Deploy the Containers in Azure Container Instance

Optional task: If time permits

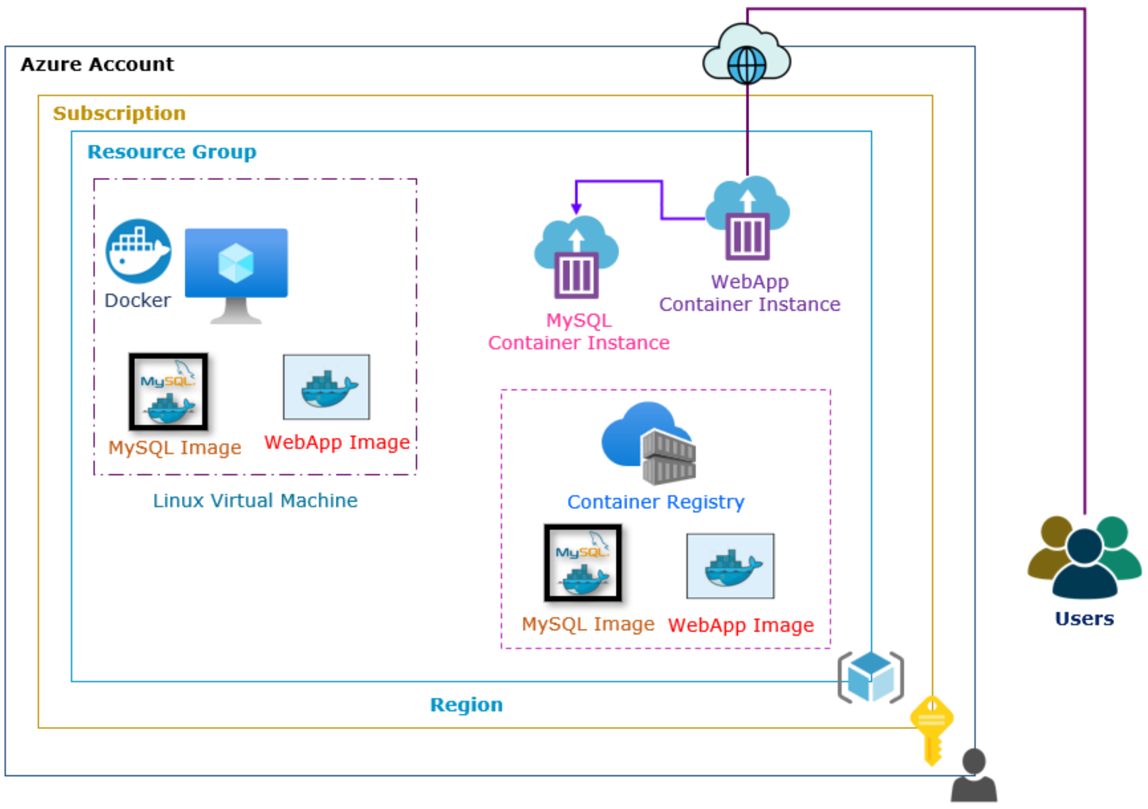
Version Control	
Document	Deploy the Container in Azure as a Service
Owner	Ahmad Majeed Zahoory
Version	2.0
Last Change	28th July 2022
Description of Change	Task steps updated

Lab Duration: **45 minutes**

Lab scenario
Your organization is seeking a way to host a web application in Container. As a proof of concept, you have decided to try creating containers from built-in images and customise images. To keep your proof of concept simple, you'll create application written that you'll deploy to your container. Your proof of concept will evaluate the Azure Container Instances.

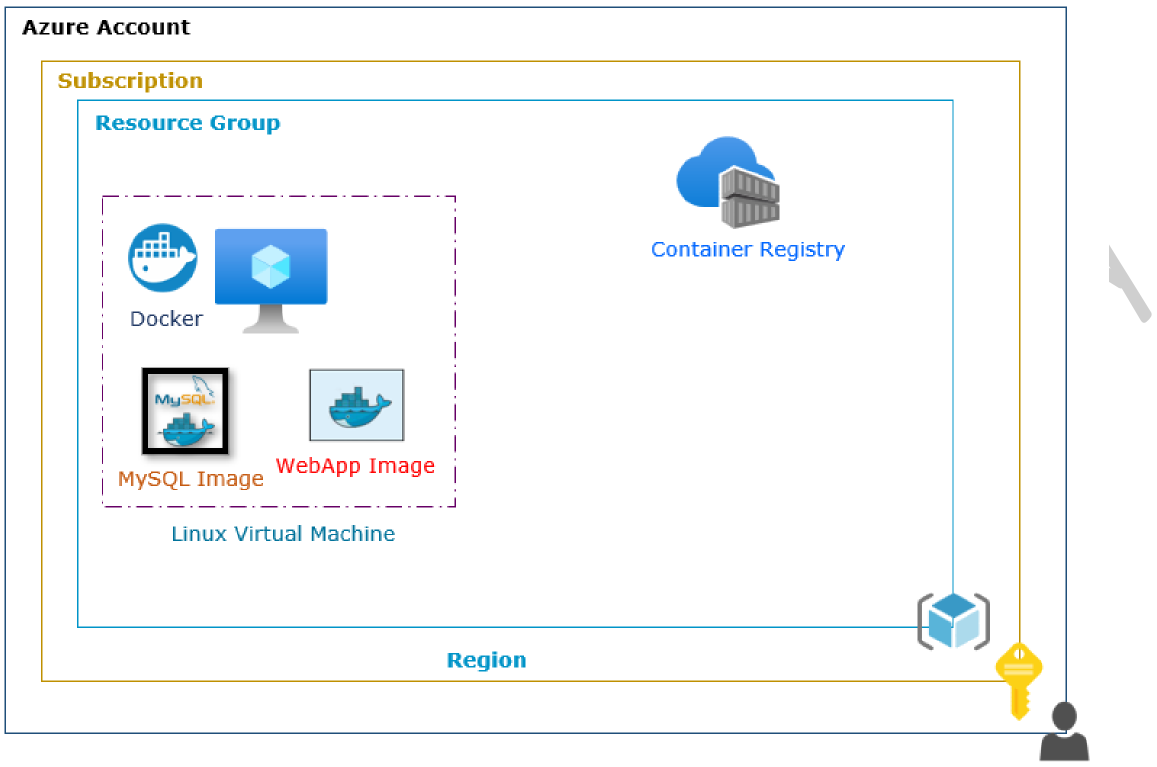
Objectives
After you complete this lab, you will be able to:

- Upload a docker container images to Azure Container Registry.
- Deploy a database container from a container image from ACR using Azure Container Instances.
- Deploy a web application container from a container image from ACR using Azure Container Instances.
- Access Web application.


Azure Account
Subscription
Resource Group
Docker
MySQL Image
WebApp Image
Linux Virtual Machine
MySQL
Container Instance
WebApp
Container Instance
Container Registry
MySQL Image
WebApp Image
Region
Users

Task 1: Create Azure Container Registry

In this task, you will create Container Registry.



Step 1: Create Azure Container Registry

1. **From** the **Azure portal**, ***go to the left***, select the **Create a resource**.
2. Search and select **Container Registry** from the list.
3. Select **Create** and configure:

    a. In the **Basic** page:

        i. **Subscription**: Select the **Default subscription**.

        ii. **Resource Group**: Select **Create new**:

            a) **Name**: Write **az-204-01-02-rg**.

            b) Select **Ok**.

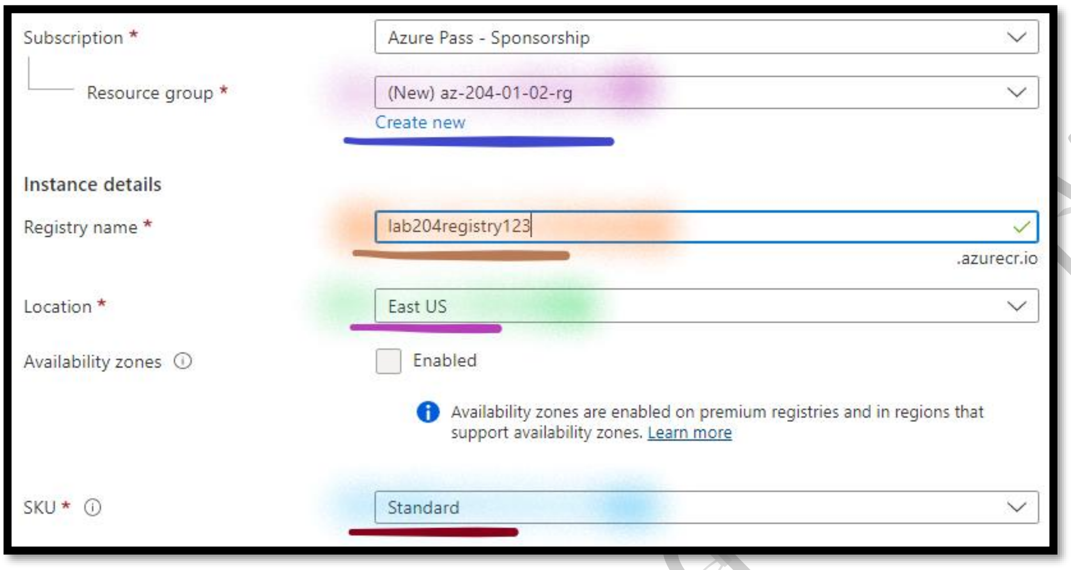
        iii. **Registry name:** Write **lab204registry123**.

Note: **Replace 123** to make registry name unique.

iv. **Location**: Select region **East US**.

v. **SKU**: Select **Standard**.

Note: Leave other details as default.



Subscription * — Azure Pass - Sponsorship

Resource group * — (New) az-204-01-02-rg

Create new

Instance details

Registry name * — lab204registry123 .azurecr.io

Location * — East US

Availability zones — ☐ Enabled

Availability zones are enabled on premium registries and in regions that support availability zones. Learn more

SKU * — Standard

vi. Select the **Next: Networking**.

b. In the **Networking** page:

i. Select **Next: Encryption**.

Note: Leave all the details as default.

c. In the **Encryption** page:

i. Select **Next: Tags**.

Note: Leave all the details as default.

d. In the **Tags** page:

i. Select **Next: Review + Create**.

Note: Leave all the details as default.

e. In the **Review + Create** page:

i. Select **Create**.

Note: **Wait** till **deployment** gets **completed**.

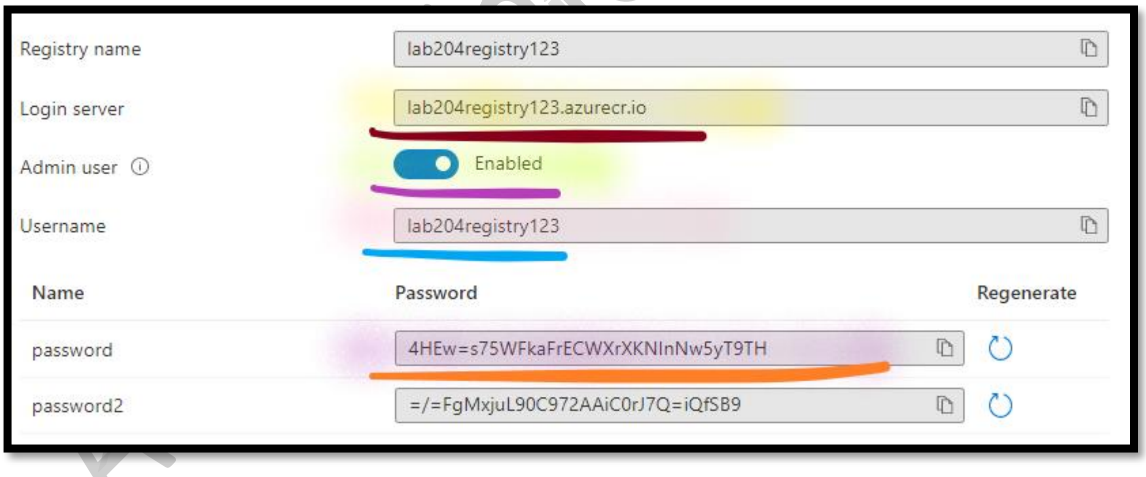
Step 2: Access the Azure Container Registry

4. **From** the **Azure portal**, ***go to the left***, select the **Resource group**.
5. Open the **az-204-01-02-rg** resource group.
6. Open the **lab204registry123** container registry.
   a. Select **Access keys** under **settings**.
      i. **Admin user**: Select the **Enable**.

Note: **Copy** the **Login server** name in the **Notepad**.

Note: **Copy** the **username** name in the **Notepad**.

Note: **Copy** the **password** in the **Notepad**.

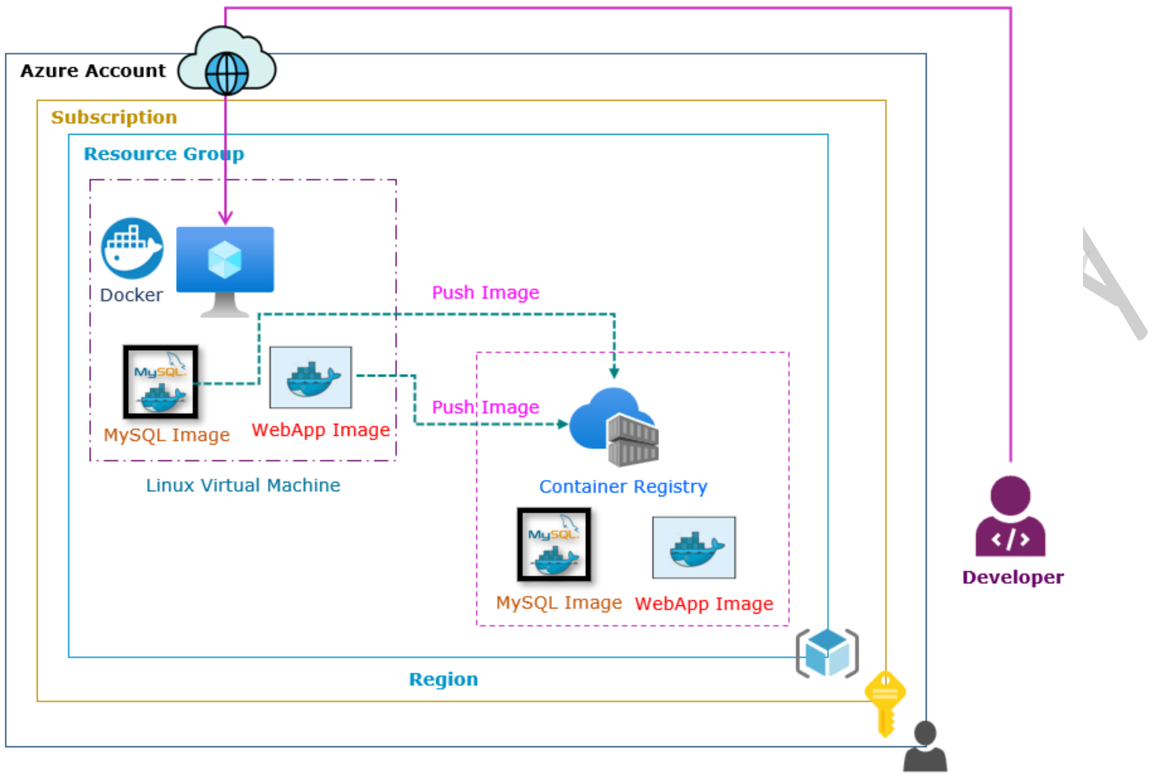


Registry name	lab204registry123	
Login server	lab204registry123.azurecr.io	
Admin user	Enabled	
Username	lab204registry123	

Name	Password	Regenerate
password	4HEw=s75WFkaFrECWXrXKNInNw5yT9TH	
password2	=/=FgMxjuL90C972AAiC0rJ7Q=iQfSB9	

Task 2: Push the Images to the ACR

In this task, you will push the container images to the Container Registry.



Step 1: Tag the Images

Tag the WebApp Image

7. **Return** to the **DevDocker** instance.

8. **From** **DevDocker** instance **terminal**:

   a. **Execute** the ***below command***, to **tag** the **webapp-image**:

```
sudo docker tag webapp-image AZURE-REGISTRY-LOGIN-SERVER-NAME/az204webapp:v1
```

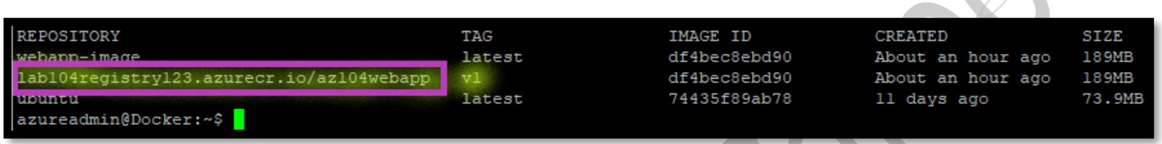
Note: **Replace** the **AZURE-REGISTRY-LOGIN-SERVER-NAME**, with the **Azure Container Registry** **Login Server name**, which you have copied in the previous step.

b. **Execute** the ***below command***, to **verify** the **image**:

```
sudo docker images
```

Note: You can see **Tagged webapp-image** image.

Note: **Copy** the **webapp** image **Repository name** in the **Notepad**.



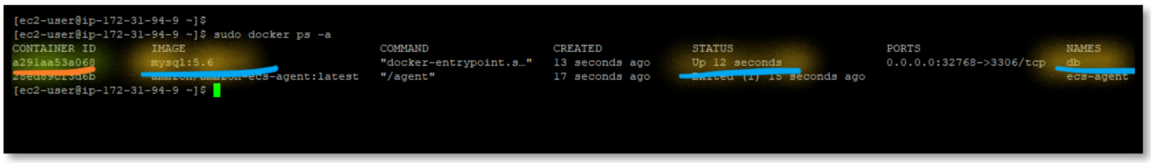
```
REPOSITORY                                   TAG          IMAGE ID       CREATED             SIZE
webapp-image                                 latest       df4bec8ebd90   About an hour ago   189MB
lab104registry123.azurecr.io/az104webapp     v1           df4bec8ebd90   About an hour ago   189MB
ubuntu                                       latest       74435f89ab78   11 days ago         73.9MB
azureadmin@Docker:~$
```

Tag the MySQL Image

c. **Execute** the following command, to **view** the **mysql container**:

```
sudo docker ps -a
```

Note: **Copy** the **db container Container ID** in the **Notepad**.



```
[ec2-user@ip-172-31-94-9 ~]$
[ec2-user@ip-172-31-94-9 ~]$ sudo docker ps -a
CONTAINER ID   IMAGE                             COMMAND                  CREATED          STATUS                      PORTS                     NAMES
a291aa53a068   mysql:5.6                         "docker-entrypoint.s…"   13 seconds ago   Up 12 seconds               0.0.0.0:32768->3306/tcp   db
28ed89c13d6b   amazon/amazon-ecs-agent:latest    "/agent"                 17 seconds ago   Exited (1) 15 seconds ago                             ecs-agent
[ec2-user@ip-172-31-94-9 ~]$
```

d. **Execute** the ***below command***, to **tag** the **mysql container**:

```
sudo docker commit DB-CONTAINER-ID AZURE-REGISTRY-LOGIN-SERVER-NAME/az204mysql:v1
```

Note: **Replace** the **AZURE-REGISTRY-LOGIN-SERVER-NAME**, with the **Azure Container Registry Login Server name**, which you have copied in the previous step.

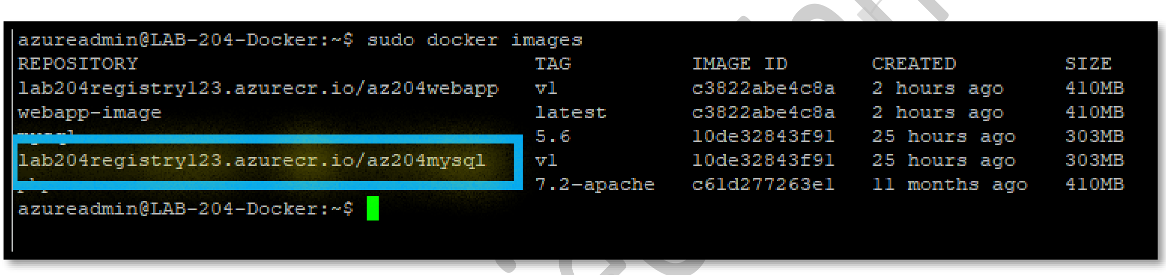
Note: **Replace** the **DB-CONTAINER-ID** with the **MySQL Container ID**, which you have copied in the previous step.

e. **Execute** the ***below command***, to **verify** the **image**:

```
sudo docker images
```

Note: You can see **Tagged mysql-image** image.

Note: **Copy** the **mysql** image **Repository name** in the **Notepad**.



```
azureadmin@LAB-204-Docker:~$ sudo docker images
REPOSITORY                                   TAG          IMAGE ID       CREATED          SIZE
lab204registry123.azurecr.io/az204webapp     v1           c3822abe4c8a   2 hours ago      410MB
webapp-image                                 latest       c3822abe4c8a   2 hours ago      410MB
mysql                                        5.6          10de32843f91   25 hours ago     303MB
lab204registry123.azurecr.io/az204mysql      v1           10de32843f91   25 hours ago     303MB
php                                          7.2-apache   c61d277263e1   11 months ago    410MB
azureadmin@LAB-204-Docker:~$
```

Step 2: Authenticate to the Azure Container Registry

9. **From** **DevDocker** instance **terminal**:

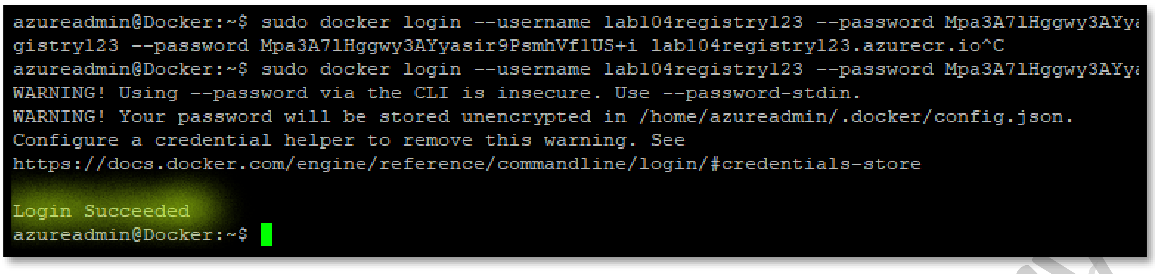
a. **Execute** the ***below command***, to **authenticate** the **docker client**:

```
sudo docker login --username USER-NAME --password PASSWORD AZURE-REGISTRY-LOGIN-SERVER-NAME
```

Note: **Replace** the **USER-NAME** and **PASSWORD** with the **Azure Container Registry** **Username** and **Password**, which you have copied in the previous step.

Note: **Replace** the **AZURE-REGISTRY-LOGIN-SERVER-NAME**, with the **Azure Container Registry** **Login Server name**, which you have copied in the previous step.

Note: If **authenticated succesfully**, you can see the **Login Succeded** message.



```
azureadmin@Docker:~$ sudo docker login --username lab104registry123 --password Mpa3A71Hggwy3AYya
gistry123 --password Mpa3A71Hggwy3AYyasir9PsmhVf1US+i lab104registry123.azurecr.io^C
azureadmin@Docker:~$ sudo docker login --username lab104registry123 --password Mpa3A71Hggwy3AYya
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /home/azureadmin/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
azureadmin@Docker:~$
```

Step 3: Push the Tag Images to the Azure Container Registry

10. **From** **DevDocker** instance **terminal**:

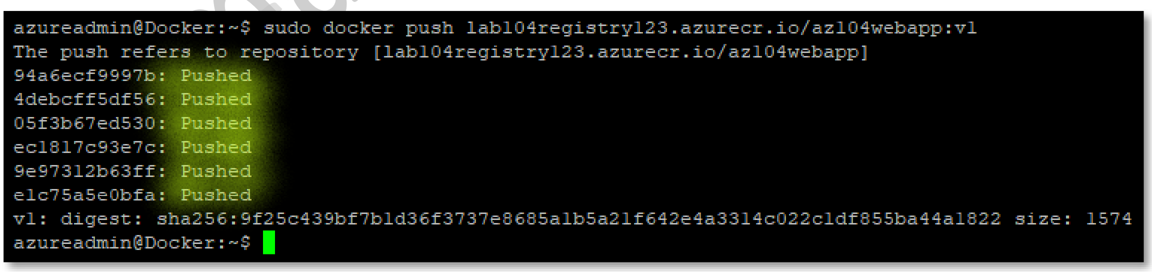
Push the WebApp Image

a. **Execute** the ***below command***, to **push** the **webapp image**:

```
sudo docker push TAG-REPOSITORY-NAME-WEBAPP:v1
```

Note: **Replace** the **TAG-REPOSITORY-NAME-WEBAPP**, with the **Tagged WebApp-Image**, which you have copied in the previous step.

Note: You can see the **Pushed** message, while pushing the image to ACR.



```
azureadmin@Docker:~$ sudo docker push lab104registry123.azurecr.io/az104webapp:v1
The push refers to repository [lab104registry123.azurecr.io/az104webapp]
94a6ecf9997b: Pushed
4debcff5df56: Pushed
05f3b67ed530: Pushed
ec1817c93e7c: Pushed
9e97312b63ff: Pushed
e1c75a5e0bfa: Pushed
v1: digest: sha256:9f25c439bf7b1d36f3737e8685a1b5a21f642e4a3314c022c1df855ba44a1822 size: 1574
azureadmin@Docker:~$
```

Push the MySQL Image

b. **Execute** the ***below command***, to **push** the **mysql image**:

```
sudo docker push TAG-REPOSITORY-NAME-MYSQL:v1
```

Note: **Replace** the **TAG-REPOSITORY-NAME-MYSQL**, with the **Tagged MySQL-Image**, which you have copied in the previous step.

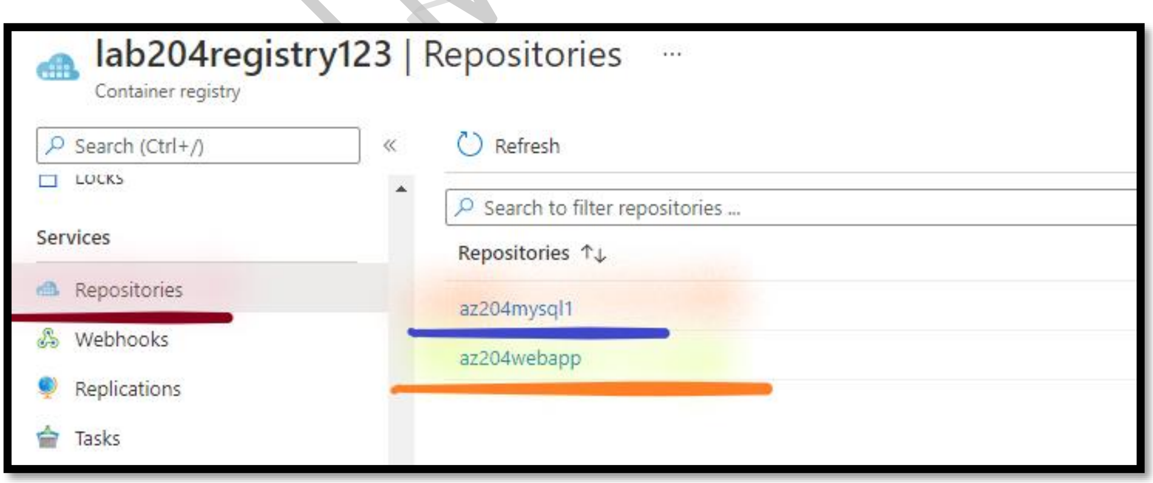
Note: You can see the **Pushed** message, while pushing the image to ACR.

Step 4: Verify the Uploaded Images from ACR

11. **From** the **Azure portal**, ***go to the left***, select the **Resource group**.
12. Open the **az-204-01-02-rg** resource group.
13. Open the **lab204registry123** container registry.

    a. Select **Repositories** under **Services**

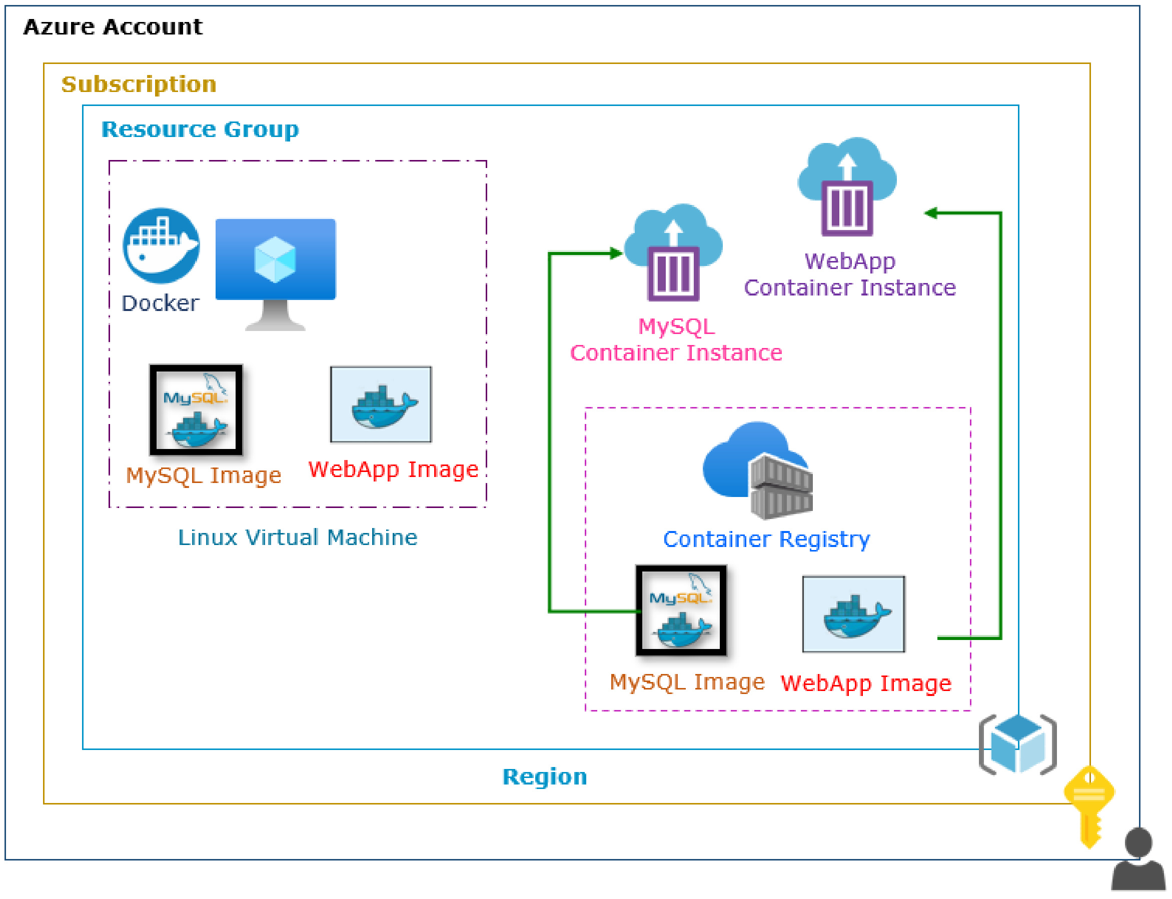
Note: You can see the **az204webapp** and **az204mysql** image, which was pushed from the Docker instance.



az204mysql1
az204webapp

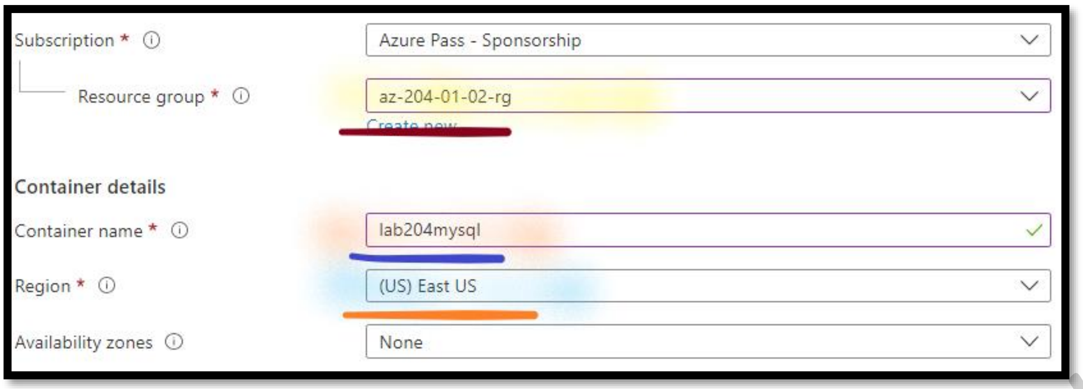
Task 3: Create the Container Instance

In this task, you will create Container Instances using the pushed the images from ACR.



Step 1: Create MySQL Container Instance

14. **From** the **Azure portal**, ***go to the left***, select the **Create a resource**.
15. Search and Select **Container Instances**.
16. Select **Create** and configure:
    a. In the **Basic** page:
        i. **Subscription**: Select **Default subscription**.
        i. **Resource group**: Dropdown and select **az-204-01-02-rg**.
        ii. **Container name**: Write **lab204mysql**.
        iii. **Region**: Dropdown and Select **East US**.


Subscription * Azure Pass - Sponsorship
Resource group * az-204-01-02-rg
Create new

Container details
Container name * lab204mysql
Region * (US) East US
Availability zones None

iv. **Image source**: Select **Azure Container Registry**.

v. **Registry**: Dropdown and Select **lab204registry123**.

vi. **Image**: Dropdown and Select **az204mysql**.

vii. **Image tag**: Dropdown and Select **v1**.

Note: Leave other details as default.

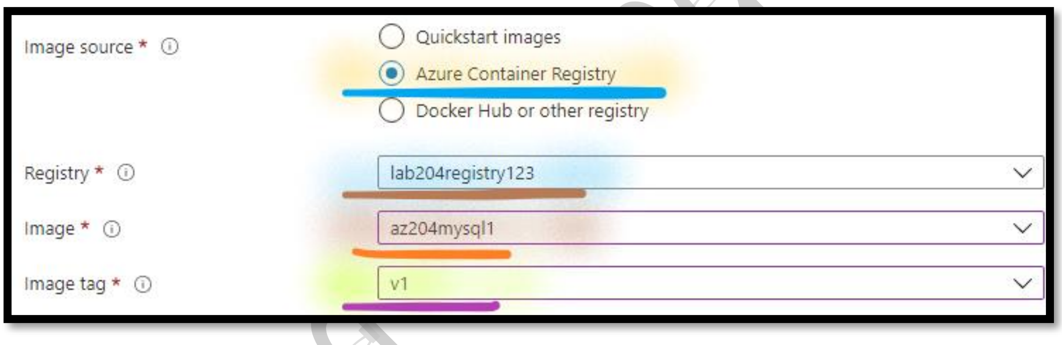


Image source * ( ) Quickstart images [✓ Azure Container Registry] ( ) Docker Hub or other registry
Registry * lab204registry123
Image * az204mysql1
Image tag * v1

viii. Select **Next: Networking**.

b. In the **Networking** page:

i. **Networking**: Select **Public**.

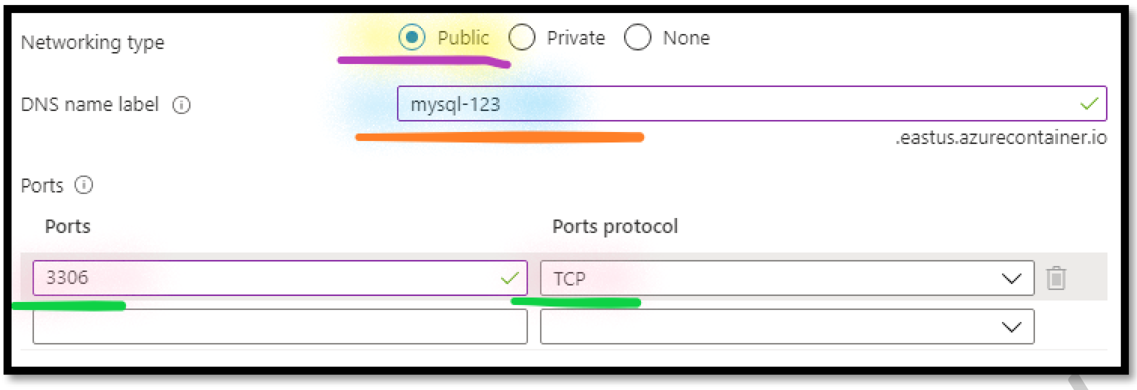
ii. **DNS label name**: Write **mysql-123**.

Note: **Replace 123** to make dns name unique.

iii. **Ports**: Write **3306**.

iv. **Ports protocol**: Dropdown and select **TCP**.

Note: Leave other details as default.



Networking type: [✓ Public] Private None

DNS name label: mysql-123 .eastus.azurecontainer.io

Ports

Ports	Ports protocol
3306	TCP

v. Select **Next: Advanced**.

c. In the **Advanced** page:

i. Select **Next: Tags**.

Note: Leave all the details as default.

d. In the **Tags** page:

i. Select **Next: Review + Create**.

Note: Leave all the details as default.

e. In the **Review + Create** page:

i. Select **Create**.

Note: **Wait** till **deployment** gets **completed**.

Step 2: Create WebApp Container Instance

17. **From** the **Azure portal**, ***go to the left***, select the **Create a resource**.
18. Search and select **Container Instances**.
19. Select **Create** and configure:

a. In the **Basic** page:

i. **Subscription**: Select **Default subscription**.

ii. **Resource group**: Dropdown and select **az-204-01-02-rg**.

iii. **Container name**: Write **lab204webapp**.

iv. **Region**: Dropdown and Select **East US**.

v. **Image source**: Select **Azure Container Registry**.

vi. **Registry**: Dropdown and select **lab204registry123**.

vii. **Image**: Dropdown and select **az204webapp**.

viii. **Image tag**: Dropdown and select **v1**.

Note: Leave other details as default.

ix. Select **Next: Networking**.

b. In the **Networking** page:

i. **Networking**: Select **Public**.

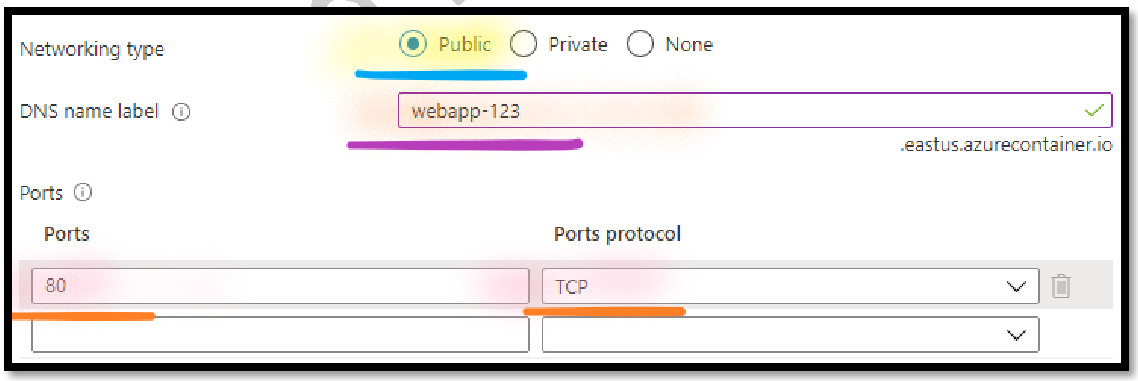
ii. **DNS label name**: Write **webapp-123**.

Note: **Replace 123** to make dns name unique.

iii. **Ports**: Write **80**.

iv. **Ports protocol**: Dropdown and select **TCP**.

Note: Leave other details as default.



Networking type [✓ Public] Private None

DNS name label: webapp-123 .eastus.azurecontainer.io

Ports

Ports	Ports protocol
80	TCP

v. Select **Next: Advanced**.

c. In the **Advanced** page:

i. Select **Next: Tags**.

Note: Leave all the details as default.

d. In the **Tags** page:

i. Select **Next: Review + Create**.

Note: Leave all the details as default.

e. In the **Review + Create** page:

i. Select **Create**.

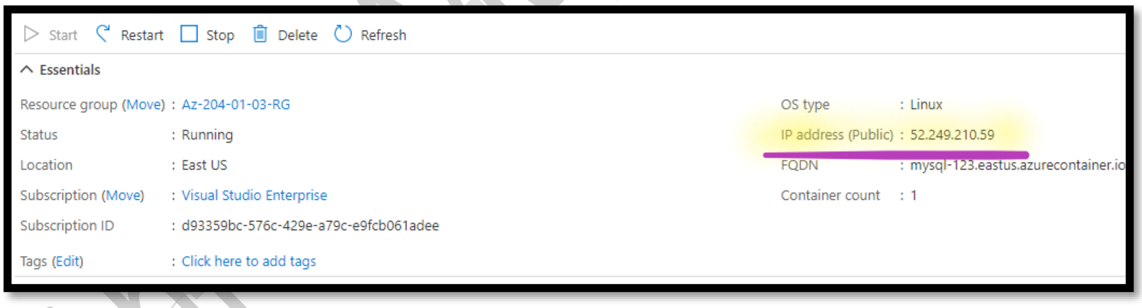
Note: **Wait** till **deployment** gets **completed**.

Step 3: Configure the MySQL Container Instance

20. **From** the **Azure portal**, ***go to the left***, select the **Resource group**.
21. Open the resource group **az-204-01-02-rg**.

a. Open the **lab204mysql** container instance.

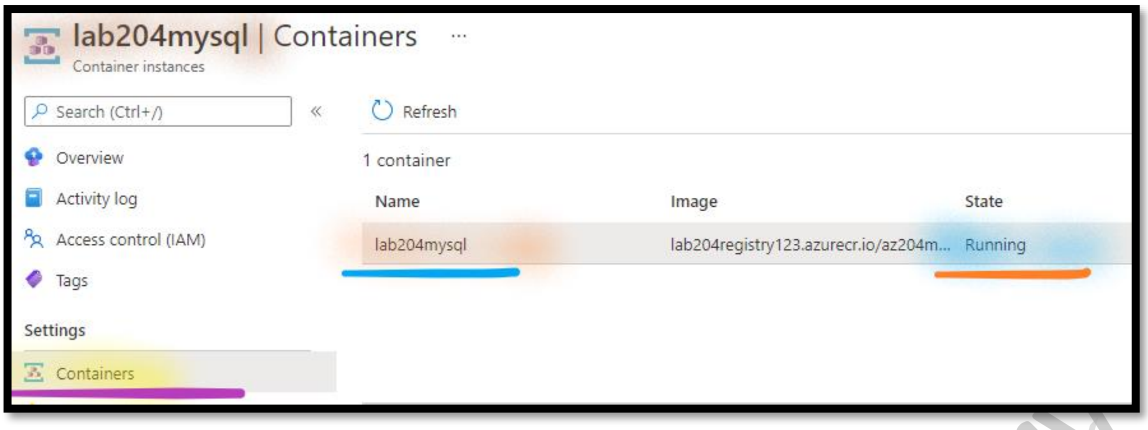
i. **Copy** the **Public IP Address** in the **Notepad**.



Resource group (Move) : Az-204-01-03-RG
Status : Running
Location : East US
Subscription (Move) : Visual Studio Enterprise
Subscription ID : d93359bc-576c-429e-a79c-e9fcb061adee
Tags (Edit) : Click here to add tags
OS type : Linux
IP address (Public) : 52.249.210.59
FQDN : mysql-123.eastus.azurecontainer.io
Container count : 1

ii. Select **Container** under **settings**.

Note: You can see the **MySQL Container** instance State as **Running**.



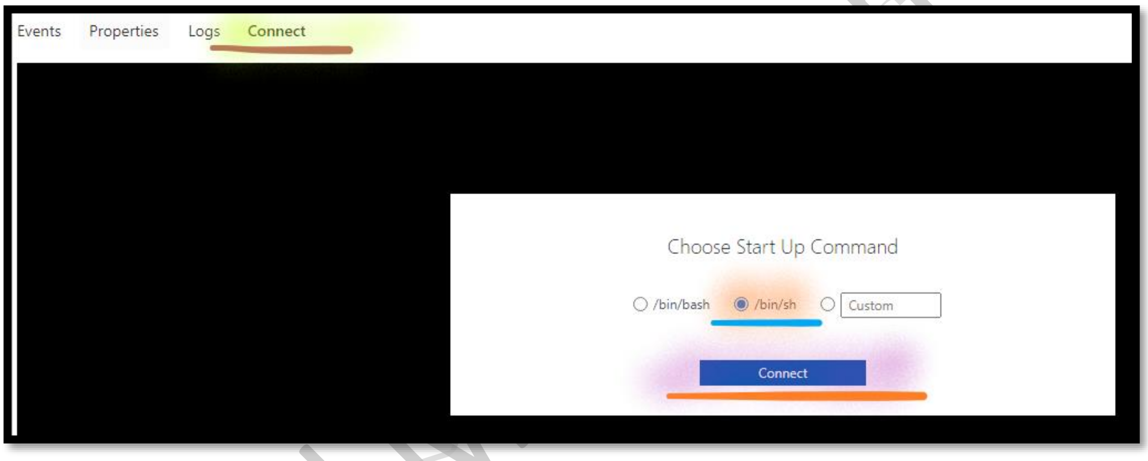
1 container

Name	Image	State
lab204mysql	lab204registry123.azurecr.io/az204m...	Running

a) Select **Connect**.

b) **Enable** the **Checkmark** against **/bin/sh**.

c) Select **Connect**.



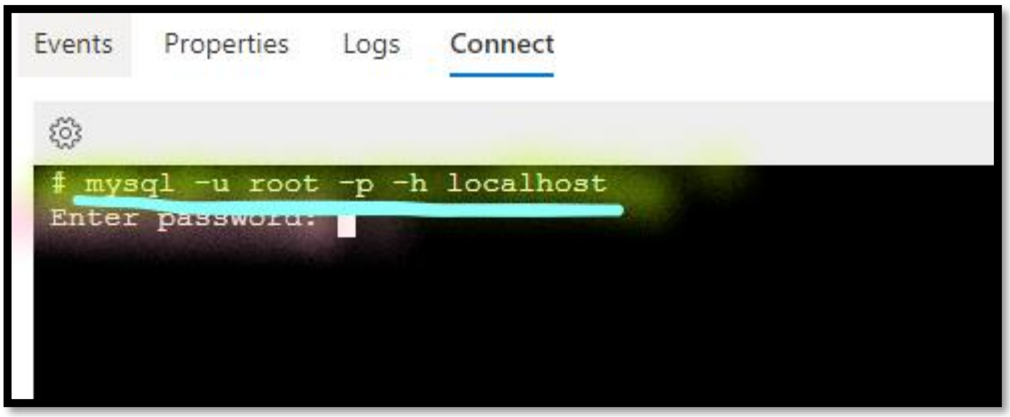
○ /bin/bash [✓ /bin/sh] ○ Custom

Note: You can see the **MySQL Container**, **prompt**.

b. **From** the **MySQL-Instance** **terminal**:

i. **Execute** the following command, to **connect** to the **MySQL**:

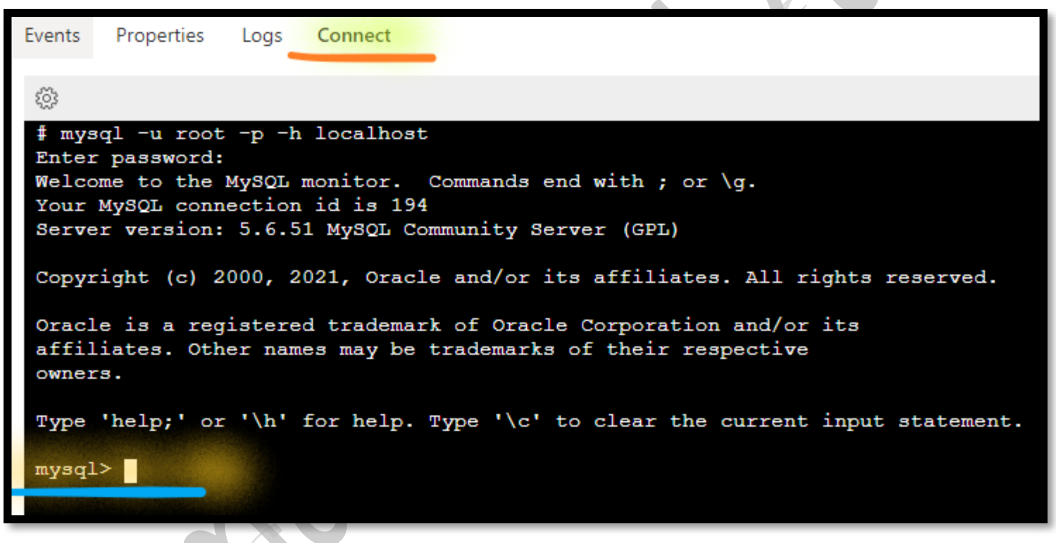
```
mysql -u root -p -h localhost
```



```
# mysql -u root -p -h localhost
Enter password:
```

a) When you **get prompt** to enter the **password**, write **password**.

Note: You can see the **MySQL Terminal**.



```
# mysql -u root -p -h localhost
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 194
Server version: 5.6.51 MySQL Community Server (GPL)

Copyright (c) 2000, 2021, Oracle and/or its affiliates. All rights reserved.

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

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

mysql>
```

ii. **From** the **MySQL terminal**:

a) **Execute** the following command, to **create database**, **prod_schema**:

```
create database prod_schema;
```

iii. **Execute** the following command, to **show databases**:

```
show databases;
```

Note: In the database, you can see the **prod_schema** database.

a) **Execute** the following command, to **use** the **prod_schema** database as the **default**:

```
use prod_schema;
```

Note: In the output, should show "**database changed**" message.

iv. **Execute** the following command, to **create table**, **products** with names of the **columns** and **datatypes**:

```
create table products (id int NOT NULL AUTO_INCREMENT, name
varchar(255), quantity varchar(255), price varchar(255), PRIMARY KEY (id));
```

Note: In the Output you can see "**Query OK, 0 rows affected**" message.

v. **Execute** the following command, to **show tables**:

```
show tables;
```

Note: In the tables, you can see the **products** table.

vi. **Execute** the following command, to **exit mysql**:

```
exit
```

Note: You can now see the **linux prompt**.

vii. **From** the **MySQL-Instance** **terminal**:

a) **Execute** the following command, to **exit**:

```
exit
```

b) **Press** the **Enter**, to ***close the connection***.

Step 4: Configure the WebApp Container Instance

22. **From** the **Azure portal**, ***go to the left***, select the **Resource group**.
23. Open the resource group **az-204-01-02-rg**.
    a. Open the **lab204webapp** container instance.
        i. Select **Container** under **settings**.

Note: You can see the **WebApp Container** instance State as **Running**.

a) Select **Connect**.
b) **Enable** the **Checkmark** against **/bin/sh**.
c) Select **Connect**.

Note: You can see the **WebApp Container**, **prompt**.

b. **From** the **WebApp-Instance** **terminal**:
    i. **Execute** the following command, to **Upgrade** the **Packages**:

```
apt-get -y upgrade
```

Note: **Wait**, for **deployment** gets **completed**.

ii. **Execute** the following command, to **Update** the **Packages**:

```
apt-get -y update
```

Note: **Wait**, for **deployment** gets **completed**.

iii. **Execute** the following command, to **Install** the **Sudo**:

```
apt-get install -y sudo
```

Note: **Wait**, for **deployment** gets **completed**.

iv. **Execute** the following command, to **Install** the **Nano**:

```
apt-get install -y nano
```

Note: **Wait**, for **deployment** gets **completed**.

c. **From** the **WebApp-Instance** **terminal**:

i. **Execute** the following command, to **list** the **file and folders**:

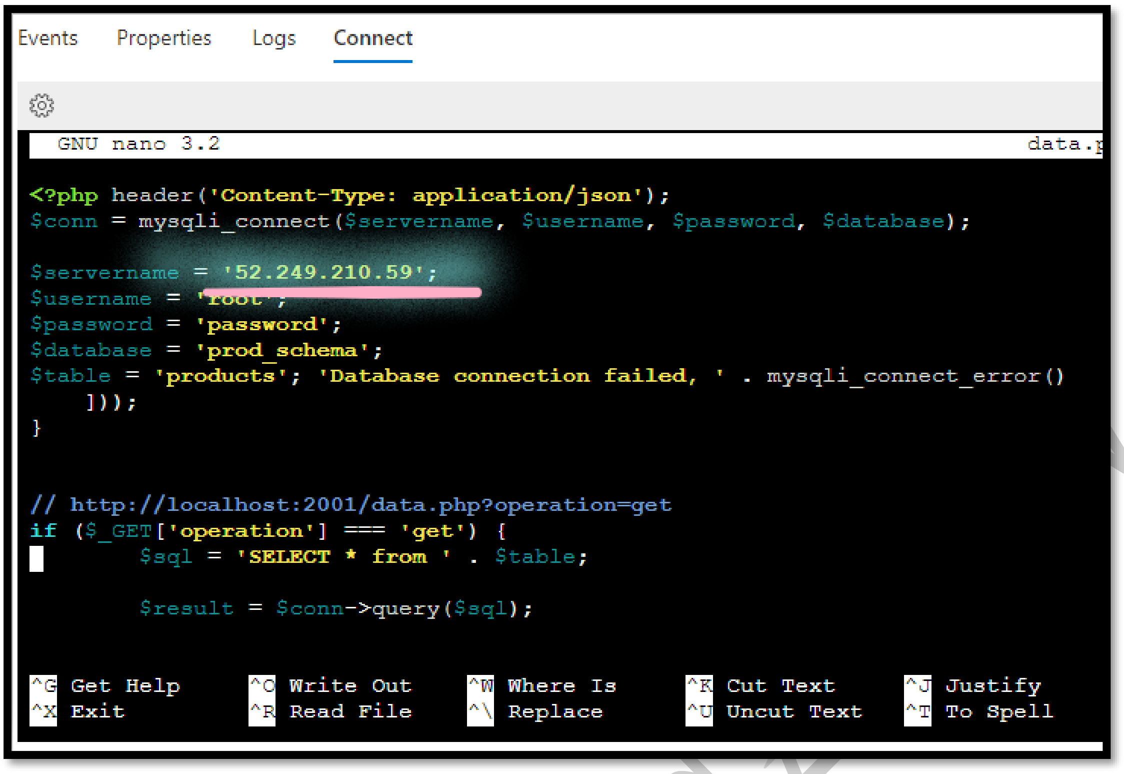
```
ls -l
```

Note: In the **Output**, you can see the **index.php** and **data.php** file.

ii. **Open** the **data.php** file in **Nano editor**:

```
sudo nano data.php
```

iii. **Update** the **Servername** with the ***MySQL-container*** **Public IP address**, which you have copied in the previous step.



```
  GNU nano 3.2                                                       data.p

<?php header('Content-Type: application/json');
$conn = mysqli_connect($servername, $username, $password, $database);

$servername = '52.249.210.59';
$username = 'root';
$password = 'password';
$database = 'prod_schema';
$table = 'products'; 'Database connection failed, ' . mysqli_connect_error()
    ]));
}


// http://localhost:2001/data.php?operation=get
if ($_GET['operation'] === 'get') {
        $sql = 'SELECT * from ' . $table;

        $result = $conn->query($sql);


^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell
```

1) Press **CTRL + O**, to ***save***.
2) Press **Enter**, ***key***.
3) Press **CTRL + X**, to ***exit***.

d. **From** the **MySQL-Instance** **terminal**:

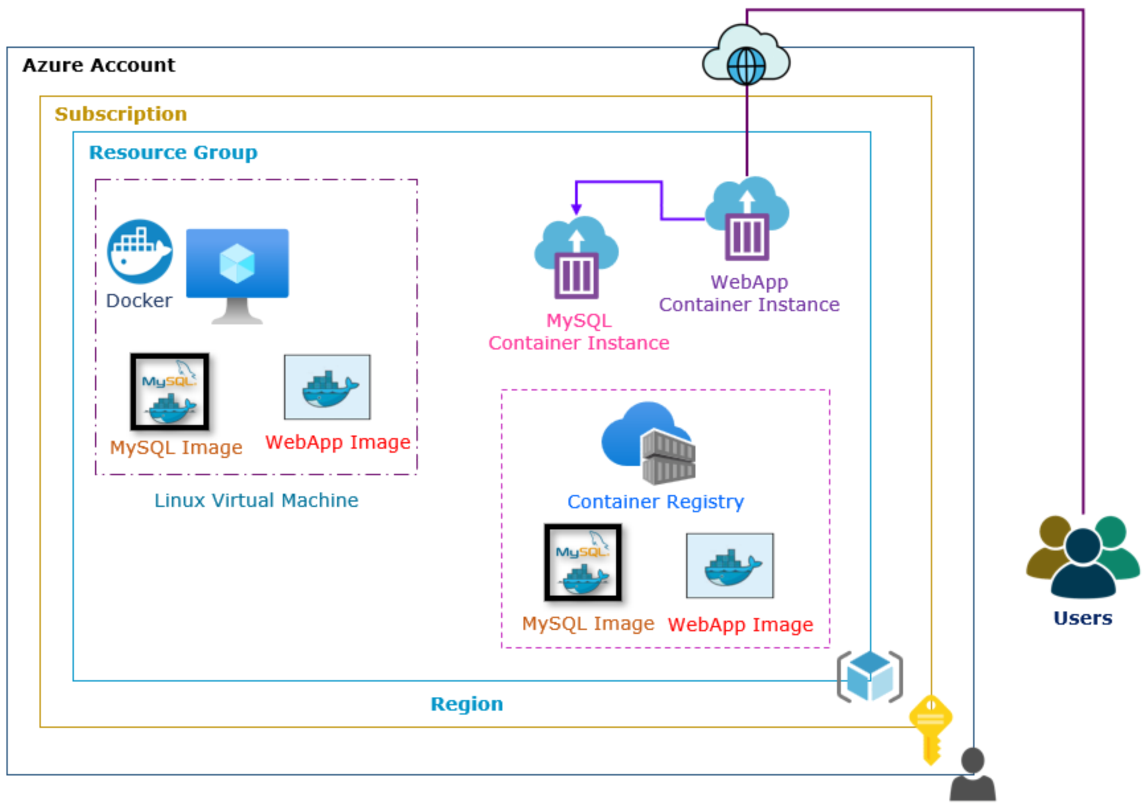
i. **Execute** the following command, to **exit**:

```
exit
```

i. **Press** the **Enter**, to ***close the connection***.

Task 4: Access Web Application

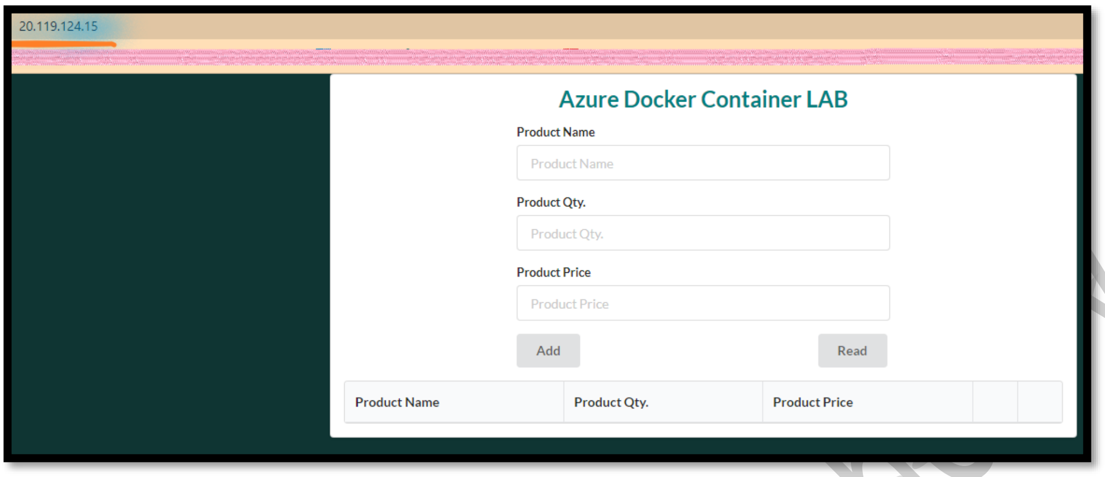
In this task, you will access web application.



Step 1: Access the WebApp

24. **From** the **Azure portal**, ***go to the left***, select the **Resource group**.
25. Open the resource group **az-204-01-02-rg**.
    a. Open the **lab204webapp** container instance.
        i. **Copy** the **Public IP Address** in the **Notepad**.

26. **From** your **Local Desktop/ Laptop**, open the **Browser**, write **Public IP Address** of the ***lab204webapp*** container instance, to access the **website**.



20.119.124.15

Azure Docker Container LAB

Product Name

Product Qty.

Product Price

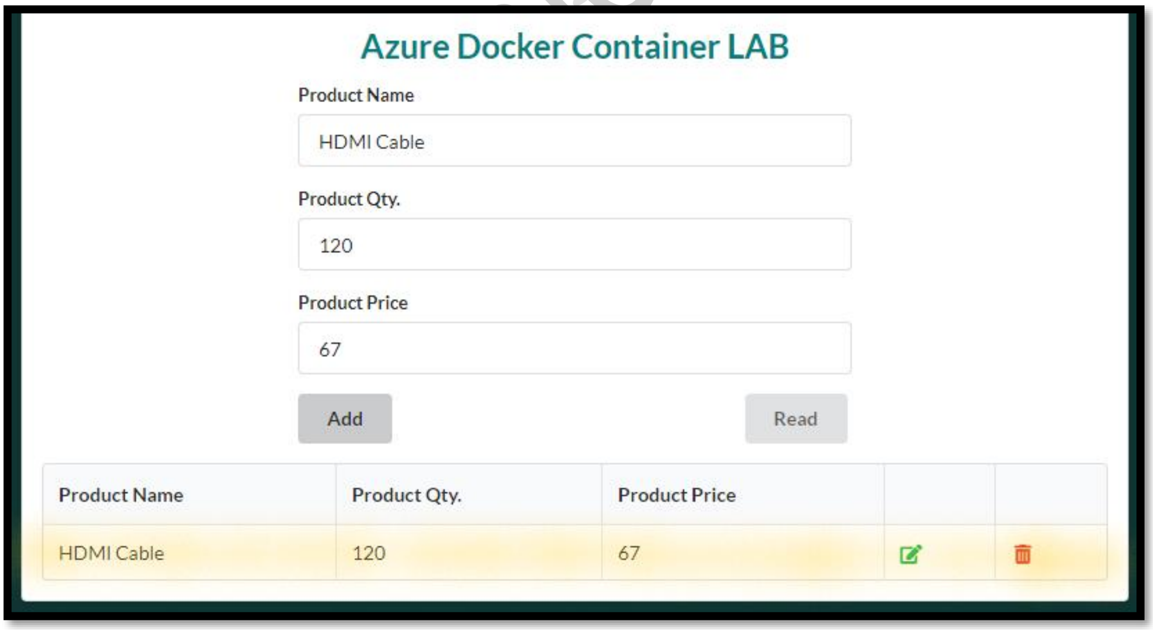
Product Name	Product Qty.	Product Price		

a. **Perform** the **CRUD operation**:

i. You can **Add** the ***Product Data***.

ii. You can **Update** the ***Product Data***.

iii. You can **Delete** the ***Product Data***.



Azure Docker Container LAB

Product Name: HDMI Cable

Product Qty.: 120

Product Price: 67

Product Name	Product Qty.	Product Price		
HDMI Cable	120	67		

Task 5: Delete the Environment

Step 1: Delete the Resource Group

27. ***Go to the left***, select **Resource group**.
    a. Select the **az-204-01-02-rg** resource group.
    b. Select the **Delete Resource Group**.
        i. Write the **az-204-01-02-RG** resource group.
        ii. Select the **Delete**.