate the available AD's**

Note: locate and use the string for the AD in the following format - **XXXX:EU-FRANKFURT-1-AD-1**

The screenshot shows a list of availability domains: AD 1 (selected), AD 2, and AD 3. The selected option is highlighted with a blue border and shows the full ID: 'IPIC:EU-FRANKFURT-1-AD-1'. There is a link 'Show advanced options' below the list.

- For this LAB, we are using AD1

5. Obtain the `tenancy_ocid`

- Navigate to **Profile > Click on Tenancy name > Locate tenancy OCID**

The screenshot shows the 'Tenancy details' section for 'okeworkshop1'. It displays the tenancy name, a green profile picture, and various settings like 'Edit object storage settings', 'Add tags', 'Rename tenancy', and 'Request tenancy deletion'. The 'Tenancy information' tab is selected, showing the OCID: 'ocid1.tenancy.oc1..aaaaaaaaaaaaq5n3yixynnkwpwlkvehz4rd4d6qogt2q5vcdj3kkkek46pyc5tuq'. To the right, there are sections for 'Object storage settings' and 'Amazon S3 compatibility API designated compartment: okeworkshop1'. On the right sidebar, there are links for 'Profile', 'My profile', 'Tenancy: okeworkshop1', 'Service user Console', 'Console settings', and 'Sign out'.

Task 4: Apply Terraform Configuration to deploy OKE Cluster

1. Initialize the directory:

```
terraform init
```

Explanation: This command initializes your Terraform configuration by downloading the required plugins and preparing the environment.

2. Execute the Terraform plan command to preview changes:

```
terraform plan
```

Explanation: This Task allows you to check for potential issues and confirm that the deployment configuration is correct.

3. Deploy the OKE cluster

```
terraform apply
```

Explanation: This command starts the deployment of the OKE cluster. Review the changes when prompted and type 'yes' to confirm.

Task 5: Validate Successful Deployment

1. Navigate to Developer Services > Kubernetes Clusters (OKE) > Click Cluster name > Review **cluster details** and

The screenshot shows the OCI Console interface. On the left, there is a sidebar with a search bar at the top and a list of services: Home, Compute, Storage, Networking, Oracle Database, Databases, Analytics & AI, and Developer Services. The 'Developer Services' option is highlighted with a blue border. The main content area has a header 'Developer Services' with a back arrow icon. Below the header, there are two sections: 'Containers & Artifacts' and 'Functions'. Under 'Containers & Artifacts', there is a 'Kubernetes Clusters (OKE)' section with links for 'Overview', 'Container Instances', 'Container Registry', and 'Artifact Registry'. Under 'Functions', there are links for 'Applications' and 'Pre-Built Functions'. To the right of the main content area, there is a separate window titled 'Create cluster' with a table. The table has columns for 'Name', 'Status', and 'Node pools'. There is one row with the values 'oke-workshop', 'Active', and '1' respectively.

Name	Status	Node pools <i>(i)</i>
oke-workshop	Active	1

Explanation: Verifying in the OCI Console ensures that the cluster has been deployed as expected.

LAB 3 - OCI Vault integration with OKE

Introduction

This lab provides the relevant configuration Tasks for Vault service integration with OCI services (e.g. Oracle Autonomous database)

Task 1: Create Dynamic Group to group the OKE cluster nodes to automatically assign policies to the group

1. Navigate to **Identity & Security > Domains > Click Domain Name > Click Dynamic groups**

The screenshot shows the OCI Identity & Security interface. The top navigation bar has 'Identity' and 'Domains' selected. Below it, 'Default domain' is shown. On the left, there's a sidebar with 'Identity domain', 'Overview' (which is active), 'Users', 'Groups', and 'Dynamic groups' (which is also active). In the main content area, there are tabs for 'Domain information' and 'Tags'. At the bottom, the OCID of the domain is displayed as 'OCID: ...4p6fqq' with 'Show' and 'Copy' buttons.

2. Click **Create dynamic group** and fill the following information
 - Name : OKE-DG
 - Description : Include OKE nodes in the same group
 - **Matching rules > Rule 1 > add the following rule :**

```
ALL {instance.compartment.id = 'insert compartment ocid'}
```

Example

The screenshot shows the 'Matching rules' section. It has a button 'Edit all matching rules'. Below it, a note says 'Instances that meet the following criteria will be included in the dynamic group: criteria defined by any of these rules'. A single rule is listed: 'ALL {instance.compartment.id = 'ocid1.tenancy.oc1..aaaaaaaaa5naxl5kdhgxyhn2jrk7fcz4spdez4umysyxnq626kf2qlcvuhx'a}'.

- Click **Create**

Task 2: Create Policy to allow the OKE cluster access the Vault resource

1. Navigate to **Identity & Security > Policies > Create Policy**

The screenshot shows the OCI Identity & Security interface. The left sidebar has a 'Policies' section selected. The main area is titled 'Policies in oraseeme' and contains a 'Create Policy' button and a table with a single row labeled 'Name'. A large red rectangle highlights the 'Name' column.

2. Add Policies:

Allow dynamic-group <Dynamic-group-name> to read secret-bundles in compartment id <compartment-ocid>

Allow dynamic-group <Dynamic-group-name> to use keys in compartment id <compartment-ocid>

Allow dynamic-group <Dynamic-group-name> to manage secret-family in compartment id <compartment-ocid>

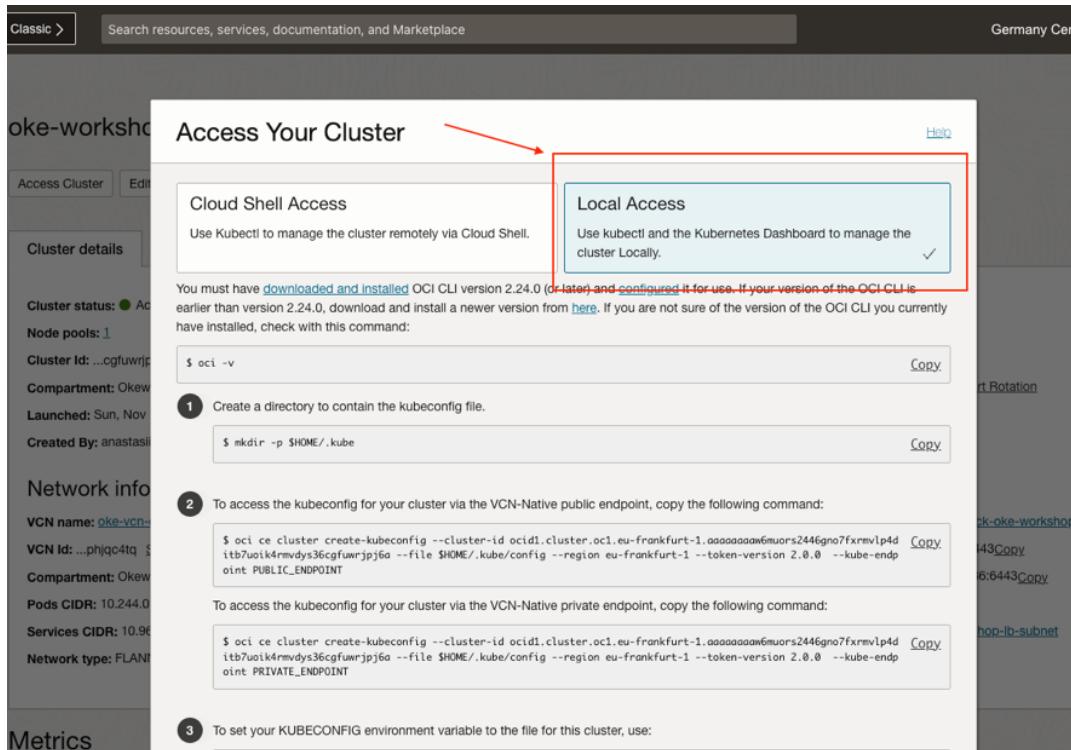
Notes:

Replace <Dynamic-group-name> with the newly created dynamic group and <compartment-ocid> with the root compartment ocid for the purpose of this lab.

3. Click Create

Task 3: Install External Secrets Operator on OKE

1. In OCI console Navigate to the **Developer Services > Kubernetes clusters (OKE)**.
2. Choose the **OKE cluster** you have previously created with terraform.
3. Follow the instruction for **Quick Access > Local Access** > Do the Action from the VM



4. Execute the following commands in the VM Terminal:

```
helm repo add external-secrets https://charts.external-secrets.io
helm install external-secrets \
    external-secrets/external-secrets \
    -n external-secrets \
    --create-namespace
```

Task 4: Configuring ESO for use with OCI Vault

1. Create a oci-secret-store.yaml:

- o Open editor in the VM terminal using VI command and create the following file:

```
apiVersion: external-secrets.io/v1beta1
kind: SecretStore
metadata:
  name: workshop-vault
spec:
  provider:
    oracle:
      vault: <vault-OCID> #Replace <vault-OCID> with the Vault ocid created by the terraform
      region: eu-frankfurt-1
```

- o Execute the following command to deploy the yaml in the oke cluster

```
kubectl create -f oci-secret-store.yaml
```

- Validate that SecretStore resource created in the cluster:

```
kubectl get secretstore -A
```

Task 5: Create secrets in OCI Vault

1. Navigate to Identity & Security > Vault

The screenshot shows the Oracle Cloud Infrastructure (OCI) Identity & Security dashboard. On the left, there's a sidebar with various service links like Home, Compute, Storage, Networking, Oracle Database, Databases, Analytics & AI, Developer Services, and Identity & Security. The Identity & Security link is highlighted with a dashed box. The main content area has several sections: Identity, Threat Intelligence, Key Management & Secret Management, Access Governance, Cloud Guard, Web Application Firewall, and Certificates. The Key Management & Secret Management section is expanded, showing sub-options like Vault, Dedicated Key Management, External Key Management, and Private Endpoints. The Vault option is also highlighted with a blue box.

2. Validate compartment name (root) and Click on the Vault instance created by terraform

The screenshot shows the details of a Vault instance within the 'okeworkshop1 (root) Compartment'. The left sidebar shows 'Key Management & Secret Management' and 'Vault' is selected. The main panel title is 'Vaults in okeworkshop1 (root) Compartment'. It says 'Vaults let you centrally manage the encryption keys that protect your data and the secret credentials that you use'. Below this is a table with two columns: 'Name' and 'State'. A single row is shown with the name 'MyOKEVault' and the state 'Active' indicated by a green dot.

3. Click Secrets

4. Create the following secrets : oadb-connection, oadb-wallet & oadb-admin

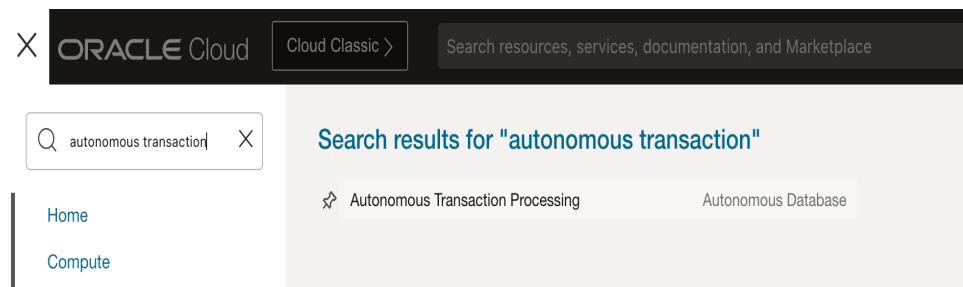
- Create oadb-admin secret as follow:
 - Name: oadb-admin
 - Compartment: Use the root for the purpose of this lab
 - Encryption Key: Choose from drop-down list
 - Secrets contents: Challenge yourself here <place-your-adb-admin-password-from-terraform-code>

Create Secret

[Help](#)

Name	oadb-admin
Description	
Encryption Key in okeworkshop1 (root) (Change compartment)	
MyVaultKey 	
<div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <p>Automatic secret generation</p> <p>Use auto secret generation to automatically generate the secret content.</p> </div> <div style="width: 45%;"> <p>Manual secret generation</p> <p>Use manual secret generation to manually provide the secret content. </p> </div> </div>	
<p>Secret Type Template</p> <p>Plain-Text </p> <p>Secret Contents</p> <p><place-your-adb-admin-password-from-terraform-code></p>	

- Create **oadb-connection** secret as follow:
 - Name: oadb-connection
 - Compartment: Use the root for the purpose of this lab
 - Encryption Key: Same as previous secret
 - Secret Type Template: < plain-text >
 - Secrets contents: Replace this value with the value of the adb-tnsname (**refer to next section Obtain the adb-tnsname to identify the value**)
 - Create secret: **Click it when you finished to fill the information above)**
- Obtain the **adb-tnsname**
 - Duplicate the OCI portal window > Navigate to the **Search bar** and search for : Autonomous transaction



The screenshot shows the Oracle Cloud search interface. At the top, there's a navigation bar with the Oracle Cloud logo, a "Cloud Classic" dropdown, and a search bar containing the text "autonomous transaction". Below the search bar, the results are displayed under the heading "Search results for 'autonomous transaction'". The results show two items: "Autonomous Transaction Processing" and "Autonomous Database". On the left side, there's a sidebar with links for "Home" and "Compute".

- Choose the Database created by the terraform

Display name	State	Compute	Storage	Workload type
My ADB AI	● Stopped	4 ECPU	1 TB	Transaction Processing

- Click on DB connection

- In the Database connection view Scroll down to **Connection Strings** > Copy TNS name_high (Do not copy the connection string)

TNS name (i)	Connection string (i)
myadb23ai_high	...ecurity=(ssl_server_dn_match=yes)) Show Copy
myadb23ai_low	...ecurity=(ssl_server_dn_match=yes)) Show Copy
myadb23ai_medium	...ecurity=(ssl_server_dn_match=yes)) Show Copy
myadb23ai_tp	...ecurity=(ssl_server_dn_match=yes)) Show Copy
myadb23ai_tpurgent	...ecurity=(ssl_server_dn_match=yes)) Show Copy

Do not close this window yet!

- Go to previous **Section** and use the copied TNS name to fill the **Secret Type Template** and Finish the Tasks to create the **oadb-connection**

Example

```
oadb_wallet_pw='your-wallet-password'
oadb_service='myadb23ai_high' #Your TNSName
```

The screenshot shows the Oracle Key Management & Secret Management interface. A large green hexagon icon with a white 'S' is visible. The secret name 'oadb-connection' is displayed above the modal window. The modal window is titled 'View Secret Contents' and contains the following information:

Secret Information	Value
OCID:	...jrqsna
Created:	Sun, Nov 12, 2023
Compartment:	...o

Below the table, there are two environment variables:

- `oadb_wallet_pw='ComplexPassword123!'`
- `oadb_service='myadb23ai_high'`

At the bottom of the modal, there is a 'Close' button.

- Download **Client Credentials (Wallet)**
- Go back to **DB connection view page**
- Click **Download wallet**

The screenshot shows the 'Database connection' page. The title is 'Database connection'. Below it, there is a section titled 'Download client credentials (Wallet)'. The text explains that to download your client credentials, you need to select the wallet type and click 'Download wallet'. It also notes that this client credential download only contains information for mTLS connections and that you do not need a wallet for TLS connections.

The 'Wallet type' dropdown is set to 'Instance wallet'. There are two buttons at the bottom: 'Download wallet' and 'Rotate wallet'. Below the buttons, a note says 'Wallet last rotated: -'.

- Set Password and click **Download**

Download wallet

[Help](#)

! Connections to your Autonomous Database are secured, and can be authorized using TLS or mTLS authentication options. TLS authentication is easier to use, provides better connection latency, and does not require you to download client credentials (wallet) if any of these is true for your connections:

- You are using JDBC Thin Client (version 12.2.0.1 or higher) with JDK 8(u163+) or higher.
- You are using the Python python-oracledb driver.
- You are using ODP.NET version 19.14 (or higher), or 21.5 (or higher).
- You are using an Oracle Call Interface based driver with Oracle Client libraries version 19.14 (or higher), or 21.5 (or higher).

[Learn more](#) about TLS authentication and how to enable it.

Please create a password for this wallet. Some database clients will require that you provide both the wallet and password to connect to your database (other clients will auto-login using the wallet without a password).

Password

Confirm password

Download [Cancel](#)

- **Upload the file to the virtual machine** In the OCI console Navigate to **Storage > Buckets** > click **Create Bucket**
 - Name: my-bucket
 - Default Storage Tier : Standard
 - Keep the default values for any additional values
 - Click **Create**

Create Bucket

bucket-20241117-1254

Default Storage Tier

- Standard
- Archive

The default storage tier for a bucket can only be specified during creation. Once set, you cannot change the storage tier in which a bucket resides. [Learn more about storage tiers](#)

Enable Auto-Tiering
Automatically move infrequently accessed objects from the Standard tier to less expensive storage. [Learn more](#)

Enable Object Versioning
Create an object version when a new object is uploaded, an existing object is overwritten, or when an object is deleted. [Learn more](#)

Emit Object Events
Create automation based on object state changes using the [Events Service](#).

Uncommitted Multipart Uploads Cleanup
Create a lifecycle rule to automatically delete uncommitted multipart uploads older than 7 days. [Learn more](#)

Encryption

- Encrypt using Oracle managed keys
Leaves all encryption-related matters to Oracle.
- Encrypt using customer-managed keys
Requires a valid key from a vault that you have access to. [Learn more](#)

Resource logging

Enable resource logging to allow resource tracking, troubleshooting, and data insights

Resource logging disabled



Create [Cancel](#)

- o **Upload the Zip folder to the bucket as describe in picture below :**

Upload Objects

Object Name Prefix *Optional*

Storage Tier

Standard

Additional checksum *Optional*

None

Choose Files from your Computer









Show Optional Response Headers and Metadata



Upload [Cancel](#)

- **From the VM execute the following command:**

```
oci os object bulk-download --bucket-name <BUCKET NAME> --download-dir
<The DIR where you want to save the folder>
```

Example:

```
oci os object bulk-download --bucket-name my-bucket --download-dir
```

- **Unzip the folder locally from the VM with command:**

```
unzip XXXX.zip
```

- **Update the oadb-wallet file: Copy the path of the directory where you save the unzipped files (execute the command pwd) .**

- **Create oadb- wallet secret as follow:**

- Name: oadb-wallet
 - Compartment: Use the root for the purpose of this lab
 - Encryption Key: Choose from drop-down list
 - Secret Type Template: < plain-text >
 - Secrets contents: The path should be the value for the oadb-wallet secret. For Example: /home/ubuntu
-

Create Secret [Help](#)

Create in Compartment
Okeworkshop2 (root)

Name
oadb-wallet

Description

Encryption Key in Okeworkshop2 (root) [\(Change compartment\)](#)
MyVaultKey

Automatic secret generation

Use auto secret generation to automatically generate the secret content.

Manual secret generation

Use manual secret generation to manually provide the secret content.

Secret Type Template
Plain-Text

Secret Contents
/home/ubuntu

Show Base64 conversion

Secret rotation

[Create Secret](#) [Cancel](#)

Task 6: Create ExternalSecret

1. Create file named oadb-admin-secret.yaml as follow:

```
apiVersion: external-secrets.io/v1beta1
kind: ExternalSecret
metadata:
  name: oci-secret-admin
spec:
  refreshInterval: 0.03m
  secretStoreRef:
    kind: SecretStore
    name: workshop-vault # Must match SecretStore name deployed on the cluster
  target:
    name: oadb-admin # Must match adb secret name downloaded in Task 4-section 9
    creationPolicy: Owner
  data:
    - secretKey: key
      remoteRef:
        key: oadb-admin
```

2. Create file named oadb-connection-secret.yaml as follow:

```
apiVersion: external-secrets.io/v1beta1
kind: ExternalSecret
metadata:
  name: oci-secret-connection
spec:
  refreshInterval: 0.03m
  secretStoreRef:
    kind: SecretStore
    name: workshop-vault # Must match SecretStore name deployed on the cluster
  target:
    name: oadb-connection # Must match adb secret name downloaded in Task 4-section 9
    creationPolicy: Owner
  data:
    - secretKey: key
      remoteRef:
        key: oadb-connection
```

3. Create file named oadb-wallet-secret.yaml as follow:

```
apiVersion: external-secrets.io/v1beta1
kind: ExternalSecret
metadata:
  name: oci-secret-wallet
spec:
  refreshInterval: 0.03m
  secretStoreRef:
    kind: SecretStore
    name: workshop-vault # Must match SecretStore name deployed on the cluster
  target:
    name: oadb-wallet # Must match adb secret name downloaded in Task4-section 9

  creationPolicy: Owner
  data:
    - secretKey: key
      remoteRef:
        key: oadb-wallet
```

4. Deploy Secrets Yaml files by executing the following commands :

```
kubectl apply -f oadb-admin-secret.yaml
kubectl apply -f oadb-connection-secret.yaml
kubectl apply -f oadb-wallet-secret.yaml
```

Task 6: Validate external secrets created

1. Execute the following commands:

```
kubectl get es
```

```
shmuel_mur@cloudshell:~ (eu-frankfurt-1)$ kubectl get es -A
NAMESPACE   NAME           STORE      REFRESH INTERVAL   STATUS      READY
default     oci-secret     oke-db-secret  0.03m          SecretSynced  True
default     oci-secret-connection oke-db-secret  0.03m          SecretSynced  True
default     oci-secret-wallet  oke-db-secret  0.03m          SecretSynced  True
```

```
kubectl get secrets ( validate that secrets created and sync with vault)
```

```
shmuel_mur@cloudshell:~ (eu-frankfurt-1)$ kubectl get secrets
NAME                           TYPE        DATA   AGE
oadb-admin                      Opaque      1      3d1h
oadb-connection                  Opaque      1      3d1h
oadb-wallet                      Opaque      1      3d1h
sh.helm.release.v1.oci-kubernetes-monitoring.v1  helm.sh/release.v1  1      3d
```

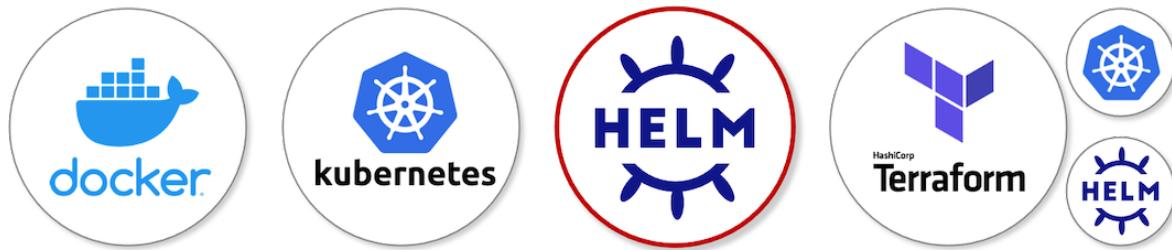
```
kubectl get secret oadb-wallet -o json | jq -r ."data.key" | base64 -d  
(validate secrets in OKE synced with value)
```

```
shmuel_mur@cloudshell:~ (eu-frankfurt-1)$ kubectl get secret oadb-wallet -o json | jq -r ."data.key" | base64 -d  
/home/shmuel_mur/walletshmuel_mur@cloudshell:~ (eu-frankfurt-1)$ █
```

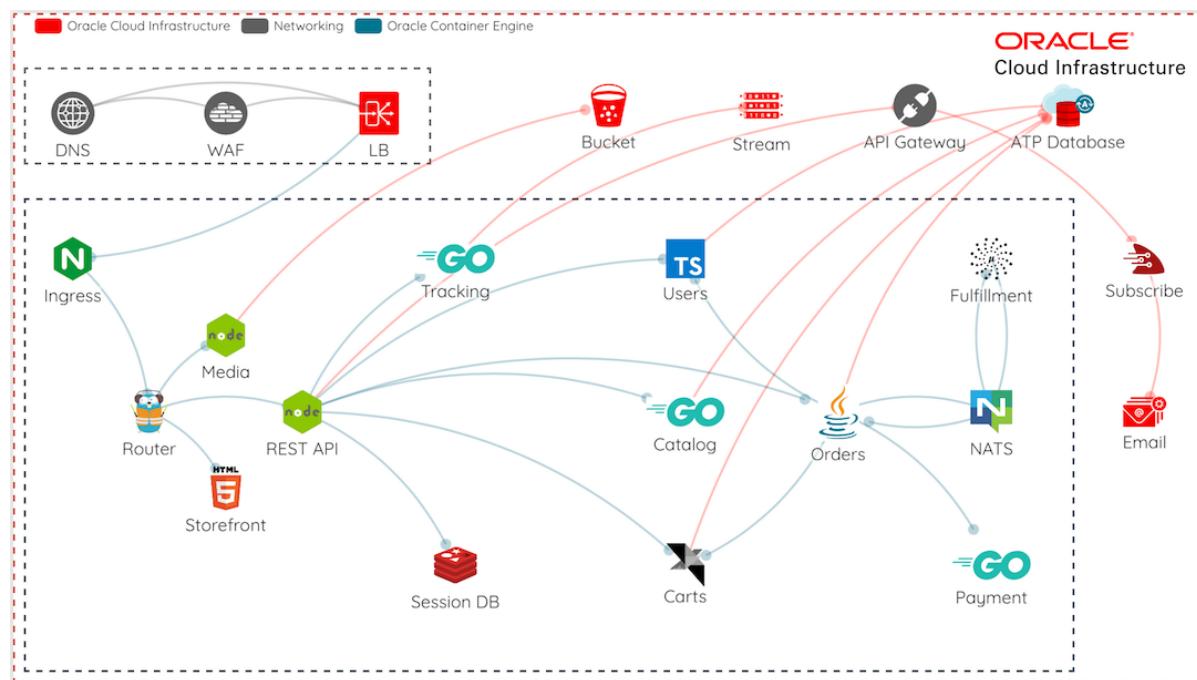
LAB 4 - Install MuShop Using Helm

Introduction

There are four options for deploying MuShop. They range from manual (docker), automated (Helm) to fully automated (Terraform).



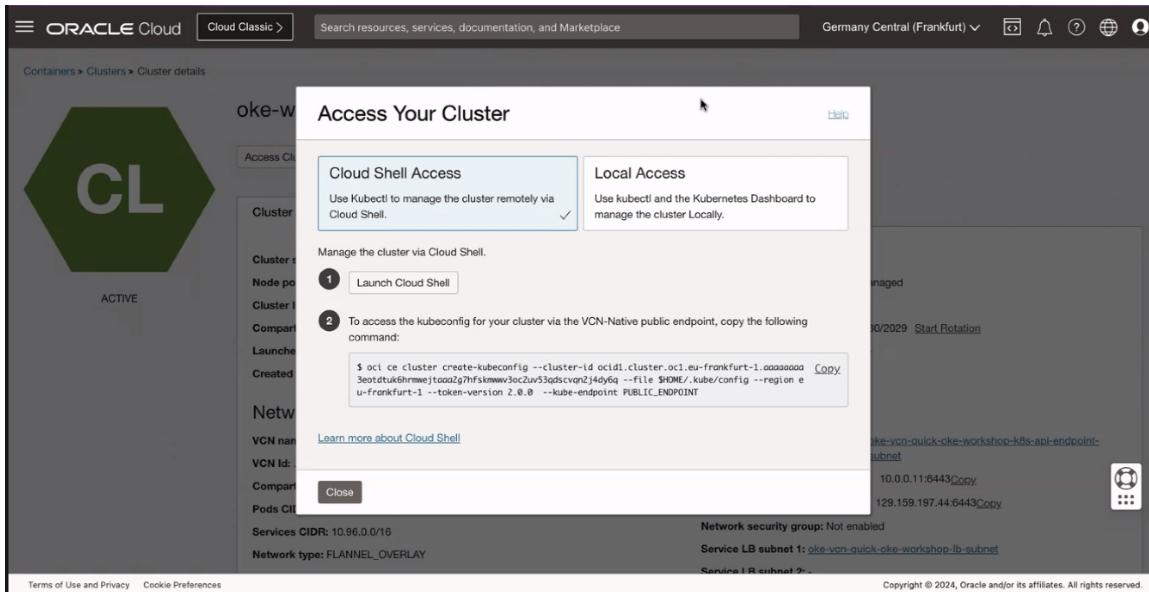
Designing in microservices offers excellent separation concerns and provides developer independence. While these benefits are clear, they can often introduce some complexity for the development environment. Services support configurations that offer flexibility, when necessary, and establish parity as much as possible. It is essential to use the same tools for development to production.



Note: This diagram contains services not covered by these labs.

Task 1: Access the OKE Cluster from Cloud Shell

1. In OCI console Navigate to the Developer Services > Kubernetes clusters (OKE).
2. Choose the OKE cluster you have previously created with terraform.
3. Connect to the OKE cluster from Quick Start with Cloud Shell.



Task 2: Obtain MuShop source code

1. Open Cloud Shell and clone the github repo.

```
git clone https://github.com/oci-oke-workshop-il/oci-cloudnative-ext.git
```

Sample response

```
livelabs main
Cloning into 'mushop'...
remote: Enumerating objects: 542, done.
remote: Counting objects: 100% (542/542), done.
remote: Compressing objects: 100% (313/313), done.
remote: Total 15949 (delta 288), reused 424 (delta 200), pack-reused 15407
Receiving objects: 100% (15949/15949), 17.59 MiB | 33.71 MiB/s, done.
Resolving deltas: 100% (9557/9557), done.
```

2. Change to the mushop directory containing the Helm charts:

```
cd oci-cloudnative-ext/deploy/complete/helm-chart/mushop/charts
```

Task 4: Update the Image Repository

In this Task we will Update the image repository in the `values.yaml` files for the following microservices - **API, Storefront & Catalogue** . The new value must point to the container repository previously created in Lab 1

1. Enter every folder in `helm-chart/mushop/charts` , For example `/api` .
2. Open the `values.yaml` file.
3. Update the `image.repository` field with the full path of the container repository created previously.
4. Update the Tag for the image that you pushed previously, For example v1

The screenshot shows the Container Registry interface for the compartment 'okeworkshop1 (root)'. On the left, there's a sidebar with 'Containers & Artifacts' and several navigation links: Overview, Kubernetes Clusters (OKE), Container Instances, **Container Registry** (which is selected), and Artifact Registry. Below that are 'List scope' and 'Compartment' dropdowns set to 'okeworkshop1 (root)'. The main area is titled 'Container Registry in okeworkshop1 (root)' and displays two large numbers: 'Repositories' (14) and 'Images' (13). A checkbox 'Include unversioned images' is checked. Below these are sections for 'Repositories and images' and a search bar. The search results list several entries, with one specific entry highlighted with a red box: 'oke-workshop-ocir/api (Public) [1 image]'. Other listed entries include 'anastasiia_oke_workshop/mushop_api (Private) [1 image]', 'anastasiia_oke_workshop/mushop_assets (Private) [1 image]', 'oke-workshop-ocir/assets (Public) [1 image]', and 'oke-Container Instances ts (Public) [1 image]'.

Example (`values.yaml` for one of the microservices):

```

replicaCount: 1

image:
  repository: iad.ocir.io/oracle/ateam/mushop-api
  tag: 2.3.2
  pullPolicy: IfNotPresent

env:
  mediaUrl: /assets
  newsletterSubscribeUrl:

resources:
  limits:
    cpu: 300m
    memory: 300Mi
  requests:
    cpu: 100m
    memory: 100Mi

securityContext:
  runAsNonRoot: true
  runAsUser: 10001
  capabilities:
    drop:
      - all
    add:
      - NET_BIND_SERVICE
  readOnlyRootFilesystem: true

hpa:
  enabled: true
  minReplicas: 1
  maxReplicas: 10
  metrics:
    - type: Resource
      resource:
        name: cpu
        target:
          type: Utilization
          averageUtilization: 70
service:
  port: 80

```

Version of the Docker Image



5. Repeat this process for API, Storefront & Catalogue microservices !!!

Task 5: Deploy the eCommerce App with Helm

1. Deploy the application by Executing the following command:

```

helm install mymushop mushop \
--namespace mushop \
--create-namespace \
--set global.mock.service=all

```

2. Please be patient. It may take a few moments to download all the application images

Execute the following command:

```
kubectl get pods -n mushop
```

Sample response to make sure all the pods are EXISTS and in the Running state

NAME	READY	STATUS	RESTARTS	AGE
mymushop-api-6ddd9588d7-tdfkd	1/1	Running	0	4m10s
mymushop-assets-79f94c8fc4-2ch4c	1/1	Running	0	4m10s
mymushop-carts-78c95c4cd8-7bsp5	1/1	Running	0	4m9s
mymushop-catalogue-5877694495-pd5r7	1/1	Running	0	4m9s
mymushop-edge-76846dc667-sfmnq	1/1	Running	0	4m10s
mymushop-events-fd4947876-7pcpk	1/1	Running	0	4m9s
mymushop-fulfillment-f5768f659-x7k7q	1/1	Running	0	4m10s
mymushop-nats-66dc5c4b59-jsqrn	2/2	Running	0	4m9s
mymushop-orders-6b5946c8b-qlbgf	1/1	Running	0	4m9s
mymushop-payment-7cffdf576-s65n9	1/1	Running	0	4m9s
mymushop-session-85b68b6c68-688ns	1/1	Running	0	4m9s
mymushop-storefront-748f549c79-m87qq	1/1	Running	0	4m9s
mymushop-user-595d9f5584-hjmch	1/1	Running	0	4m10s

LAB 5 - Install Ingress Controller

Prerequisites : Running Kubernetes cluster ; kubectl ; Helm installed.

Task 1: Add NGINX Ingress Helm Repository

3. Execute the following command:

```
helm repo add ingress-nginx https://kubernetes.github.io/ingress-nginx
helm repo update
```

Task 2: Create a namespace for NGINX Ingress

1. Execute the following command:

```
kubectl create namespace nginx-ingress
```

Task 3: Install NGINX Ingress Controller using Helm

1. Execute the following command:

```
helm install nginx-ingress ingress-nginx/ingress-nginx --namespace nginx-ingress
```

Task 4: Validate installation:

1. Execute the following command:

```
kubectl get pods -n nginx-ingress
```

Sample response

NAME	READY	
STATUS	RESTARTS	AGE
nginx-ingress-ingress-nginx-controller-xxxxxx	1/1	
Running	0	2m

LAB 6 - Install CertManager

Task 1: Add Cert-Manager to OKE via Add-on:

1. Navigate to **Developer Services > OKE** > Click on **Cluster Name** > Click **Add-ons** > Click **Manage add-ons**

The screenshot shows the 'Add-ons' section of the OKE cluster management interface. On the left, there's a sidebar with links like 'Metrics', 'Node pools', 'Work requests', 'Image verification', 'Quick Start', and 'Path analysis tests'. Below these is a section titled 'Add-ons' which is currently selected. On the right, there's a main panel with a title 'Add-ons' and a 'Manage add-ons' button. A list of installed add-ons is shown, each with a checkbox and a 'Delete' button. The add-ons listed are: CertManager, ClusterAutoscaler, CoreDNS, Flannel, and KubeProxy.

Name	Action
CertManager	<input type="checkbox"/> Delete
ClusterAutoscaler	<input type="checkbox"/> Delete
CoreDNS	<input type="checkbox"/> Delete
Flannel	<input type="checkbox"/> Delete
KubeProxy	<input type="checkbox"/> Delete

Task 2: Enable CertManager

The screenshot shows the 'Edit Certificate manager' configuration page. On the left, there's a sidebar with a 'Certificate manager' section containing a brief description of Cert-manager. The main area has a title 'Edit Certificate manager' and a 'Help' link. It contains several configuration options:

- A checked checkbox for 'Enable Certificate manager' with a tooltip explaining its function.
- A radio button for 'Automatic Updates' with a tooltip.
- A radio button for 'Choose a version' with a tooltip.
- A large 'Add configuration' button at the bottom right.

Task 3: Validate CertManager installation

1. Verify that CertManager pods are running

```
kubectl get pods -n cert-manager
```

Sample response

NAME	READY	STATUS	
RESTARTS	AGE		
<code>cert-manager-xxxxxx</code>	1/1	Running	0
2m			
<code>cert-manager-cainjector-xxxxxx</code>	1/1	Running	0
2m			
<code>cert-manager-webhook-xxxxxx</code>	1/1	Running	0
2m			

Task 4: Create Let's Encrypt ClusterIssuer

1. Create a YAML file named : **letsencrypt-prod-clusterissuer.yaml**

```
vi letsencrypt-prod-clusterissuer.yaml
```

2. Enter the following:

```
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: letsencrypt-prod
spec:
  acme:
    server: https://acme-v02.api.letsencrypt.org/directory
    email: your-email@example.com # Replace with your email
    privateKeySecretRef:
      name: letsencrypt-prod
    solvers:
    - http01:
        ingress:
          class: nginx
```

Task 5: Apply the ClusterIssuer:

```
kubectl apply -f letsencrypt-prod-clusterissuer.yaml
```

Task 6: Validate ClusterIssuer created:

```
kubectl get clusterissuer
```

Sample response

NAME	READY	AGE
letsencrypt-prod	True	2m

Task 7: Configure the MuShop Ingress Resource with TLS

1. Create YAML file named: mushop-dev-ingress.yaml

```
vi mushop-dev-ingress.yaml
```

2. Add the following content:

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: mushop-dev
  namespace: mushop
  annotations:
    kubernetes.io/ingress.class: nginx
    cert-manager.io/cluster-issuer: letsencrypt-prod
spec:
  tls:
  - secretName: tls-secret
  rules:
  - http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: edge
            port:
              number: 8080
```

Task 8: Apply the Ingress Resource:

1. Apply the Ingress resource to your mushop namespace:

```
kubectl apply -f mushop-dev-ingress.yaml
```

Task 9: Validate Ingress Setup:

2. Execute the following command:

```
kubectl get ingress mushop-dev -n mushop
```

Expected results after LoadBalancer deployment:

- Ingress resource and an external IP
- hostname

Task 10: Test HTTPs Access:

1. Find the **External IP** assigned to the ingress controller.

```
kubectl get svc -n nginx-ingress
```

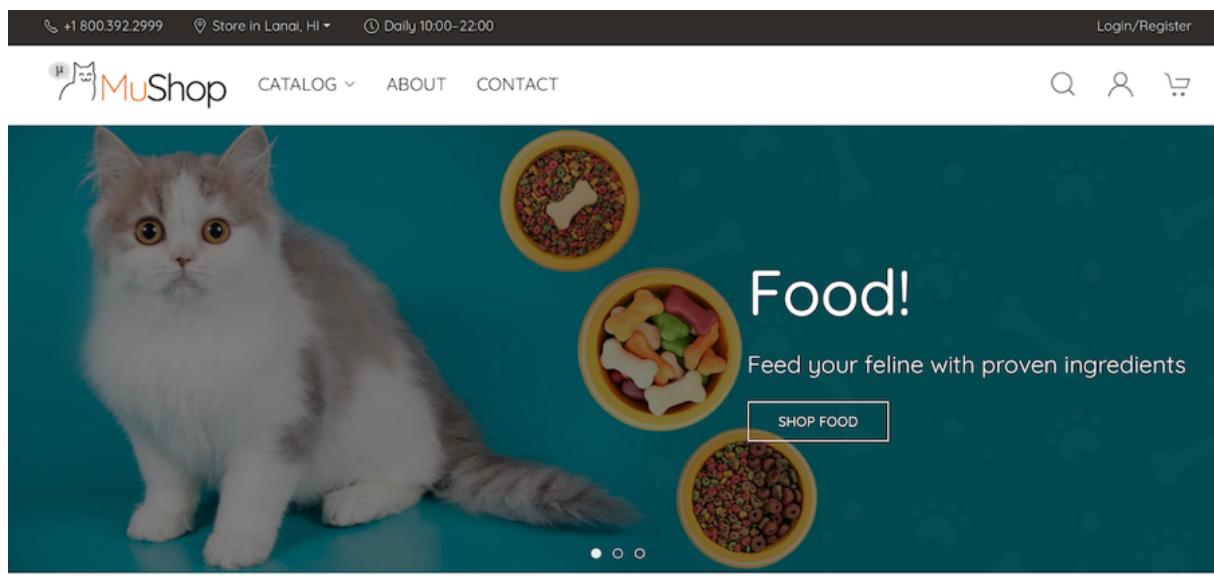
Look for the EXTERNAL-IP in the output:

Sample response

NAME	CLUSTER-IP	EXTERNAL-IP	PORT(S)	TYPE
nginx-ingress-ingress-nginx-controller	10.96.25.104	your-external-ip	80:32456/TCP,443:31578/TCP	LoadBalancer

Task 11: Application Access:

1. Open to the MuShop Storefront by using your browser connecting to <https://EXTERNAL-IP>



The screenshot shows the MuShop storefront. At the top, there are links for phone number (+1 800.392.2999), store location (Store in Lanai, HI), and daily hours (Daily 10:00-22:00). On the right, there are buttons for Login/Register, a search icon, a user icon, and a shopping cart icon. The main header features the MuShop logo with a cat icon. Below the header, there are navigation links for CATALOG, ABOUT, and CONTACT. The main visual area features a large image of a fluffy white and brown cat sitting next to three bowls of colorful cat food. The word "Food!" is prominently displayed in the center, with the subtext "Feed your feline with proven ingredients". A "SHOP FOOD" button is located below the bowls. Below this section, there is a "Trending Now" heading followed by five product cards:

- Litter Trapper** (Cat Litter) - \$28.99
- Mu BroomKit** (Cat Litter Accessories) - \$7.99
- Mu DeoSpray** (Cat Litter Deodorizers) - \$4.99
- Mu O-DeoSpray** (Cat Litter Deodorizer) - \$4.99
- Mu LitterBox** (Cat Litter Boxes) - \$9.50

A cookie consent message is overlaid on the first product card:
This website uses cookies to ensure proper functionality, but does not use personal information or track usage. By using the website, you agree to our use of cookies.
OK

LAB 7 – Deployment Autoscaling

Introduction

Scaling out a Deployment will ensure new Pods are created and scheduled to Nodes with available resources. Scaling will increase the number of Pods to the new desired state. Kubernetes also supports autoscaling of Pods, but it is outside of the scope of this tutorial. Scaling to zero is also possible, and it will terminate all Pods of the specified Deployment.

This step showcases the [Horizontal Pod Autoscaling](#) configurations deployed with the MuShop application.

Task 1: Create Policy to work with Autoscaler

1. Setting Up Instance Principal to Enable the Cluster Autoscaler Add-on to Access to Node Pools
 - Obtain the Cluster OCID (Cluster ID)



- Obtain Compartment OCID - Navigate to **Identity & Security > compartments**

Compartments

Create Compartment			
Name	Status		
okeworkshop1	● Active	...yc5tuq	Yes

2. Create the policy

- Navigate to Identity & Security > Policies > Create Policy

Identity

Policies *in okeworkshop1 (root)*

Overview Domains Network Sources Policies

Create Policy Delete

<input type="checkbox"/>	Name
<input type="checkbox"/>	logging_analytics_automatic_service_policies
<input type="checkbox"/>	Docker VM

- Fill the policy name (e.g cluster-autoscaler-policy)

Create Policy

Name

cluster-autoscaler-policy

No spaces. Only letters, numerals, hyphens, periods, or underscores.

- Enter policy statements to allow node pool management as follow:

Allow dynamic-group <dynamic-group-name> to manage cluster-node-pools in compartment id <compartment-ocid> where ALL {request.principal.type='workload', request.principal.namespace ='kube-system', request.principal.service_account = 'cluster-autoscaler', request.principal.cluster_id = '<cluster-ocid>'}

Allow dynamic-group <dynamic-group-name> to use subnets in compartment id <compartment-ocid> where ALL {request.principal.type='workload', request.principal.namespace ='kube-system', request.principal.service_account = 'cluster-autoscaler', request.principal.cluster_id = '<cluster-ocid>'}

Allow dynamic-group <dynamic-group-name> to read virtual-network-family in compartment id <compartment-ocid> where ALL {request.principal.type='workload', request.principal.namespace ='kube-system', request.principal.service_account = 'cluster-autoscaler', request.principal.cluster_id = '<cluster-ocid>'}

Allow dynamic-group <dynamic-group-name> to use vnics in compartment id <compartment-ocid> where ALL {request.principal.type='workload', request.principal.namespace ='kube-system', request.principal.service_account = 'cluster-autoscaler', request.principal.cluster_id = '<cluster-ocid>'}

Allow dynamic-group <dynamic-group-name> to inspect compartments in compartment id <compartment-ocid> where ALL {request.principal.type='workload', request.principal.namespace ='kube-system', request.principal.service_account = 'cluster-autoscaler', request.principal.cluster_id = '<cluster-ocid>'}
--

Allow dynamic-group <dynamic-group-name> to manage all-resources in tenancy

Replace:

- <cluster-ocid> with the cluster OCID obtained previously.
- <compartment-ocid> with the cluster OCID obtained previously.
- <dynamic-group-name> with the Dymanic group created on LAB 3 (Vault)

Task 2: Create the Cluster Autoscaler Add-on configuration

1. In OCI console navigate to **OKE cluster > Add-on > Manage Add-ons** > Click on **Cluster Autoscaler** > Click “**Enable Cluster Autoscaler**” > Choose **nodes** from the option drop-down list > Fill the “Value” parameters

The nodes parameter value has the following format:



2. For this lab Replace :

- o <min-nodes> = 1 , The minimum number of nodes allowed in the node pool. The Kubernetes Cluster Autoscaler will not reduce the number of nodes below this number.
- o <max-nodes> = 5 , The maximum number of nodes allowed in the node pool. The Kubernetes Cluster Autoscaler will not increase the number of nodes above this number. Make sure the maximum number of nodes you specify does not exceed the tenancy limits for the worker node shape defined for the node pool.
- o <nodepool-ocid> with the node pool OCIDs.

Task 3: Validate Cluster Autoscaler successfully deployed

```
kubectl get deployments -n kube-system | grep cluster-autoscaler
```

```
shmuell_mur@cloudshell:~$ kubectl get deployments -n kube-system | grep cluster-autoscaler
cluster-autoscaler   1/1     1           1          2m39s
```

Task 4: Change the Value of the HPA autoscaaller

1. Execute the command:

```
kubectl get hpa -n mushop
```

2. Edit the value of the HPA with the command :

```
kubectl edit hpa mymushop-api -n mushop
```

```

ubuntu@instance-oke-workshop:~/valult$ kubectl get hpa -n mushop
NAME             REFERENCE          TARGETS          MINPODS   MAXPODS   REPLICAS
mymushop-api     Deployment/mymushop-api    cpu: <unknown>/70%  1         50        50
mymushop-assets  Deployment/mymushop-assets  cpu: <unknown>/70%  1         10        1
mymushop-catalogue Deployment/mymushop-catalogue  cpu: <unknown>/70%  1         10        1
mymushop-edge    Deployment/mymushop-edge    cpu: <unknown>/70%  1         10        1
mymushop-events  Deployment/mymushop-events  cpu: <unknown>/70%  1         10        1
mymushop-storefront Deployment/mymushop-storefront  cpu: <unknown>/70%  1         10        1
mymushop-user    Deployment/mymushop-user    cpu: <unknown>/70%  1         10        1
ubuntu@instance-oke-workshop:~/valult$ kubectl edit hpa mymushop-api -n mushop
Edit cancelled, no changes made.
ubuntu@instance-oke-workshop:~/valult$ █

```

3. Change the value to 50

```

█ Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
  annotations:
    meta.helm.sh/release-name: mymushop
    meta.helm.sh/release-namespace: mushop
  creationTimestamp: "2024-11-17T13:22:46Z"
  labels:
    app.kubernetes.io/managed-by: Helm
  name: mymushop-api
  namespace: mushop
  resourceVersion: "19587"
  uid: 0e275e5c-b6cb-4a2c-8bae-b370afae3a32
spec:
  maxReplicas: 50
  metrics:
    - resource:
        name: cpu
        target:
          averageUtilization: 70
          type: Utilization
        type: Resource
      minReplicas: 1
    scaleTargetRef:
      apiVersion: apps/v1
      kind: Deployment
      name: mymushop-api
  status:
    conditions:
      - lastTransitionTime: "2024-11-17T13:23:01Z"
        message: the HPA controller was able to get the target's current scale
        reason: SucceededGetScale

```

4. Save the changed with command :wq

Task 5: Increase replica count in the Mushop API service deployment to 50

1. Execute the following command:

```
kubectl scale deployment mymushop-api --replicas=50 -n mushop
```

Example – Status should be “Running” for all

```
ubuntu@uservm:~/oci_workshop$ kubectl scale deployment mymushop-api --replicas=50 -n mushop
deployment.apps/mymushop-api scaled
ubuntu@uservm:~/oci_workshop$ kubectl get pods -n mushop
NAME                  READY   STATUS    RESTARTS   AGE
mymushop-api-5984fc895b-2hc4d   1/1    Running   0          20s
mymushop-api-5984fc895b-4tswj   1/1    Running   0          20s
mymushop-api-5984fc895b-558xd   1/1    Running   0          20s
mymushop-api-5984fc895b-55hsf   0/1    Pending   0          19s
mymushop-api-5984fc895b-5bvdb   1/1    Running   0          20s
mymushop-api-5984fc895b-6v8mv   0/1    Pending   0          19s
```

2. Where Pod Status is “Pending” you might verify the reseaon by executing the following command :

```
kubectl describe pod <add pod name> -n mushop
```

Example – Status should be “Running” for all

```
ubuntu@uservm:~/oci_workshop$ kubectl describe pod mymushop-api-5984fc895b-7rw89 -n mushop
Name:           mymushop-api-5984fc895b-7rw89
Namespace:      mushop
Priority:       0
Service Account: default
Node:           <none>
Labels:         app=api
Events:          (Last 10 events) (most recent failure first)
                Type  Reason     Age   From           Message
                ----  -----     --   --            --
                Warning FailedScheduling 105s  default-scheduler  0/3 nodes are available: 3 Insufficient cpu. preemption: 0/3 nodes are available: 3 No preemption victims found for incoming pod.
                Normal  TriggeredScaleUp 98s   cluster-autoscaler  pod triggered scale-up: [{"ocid1.nodepool.oc1.eu-frankfurt-1.aaaaaaaaarjx2636uyy7a16o6vpuz3asizet2etvsu53ulubny4xa2x4qmq 3->4 (max: 5)}]
ubuntu@uservm:~/oci_workshop$ kubectl describe pod mymushop-api-5984fc895b-7rw89 -n mushop
```

Task 6: Validate replica size and new node were created in the nodepool

1. In the CLI Execute the following command:

```
Kubectl get pods -n mushop
```

2. In OCI console navigate to **Developer services > Kubernetes Clusters (OKE) > click Cluster Name > click Node pools > click Node pool name > validate the number of Nodes > 3**

Nodes					
 ⓘ In order to access your private nodes with a public SSH key you will need to set up a bastion host (a.k.a. jump box). Learn more about setting up a bastion host Dismiss					
Node name	Private IP	Kubernetes node condition ⓘ	Node state ⓘ	Kubernetes version	
oke-cerlapuzffa-ny4xa2x4qmg-son3j74axaa-0	10.0.10.212	● Ready	● Active	✓ v1.30.1	▼
oke-cerlapuzffa-ny4xa2x4qmg-son3j74axaa-1	10.0.10.211	● Ready	● Active	✓ v1.30.1	▼
oke-cerlapuzffa-ny4xa2x4qmg-son3j74axaa-2	10.0.10.176	● Ready	● Active	✓ v1.30.1	▼
oke-cerlapuzffa-ny4xa2x4qmg-son3j74axaa-3	10.0.10.189	● Not ready	● Active	✓ v1.30.1	▼

Showing 4 items < 1 of 1 >

Task 7: Revert replica size

1. Execute the following command and validate that pods are terminated

```
kubectl scale deployment mymushop-api --replicas=1 -n mushop
```

2. Validate the number of nodes in the OCI console as described in task 6.

Task 8: validate # pod revert to the original count

```
kubectl get pods -n mushop
```

LAB 8 – Monitoring the deployment

Introduction

Observability helps to understand and explain the state of your systems using a combination of logs, metrics and traces. It helps bringing better visibility into systems.

Pre-requisites: OCI CLI & Kubectl installed

Task 1: OKE Management Pack Deployment

1. Navigate to **Observability & Management > Logging Analytics > Click on Solutions**

The screenshot shows the Oracle Cloud interface for Observability & Management. The left sidebar lists various services: Home, Compute, Storage, Networking, Oracle Database, Databases, Analytics & AI, Developer Services, and Identity & Security. Below this is a section for **Observability & Management**. The main content area is titled **Observability & Management** and contains three main sections: **Application Performance Monitoring**, **Logging Analytics**, and **Events Service**. Under **Logging Analytics**, the **Solutions** link is highlighted with a green circle. Other options under Logging Analytics include Home, Log Explorer, Dashboards, Administration, and Rules. The **Events Service** section includes Rules and Metrics. The **Database Management** section includes Overview and Diagnostics & Management.

2. Choose **Kubernetes** solution

The screenshot shows the **Logging Analytics** section of the Oracle Cloud interface. The left sidebar shows the **Solutions** section expanded, with **Kubernetes** selected. A green box highlights the **Kubernetes** card, which features the Kubernetes logo and the word "Kubernetes". Other options in the **Solutions** section include Home, Log Explorer, Dashboards, and Administration.

3. Click on Connect clusters

The screenshot shows the Oracle Cloud interface for monitoring Kubernetes. At the top, there's a navigation bar with 'ORACLE Cloud', 'Cloud Classic >', a search bar, and account information ('Germany Central (Frankfurt)'). Below the navigation is a dropdown menu for 'Logging Analytics' and a breadcrumb trail: 'Solutions > Kubernetes'. The main content area is titled 'Kubernetes' and contains a brief description of the monitoring solution. To the right, there's a 'Get started' section with a 'Connect clusters' button, which is highlighted with a green box. Below this is a table with columns: Name, Type, Cluster Connected, Cluster Created, Kubernetes Version, CPU, Memory, Pods, and Latest telemetry. The table displays 'No data to display.' At the bottom, there are pagination controls: 'Page 1 (0 of 0 items)', a search icon, and navigation arrows.

4. Add Data > Monitor Kubernetes > Oracle OKE

The screenshot shows the 'Add Data' page in Oracle Cloud. The top navigation bar includes 'ORACLE Cloud', 'Cloud Classic >', and a search bar. The main content area has a title 'Add Data' and several sections: 'Monitor OCI resources', 'Monitor with management agent', and 'Monitor Kubernetes'. Under 'Monitor Kubernetes', there's an option for 'Oracle OKE', which is highlighted with a green box. The Oracle OKE icon consists of a blue cloud with a gear and a wrench.

5. Select the **OKE cluster** from the clusters list and click **Next**

Cluster name	Cluster Created	Node pools	Subnet	Kubernetes Version	OKE Compartment
oke-workshop	Wed, Oct 30, 2024, 16:31:41 UTC	pool1	oke-vcn-quick-oke-workshop-node-subnet	v1.30.1	okeworkshop1 (root)

6. Validate all the details based on the following screen

Cluster name	Cluster Created	Node pools	Subnet	Kubernetes Version	OKE Compartment
scaler-cluster		pool1			

7. The following page after assure that OKE monitoring environment configured successfully

Completed setup for cluster **scaler-cluster**.

- Created Logging Analytics Entity: [scaler-cluster](#)
- Created Logging Analytics Log Group: [scaler-cluster](#)
- Created Resource Manager Stack: [oci-kubernetes-monitoring-1728288428186](#)
- Resource Manager plan job completed.
- Resource Manager apply job completed.

8. Validate OKE components deployed successfully via CLI

```
kubectl get pods -n oci-onm
```

Sample response

NAME	READY	STATUS	RESTARTS	AGE
oci-onm-discovery-28804810-kvcq8	0/1	Completed	0	3m16s
oci-onm-logan-7cq8z	1/1	Running	0	4m19s
oci-onm-logan-bn527	1/1	Running	0	4m19s
oci-onm-mgmt-agent-0	1/1	Running	0	4m19s

Task 2: Create (import) OKE Dashboards

1. Go into the cloned GitHub repository ‘oci-cloudnative-ext’ or clone it again as follow :

```
git clone https://github.com/oci-oke-workshop-il/oci-cloudnative-ext.git
```

2. Execute the following command

```
cd oci- cloudnative-ext/dashboards_json
```

3. Obtain the **OCID** of the compartment, where the dashboards need to be imported (same as the OKE cluster’s compartment).

`${compartment_ocid}` - Replace all the instances of the keyword in the JSON with Compartment OCID

4. The Following command is for quick reference that can be used in a linux/cloud-shell **environment**:

```
sed -i "s/\${compartment_ocid}/<Replace-with-Compartment-OCID>/g" *.json
```

5. Execute the following commands to import the dashboards.

```
oci management-dashboard dashboard import --from-json file://cluster.json  
oci management-dashboard dashboard import --from-json file://node.json  
oci management-dashboard dashboard import --from-json file://workload.json  
oci management-dashboard dashboard import --from-json file://pod.json  
oci management-dashboard dashboard import --from-json file://service-type-lb.json
```

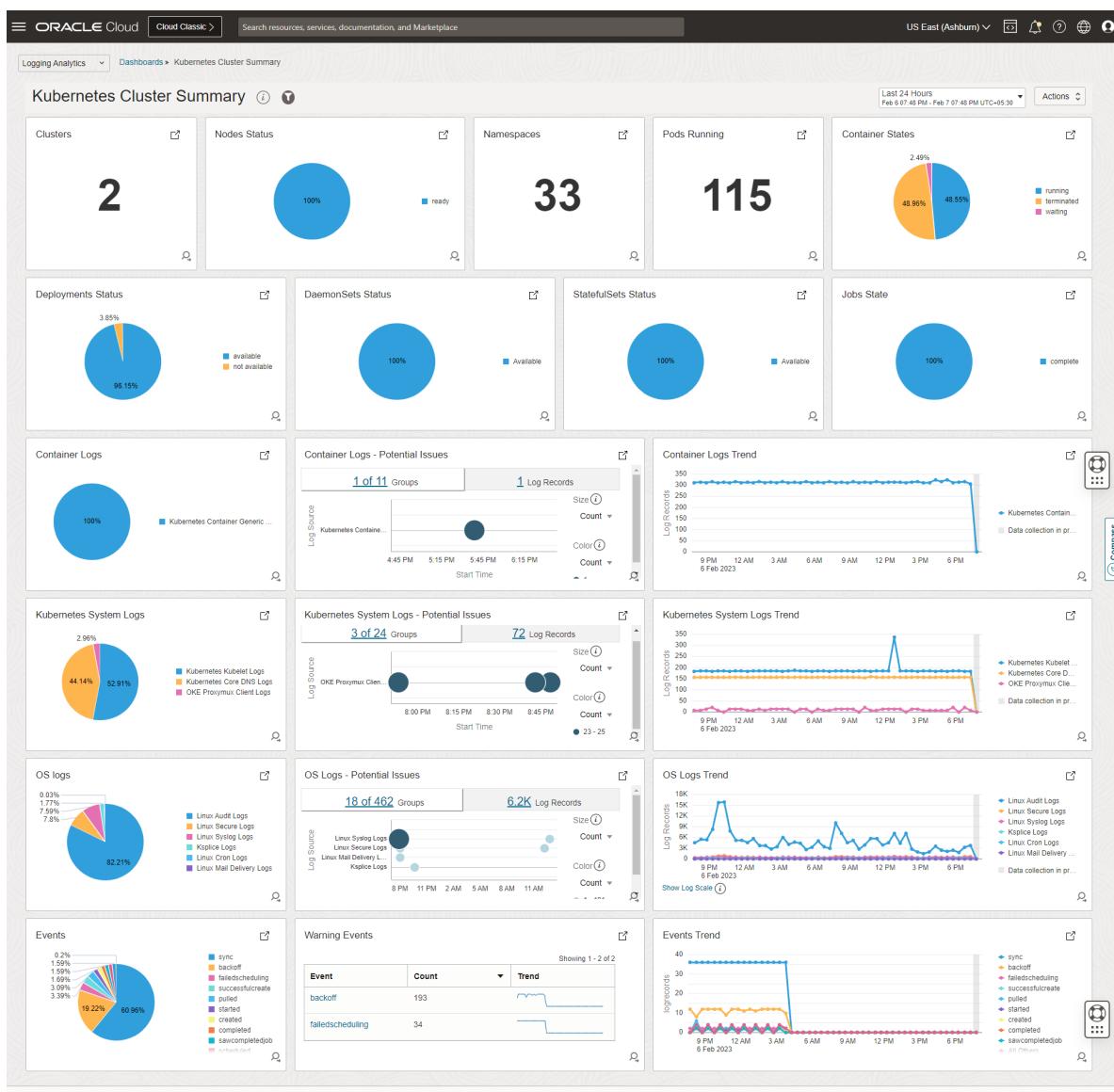
Task 3: Review the OKE Dashboard

1. Navigate to Observability & Management > Logging Analytics > Click on Dashboards

The screenshot shows the 'Dashboards' section of the Logging Analytics interface. On the left, there's a sidebar with links like Home, Log Explorer, Dashboards (which is selected), Saved Searches, Widgets, Filters, Solutions, and Administration. The main area has tabs for 'Create dashboard' and 'Import dashboards'. A search bar at the top right is set to 'kubernetes'. Below these are five listed dashboards:

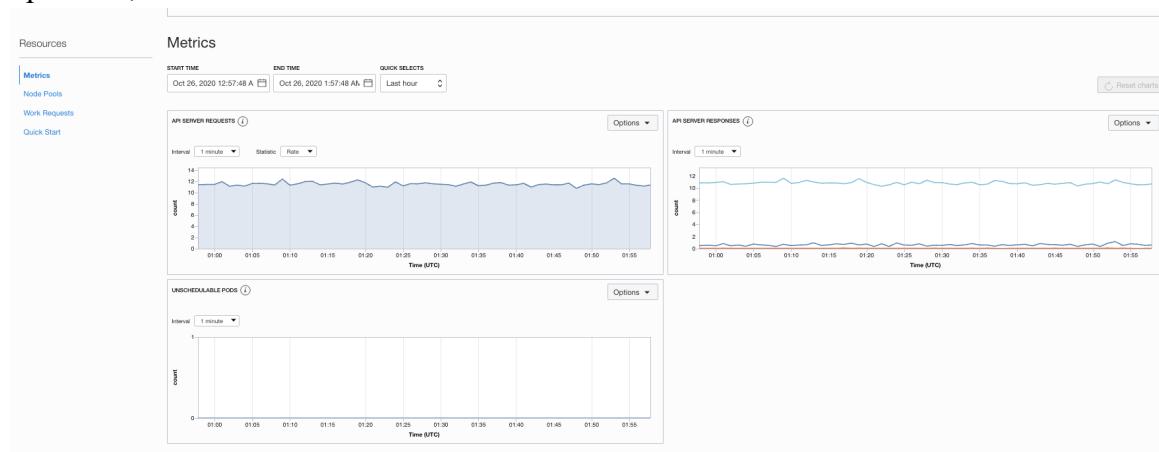
Name	Description	Services	Created by	Updated by	Last updated
Kubernetes Services (Type: Load Balancer)	Kubernetes Services (Type: Load Balancer)	Logging Analytics			
Kubernetes Pods	Kubernetes Pods	Logging Analytics			
Kubernetes Workloads	Kubernetes Workloads	Logging Analytics			
Kubernetes Nodes	Kubernetes Nodes	Logging Analytics			
Kubernetes Cluster Summary	Kubernetes Cluster Summary	Logging Analytics			

2. Click on the dashboards to view the cluster metrics, logs and status:



Task 3: Review OKE Metrics

1. OKE Cluster Metrics: Navigate to **Developer Services > Kubernetes Clusters >**
2. Under **Resources -> Metrics** observe the following metrics:
 - o Unschedulable pods, which can be used to trigger node pool scale operations when there are insufficient resources on which to schedule pods.
 - o API Server requests per second, which is helpful to understand any underlying performance issues seen in the Kubernetes API server.
3. These metrics can also be viewed from OCI Monitoring console under “oci_oke” namespace. Additionally, alarms can be created using industry standard statistics, trigger operators, and time intervals.



4. OKE Node Pool Metrics: Navigate to **Developer Services > Kubernetes Clusters > Node Pools >**

Observe the following node pool metrics:

- o Node State (If your worker nodes are in Active state as indicated by OCI Compute Service)
- o Node condition (If your worker node are in Ready state as indicated by OKE API server)



5. OKE Worker Node Metrics: Navigate to **Developer Services > Kubernetes Clusters > Node Pools > Nodes** >

Observe the following node metrics:

- Activity level from CPU. Expressed as a percentage of total time (busy and idle) versus idle time. A typical alarm threshold is 90 percent.
- Space currently in use. Measured by pages. Expressed as a percentage of used pages versus unused pages. A typical alarm threshold is 85 percent.
- Activity level from I/O reads and writes. Expressed as reads/writes per second.
- Read/Write throughput. Expressed as bytes read/Write per second.
- Network receipt/transmit throughput. Expressed as bytes received/transmit per second.



LAB – 9 - Create Alarms to Detect unscheduled Pods

Task 1: Create an OCI Topic & Subscription for email notifications

1. Search for **notifications** > click of Notifications under Application Integration

The screenshot shows the Oracle Cloud search interface. The search bar at the top contains the text "notifications". Below the search bar, the results are displayed under the heading "Search results for \"notifications\"". There are two main categories: "Notifications" and "Application Integration". Under "Notifications", there is a single result labeled "Notifications". Under "Application Integration", there is a result labeled "License Manager". On the left side of the page, there is a sidebar with links for "Home" and "Compute".

2. Click **Create topic** and name it – **EmailsTopic**

The screenshot shows the "Create Topic" form in the Oracle Cloud interface. The form has a "Name" field where "EmailsTopic" is entered. There is also a "Description" field which is currently empty. The left sidebar shows a navigation menu with "Topics" selected, and the "EmailsTopic" resource is listed under the "Topics" section.

3. Click on the created topic > Select **Create Subscription** > Protocol Email > Email {Your-Email-Address} > Click **Create**

The screenshot shows the "Create Subscription" form for the "EmailsTopic" topic. In the "Protocol" field, "Email" is selected. In the "Email" field, "user@example.com" is entered. A note at the bottom of the form states: "Email notifications use the sender "noreply" at a recipient address. Example sender: noreply@notification.us-ashburn-creating-a-subscription-for-email." A link "Show advanced options" is visible at the bottom right.

4. Go to your **email inbox** > **Confirm** the subscription.

Oracle Cloud Infrastructure Notifications Service Subscription Confirmation

From: noreply@notification.eu-frankfurt-1.oci.oraclecloud.com <nore...> To: [REDACTED] Today at 10:46

You have chosen to subscribe to the topic:
EmailsTopic (Topic OCID: ocid1.onstopic.oc1.eu-frankfurt-1.amaaaaaaa6tgjodqarfwlze5qwxkfvpk7pufcdxaygfipmfthjfke6z4yuoa)

To confirm this subscription, click or visit the link below (If this was in error, you can ignore this message):
[Confirm subscription](#)

--
Please do not reply directly to this email. If you have any questions or comments regarding this email, contact your account administrator.

5. Validate Created Subscription in **Active** state

Subscriptions

Create Subscription	
Subscription OCID	State
ocid.....jse3og4q	● Active

Task 2: Configure Alarm rule

The following alarm will generate a notification when the number of Kubernetes Nodes will be less than 3 Nodes which indicates a non-stable environment!

1. Search for **alarm** > click of **Alarm Definitions** under **Monitoring**

Search resources, services, documentation, and Marketplace

alarm

Home

Compute

Storage

Networking

Oracle Database

Metrics and Alarms

Alarm Status

Alarm Definitions

Overview

Budgets

OS Management

Monitoring

Monitoring

Cost Management

Cost Management

2. Click Create Alarm

3. Enter the following information:

- Alarm Name: NODECONDITION

Define alarm

Alarm name

NODECONDITION

- Metric Description:

- Compartment: <Your-Compartment>
- Metric Namespace: oci_oke
- Metric name: KubernetesNodeCondition

Metric description

The metric to evaluate for the alarm.

Compartment: okeworkshop1 (root)

Metric namespace: oci_oke

Resource group: Optional (No resource group selected)

Metric name: KubernetesNodeCondition

Interval: 1 minute

Statistic: Mean

- **Metric dimensions:**

- Dimension name: `clustereId`
- Dimension value: <OKE-Cluster-OCID>
- NodePoolId: leave blank

Metric dimensions

Dimension name ⓘ
clusterId

Dimension value ⓘ
ocid1.cluster.oc1.eu-frankfurt

nodePoolId ⓘ
Select a value

- **Trigger rule 1 Values:**

- Operator: Less than
- Value: 3
- Trigger delay minutes: 1 (default)
- Alarm severity: Critical

Trigger rule 1

The base condition for putting the alarm in the firing state.

Operator ⓘ
less than

Value ⓘ
3

Trigger delay minutes ⓘ
1

Unit: count

Alarm severity ⓘ
Critical

- **Destination:**

- Destination Service: Notifications
- Compartment: <Your-Compartment>
- Topic: EmailsTopic

Destination

Destination service ⓘ
Notifications

Compartment ⓘ
okeworkshop1 (root)

Topic ⓘ
EmailsTopic

- **Click Save alarm**

Task 3: Trigger Alarm manually by stopping one of the nodes in the pool

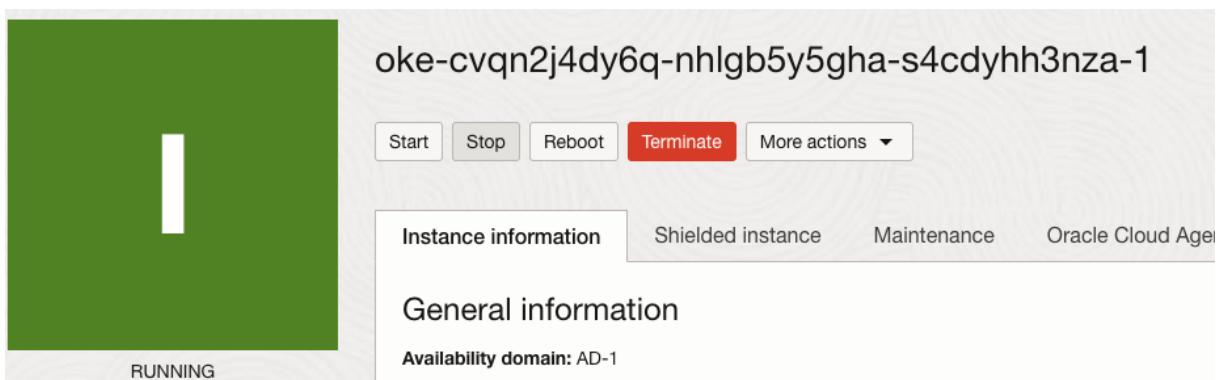
1. Navigate to Developer Services > Kubernetes Clusters (OKE) > Click Node Pools

Name	Status	Node pools <i>i</i>	VCN	Version
oke-workshop	● Active	1	oke-vcn-quick-oke-workshop	✓ v1.30.1

2. Click Pool Name

Add node pool	Node pool	Node type	Node pool status	Kubernetes version
	pool1	managed	● Active	✓ v1.30.1

3. Click on one of the nodes > Click Stop



4. Go back to Alarm view to validate the system identify the false Node and trigger and Alarm

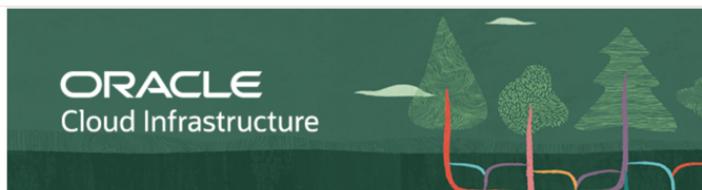
Monitoring	Alarm Status in okeworkshop1 (root) Compartment						
This page lists firing alarms and associated metric streams .							
Service Metrics	Critical alarms		Error alarms		Warning alarms		Informational alarms
	1		0		0		0
Create Alarm	Actions ▾		Search				
Alarm name	Severity	Alarm summary	Metric streams	Resource group	Triggered time		Metric namespace
NODECONDITION	Critical	...) < 3°, with a trigger delay of 1 minute	Show	Copy	-	-	Nov 14, 2024, 09:33:00 UTC

5. Validate Email notification service works correctly

Go to the **Email** and validate the following **email received**:

Alarm: OK_TO_FIRING | CRITICAL | NODECONDITION | 2024-11-14T09:33:00Z

N noreply@notification.eu-frankfurt-1.oci.oraclecloud.com <noreply@notification.eu-frankfurt-1.oci.oraclecloud.com>
To: [REDACTED]



Alarm: OK_TO_FIRING | CRITICAL | NODECONDITION | 2024-11-14T09:33:00Z

Name
NODECONDITION

State
FIRING

Type
OK_TO_FIRING

Severity
CRITICAL

Notification type
Grouped messages across metric streams

Alarm Summary
Alarm "NODECONDITION" is in a "FIRING" state; because 15 metrics meet the trigger rule:
"KubernetesNodeCondition(1m){clusterId = \"ocid1.cluster.oc1.eu-frankfurt-1.aaaaaaaa3eotduk6hrmwejaaa2g7hfskmwww3oc2uv53qdscvqn2j4dy6q\"}.mean() < 3", with a trigger delay of 1 minute

LAB 10 - Use Taints and Tolerations for Resource Allocation (Optional)

Open the following file:

```
deploy/complete/helm-chart/mushop/charts/catalogue/values.yaml
```

Add the toleration parameters:

tolerations:

```
key: "dedicated"  
value: "test-taint"  
effect: "NoSchedule"
```

Open the following file:

```
deploy/complete/helm-chart/mushop/charts/catalogue/templates/catalogue-deployment.yaml
```

Add the toleration parameters:

tolerations:

```
{ {- toYaml .Values.tolerations | nindent 8 } }
```

Redeploy the service and validate the following expected results:

```
Conditions:  
Type PodReadyToStartContainers Status  
Initialized True  
Ready True  
ContainersReady True  
PodScheduled True  
Volume:  
wallet:  
  Type: Secret (a volume populated by a Secret)  
  SecretName: catalogue-oadb-wallet  
  Optional: false  
initdb:  
  Type: ConfigMap (a volume populated by a ConfigMap)  
  Name: mymushop-catalogue-init  
  Optional: false  
  QoS Class: Burstable  
  Node-Selectors:  
    kube-api-access-7977z:  
      token:  
        TokenExpirationSeconds: 3607  
        ConfigMapName: kube-root-ca.crt  
        ConfigMapOptional: <nil>  
        DownwardAPI: true  
        QoS Class: Burstable  
        Node-Selectors: <none>  
Tolerations:  
  dedicated:test-taint:NoSchedule  
    node.kubernetes.io/not-ready:NoExecute op=Exists for 300s  
    node.kubernetes.io/unreachable:NoExecute op=Exists for 300s  
  
Events:  
  Type Reason Age From Message  
  Normal Scheduled 41s default-scheduler Successfully assigned mushop1/mymushop-catalogue-6bbff74ffd-6g9pg to 10.0.10.242  
  Normal Pulling 40s kubelet Pulling image "iad.ocir.io/oracle/ateam/mushop-catalogue:1.4"  
  Normal Pulled 38s kubelet Successfully pulled image "iad.ocir.io/oracle/ateam/mushop-catalogue:1.4" in 1.546s (1.546s including waiting). Image size: 423480299 bytes.  
  Normal Created 38s kubelet Created container catalogue  
  Normal Started 38s kubelet Started container catalogue  
anastasia@cloudshell:mushop (il-jerusalem-1)$
```