

Docker assignment

Test:-1

1. Pull Ubuntu container
2. Run this container and map port 80 on the local
3. Install Apache2 on this container
4. Check if you are able to access the Apache page on your browser

Answer:

First docker to pull an ubuntu

- Docker pull ubuntu

Then we have to run container with given port 80

- Docker run -d -p 80:80 -- ubuntu -apache ubuntu

Let install the Apache2 inside the container

- Docker exec -it ubuntu-apache /bin/bash

Install apache command to used

- apt-get update
- apt-get install apache2
- bash : systemctl status apache2

let check out server

- docker exec -it ubuntu-apache systemctl status apache2

check out the browser `http://local host given port` and run it

The screenshot shows the AWS Management Console interface. At the top, there's a navigation bar with the AWS logo, 'Services' dropdown, and search bar. Below it, a list of services is visible: VPC, EC2, Amazon Pinpoint, IAM, S3, CloudWatch, Route 53, RDS, and CloudFormation. The main area displays a terminal window for an Ubuntu instance with IP 172.31.11.100. The terminal shows a series of commands for installing Jenkins, including downloading a key, adding a repository, updating apt, and installing Jenkins and Java. A notification banner at the bottom of the terminal window provides the instance ID 'i-08655f5dfd9055307 (docker assignment)' and IP addresses: PublicIPs: 65.1.112.34 and PrivateIPs: 172.31.11.100.

```
2 curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee
do tee /usr/share/keyrings/jenkins-keyring.asc > /dev/null
3 echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://p
kg.jenkins.io/debian-stable binary/ | sudo tee /etc/apt/sources.list.d/jenkin
s.list > /dev/null
4 sudo apt-get install jenkins
5 sudo apt-get update
6 curl -fsSL https://pkg.jenkins.io/debian/jenkins.io-2023.key | sudo tee
/usr/share/keyrings/jenkins-keyring.asc > /dev/null
7 echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://p
kg.jenkins.io/debian binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list
> /dev/null
8 sudo apt-get update
9 sudo apt-get install jenkins
10 ls
11 which jenkins
12 jenkins --version
13 sudo apt-get install jdk-11-jdk
14 sudo apt-get update
15 sudo apt install default-jre
16 java -version
17 sudo apt-get update
18 sudo apt-get install apache2
19 http://local.server.ip
20 history
ubuntu@ip-172-31-11-100:~$
```


i-08655f5dfd9055307 (docker assignment) X

PublicIPs: 65.1.112.34 PrivateIPs: 172.31.11.100

Check it in ip address

← ↻ 🏠 ⚠ Not secure | 65.1.112.34 🔊 📖 ☆ 🗑 ...

🌱 Spring Initializr 🖨 Instances 🔍 ndhm.gov.in - Bing 🏠 Home | e-SHRAM 🇮🇳 https://www.revisitc... >



Apache2 Default Page

Ubuntu

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installing Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache package you can read this page, it means that the Apache HTTP server installed at this site is working properly. **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and scripts are optimized for interaction with Ubuntu tools. The configuration system is **fully documented in** [/usr/share/doc/apache2/README.Debian.gz](#). Refer to this for the full documentation. Documentation for the server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this system.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|   |-- ports.conf
|-- mods-enabled
|   |-- *.load
|   |-- *.conf
|-- conf-enabled
|   |-- *.conf
|-- sites-enabled
|   |-- *.conf
```

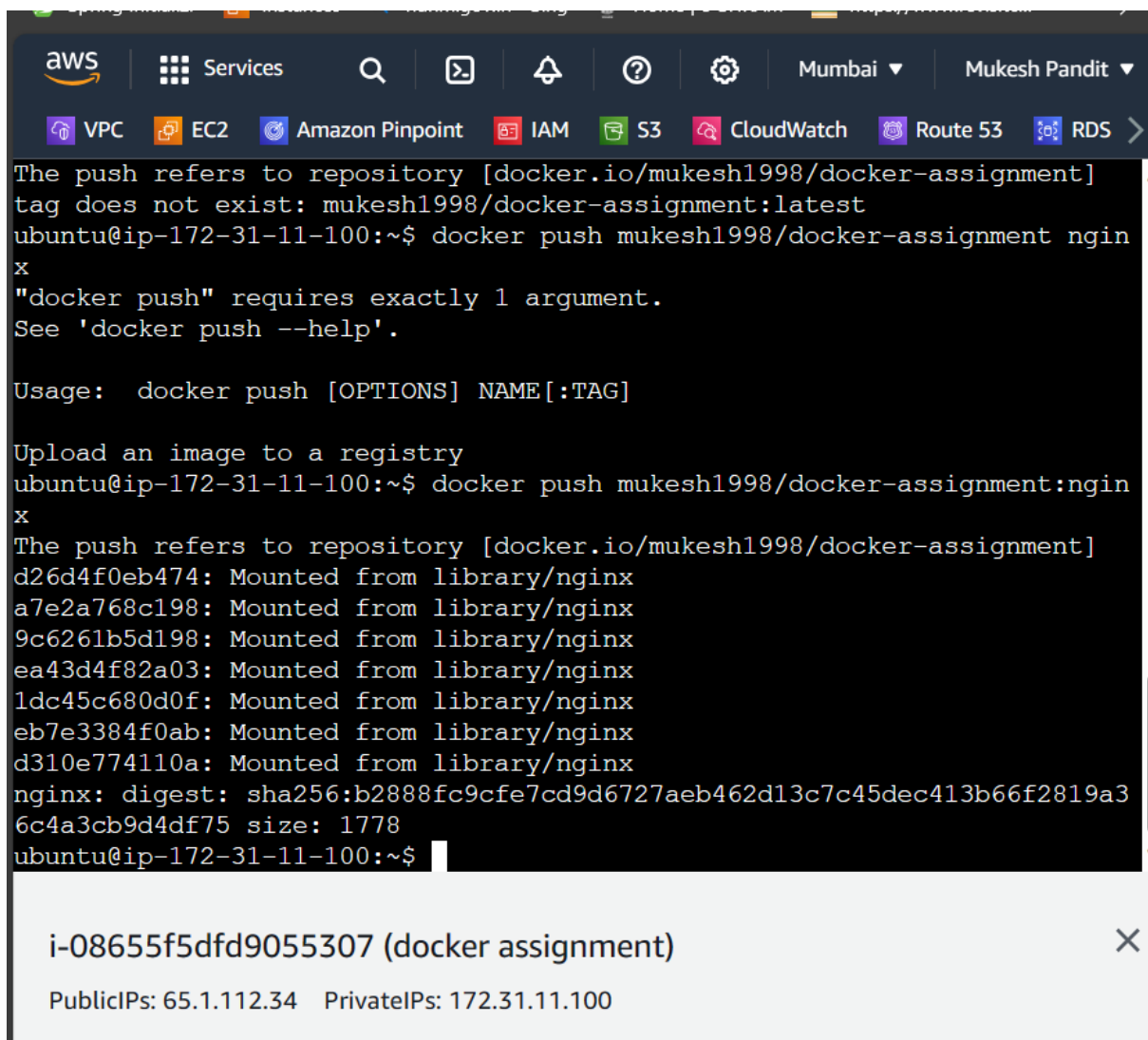
Test :- 2

Tasks To Be Performed:

1. Save the image created in assignment 1 as a Docker image
2. Launch container from this new image and map the port to 81
3. Go inside the container and start the Apache2 service
4. Check if you are able to access it on the browser

Create the docker Images with specify the images

New images to launch



```
aws
Services
Mumbai
Mukesh Pandit
VPC EC2 Amazon Pinpoint IAM S3 CloudWatch Route 53 RDS
The push refers to repository [docker.io/mukesh1998/docker-assignment]
tag does not exist: mukesh1998/docker-assignment:latest
ubuntu@ip-172-31-11-100:~$ docker push mukesh1998/docker-assignment nginx
x
"docker push" requires exactly 1 argument.
See 'docker push --help'.

Usage:  docker push [OPTIONS] NAME[:TAG]

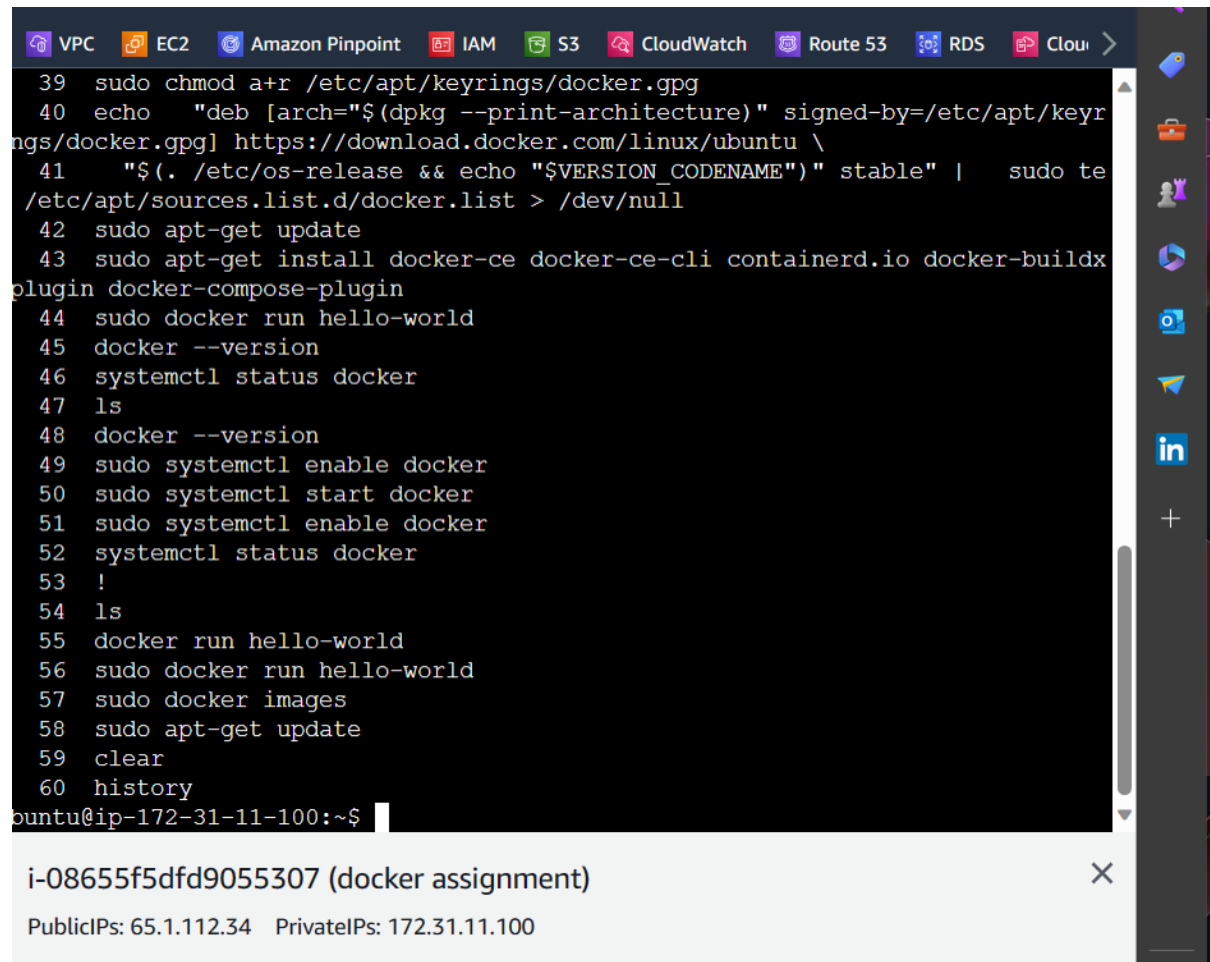
Upload an image to a registry
ubuntu@ip-172-31-11-100:~$ docker push mukesh1998/docker-assignment:nginx
x
The push refers to repository [docker.io/mukesh1998/docker-assignment]
d26d4f0eb474: Mounted from library/nginx
a7e2a768c198: Mounted from library/nginx
9c6261b5d198: Mounted from library/nginx
ea43d4f82a03: Mounted from library/nginx
1dc45c680d0f: Mounted from library/nginx
eb7e3384f0ab: Mounted from library/nginx
d310e774110a: Mounted from library/nginx
nginx: digest: sha256:b2888fc9cfe7cd9d6727aeb462d13c7c45dec413b66f2819a3
6c4a3cb9d4df75 size: 1778
ubuntu@ip-172-31-11-100:~$
```

i-08655f5dfd9055307 (docker assignment) X

PublicIPs: 65.1.112.34 PrivateIPs: 172.31.11.100

Install the apache 2

Then check out the imgs to create the docker images



```
39 sudo chmod a+r /etc/apt/keyrings/docker.gpg
40 echo "deb [arch=$(dpkg --print-architecture)] signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
41     '$(. /etc/os-release && echo "$VERSION_CODENAME")' stable" | sudo tee
/etc/apt/sources.list.d/docker.list > /dev/null
42 sudo apt-get update
43 sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx
plugin docker-compose-plugin
44 sudo docker run hello-world
45 docker --version
46 systemctl status docker
47 ls
48 docker --version
49 sudo systemctl enable docker
50 sudo systemctl start docker
51 sudo systemctl enable docker
52 systemctl status docker
53 !
54 ls
55 docker run hello-world
56 sudo docker run hello-world
57 sudo docker images
58 sudo apt-get update
59 clear
60 history
ubuntu@ip-172-31-11-100:~$
```

i-08655f5dfd9055307 (docker assignment) X

PublicIPs: 65.1.112.34 PrivateIPs: 172.31.11.100

Installing the docker

ls

- 48 docker --version
- 49 sudo systemctl enable docker
- 50 sudo systemctl start docker
- 51 sudo systemctl enable docker
- 52 systemctl status docker
- 53 !
- 54 ls
- 55 docker run hello-world
- 56 sudo docker run hello-world
- 57 sudo docker images

58 sudo apt-get update

59 clear

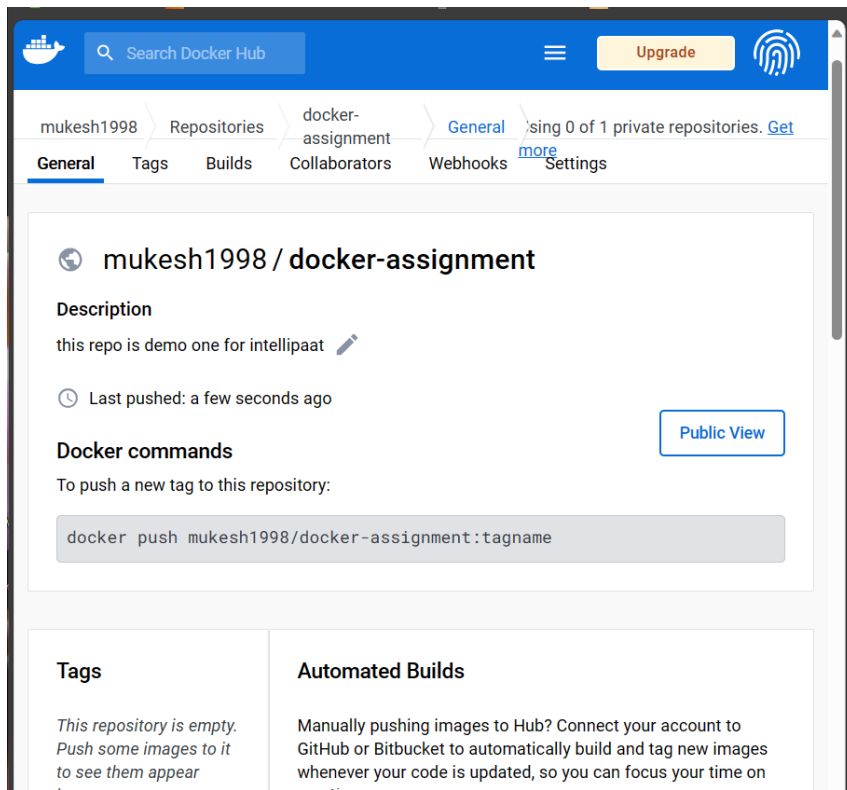
60 history

```
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:12:af:00:2e brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever
    inet6 fe80::42:12ff:feaf:2e/64 scope link
        valid_lft forever preferred_lft forever
ubuntu@ip-172-31-11-100:~$ docker run -d --name democontainernginx nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
a803e7c4b030: Pull complete
8b625c47d697: Pull complete
4d3239651a63: Pull complete
0f816efa513d: Pull complete
01d159b8db2f: Pull complete
5fb9a81470f3: Pull complete
9b1e1e7164db: Pull complete
Digest: sha256:32da30332506740a2f7c34d5dc70467b7f14ec67d912703568daff790ab3f755
Status: Downloaded newer image for nginx:latest
b33b55fe52913a5fe9ed7247c020b01683cbc2f5ab3e3bf828c7ab42203da300
ubuntu@ip-172-31-11-100:~$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS
PORTS         NAMES
b33b55fe5291   nginx     "/docker-entrypoint...." 14 seconds ago Up 13 seconds
80/tcp        democontainernginx
ubuntu@ip-172-31-11-100:~$
```

i-08655f5dfd9055307 (docker assignment)

PublicIPs: 65.1.112.34 PrivateIPs: 172.31.11.100

Pull the docker images to ubuntu



Push the docker to docker hub

aws

Services

Q

Mumbai ▾

Mukesh Pandit ▾

VPC

EC2

Amazon Pinpoint

IAM

S3

CloudWatch

Route 53

RDS

>

```
The push refers to repository [docker.io/mukesh1998/docker-assignment]
tag does not exist: mukesh1998/docker-assignment:latest
ubuntu@ip-172-31-11-100:~$ docker push mukesh1998/docker-assignment nginx
x
"docker push" requires exactly 1 argument.
See 'docker push --help'.

Usage:  docker push [OPTIONS] NAME[:TAG]


Upload an image to a registry
ubuntu@ip-172-31-11-100:~$ docker push mukesh1998/docker-assignment:nginx
x
The push refers to repository [docker.io/mukesh1998/docker-assignment]
d26d4f0eb474: Mounted from library/nginx
a7e2a768c198: Mounted from library/nginx
9c6261b5d198: Mounted from library/nginx
ea43d4f82a03: Mounted from library/nginx
1dc45c680d0f: Mounted from library/nginx
eb7e3384f0ab: Mounted from library/nginx
d310e774110a: Mounted from library/nginx
nginx: digest: sha256:b2888fc9cfe7cd9d6727aeb462d13c7c45dec413b66f2819a3
6c4a3cb9d4df75 size: 1778
ubuntu@ip-172-31-11-100:~$
```

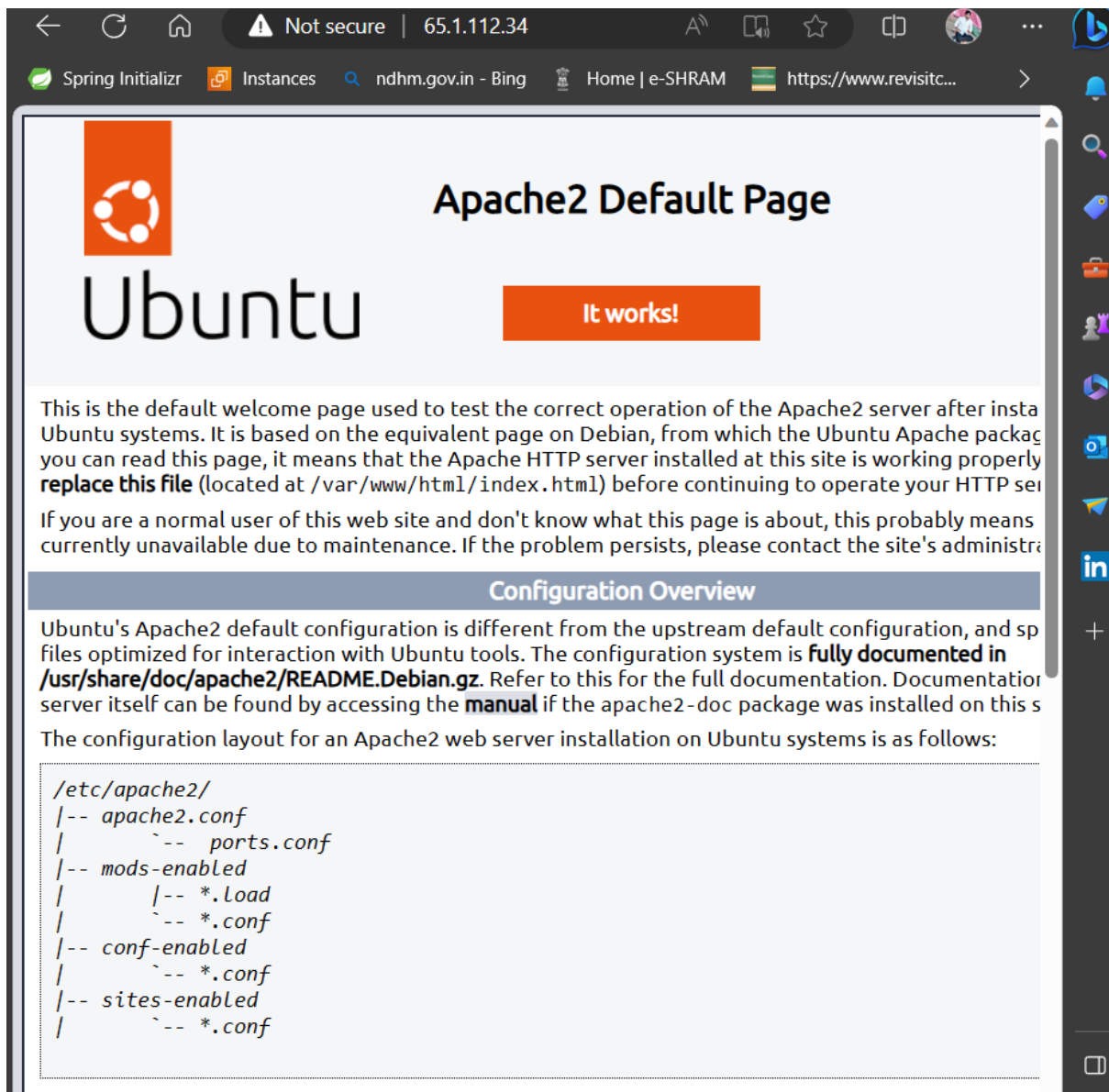
i-08655f5dfd9055307 (docker assignment)

PublicIPs: 65.1.112.34 PrivateIPs: 172.31.11.100

×

Available with Pro, Team and Business subscriptions. [Read more about automated builds](#) .

Available with Pro, Team and Business subscriptions. [Read more about automated builds](#) .



```
docker run -d -p nginx
```

```
docker ps
```

```
docker run -d -p nginx
```

```
docker run -d -p ubuntu
```

```
docker ps
```

```
netstat -an | grep -i 32768
```

```
docker port
```

Let go the images to ip address in the ip address

```
netstat -an | grep -i 9876
```

```
134 netstat -an | grep -i 4321
```

```
135 docker ps
```

```
136 docker port test
```

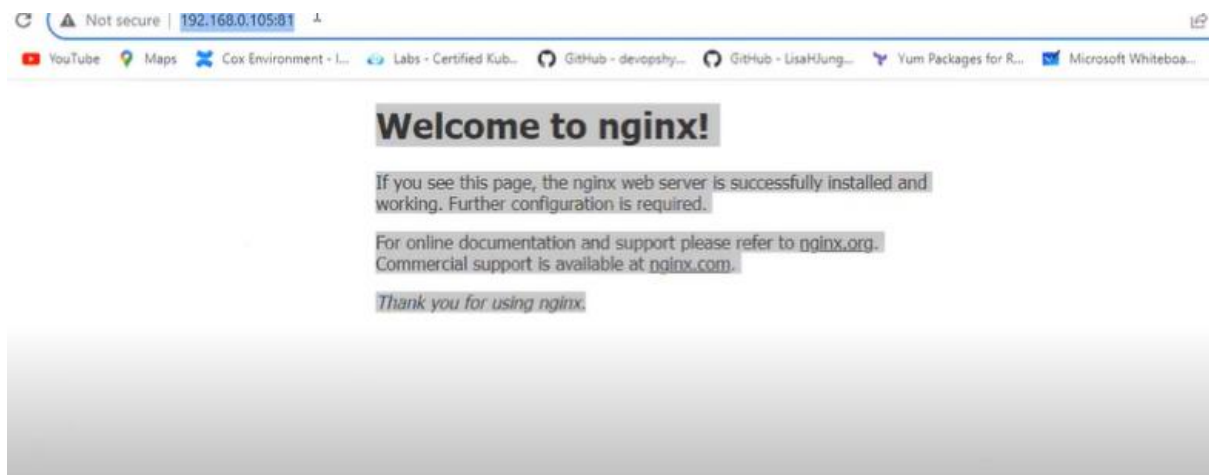
```
137 172-31-11-100
```

```
138 docker run -d -P
```

```
139 docker run -d -p 81:80 nginx
```

```
140 ip adr
```

```
141 ip addr
```



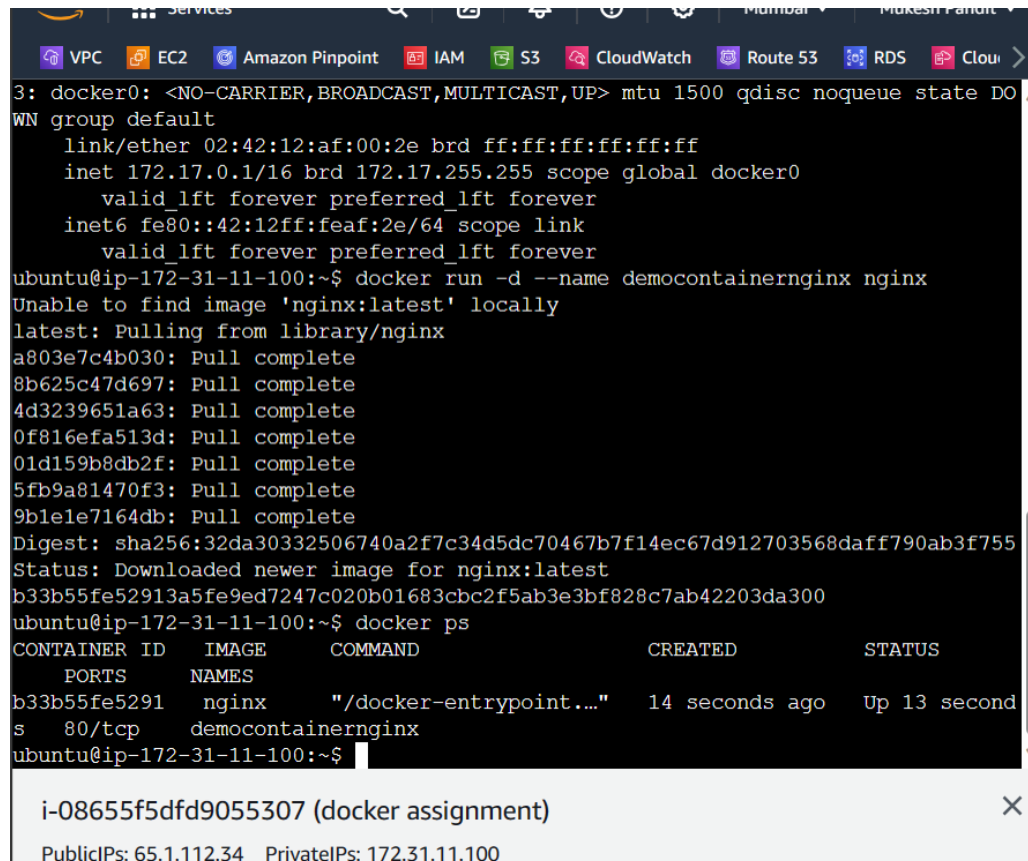
Test:-3

Tasks To Be Performed:

1. Use the saved image in the previous assignment
2. Upload this image on Docker Hub
3. On a separate machine pull this Docker Hub image and launch it on port 80
4. Start the Apache2 service
5. Verify if you are able to see the Apache2 service

First go to :

docker login

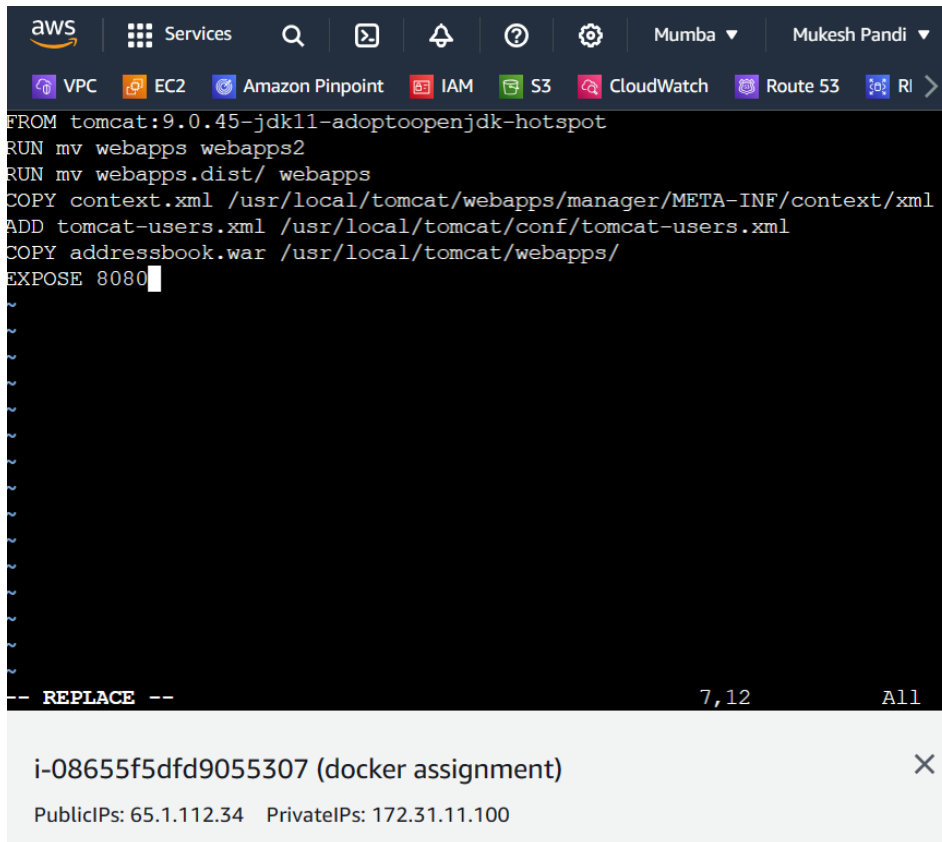


```
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DO
WN group default
    link/ether 02:42:12:af:00:2e brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever
    inet6 fe80::42:12ff:feaf:2e/64 scope link
        valid_lft forever preferred_lft forever
ubuntu@ip-172-31-11-100:~$ docker run -d --name democontainernginx nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
a803e7c4b030: Pull complete
8b625c47d697: Pull complete
4d3239651a63: Pull complete
0f816efa513d: Pull complete
01d159b8db2f: Pull complete
5fb9a81470f3: Pull complete
9b1e1e7164db: Pull complete
Digest: sha256:32da30332506740a2f7c34d5dc70467b7f14ec67d912703568daff790ab3f755
Status: Downloaded newer image for nginx:latest
b33b55fe52913a5fe9ed7247c020b01683cbc2f5ab3e3bf828c7ab42203da300
ubuntu@ip-172-31-11-100:~$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS
PORTS         NAMES
b33b55fe5291   nginx    "/docker-entrypoint. .... 14 seconds ago Up 13 second
s 80/tcp       democontainernginx
ubuntu@ip-172-31-11-100:~$
```

i-08655f5dfd9055307 (docker assignment) X

PublicIPs: 65.1.112.34 PrivateIPs: 172.31.11.100

docker tag ubuntu-apache:latest <your-docker-hub-username>/<your-repository-name>:latest



The screenshot shows the AWS IAM console interface. At the top, there's a navigation bar with the AWS logo, 'Services' dropdown, a search bar, and user information 'Mumba' and 'Mukesh Pandi'. Below the navigation bar, there's a list of services: VPC, EC2, Amazon Pinpoint, IAM, S3, CloudWatch, Route 53, and RI. The main content area displays a Docker assignment for an EC2 instance. The assignment is named 'i-08655f5dfd9055307 (docker assignment)'. It shows the public IP as 65.1.112.34 and the private IP as 172.31.11.100. The Docker command being executed is: FROM tomcat:9.0.45-jdk11-adoptopenjdk-hotspot, RUN mv webapps webapps2, RUN mv webapps.dist/ webapps, COPY context.xml /usr/local/tomcat/webapps/manager/META-INF/context.xml, ADD tomcat-users.xml /usr/local/tomcat/conf/tomcat-users.xml, COPY addressbook.war /usr/local/tomcat/webapps/, and EXPOSE 8080. At the bottom, there's a 'REPLACE' button and a '7,12 All' indicator.

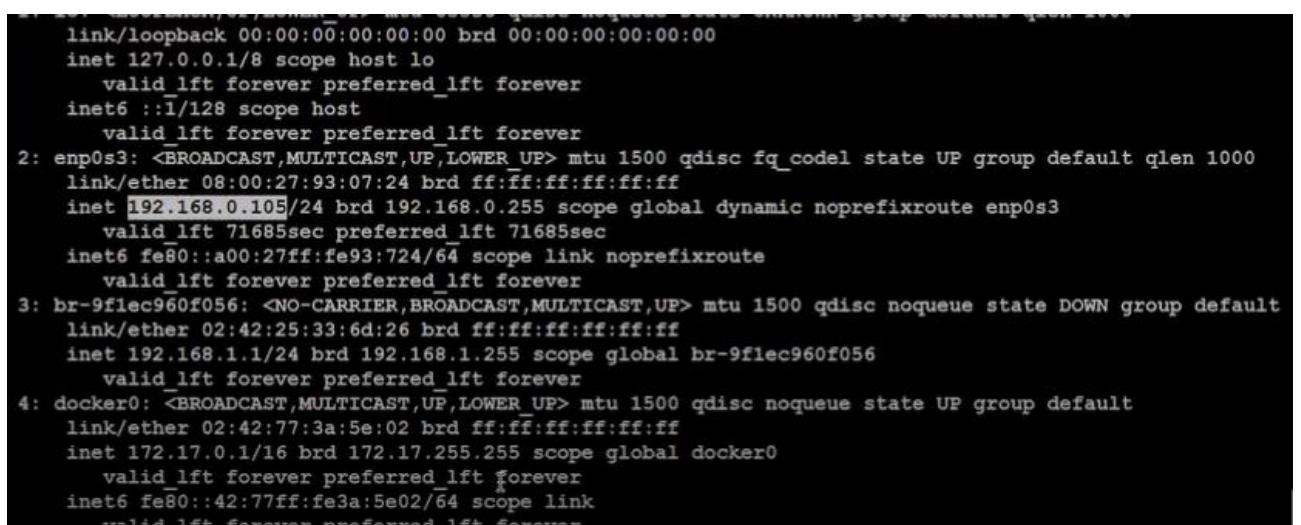
```
FROM tomcat:9.0.45-jdk11-adoptopenjdk-hotspot
RUN mv webapps webapps2
RUN mv webapps.dist/ webapps
COPY context.xml /usr/local/tomcat/webapps/manager/META-INF/context.xml
ADD tomcat-users.xml /usr/local/tomcat/conf/tomcat-users.xml
COPY addressbook.war /usr/local/tomcat/webapps/
EXPOSE 8080
```

-- REPLACE -- 7,12 All

i-08655f5dfd9055307 (docker assignment) X

PublicIPs: 65.1.112.34 PrivateIPs: 172.31.11.100

- docker push <your-docker-hub-username>/<your-repository-name>:latest
- docker run -d -p 8080:80 <your-docker-hub-username>/<your-repository-name>:latest



The screenshot shows a network configuration file with details for several interfaces. The interfaces are listed with their names, MAC addresses, IP addresses, and other configuration parameters. The interfaces are: link/loopback, enp0s3, br-9f1ec960f056, and docker0. The configuration includes details such as MTU, Qdisc, state, group, and link type. The configuration is as follows:

```
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
    valid_lft forever preferred_lft forever
inet6 ::1/128 scope host
    valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:93:07:24 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.105/24 brd 192.168.0.255 scope global dynamic noprefixroute enp0s3
        valid_lft 71685sec preferred_lft 71685sec
    inet6 fe80::a00:27ff:fe93:724/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: br-9f1ec960f056: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:25:33:6d:26 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.1/24 brd 192.168.1.255 scope global br-9f1ec960f056
        valid_lft forever preferred_lft forever
4: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:77:3a:5e:02 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever
    inet6 fe80::42:77ff:fe3a:5e02/64 scope link
        valid_lft forever preferred_lft forever
```


docker.file

<http://192.168.1.100:8080>

Home Documentation Configuration Examples Wiki Mailing Lists Find Help

Apache Tomcat/9.0.45

If you're seeing this, you've successfully installed Tomcat. Congratulations!



Recommended Reading:

- [Security Considerations How-To](#)
- [Manager Application How-To](#)
- [Clustering/Session Replication How-To](#)

Server Status
Manager App
Host Manager

Developer Quick Start

[Tomcat Setup](#) [First Web Application](#) [Realms & AAA](#) [JDBC DataSources](#) [Examples](#) [Servlet Specifications](#) [Tomcat Versions](#)

Managing Tomcat

For security, access to the [manager webapp](#) is restricted. Users are defined in:

```
$CATALINA_HOME/conf/tomcat-users.xml
```

In Tomcat 9.0 access to the manager application is split between different users. [Read more...](#)

Documentation

[Tomcat 9.0 Documentation](#)
[Tomcat 9.0 Configuration](#)
[Tomcat Wiki](#)

Find additional important configuration information in:

```
$CATALINA_HOME/README.txt
```

Getting Help

FAQ and Mailing Lists

The following mailing lists are available:

- [tomcat-announce](#)
Important announcements, releases, security vulnerability notifications. (Low volume)
- [tomcat-users](#)
User support and discussion

Verify the apache 2

<http://-ip:8080>

Now we are apache tomcat

My contacts...

First Name	Last Name	Email
George	Wong	george@wong.com
Daniel	Thompson	daniel@thompson.com
Emily	Jones	emily@jones.com
Peter	Wilson	peter@wilson.com
Dan	Robinson	dan@robinson.com
Dan	Davis	dan@davis.com
Olivia	Davis	olivia@davis.com
Dan	Smith	dan@smith.com
Daniel	Anderson	daniel@anderson.com
Alice	Thomas	alice@thomas.com
Frank	Thompson	frank@thompson.com
Daniel	Robinson	daniel@robinson.com

New Contact

Save Cancel

First name:

Last name:

Phone:

Email:

Birth date:

My contacts...			New contact
First Name	Last Name	Email	
Omar	Hussain	abobad999@gmail.com	
George	White	george@white.com	
Daniel	Thompson	daniel@thompson.com	
Timothy	Jones	timothy@jones.com	
Peter	Wilson	peter@wilson.com	
Dan	Robinson	dan@robinson.com	
Dan	Davis	dan@davis.com	
Olivia	Davis	olivia@davis.com	
Dan	Smith	dan@smith.com	
Daniel	Anderson	daniel@anderson.com	
Alice	Thomas	alice@thomas.com	
Linda	Harris	linda@harris.com	
Daniel	Robinson	daniel@robinson.com	
Mike	Young	mike@young.com	

Test: 4

Tasks To Be Performed:

1. Create a Dockerfile with the following specs:

- Ubuntu container
- Apache2 installed
- Apache2 should automatically run once the container starts

2. Submit the Dockerfile for assignment completion

Use the official Ubuntu as the base image

FROM ubuntu:latest

Update the package list and install Apache2

RUN apt-get update && \

apt-get install -y apache2

Start Apache2 when the container starts

CMD ["apachectl", "-D", "FOREGROUND"]

Expose port 80 for HTTP traffic

EXPOSE 80

```
docker run -d -p 8080:80 ubuntu-apache
```

```

root@af97cef0a16f: /var/www/html
[centos@localhost ~]$ docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED        STATUS
af97cef0a16f   customizednginx:1.1   "/bin/sh -c '/usr/sb..." About a minute ago Up 3 seconds
[centos@localhost ~]$ docker exec -it af97cef0a16f bash
root@af97cef0a16f:/var/www/html# ls -al
total 4
drwxr-xr-x. 1 root root 37 Sep  9 16:52 .
drwxr-xr-x. 1 root root 18 Sep  9 16:50 ..
-rw-r--r--. 1 root root 34 Sep  9 16:50 index.html
root@af97cef0a16f:/var/www/html#

```

```


inet 127.0.0.1/8 scope host lo
    valid_lft forever preferred_lft forever
inet6 ::1/128 scope host
    valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default c
    link/ether 08:00:27:93:07:24 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.105/24 brd 192.168.0.255 scope global dynamic noprefixroute enp0s3
        valid_lft 72507sec preferred_lft 72507sec
    inet6 fe80::a00:27ff:fe93:724/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: br-9f1ec960f056: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN gro
    link/ether 02:42:25:33:6d:26 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.1/24 brd 192.168.1.255 scope global br-9f1ec960f056
        valid_lft forever preferred_lft forever
4: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:77:3a:5e:02 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever
    inet6 fe80::42:77ff:fe3a:5e02/64 scope link
        valid_lft forever preferred_lft forever

```

docker run -d -p 8080:80 ubuntu-apache

← ↻ 🏠 ⚠️ Not secure | 65.1.112.34 🔊 📖 ☆ 🗑️ 👤

🌱 Spring Initializr 🖨️ Instances 🔍 ndhm.gov.in - Bing 🏠 Home | e-SHRAM 🇮🇳 https://www.revisitc... >



Apache2 Default Page

Ubuntu

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installing Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache package you can read this page, it means that the Apache HTTP server installed at this site is working properly. **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and scripts are optimized for interaction with Ubuntu tools. The configuration system is **fully documented in** [/usr/share/doc/apache2/README.Debian.gz](#). Refer to this for the full documentation. Documentation for the server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this system.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|   |-- ports.conf
|-- mods-enabled
|   |-- *.load
|   |-- *.conf
|-- conf-enabled
|   |-- *.conf
|-- sites-enabled
|   |-- *.conf
```

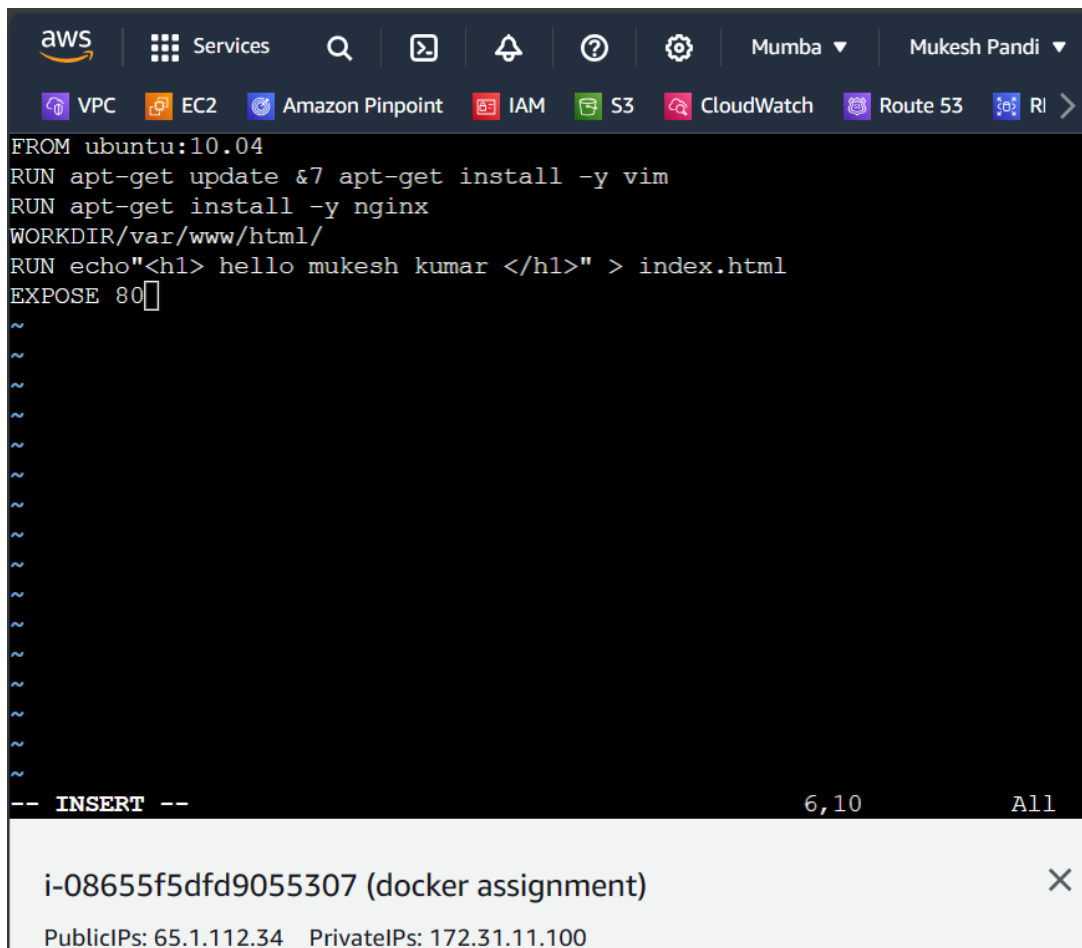
Test:5

Tasks To Be Performed:

1. Create a sample HTML file
2. Use the Dockerfile from the previous task
3. Replace this sample HTML file inside the Docker container with the default page

Assignment 1,2,3 are same and last step to do html file in docker container to creat in page
docker build -t my-web-app .

docker run -d -p 8080:80 my-web-app



```
FROM ubuntu:10.04
RUN apt-get update &7 apt-get install -y vim
RUN apt-get install -y nginx
WORKDIR /var/www/html/
RUN echo "<h1> hello mukesh kumar </h1>" > index.html
EXPOSE 80
```

-- INSERT -- 6,10 All

i-08655f5dfd9055307 (docker assignment) ×

PublicIPs: 65.1.112.34 PrivateIPs: 172.31.11.100

Loclhost:8080

PROJECT 1

Case study -containerization using docker

Part -1

DevOps Certification Training Problem Statement:

You work as a DevOps Engineer in a leading software company. You have been asked to Dockerize the applications on the production server. The company uses custom software. Therefore, there is no pre-built container which can be used.

Assume the following things:

1. Assume the software to be installed is Apache
2. Use an Ubuntu container

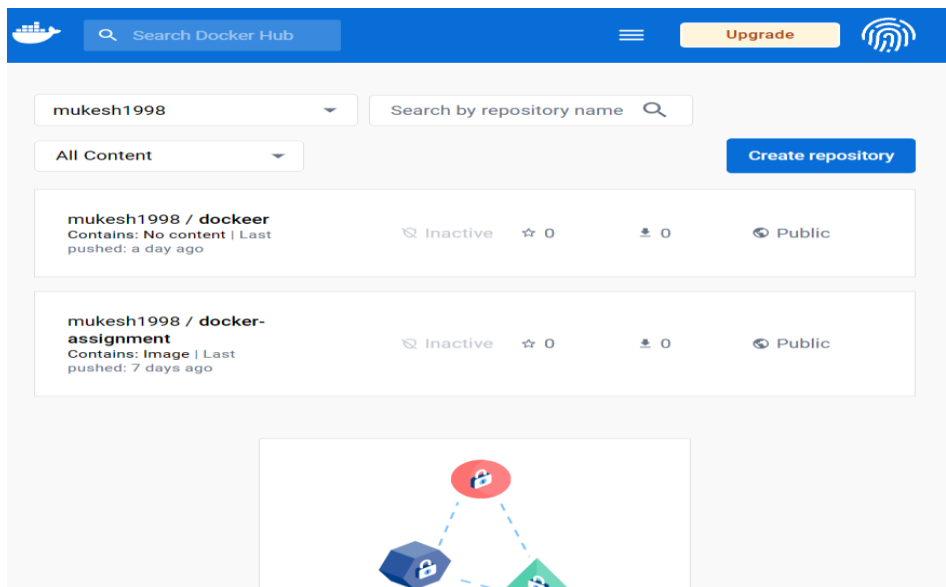
The company wants the following things:

1. Push a container to Docker Hub with the above config
2. The developers will not be working with Docker, hence from their side you will just get the code. Write a Dockerfile which could put the code in the custom image that you have built.

STEP:1

First create the docker installed on the system

Login to dockerhub account



Now go to aws creat the docker master and slave m/c

This command to customer apache application.

```
Enable ESM Apps to receive additional future security updates.  
See https://ubuntu.com/esm or run: sudo pro status  
[  
Y  
a Last login: Mon Oct 16 10:25:24 2023 from 13.233.177.4  
p ubuntu@ip-172-31-8-15:~$ history  
A  
1 1 ls  
2 2 sudo yum install docker  
1 3 apt-get install docker  
2 4 sudo yum update  
T 5 ls  
1 6 sudo apt update  
2 7 sudo apt install apt-transport-https ca-certificates curl sof  
W e-properties-common  
3 8 curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sud  
4 t-key add -  
5 9 sudo add-apt-repository "deb [arch=amd64] https://download.do  
6 .com/linux/ubuntu focal stable"  
7 10 apt-cache policy docker-ce  
8 11 sudo apt install docker-ce  
9 12 sudo systemctl status docker  
10 13 sudo usermod -aG docker ${USER}
```

i-0aa52e3387e76a421 (project1)

```
17 sudo usermod -aG docker username  
18 docker  
19 docker docker-subcommand --help  
20 docker info  
21 docker run hello-world  
22 docker search ubuntu  
23 sudo add-apt-repository "deb [arch=amd64] https://download.docker  
.com/linux/ubuntu focal stable"  
24 docker search ubuntu  
25 docker pull ubuntu  
26 sudo usermod -aG docker mukesh1998  
27 sudo usermod -aG docker ${mukesh1998; .; /; /.; />; <..; 1q; .  
28 history  
ubuntu@ip-172-31-8-15:~$
```

i-0aa52e3387e76a421 (project1) X

PublicIPs: 13.126.153.207 PrivateIPs: 172.31.8.15

Feedback Privacy Terms Cookie preferences

2 step:

Create the docker file with this command to use

```
ubuntu@ip-172-31-8-15:~$ vi dockerfile.sh
ubuntu@ip-172-31-8-15:~$
ubuntu@ip-172-31-8-15:~$ ls
dockerfile.sh
ubuntu@ip-172-31-8-15:~$
```

CREATE FILE WITH DOCKER FILE.SH

RUN apt-get update && apt-get install -y apache2

Then EXPOSE 80

CMD ["apache2ectl", "-d", "FOREGROUND"]

Step:3

Create the docker images in dockerfile

- Docker build -t custom=apache-image

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
b33b55fe5291	nginx	"/docker-entrypoint..."	14 seconds ago	Up 13 seconds
80/tcp	democontainernginx			

Create the directory for code

- Mkdir developer-code
- Ls

```
ubuntu@ip-172-31-8-15:~$ ls
developer-code  dockerfile.sh
```

```
ubuntu@ip-172-31-8-15:~$ docker build -t custom-apache-image .
[+] Building 0.1s (2/2) FINISHED
=> [internal] load build definition from Dockerfile 0.1s
=> => transferring dockerfile: 2B 0.0s
=> [internal] load .dockerignore 0.1s
=> => transferring context: 2B 0.0s
```

After words create a

- docker tag custom-apache-image your-docker-hub-mukesh1998/docker-apache:1.0


```

Processing triggers for ufw (0.36.1-4build1) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this
host.
ubuntu@ip-172-31-8-15:~$ http://local.service.ip
-bash: http://local.service.ip: No such file or directory
ubuntu@ip-172-31-8-15:~$ apache2 --version
[Mon Oct 16 11:03:21.639640 2023] [core:warn] [pid 12292] AH00111: C
g variable ${APACHE_RUN_DIR} is not defined

```

```

ubuntu@ip-172-31-8-15:~/dockerfile$ ls
ubuntu@ip-172-31-8-15:~/dockerfile$ docker pull mukesh1998/docker-assignment:nginx
nginx: Pulling from mukesh1998/docker-assignment:mukesh1998/docker-assignment:nginx
a803e7c4b030: Pull complete
8b625c47d697: Pull complete
4d3239651a63: Pull complete
0f816efa513d: Pull complete
01d159b8db2f: Pull complete
5fb9a81470f3: Pull complete
9b1e1e7164db: Pull complete
Digest: sha256:b2888fc9cfe7cd9d6727aeb462d13c7c45dec413b66f2819a36c4a3cb9d4df75
Status: Downloaded newer image for mukesh1998/docker-assignment:nginx
docker.io/mukesh1998/docker-assignment:nginx

```

So we have done Dockerized the custom apache application in docker hub we provide the docker image

Apache2 Default Page

Ubuntu **It works!**

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in** `/usr/share/doc/apache2/README.Debian.gz`. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```

/etc/apache2/
|-- apache2.conf
|   |-- ports.conf
|-- mods-enabled
|   |-- *.load
|   |-- *.conf
|-- conf-enabled
|   |-- *.conf
|-- sites-enabled
|   |-- *.conf

```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining

Module 3:

Docker implementation for website deployment

Tasks To Be Performed:

1. Fork the given repository and make your own github repository.
2. Design a Dockerfile to build an image with Nginx for hosting a website.
3. Implement commands in the Dockerfile to clone your github repository securely.
4. Configure the Nginx server within the Dockerfile to serve the cloned website.
5. Validate the Docker image by building and running a container to confirm successful image creation and website accessibility.

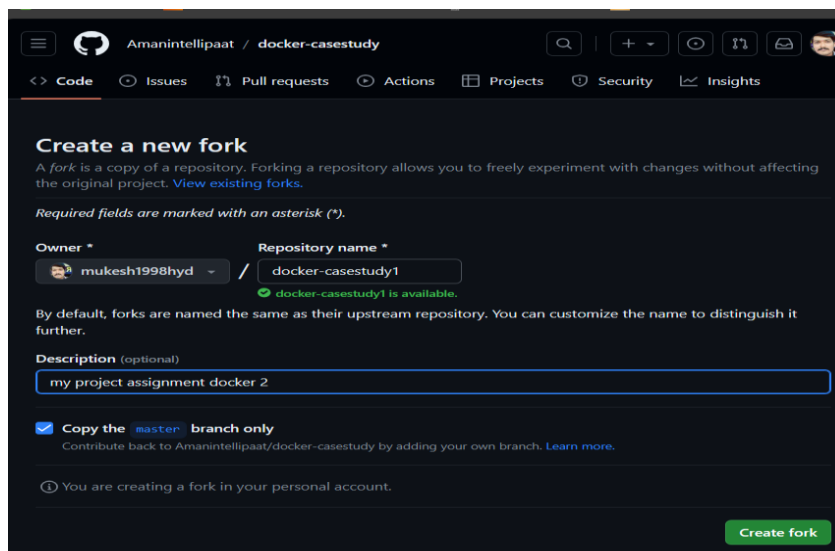
To submit the assignment, upload the screenshots of the executed steps and the Dockerfile to your GitHub repository .After that,share the repository with us.

Answer :

Step:1:

Open this link and fork the account :- : <https://github.com/Amanintellipaat/docker-casestudy>

First go to github create the fork to our project name .



Now install the latest nginx

```

ubuntu@ip-172-31-8-15:~$ ls
developer-code  dockerfile  dockerfile.sh
ubuntu@ip-172-31-8-15:~$ docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
a378f10b3218: Pull complete
4dfff0708538: Pull complete
2135e49ace4b: Pull complete
c843f6b280ce: Pull complete
6f35ab6f1400: Pull complete
6c538b49fa4a: Pull complete
d57731fb9008: Pull complete
Digest: sha256:b4af4f8b6470febf45dc10f564551af682a802eda1743055a7dfc8332dffa595
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest
ubuntu@ip-172-31-8-15:~$

```

Nginx latest version install docker file

```

ubuntu@ip-172-31-8-15:~$ cd docker-casestudy
ubuntu@ip-172-31-8-15:~/docker-casestudy$ ls
dist  index.html  src
ubuntu@ip-172-31-8-15:~/docker-casestudy$ index.html
index.html: command not found
ubuntu@ip-172-31-8-15:~/docker-casestudy$

```

Given link customer link with docker file and github account

```

ubuntu@ip-172-31-8-15:~/docker-casestudy$ sudo apt-get update && apt-get install -y git
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu focal InRelease
Get:5 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Fetched 110 kB in 1s (139 kB/s)
Reading package lists... Done

```

Docker vuilt the -t docker file

```

ee 'docker run --help'.
ubuntu@ip-172-31-8-15:~$ docker build -t docker-casestudy .
[+] Building 0.1s (2/2) FINISHED
                                docker:co
fault
=> [internal] load build definition from dockerfile
0.0s
=> => transferring dockerfile: 306B
0.0s
=> [internal] load .dockerignore
0.0s
=> => transferring context: 2B
0.0s
ERROR: failed to solve: failed to read dockerfile: read /var/lib/docker/
/var/lib/buildkit-mount671476311/dockerfile: is a directory

```

Create the container with nginx/html

Traffic with

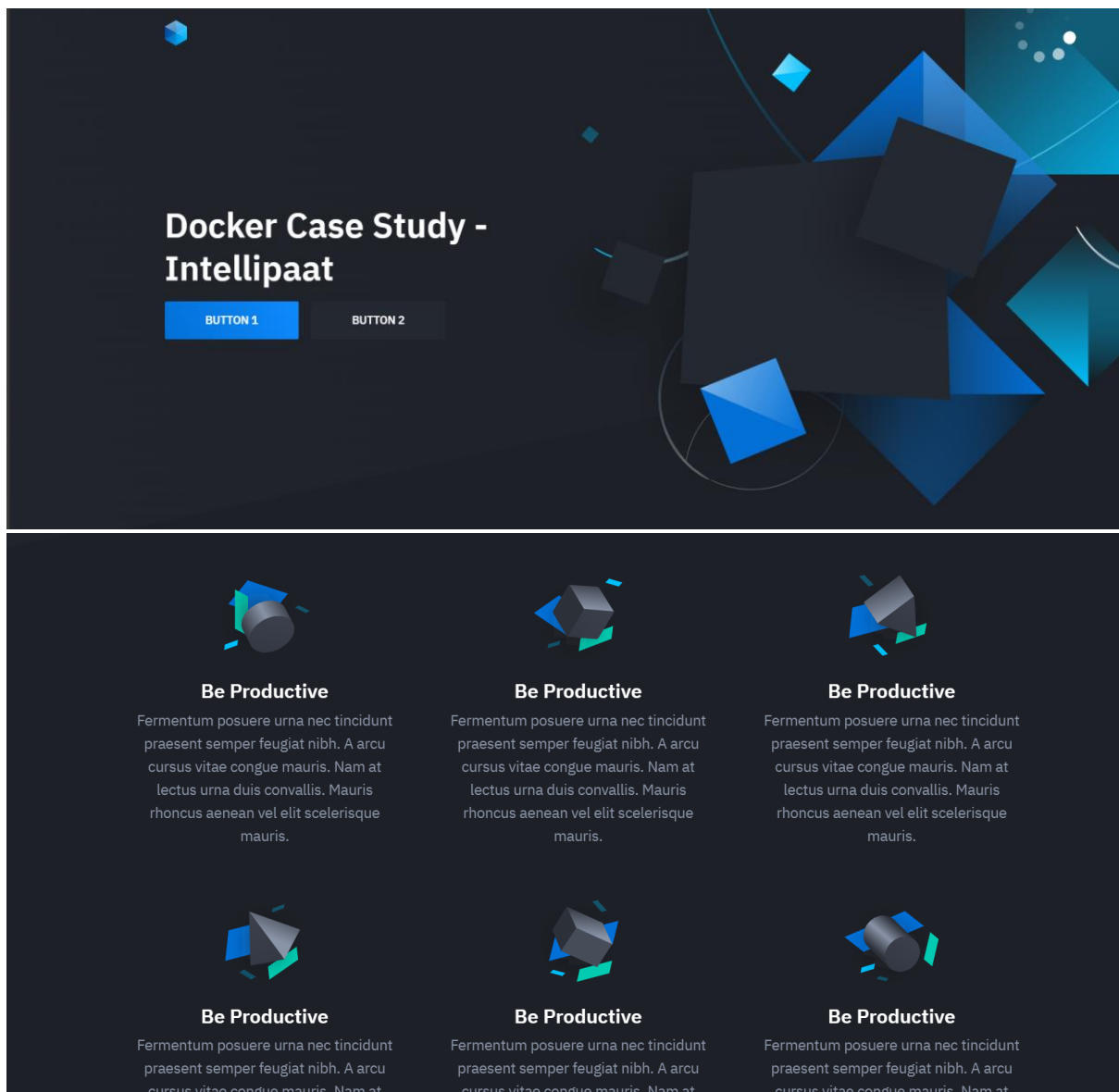
- EXPOSE 80

USE THE NGINX IMAGE BASE ON THE DOCKER ACCOUNT

BASE IMAGE

WHERE WE CAN SEE DOCKER FILE CRETE THEN LOCATION

So we can output the website



You are Done , You can leave this Page.

*****Completed the assignment*****