Capstone project documentation:

1. React application:

It is a web application built using React, which is a JavaScript library for building user interfaces. Developed and maintained by Facebook, React allows developers to create large web applications that can update and render efficiently in response to data changes.

React applications are known for their performance, modularity, and ease of maintenance.

They are widely used for developing modern web applications, including social media platforms, e-commerce sites, dashboards, and more.

2. Node.js :

1. Node.js is a JavaScript-based platform for server-side and networking applications.

2. It is a runtime environment that allows developers to run JavaScript code on the server side.

3. Node.js is commonly used for building web servers, real-time applications (like chat applications), APIs, and microservices, among other types of applications.

4. Node.js is a software platform for scalable server-side and networking applications.

5. Node.js applications are designed to maximize throughput and efficiency.

6. Node.js internally uses the Google V8 JavaScript engine to execute code.

Developer gives

1. .json file →it denotes write docker file for node.js
2. .txt file → it denotes write docker file for python
3. .jar file → it denotes write docker file for java
4. .xml file → it denotes write docker file for java.

→ But developer does not give .jar file. We can create .jar file using build tool [maven /gradel]

1. Maven [pom.xml]
2. Gradel [build.gradle]

File format	Docker file	Package management tool
package. json	Node.js	Npm [Node Package Manager]
requirement.txt	python	pip
app.jar pom.xml	java	Maven [pom.xml] Gradel [build.gradle]

Suppose package.json file [default file name] not given in github repo, it means it's an already build application. For run that application, we need web server [nginx or Apache]

difference between docker file and docker compose file

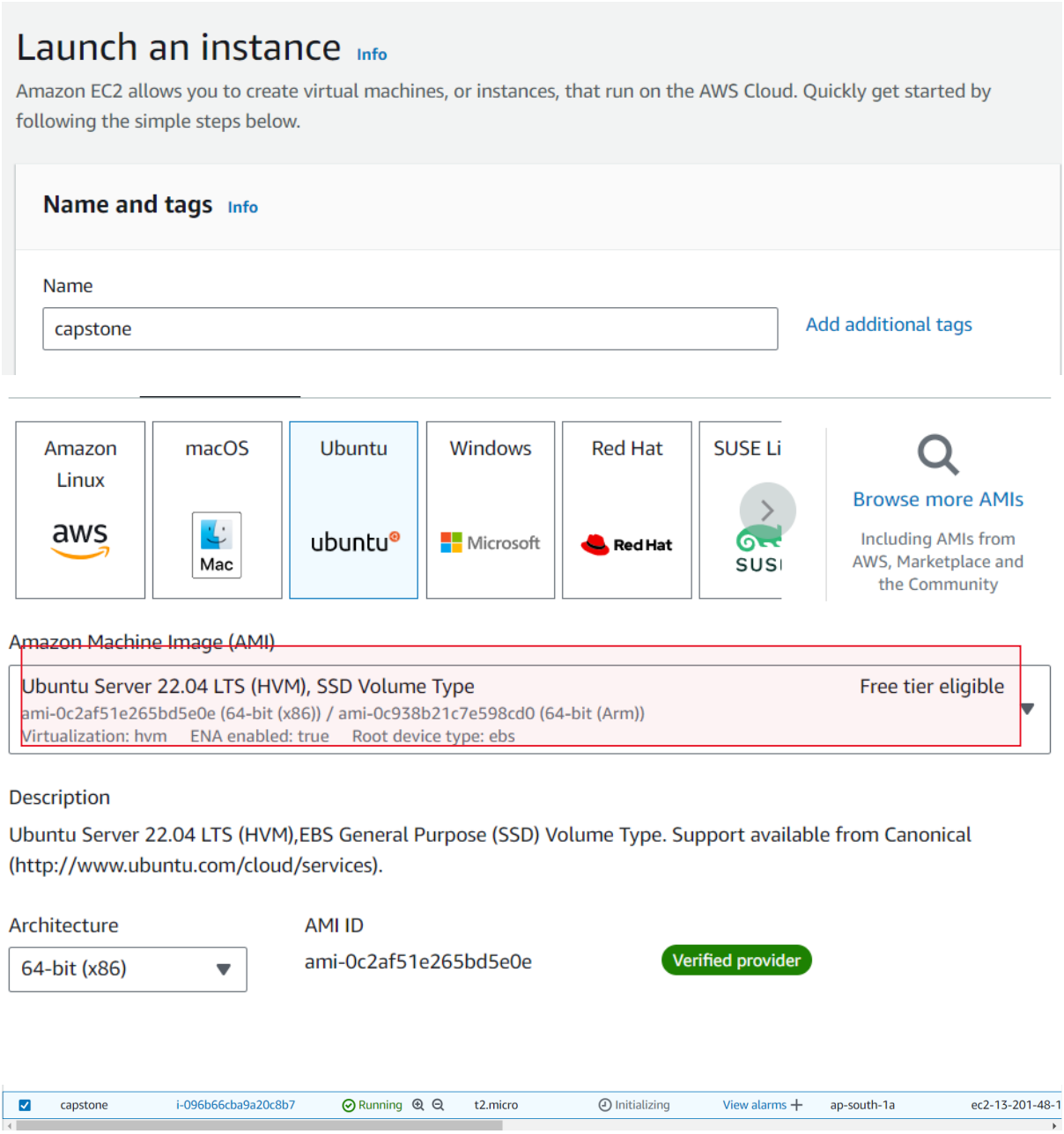
	Docker file	**Docker compose file**
purpose	Defines how to build a single Docker image.	Defines how to run multi-container applications.it is used for running multiple containers.
File type	Text file	.yaml / yml file
syntax	FROM, RUN, COPY, etc	YAML syntax to define services, networks, and volumes.
Content	It Contains instructions to set up an environment inside a container	It contains service definitions and configurations for running containers

By understanding these differences, you can effectively use both Docker file and Docker Compose together to build and deploy complex Docker-based applications.

Installing the necessary software's & services for this task:

1. **Git**
2. **Docker**
3. **Docker compose**
4. **Java**
5. **Jenkins**

Before install git. we have to launch an instance. select ubuntu



Connect that instance

1. Git installation:

sudo apt update

sudo apt install git -y

git –version

sudo apt update

```
Get:39 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [3668 B]
Get:40 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [249 kB]
Get:41 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [108 kB]
Get:42 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8632 B]
Get:43 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [9220 B]
Get:44 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [208 kB]
Get:45 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [40.7 kB]
Get:46 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [420 B]
Get:47 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [10.6 kB]
Get:48 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [2808 B]
Get:49 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Get:50 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [344 B]
Fetched 28.2 MB in 6s (5082 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
47 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-172-31-32-39:~$ sudo apt install git -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.43.0-1ubuntu7.1).
git set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 47 not upgraded.
ubuntu@ip-172-31-32-39:~$ git --version
git version 2.43.0
ubuntu@ip-172-31-32-39:~$
```

Clone the given repo

git clone https://github.com/sriram-R-krishnan/devops-build

```
ubuntu@ip-172-31-32-39:~$ git clone https://github.com/sriram-R-krishnan/devops-build
Cloning into 'devops-build'...
remote: Enumerating objects: 21, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 21 (delta 0), reused 0 (delta 0), pack-reused 18
Receiving objects: 100% (21/21), 720.09 KiB | 15.00 MiB/s, done.
ubuntu@ip-172-31-32-39:~$
```

```
ubuntu@ip-172-31-32-39:~$ mkdir caspstone1
ubuntu@ip-172-31-32-39:~$ cd caspstone1/
ubuntu@ip-172-31-32-39:~/caspstone1$
```

2. Docker installation:

sudo apt update

sudo apt install docker.io

sudo docker version

sudo usermod -aG docker ubuntu

sudo systemctl status docker

sudo systemctl start docker

after that, I got some error regarding permission.so I give below commands. search the below commands in chatgpt

sudo usermod -aG docker ubuntu

newgrp docker

sudo systemctl start docker

sudo systemctl enable docker

just check, docker can pull the image from docker hub. This command not needed for this task

sudo docker run hello-world

sudo apt update

```
ubuntu@ip-172-31-32-39:~$ sudo apt update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
47 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

sudo apt install docker.io

```
ubuntu@ip-172-31-32-39:~$ sudo apt install docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-buildx docker-compose-v2 docker-doc rinse zfs-
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io pigz runc ubuntu-fan
0 upgraded, 8 newly installed, 0 to remove and 47 not upgraded.
Need to get 76.8 MB of archives.
After this operation, 289 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 bridge-utils amd64 1.7.1-1ubuntu2 [33.9 kB]
Get:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 runc amd64 1.1.12-0ubuntu3 [8599 kB]
Get:4 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 containerd amd64 1.7.12-0ubuntu4 [38.6 MB]
Get:5 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 dns-root-data all 2023112702~willsync1 [4450
Get:6 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 dnsmasq-base amd64 2.90-2build2 [375 kB]
Get:7 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 docker.io amd64 24.0.7-0ubuntu4 [29.1 MB]
Get:8 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 ubuntu-fan all 0.12.16 [35.2 kB]
Fetched 76.8 MB in 1s (56.2 MB/s)
Preconfiguring packages ...
Selecting previously unselected package pigz.
(Reading database ... 67739 files and directories currently installed.)
```

sudo docker version

```
ubuntu@ip-172-31-32-39:~$ sudo docker version
Client:
 Version:           24.0.7
 API version:       1.43
 Go version:        go1.22.2
 Git commit:        24.0.7-0ubuntu4
 Built:             Wed Apr 17 20:08:25 2024
 OS/Arch:           linux/amd64
 Context:           default

Server:
 Engine:
  Version:          24.0.7
  API version:      1.43 (minimum version 1.12)
  Go version:       go1.22.2
  Git commit:       24.0.7-0ubuntu4
  Built:            Wed Apr 17 20:08:25 2024
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.7.12
  GitCommit:
 runc:
  Version:          1.1.12-0ubuntu3
  GitCommit:
 docker-init:
```

sudo usermod -aG docker ubuntu

```
  GitCommit:
ubuntu@ip-172-31-32-39:~$ sudo usermod -aG docker ubuntu
ubuntu@ip-172-31-32-39:~$
```

sudo systemctl status docker

```
ubuntu@ip-172-31-32-39:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: enabled)
     Active: active (running) since Thu 2024-08-08 06:50:22 UTC; 28min ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 2556 (dockerd)
      Tasks: 8
     Memory: 32.6M (peak: 33.2M)
        CPU: 442ms
     CGroup: /system.slice/docker.service
             └─2556 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.soc
```

sudo systemctl start docker

```
ubuntu@ip-172-31-46-99:~$ sudo usermod -aG docker ubuntu
ubuntu@ip-172-31-46-99:~$ newgrp docker
ubuntu@ip-172-31-46-99:~$ sudo systemctl start docker
ubuntu@ip-172-31-46-99:~$ sudo systemctl enable docker
ubuntu@ip-172-31-46-99:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:1408fec50309afee38f3535383f5b09419e6dc0925bc69891e79d84cc4cdcec6
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.
```

```
To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

ubuntu@ip-172-31-46-99:~$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
hello-world   latest    d2c94e258dcb   15 months ago   13.3kB
```

docker file:

- In our task given application was already built.
- So, we did not need to write node.js docker file. Just write docker file for nginx server.
- For running the application, we need web server. So, write docker file for nginx web server.
- Because in that server we have to run an already built application.
- **Docker file written in the folder devops-build [in given github this folder available].**
- **so, get into the folder [cd devops-build]**

Explanation of the Dockerfile:

1. **FROM nginx:latest**:
    - This line specifies the base image to use for the Docker image. In this case, it pulls the latest version of the official Nginx image from Docker Hub.
2. **WORKDIR /usr/share/nginx/html**:
    - This sets the working directory inside the container to /usr/share/nginx/html. This is the default directory where Nginx serves static files (like HTML, CSS, and JavaScript).
3. **COPY build/ .**:
    - This copies the contents of the build/ directory [this dir in the given repo]on your host machine into the current working directory in the container (/usr/share/nginx/html). This typically contains the static files that your web application will serve.
4. **EXPOSE 80**:
    - This line informs Docker that the container will listen on port 80 at runtime. Port 80 is the default HTTP port, so this is where Nginx will serve the web application.
5. **CMD ["nginx", "-g", "daemon off;"]**:
    - This command runs Nginx in the foreground (instead of in the background as a daemon), which is necessary for Docker containers so that the container doesn't exit immediately after starting. The command starts Nginx and keeps it running.

vi dockerfile

```
FROM nginx:latest

WORKDIR /usr/share/nginx/html

COPY build/ .

EXPOSE 80

CMD ["nginx", "-g", "daemon off;"]
```

Convert docker file into docker image:

docker build -t nginximage . **[nginximage -we give that image name for our choice]. This command used for build docker image**

```
ubuntu@ip-172-31-46-99:~$ vi dockerfile
ubuntu@ip-172-31-46-99:~$ cd devops-build/
ubuntu@ip-172-31-46-99:~/devops-build$ vi dockerfile
ubuntu@ip-172-31-46-99:~/devops-build$ docker build -t nginximage .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  3.415MB
Step 1/5 : FROM nginx:latest
 ---> a72860cb95fd
Step 2/5 : WORKDIR /usr/share/nginx/html
 ---> Using cache
 ---> 834607514a93
Step 3/5 : COPY build/ .
 ---> 9a869321475c
Step 4/5 : EXPOSE 80
 ---> Running in 2db9e8a75aea
Removing intermediate container 2db9e8a75aea
 ---> e144c57ce161
Step 5/5 : CMD ["nginx", "-g", "daemon off;"]
 ---> Running in 60c950b6a23f
Removing intermediate container 60c950b6a23f
 ---> 6ba82f9c1767
Successfully built 6ba82f9c1767
Successfully tagged nginximage:latest
ubuntu@ip-172-31-46-99:~/devops-build$ docker images
```

docker images

```
ubuntu@ip-172-31-46-99:~/devops-build$ docker images
REPOSITORY    TAG             IMAGE ID       CREATED          SIZE
nginximage    latest          6ba82f9c1767   25 seconds ago   190MB
nodeimage     latest          d44d4a7a0b69   27 minutes ago   188MB
<none>        <none>          ae52f8af4119   27 minutes ago   81.1MB
nginx         latest          a72860cb95fd   6 weeks ago      188MB
hello-world   latest          d2c94e258dcb   15 months ago    13.3kB
node          12.2.0-alpine   f391dabf9dce   5 years ago      77.7MB
```

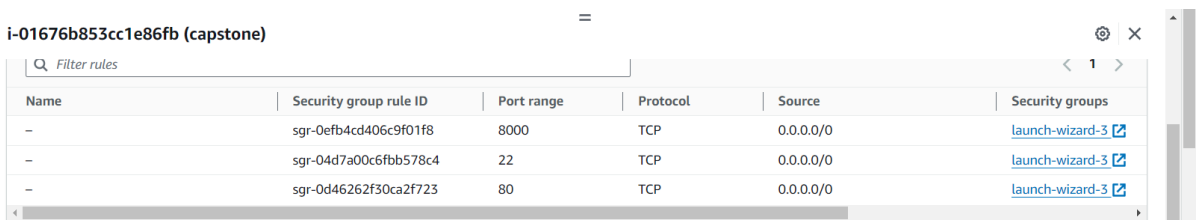
docker run -d –name mynignxcointainer -p 80:80 nginximage

```
ubuntu@ip-172-31-46-99:~/devops-build$ docker run -d --name mynginxcointainer -p 80:80 nginximage
1c82a7a8e8f05bcd8fce44d66e1bdae6fcd163163fe1cad573c78c81a4adb25f
```

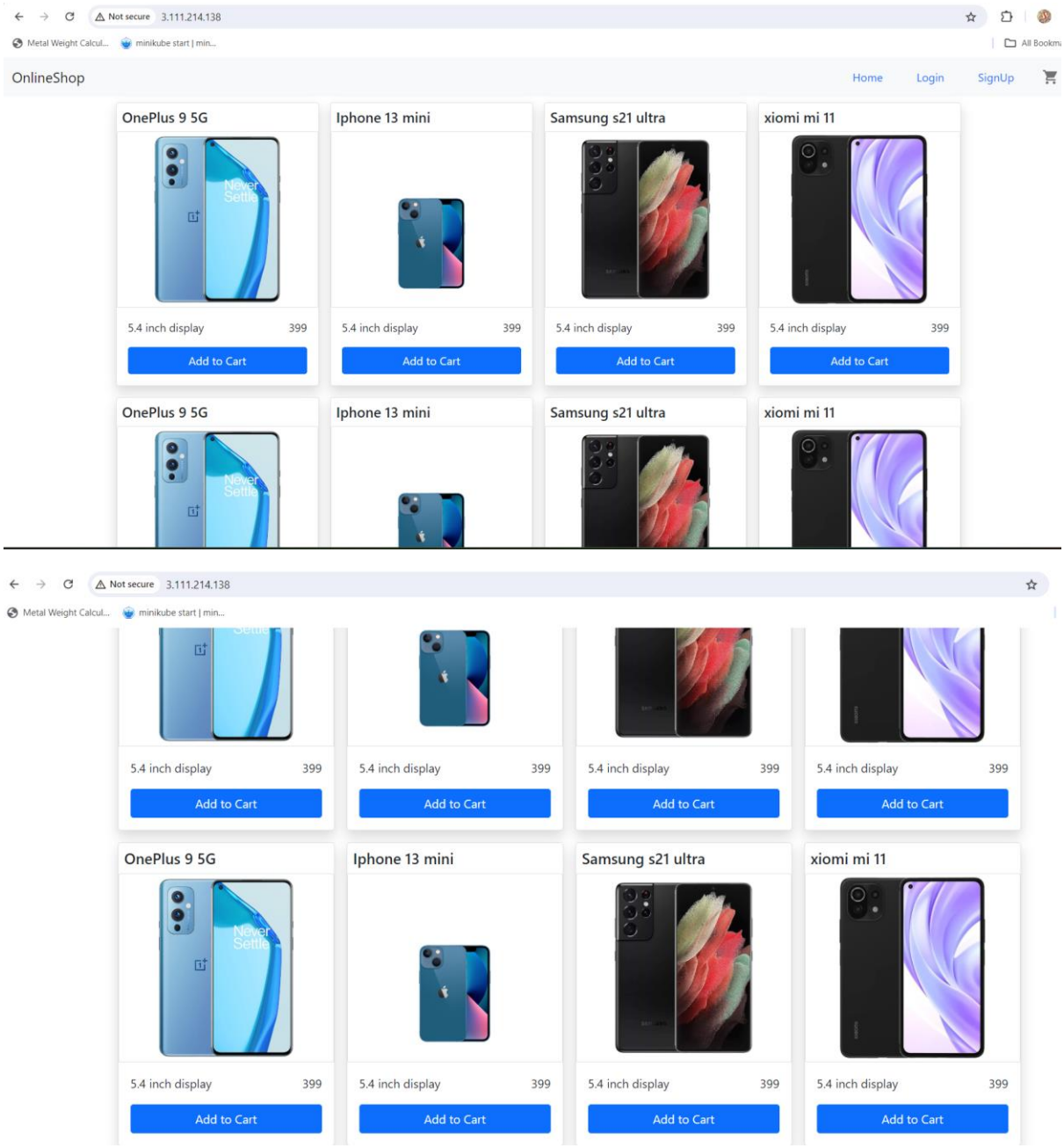
docker ps

```
ubuntu@ip-172-31-46-99:~/devops-build$ docker ps
CONTAINER ID   IMAGE        COMMAND                  CREATED          STATUS          PORTS                                   NAMES
1c82a7a8e8f0   nginximage   "/docker-entrypoint.…"   5 seconds ago    Up 3 seconds    0.0.0.0:80->80/tcp, :::80->80/tcp       mynginxcointainer
ae71713569e8   nodeimage    "/docker-entrypoint.…"   25 minutes ago   Up 25 minutes   0.0.0.0:8000->80/tcp, :::8000->80/tcp   nginxcointainer
```

Open the port no :80 in ec2. Copy and paste the ip address



Output page of given application:



3. **Docker compose installation**:

sudo apt update

sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

sudo chmod +x /usr/local/bin/docker-compose

docker-compose –version

sudo apt update

```
ubuntu@ip-172-31-46-99:~$ sudo apt update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Get:5 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1930 kB]
Get:6 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [341 kB]
Get:7 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [17.8 kB]
Get:8 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [2301 kB]
Get:9 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted Translation-en [396 kB]
Get:10 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1110 kB]
Get:11 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 c-n-f Metadata [25.9 kB]
Get:12 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [43.3 kB]
Get:13 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [1712 kB]
Get:14 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [283 kB]
Get:15 http://security.ubuntu.com/ubuntu jammy-security/main amd64 c-n-f Metadata [13.1 kB]
Get:16 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [2226 kB]
Get:17 http://security.ubuntu.com/ubuntu jammy-security/restricted Translation-en [383 kB]
Fetched 11.0 MB in 3s (3177 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
29 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

```
ubuntu@ip-172-31-46-99:~$ sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docke
r-compose
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100 12.1M  100 12.1M    0     0  8033k      0  0:00:01  0:00:01 --:--:-- 12.3M
```

sudo chmod +x /usr/local/bin/docker-compose

```
ubuntu@ip-172-31-46-99:~$ sudo chmod +x /usr/local/bin/docker-compose
ubuntu@ip-172-31-46-99:~$ docker-compose --version
docker-compose version 1.29.2, build 5becea4c
```

docker-compose –version

Docker compose file for node js:

getinto folder devops-build. Then write this

```
ubuntu@ip-172-31-46-99:~$ cd devops-build/
ubuntu@ip-172-31-46-99:~/devops-build$ vi docker-compose.yaml
```

vi docker-compose.yaml

```
version: '3'

services:
 webcointainer:
  image: nginximage
  ports:
   - "80:80"
```

version: '3'

services:

webcointainer:

image: nginximage [already we created docker image. give that image name]

ports:

- “80:80”

Give permission for this file:

Chmod 777 docker-compose.yaml

```
ubuntu@ip-172-31-46-99:~/devops-build$ vi docker-compose.yaml
ubuntu@ip-172-31-46-99:~/devops-build$ chmod 777 docker-compose.yaml
```

Execute the docker-compose.yaml :

docker-compose up -d

```
ubuntu@ip-172-31-46-99:~/devops-build$ docker-compose up -d
Starting devops-build_webcointainer_1 ... done
```

docker ps

```
ubuntu@ip-172-31-46-99:~/devops-build$ docker ps
CONTAINER ID   IMAGE         COMMAND                  CREATED          STATUS          PORTS                                   NAMES
4e1ec10be231   nginximage    "/docker-entrypoint.…"   10 minutes ago   Up 10 seconds   0.0.0.0:80->80/tcp, :::80->80/tcp   devops-build_webcointainer_1
```