

School of Computer Science
UNIVERSITY OF PETROLEUM AND ENERGY STUDIES
DEHRADUN, UTTARAKHAND



**Container Orchestration and infrastructure Automation Lab
(CSVT3109)
Lab File
(2023)**

for

5th Semester

Submitted To:

Mr. Abhirup Khanna
Assistant Professor (SS)
Cloud Software Operations Cluster
School of Computer Science
UPES

Submitted By:

Hitendra Sisodia
B. Tech. CSE spl CCVT[5th Semester]
500091910
R2142210352
Batch No: 2

Experiment No.	Description	Page No	Date
1.	Docker Installation	3-6	17-08-23
2.	Docker Lifecycle	6-9	24-08-23
3.	Running Apache web server	10-11	31-08-23
3.2.	Creating a custom container: JDK	12-13	31-08-23
4.	Docker file	14-15	07-09-23
5.	Data management in Docker: Volume	16-17	07-09-23
5.2.	Data management in Docker: Bind mount	18-18	07-09-23
6.	Docker Networking	19-24	14-09-23
7.	Docker Compose	25-28	21-09-23
8.	Docker Swarm	29-33	05-10-23

Experiment 1 - Docker Installation

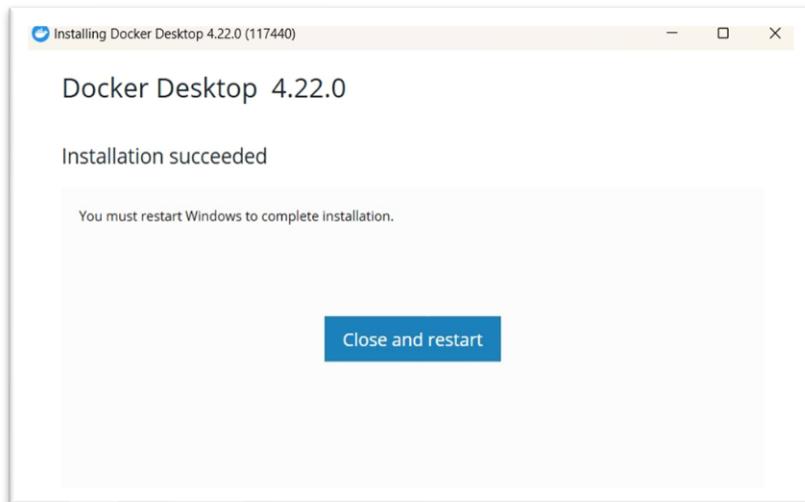
Step1 - Download Docker Desktop from the internet



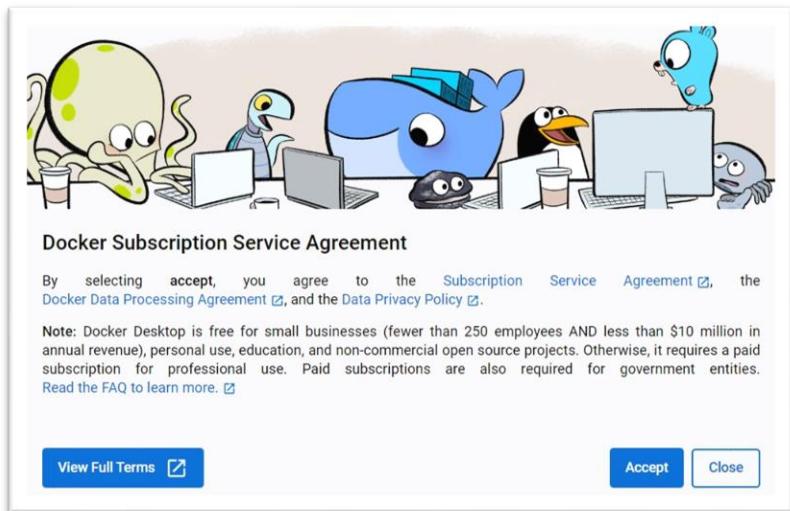
Step2 - Select Docker Desktop Installer to start installing



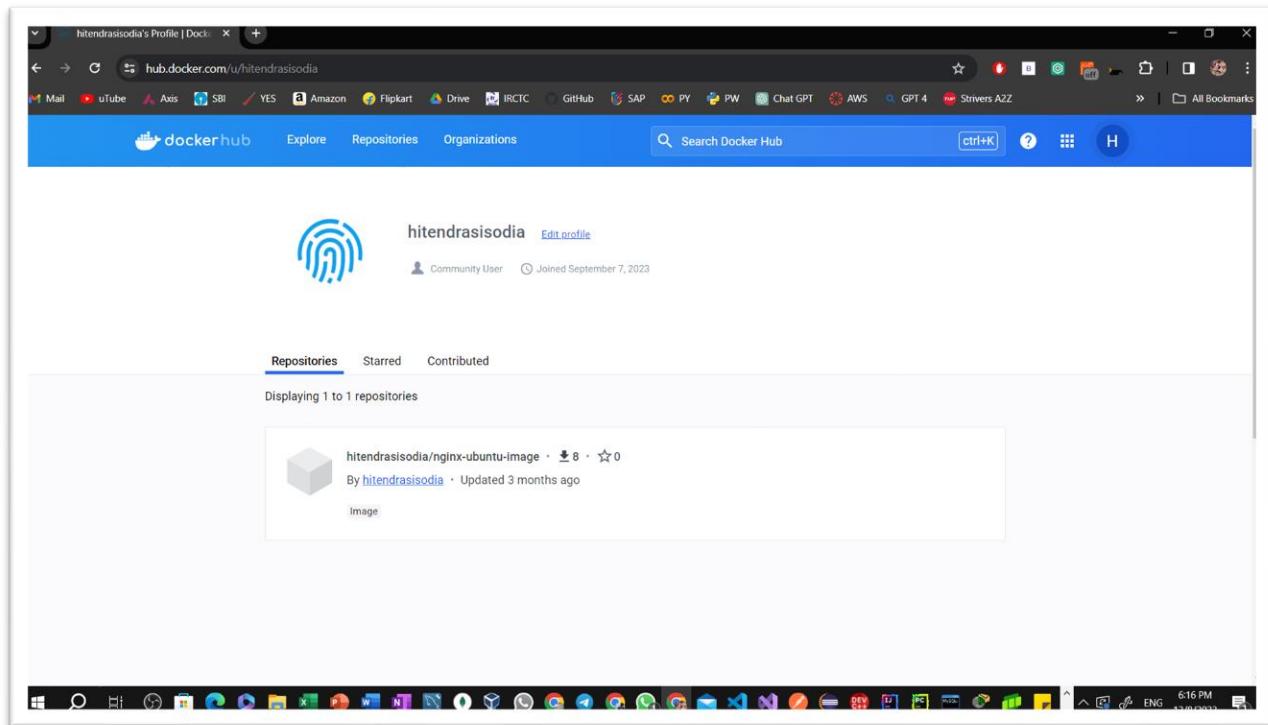
Step3 - After Finish installation click on “Close and Restart” Button



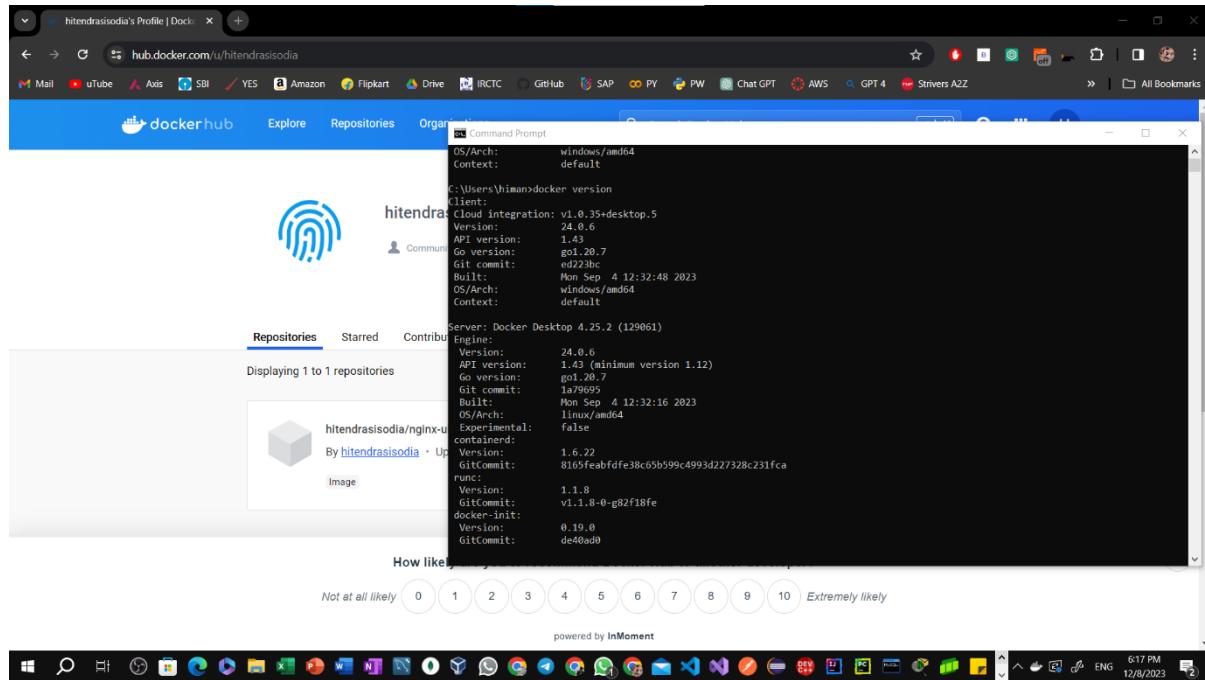
Step4 - Accept the terms and agreements



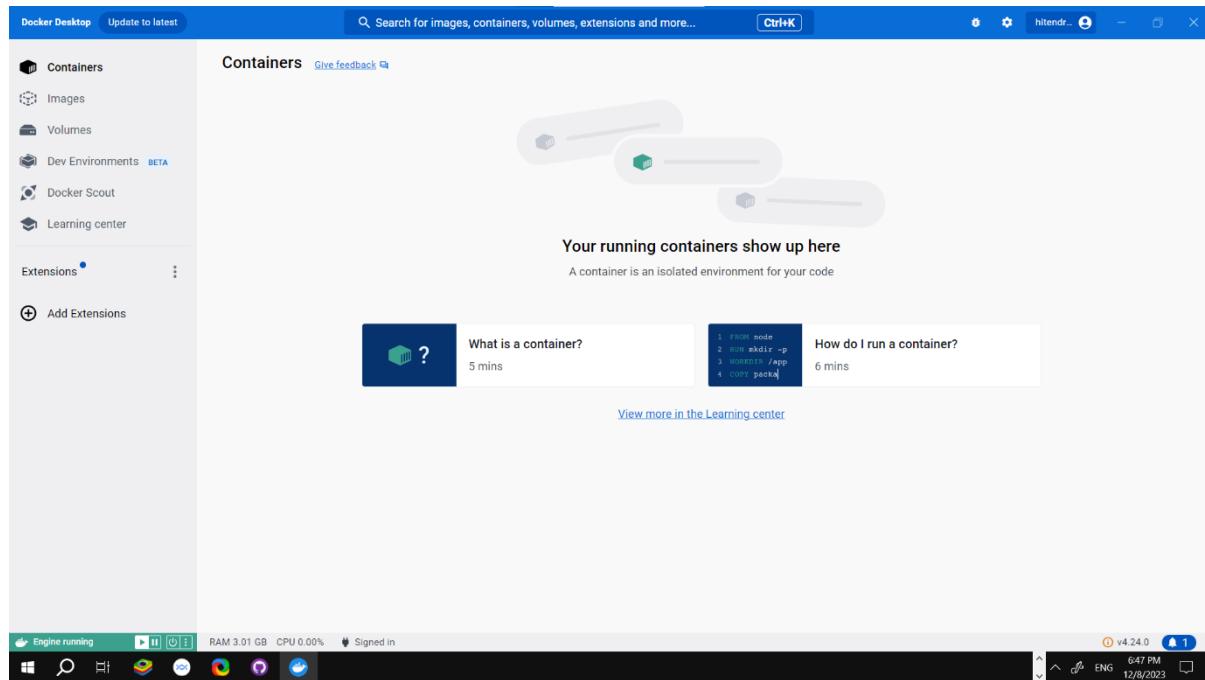
Step5 - Create an account in the docker hub



Step6 - Open windows CMD and type command- docker version. The version command prints the current version number for all independently versioned Docker components.

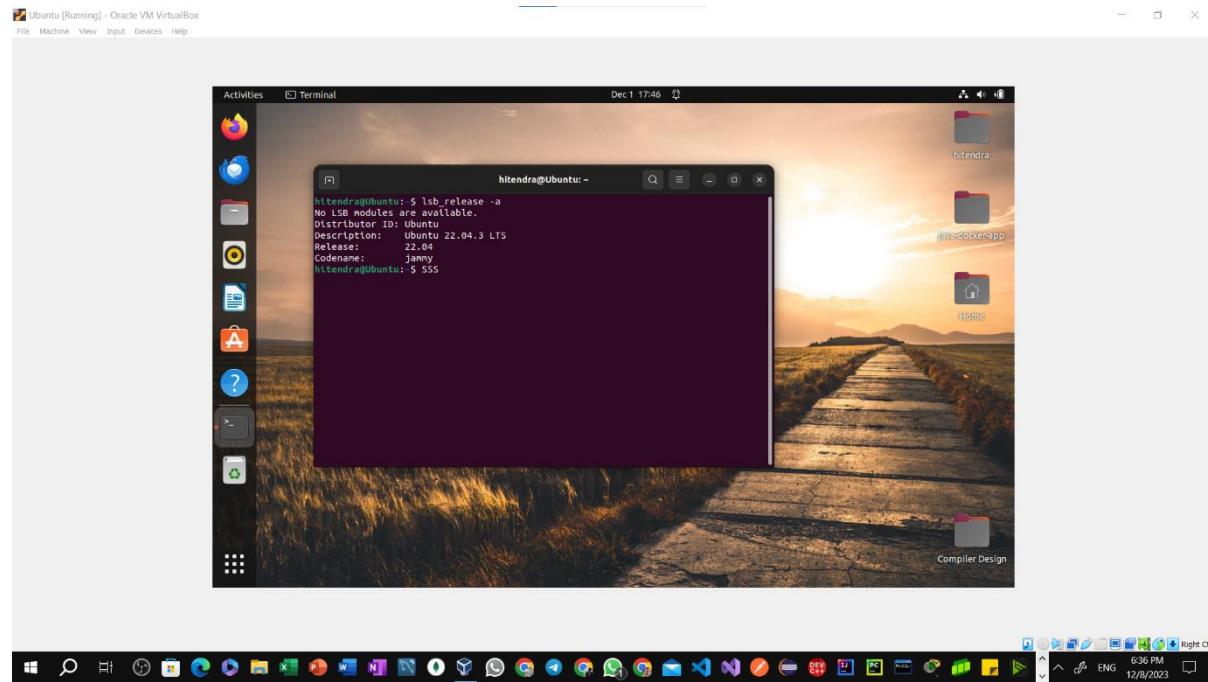


Step7 - Open Docker Desktop

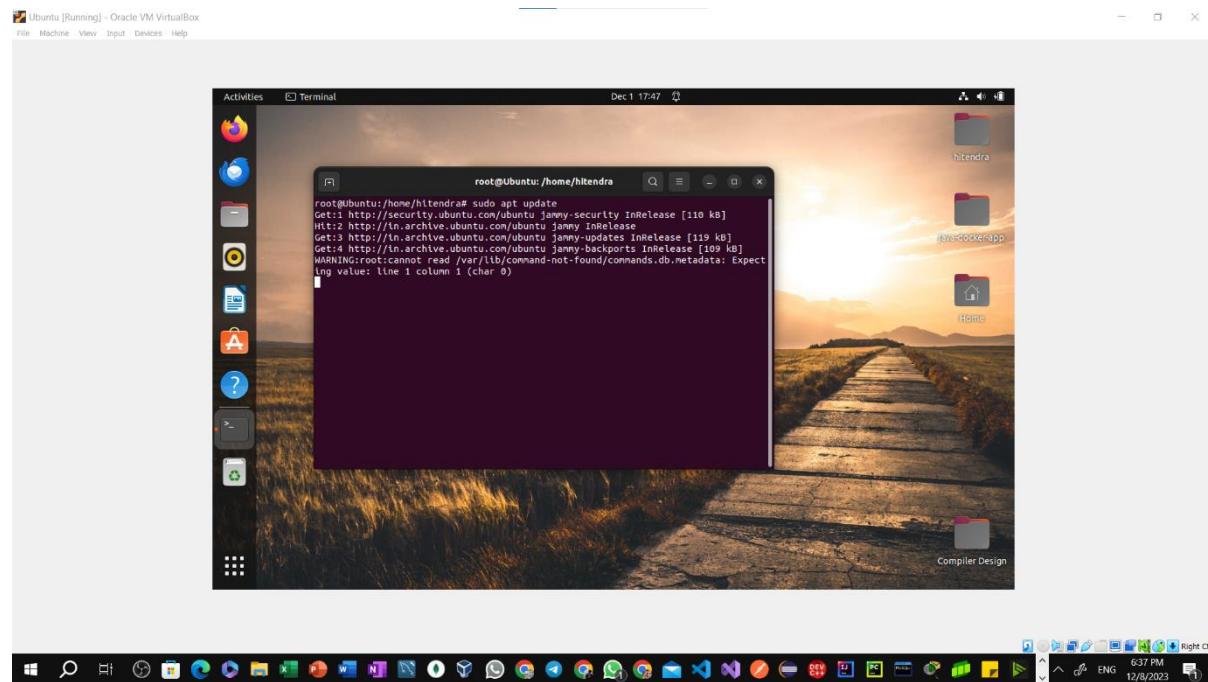


Installing Docker on Ubuntu

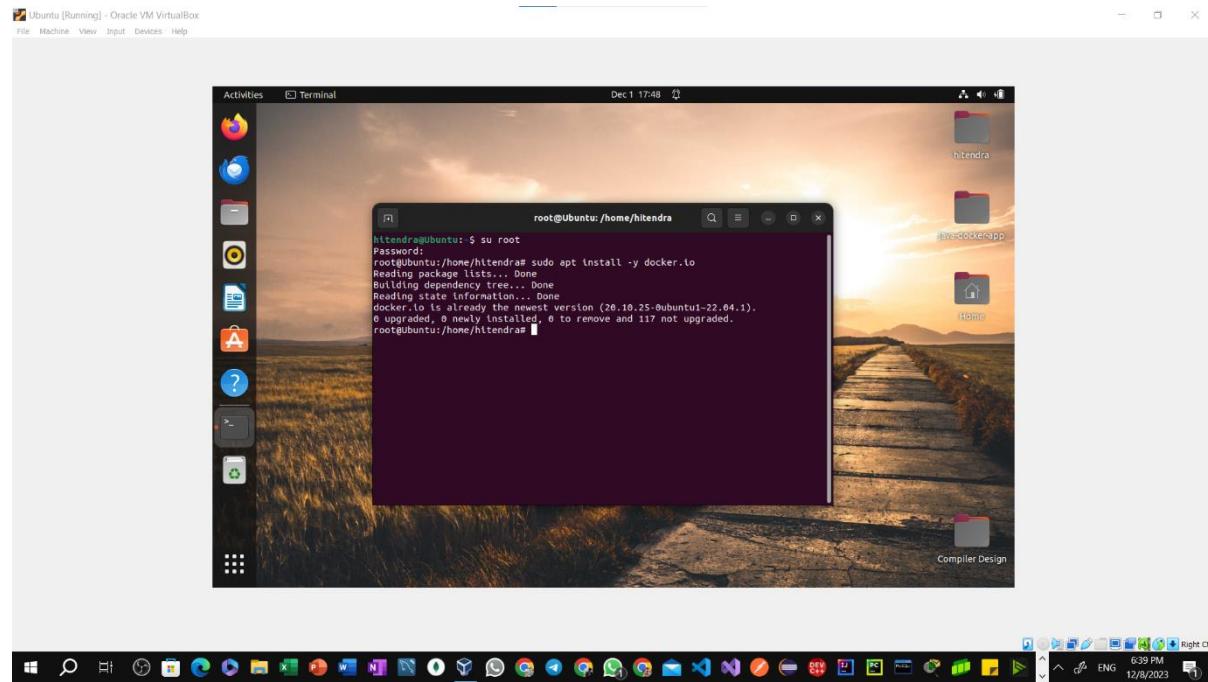
Step1: Make sure that the oracle vm is the latest version and ubuntu version is above 20.



Step3: After reboot again login the terminal and type the `sudo apt update` command

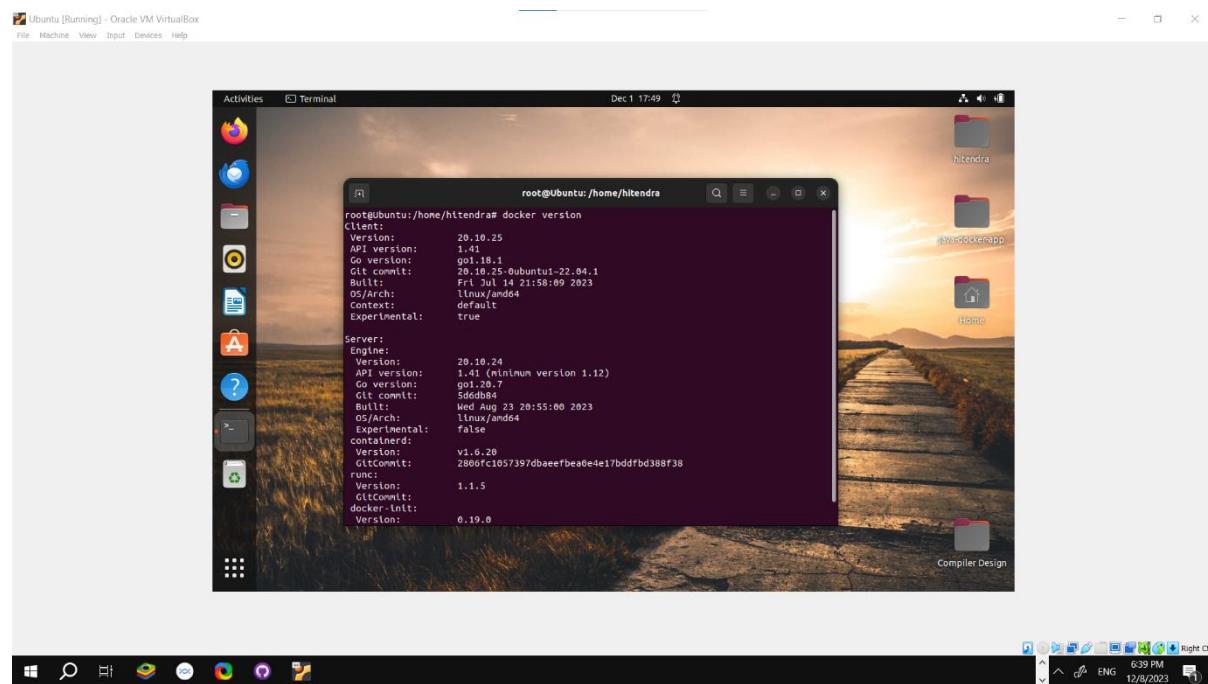


Step4: Enter the command: sudo apt install -y docker.io to install docker.

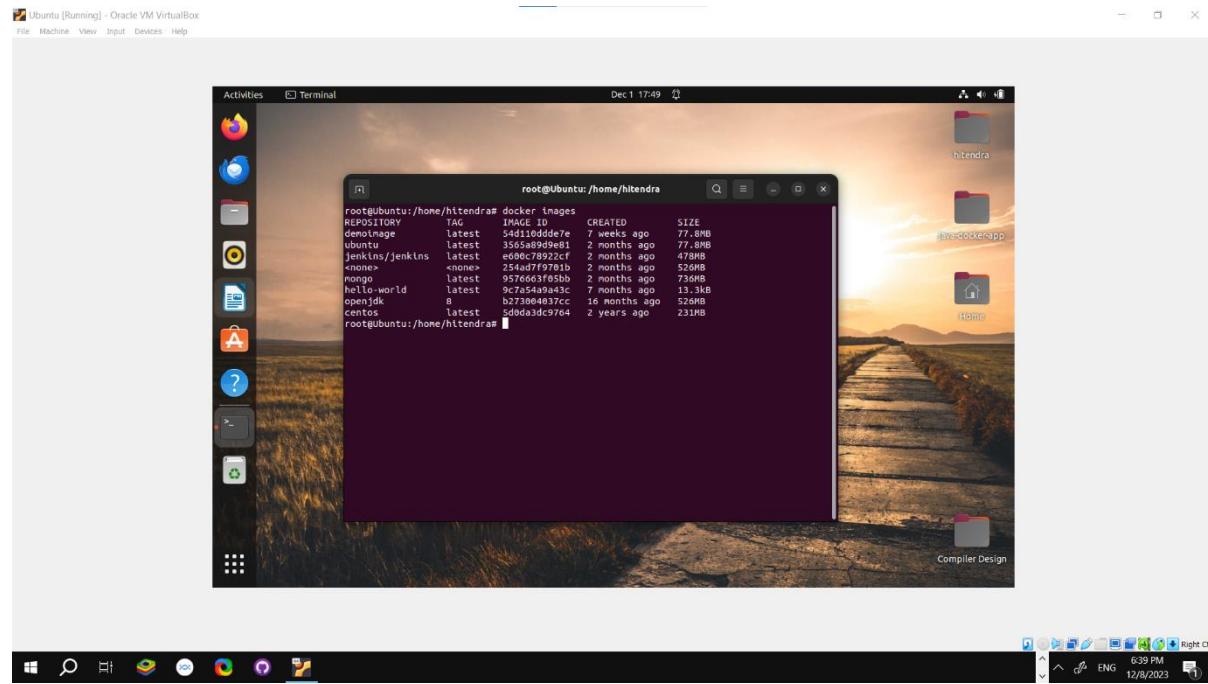


Verification steps:

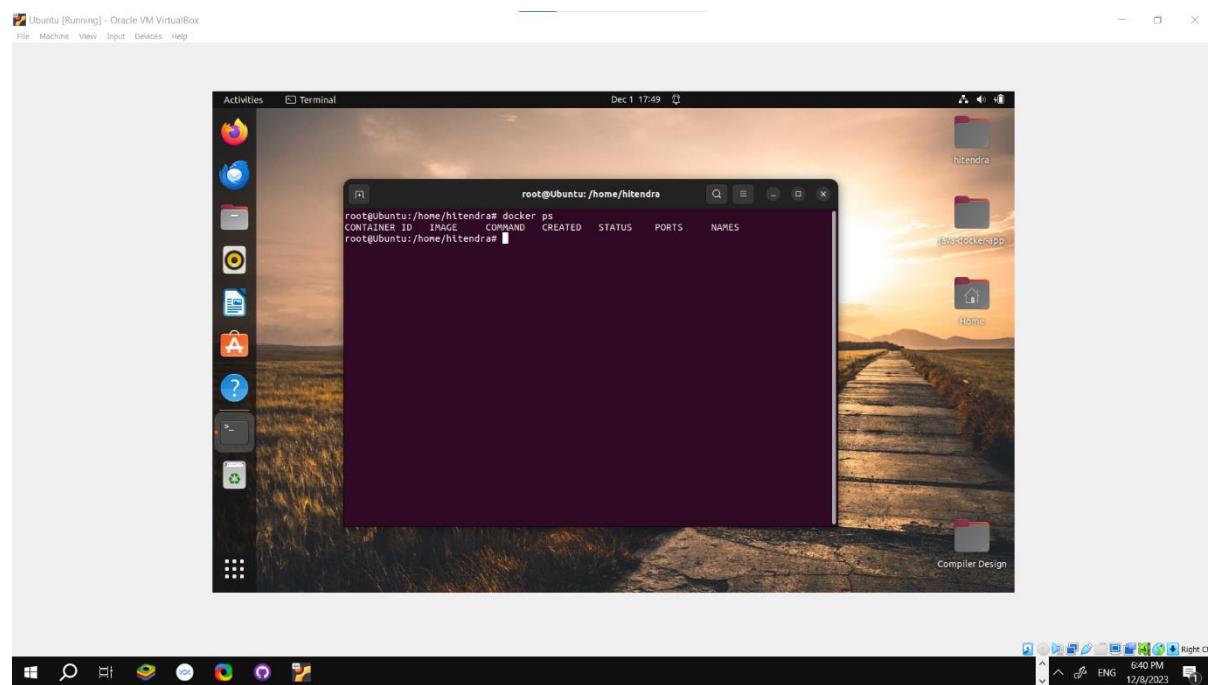
Step8: Type the command: docker version



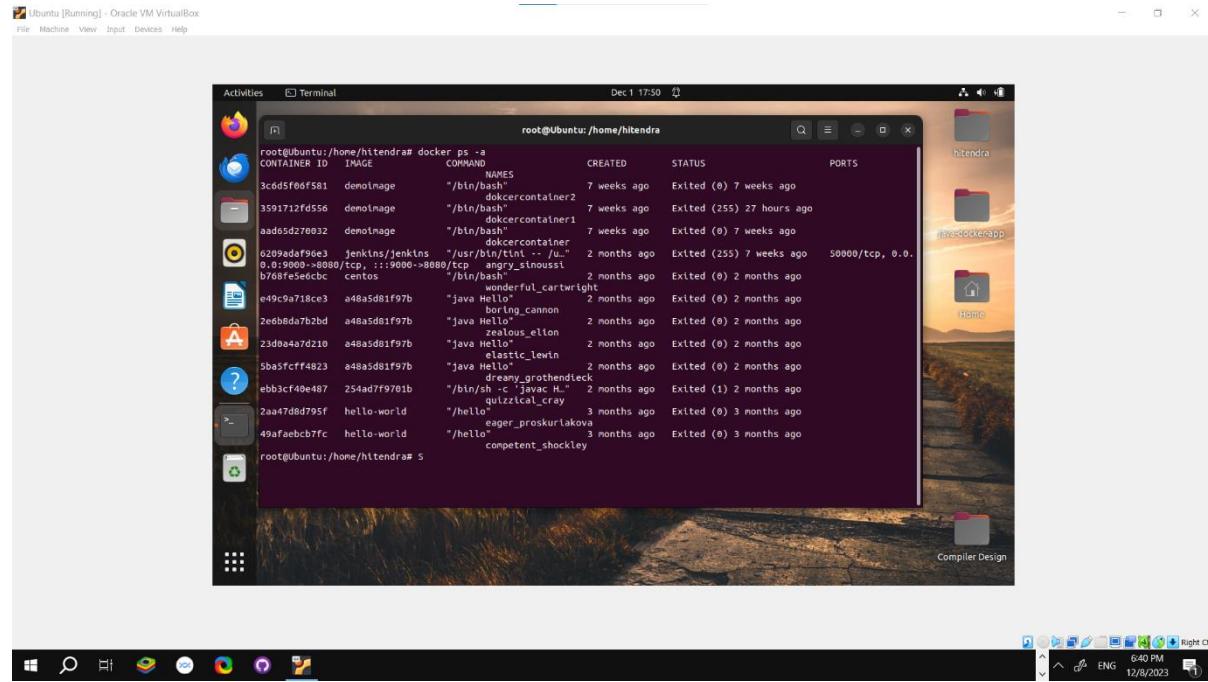
Step9: Command: docker images



Step10: Command: docker ps

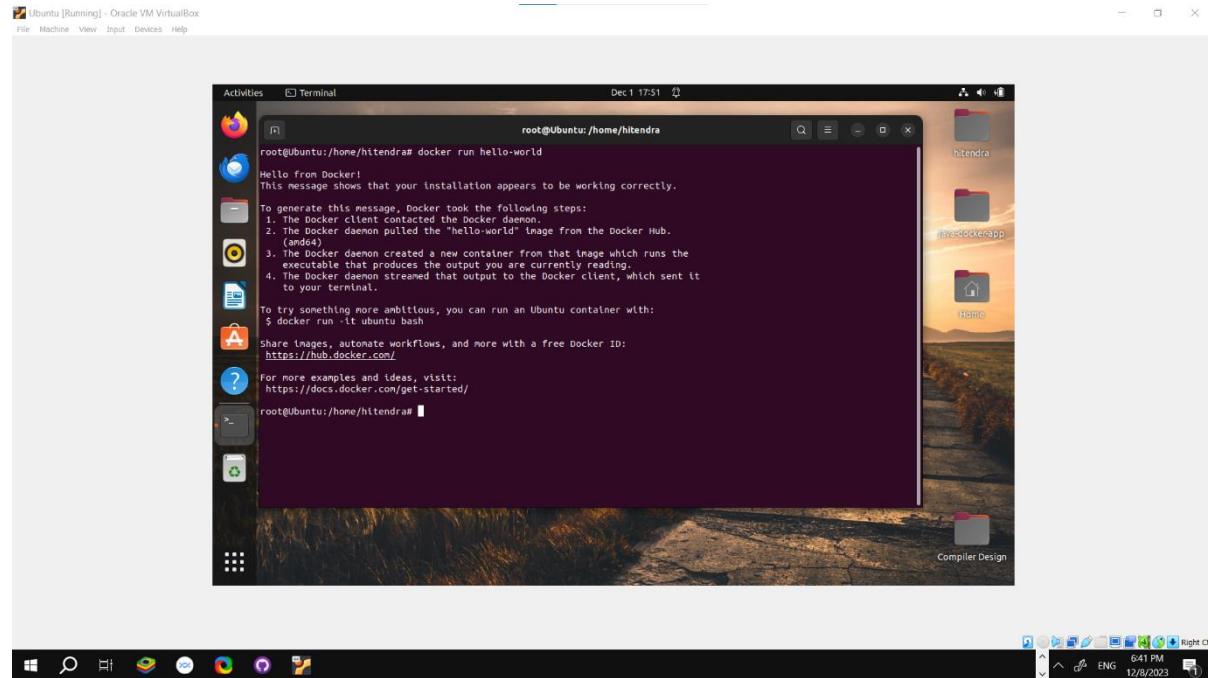


Step11: Command: docker ps -a



Test docker

Step - 1Command: docker run hello-world



Experiment - 2

Q1- Search Apache image. Pull appropriate image. Check whether the image is pulled or not. Run an apache container and map the port with the host machine using –p flag.

Search and pull the appropriate apache image using:

docker search apache.

docker pull httpd.

```
Command Prompt x + 
Microsoft Windows [Version 10.0.22621.2215]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ACER>docker search apache
NAME          DESCRIPTION                                     STARS   OFFICIAL  AUTOMATED
httpd         The Apache HTTP Server Project               4521    [OK]
tomcat        Apache Tomcat is an open source implementati... 3578    [OK]
maven         Apache Maven is a software project managemen... 1512    [OK]
zookeeper     Apache ZooKeeper is an open-source server wh... 1381    [OK]
cassandra     Apache Cassandra is an open-source distribut... 1485    [OK]
solr          Apache Solr is the popular, blazing-fast, op... 963     [OK]
flink         Apache Flink® is a powerful open-source dist... 400     [OK]
groovy        Apache Groovy is a multi-faceted language fo... 143     [OK]
tomee         Apache TomEE is an all-Apache Java EE certif... 110     [OK]
storm          Apache Storm is a free and open source distr... 194     [OK]
spark          Apache Spark - A unified analytics engine fo... 10      [OK]
apachepinot/pinot      Pinot is a real-time distributed OLAP dataset... 9
bitnami/apache      Bitnami Apache Docker Image           88     [OK]
apachepinot/pinot-presto  This image is a presto image built-in with p... 1
apachepinot/pinot-superset  1
apache/airflow       Apache Airflow                         467
apachepinot/pinot-base-runtime  The base image for Apache Pinot runtime       0
apache/superset      Apache Superset                         227
apachepinot/pinot-base-build  The base image for Apache Pinot build          0
apache/tika          Apache Tika Server - the content analysis to... 32
apache/apisix        Apache APISIX: Cloud-Native API Gateway       80
apachepulsar/pulsar  Apache Pulsar - Distributed pub/sub messagin... 79
apachepinot/thirdeye Docker Image for Apache Thirdeye        1
apachepulsar/pulsar-all  Apache Pulsar - Distributed pub/sub messagin... 21
apache/nifi          Unofficial convenience binaries and Docker i... 277    [OK]

C:\Users\ACER>docker pull httpd
Using default tag: latest
latest: Pulling from library/httpd
Digest: sha256:333f7bc9fb7224bf301fd2ae4892b86d36cc2fdf4c6aa49a6700f27c8e06daf
Status: Image is up to date for httpd:latest
docker.io/library/httpd:latest
```

Check whether the image is pulled or not using this command: docker images.

```
Command Prompt x + 
Status: Image is up to date for httpd:latest
docker.io/library/httpd:latest

What's Next?
View summary of image vulnerabilities and recommendations → docker scout quickview httpd

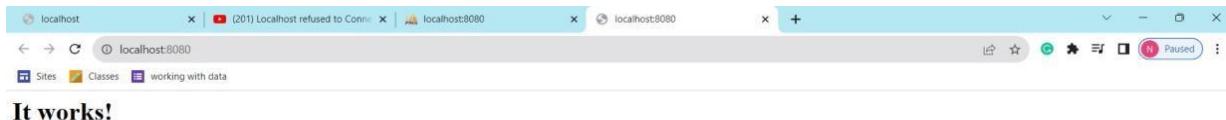
C:\Users\ACER>docker images
REPOSITORY    TAG      IMAGE ID      CREATED      SIZE
httpd        latest   76e5ad98b58e  2 weeks ago  168MB
```

Create a container using the pulled image and port it using –p flag

```
C:\Users\ACER>docker run -dt --name=mywebserver -p8080:80 httpd
2ff846eff12665436d34cd704dbf8ea7e97b18dd7fb4cac2b04cb6f377390989

C:\Users\ACER>docker ps
CONTAINER ID  IMAGE      COMMAND      CREATED      STATUS      PORTS      NAMES
2ff846eff126  httpd      "httpd-foreground"  3 seconds ago  Up 2 seconds  0.0.0.0:8080->80/tcp  mywebserver

C:\Users\ACER>curl localhost:8080
<html><body><h1>It works!</h1></body></html>
```

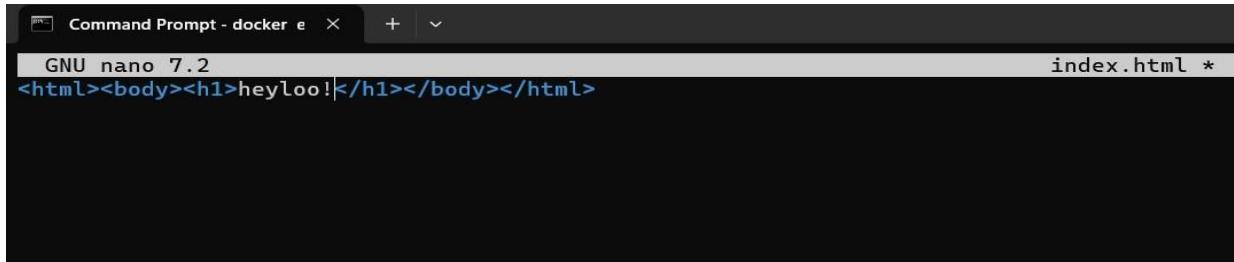
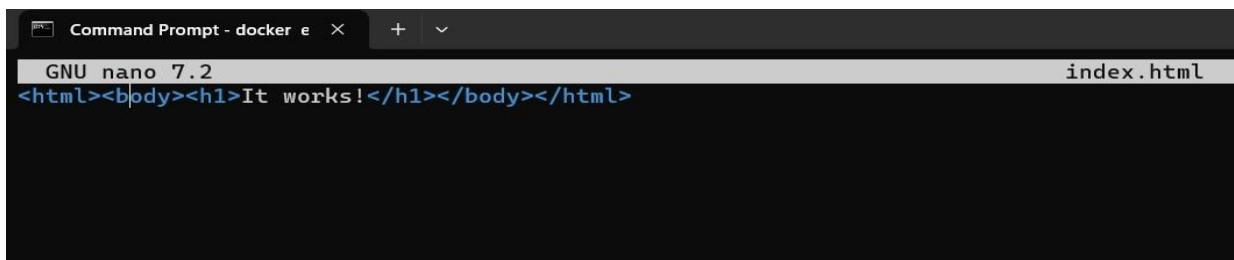


Get into the container bash and install the Nano editor.

```
C:\Users\ACER>docker exec -it mywebserver bash
root@2ff846eff126:/usr/local/apache2# apt-get update
Get:1 http://deb.debian.org/debian bookworm InRelease [151 kB]
Get:2 http://deb.debian.org/debian bookworm-updates InRelease [52.1 kB]
Get:3 http://deb.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Get:4 http://deb.debian.org/debian bookworm/main amd64 Packages [8906 kB]
Get:5 http://deb.debian.org/debian bookworm-updates/main amd64 Packages [4952 B]
Get:6 http://deb.debian.org/debian-security bookworm-security/main amd64 Packages [58.3 kB]
Fetched 9221 kB in 2s (5626 kB/s)
Reading package lists... Done
root@2ff846eff126:/usr/local/apache2# apt-get install -y nano
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libgpm2 libncursesw6
```

Now using the Nano editor edit the text in html file.

```
Processing triggers for libc-bin (2.36-9+deb12u1) ...
root@2ff846eff126:/usr/local/apache2# ls
bin build cgi-bin conf error htdocs icons include logs modules
root@2ff846eff126:/usr/local/apache2# cd htdocs
root@2ff846eff126:/usr/local/apache2/htdocs# ls
index.html
root@2ff846eff126:/usr/local/apache2/htdocs# nano index.html
root@2ff846eff126:/usr/local/apache2/htdocs# exit
exit
```



Now remove the image and the container.

Experiment - 3

Q1- Pull an Ubuntu image from the docker hub and check whether its in the system or not.

Run a container from it and install jdk on top of it.

Now, write a hello world java program. Save this entire setup as an image. Push the image to the docker hub repository. Now, download the image again and run a docker container from it.

Search the Ubuntu image.

```
root@cccfde17a678:/ Microsoft Windows [Version 10.0.22621.2215]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ACER>docker login
Authenticating with existing credentials...
Login Succeeded

Logging in with your password grants your terminal complete access to your account.
For better security, log in with a limited-privilege personal access token. Learn more at https://docs.docker.com/go/access-tokens/

C:\Users\ACER>docker search ubuntu
NAME                  DESCRIPTION                               STARS      OFFICIAL      AUTOMATED
ubuntu                Ubuntu is a Debian-based Linux operating sys...  16344      [OK]
websphere-liberty     WebSphere Liberty multi-architecture images ...  296       [OK]
open-liberty          Open Liberty multi-architecture images based...  61        [OK]
neurodebian           NeuroDebian provides neuroscience research s...  103      [OK]
ubuntu-debootstrap    DEPRECATED; use "ubuntu" instead            52        [OK]
ubuntu-upstart         DEPRECATED; as is Upstart (find other process  115      [OK]
```

Now create a container from it and get into its bash.

```
root@cccfde17a678:/ ~ + - x
latest: Pulling from library/ubuntu
b237fe92c417: Pull complete
Digest: sha256:ec050c32e4a6085b423d36ecd025c0d3ff00c38ab93a3d71a460ff1c44fa6d77
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest

What's Next?
View summary of image vulnerabilities and recommendations → docker scout quickview ubuntu

C:\Users\ACER>docker run -dt --name=ub1 ubuntu
cccfde17a678caf4dee6827cdf8f2581794857302c505c23fc79c4d8a7004b25

C:\Users\ACER>docker ps
CONTAINER ID   IMAGE      COMMAND       CREATED      STATUS      PORTS     NAMES
cccfde17a678   ubuntu     "/bin/bash"   4 seconds ago   Up 3 seconds   ub1

C:\Users\ACER>docker exec -it ub1 bash
root@cccfde17a678:/# apt-get update
Get:1 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
```

Now install idk using this command:

```
apt-get install -y default-jdk
```

```
[root@cccfde17a678:/] x + v
Get:18 http://archive.ubuntu.com/ubuntu jammy-backports/main amd64 Packages [49.2 kB]
Fetched 26.8 MB in 38s (703 kB/s)
Reading package lists... Done
root@cccfde17a678:/# apt-get install -y default-jdk
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
The following packages have unmet dependencies:
default-jdk : Depends: oracle-java8-installer (>= 8.0+112-0~jammy) but it is not going to be installed
E: Unmet dependencies. Try 'apt-get -f install' with no packages (or specify a solution)
```

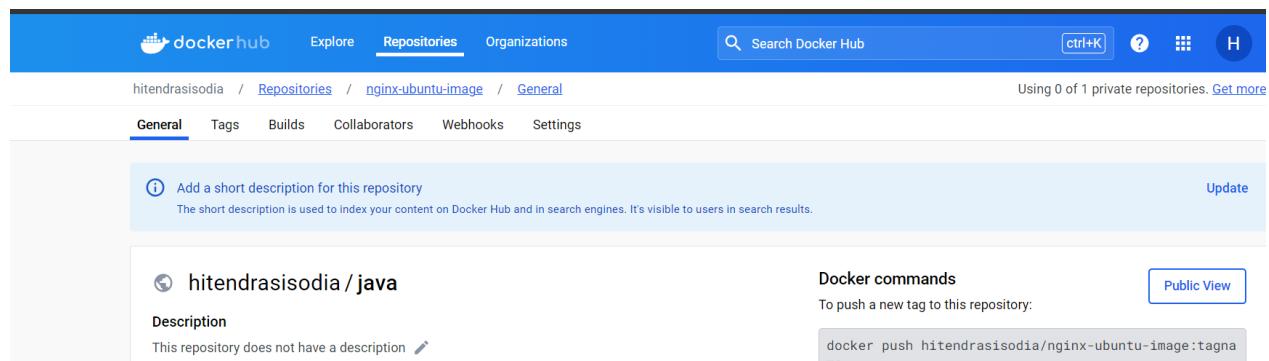
Now using the cat command create a Hello.java file.

```
root@cccfde17a678:/# cat >hello.java
class Hello{
    public static void main(String args[]){
        System.out.println("hello world");
    }
}
root@cccfde17a678:/# java Hello
Error: Could not find or load main class Hello
Caused by: java.lang.ClassNotFoundException: Hello
root@cccfde17a678:/# java hello.java
hello world
root@cccfde17a678:/# |
```



Using the commit and push command push this setup to the docker hub repository.

```
C:\Users\ACER>docker push hitendrasisodia18/java
Using default tag: latest
The push refers to repository [docker.io/hitendrasisodia18/java]
f39d2b7d442b: Pushed
bce45ce613d3: Pushed
latest: digest: sha256:21f175ef17be211632c6ddf175facfb117faccf968f53c6a329c3a6359f11ad6 size: 742
C:\Users\ACER>
```



The screenshot shows the Docker Hub interface for the repository `hitendrasisodia / java`. The repository has 0 private repositories. A description input field is present, and a Docker commands section shows the command `docker push hitendrasisodia/nginx-ubuntu-image:tagname`.

Now pull the image back.

```
C:\Users\ACER>docker pull hitendrasisodia/java:latest
latest: Pulling from hitendrasisodia18/java
Digest: sha256:21f175ef17be211632c6ddf175facfb117faccf968f53c6a329c3a6359f11ad6
Status: Downloaded newer image for hitendrasisodia/java:latest
docker.io/nikhil0803/java:latest
```

Finally Create a container from this pulled image.

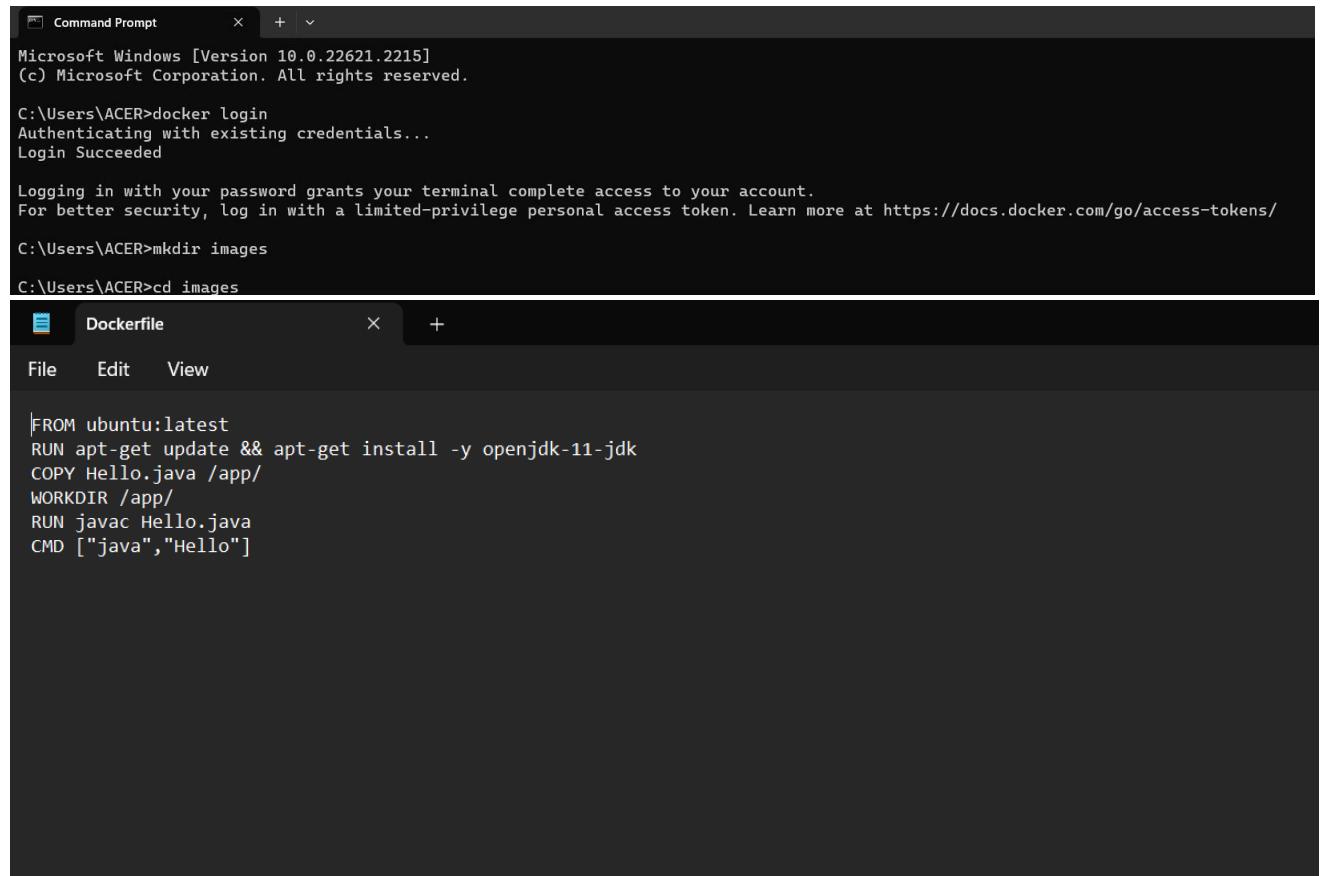
```
C:\Users\ACER>docker run -dt --name=ub2 hitendrasisodia/java:latest
8ca62c912458a549d912eb2d87e099f32d3f4b422e0b0e7d09e7cc6deb4a2a9c

C:\Users\ACER>docker exec -it ub2 bash
root@8ca62c912458:/# java hello.java
hello world
root@8ca62c912458:/#
```

Experiment - 4

Q1.Create a dockerfile and run. Create an image using the dockerfile. Run a container from it. Now , push the image into the dockerhub repository and pull it back. Now create a container from it.(also install jdk and write a Hello world program when creating the dockerfile)

Create a directory images and create the hello world program also create a docker file in it using the notepad.



The screenshot shows two windows side-by-side. The left window is a Command Prompt with the following text:

```
Microsoft Windows [Version 10.0.22621.2215]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ACER>docker login
Authenticating with existing credentials...
Login Succeeded

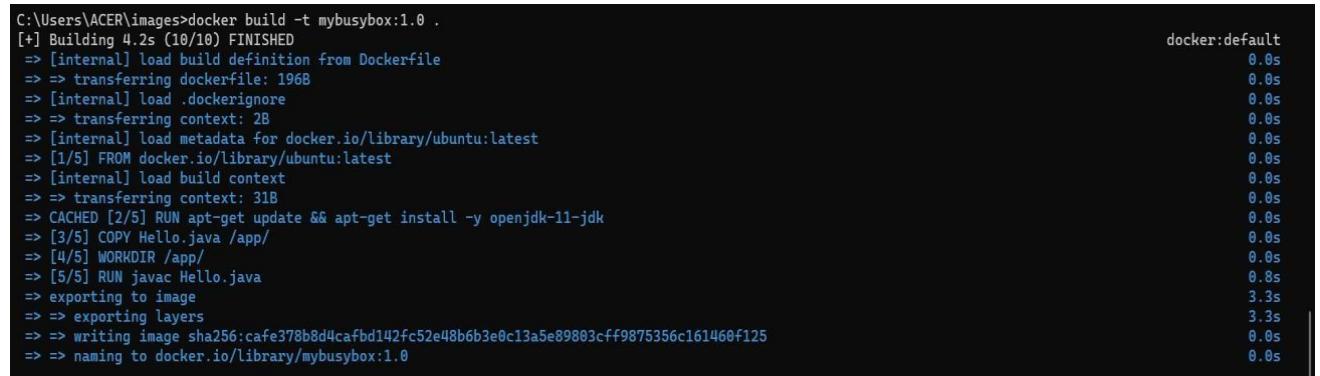
Logging in with your password grants your terminal complete access to your account.
For better security, log in with a limited-privilege personal access token. Learn more at https://docs.docker.com/go/access-tokens/

C:\Users\ACER>mkdir images
C:\Users\ACER>cd images
```

The right window is a Notepad titled "Dockerfile" with the following content:

```
FROM ubuntu:latest
RUN apt-get update && apt-get install -y openjdk-11-jdk
COPY Hello.java /app/
WORKDIR /app/
RUN javac Hello.java
CMD ["java","Hello"]
```

Now use the build command to create the image from the dockerfile.



The screenshot shows a Command Prompt window with the following text output:

```
C:\Users\ACER\images>docker build -t mybusybox:1.0 .
[+] Building 4.2s (10/10) FINISHED
--> [internal] load build definition from Dockerfile
--> transferring dockerfile: 196B
--> [internal] load .dockerignore
--> transferring context: 2B
--> [internal] load metadata for docker.io/library/ubuntu:latest
--> [1/5] FROM docker.io/library/ubuntu:latest
--> [internal] load build context
--> transferring context: 31B
--> CACHED [2/5] RUN apt-get update && apt-get install -y openjdk-11-jdk
--> [3/5] COPY Hello.java /app/
--> [4/5] WORKDIR /app/
--> [5/5] RUN javac Hello.java
--> exporting to image
--> --> exporting layers
--> --> writing image sha256:cafe378b8d4cafbd142fc52e48b6b3e0c13a5e89803cff9875356c161460f125
--> --> naming to docker.io/library/mybusybox:1.0
```

Now create a container from built image.

Also, make an image from it using commit command and push the image to the docker hubrepository

```
C:\Users\ACER\images>docker run -it --name=jdktest mybusybox:1.0
Hello, World!

C:\Users\ACER\images>docker commit jdktest hitendrasisodia18/jdktest
sha256:ed9f21326644e1e6bf65a1a2f5e768b672f8cf3d073461fac4e272066efaa46b

C:\Users\ACER\images>docker push hitendrasisodia18/jdktest
Using default tag: latest
The push refers to repository [docker.io/hitendrasisodia18/jdktest]
3b5b5b5545b0: Pushed
1cb1e9d2c2d3: Pushed
5f70bf18a086: Pushed
d953a7ae618e: Pushed
75cc79dc78c3: Pushed
bce45ce613d3: Mounted from hitendrasisodia18/java
latest: digest: sha256:21a40a456dc307b6f95b8b65b8664e93499f7305161bffb4d58b426b9b4a7a4 size: 1569

C:\Users\ACER\images>
```

Now, pull this image back.

```
C:\Users\ACER\images>docker pull hitendrasisodia18/jdktest
Using default tag: latest
latest: Pulling from hitendrasisodia18/jdktest
Digest: sha256:21a40a456dc307b6f95b8b65b8664e93499f7305161bffb4d58b426b9b4a7a4
Status: Image is up to date for hitendrasisodia18/jdktest
```

Again, create a container form this image.

```
See "docker run --help".

C:\Users\ACER\images>docker run -dt --name=jdktest1 hitendrasisodia18/jdktest
b13abe31e0f655121f8ac699a8e1b1af4bfe0e336b50c60e306ddef9414d40f9

C:\Users\ACER\images>docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
b13abe31e0f6 nikhil0803/jdktest "java Hello" 12 seconds ago Exited (0) 11 seconds ago jdktest1
74c682a33401 mybusybox:1.0 "java Hello" 19 minutes ago Exited (0) 19 minutes ago jdktest
8ca62c912458 nikhil0803/java "/bin/bash" 4 days ago Exited (137) 4 days ago ub2
2ff846eff126 httpd "httpd-foreground" 4 days ago Exited (0) 4 days ago mywebserver

C:\Users\ACER\images>
```

Experiment - 5

Q1. Use the `-v` and `--mount` flags to attach a volume and bind mount to the container.

Volumes-

--mount:

Create a container with volume using `--mount` flag

```
C:\Users\ACER>docker run -dt --name=newcon --mount source=newvol,target=/app\ nginx:latest
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
360eba32fa65: Pull complete
c5903f3678a7: Pull complete
27e923fb52d3: Pull complete
72de7dice3a4: Pull complete
94f34d60e454: Pull complete
e42dcfe1730b: Pull complete
907d1bb4e931: Pull complete
Digest: sha256:6926dd802f40e5e7257fded83e0d8030039642e4e10c4a98a6478e9c6fe06153
Status: Downloaded newer image for nginx:latest
756b6682054e48882c92a8ca908ff5c3f20bad1baa687eaaab46f99ae80da8bb

C:\Users\ACER>docker inspect newcon
```

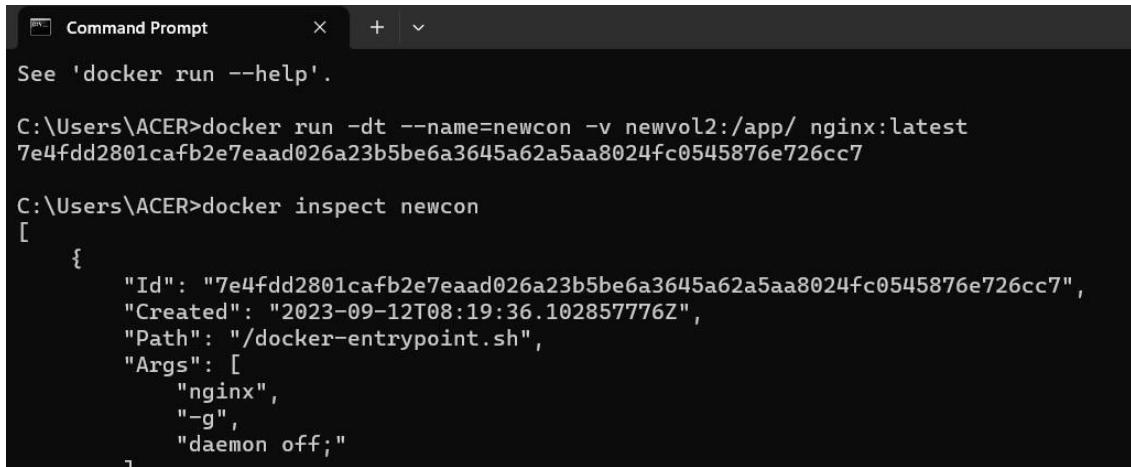
Inspect the container to view the attached volume

```
        "Name": "overlay2"
    },
    "Mounts": [
        {
            "Type": "volume",
            "Name": "newvol",
            "Source": "/var/lib/docker/volumes/newvol/_data",
            "Destination": "/app\\",
            "Driver": "local",
            "Mode": "z",
            "RW": true,
            "Propagation": ""
        }
    ]
}
```

Go into the container's bash and create a file in it.

```
C:\Users\ACER>docker exec -it newcon bash
root@756b6682054e:/# ls
'app\'  boot  docker-entrypoint.d  etc    lib    lib64   media  opt    root   sbin   sys    tmp    var
bin     dev   docker-entrypoint.sh  home   lib32  libx32  mnt    proc   run    srv    test.txt  usr
root@756b6682054e:/# cd 'app\' 
root@756b6682054e:/app\# cat >test.txt
hello
C:\Users\ACER>docker exec -it newcon bash
root@756b6682054e:/# ls var/lib/docker/docker/volume/-data
ls: cannot access 'var/lib/docker/docker/volume/-data': No such file or directory
root@756b6682054e:/# exit
exit
```

-v: Now again create a container with a volume using -v flag

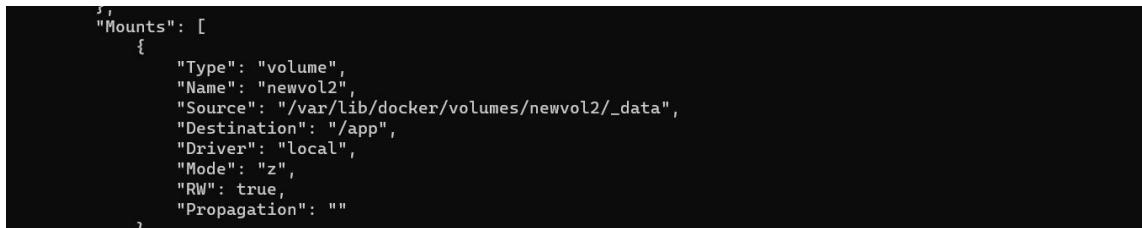


```
Command Prompt
See 'docker run --help'.

C:\Users\ACER>docker run -dt --name=newcon -v newvol2:/app/ nginx:latest
7e4fdd2801cafb2e7eaad026a23b5be6a3645a62a5aa8024fc0545876e726cc7

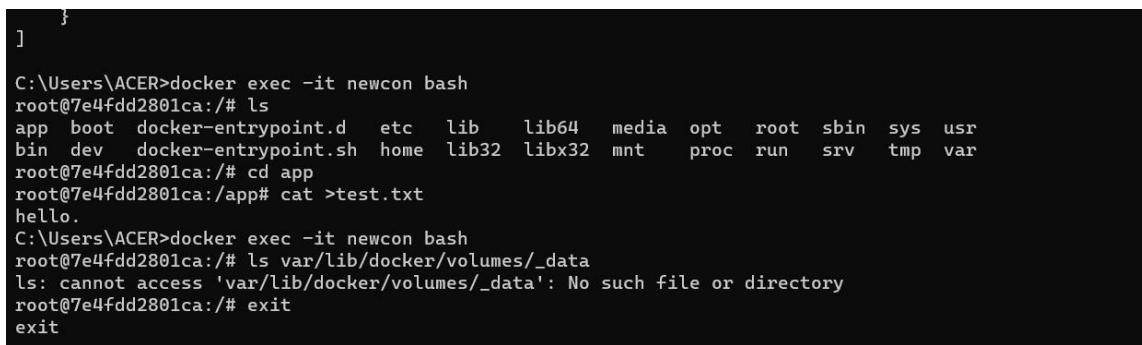
C:\Users\ACER>docker inspect newcon
[{"Id": "7e4fdd2801cafb2e7eaad026a23b5be6a3645a62a5aa8024fc0545876e726cc7", "Created": "2023-09-12T08:19:36.102Z", "Path": "/docker-entrypoint.sh", "Args": ["nginx", "-g", "daemon off;"]}
```

Inspect it to view the attached volume.



```
{
  "Mounts": [
    {
      "Type": "volume",
      "Name": "newvol2",
      "Source": "/var/lib/docker/volumes/newvol2/_data",
      "Destination": "/app",
      "Driver": "local",
      "Mode": "z",
      "RW": true,
      "Propagation": ""
    }
  ]
}
```

Go into the container's bash and create a file in it.



```
]
C:\Users\ACER>docker exec -it newcon bash
root@7e4fdd2801ca:/# ls
app  boot  docker-entrypoint.d  etc  lib  lib64  media  opt  root  sbin  sys  usr
bin  dev   docker-entrypoint.sh  home  lib32  libx32  mnt   proc  run   srv   tmp  var
root@7e4fdd2801ca:/# cd app
root@7e4fdd2801ca:/app# cat >test.txt
hello.
C:\Users\ACER>docker exec -it newcon bash
root@7e4fdd2801ca:/# ls var/lib/docker/volumes/_data
ls: cannot access 'var/lib/docker/volumes/_data': No such file or directory
root@7e4fdd2801ca:/# exit
exit
```

Bind Mounts-

-v: Create a container with Bind mount using -v flag

```
C:\Users\ACER>docker run -d --name=devtest -v "$(pwd)"/target,target=/app nginx:latest  
27b23656a6e7f7a54efe709fddc5394c6552df39b5cf6aa68e4d96bc31cc6ad5
```

Go into the container's bash and create a file in it.

```
C:\Users\ACER>docker exec -it devtest bash  
root@27b23656a6e7:/# ls  
'$(pwd)'  boot  docker-entrypoint.d  etc   lib    lib64  media  opt   root  sbin  sys   usr  
bin      dev   docker-entrypoint.sh  home  lib32  libx32 mnt   proc  run   srv   tmp   var  
root@27b23656a6e7:/# ls '$(pwd)'  
'target,target='  
root@27b23656a6e7:/# cd '$(pwd)'  
root@27b23656a6e7:/$(pwd)# cat >hello.txt  
hiieroot@27b23656a6e7:/$(pwd)# ls 'target,target='  
app  
root@27b23656a6e7:/$(pwd)# cd app  
bash: cd: app: No such file or directory  
root@27b23656a6e7:/$(pwd)# cd 'target,target='  
root@27b23656a6e7:/$(pwd)/target,target=# cd app  
root@27b23656a6e7:/$(pwd)/target,target=/app# cat >hello.txt  
hiieroot@27b23656a6e7:/$(pwd)/target,target=/app# ls app  
ls: cannot access 'app': No such file or directory  
root@27b23656a6e7:/$(pwd)/target,target=/app# ls app  
ls: cannot access 'app': No such file or directory  
root@27b23656a6e7:/$(pwd)/target,target=/app# ls  
hello.txt  
root@27b23656a6e7:/$(pwd)/target,target=/app# exit  
exit
```

--mount:

Create a container with bind mount using the --mount flag

```
C:\Users\ACER>docker exec -it devtest bash  
root@84ec6e9c56a2:/# ls  
app  boot  docker-entrypoint.d  etc   lib    lib64  media  opt   root  sbin  sys   usr  
bin  dev   docker-entrypoint.sh  home  lib32  libx32 mnt   proc  run   srv   tmp   var  
root@84ec6e9c56a2:/# cd app  
root@84ec6e9c56a2:/app# ls  
'3D Objects'  
  AppData  
'Application Data'  
'Cisco Packet Tracer 8.2.1'  
  Contacts  
  Cookies  
  Documents  
  Downloads  
  Favorites  
  Hello.java  
  Links  
'Local Settings'  
  Music  
'My Documents'  
  NTUSER.DAT  
  NTUSER.DAT{a2332f18-cdbf-11ec-8680-002248483d79}.TM.blf  
  NTUSER.DAT{a2332f18-cdbf-11ec-8680-002248483d79}.TMContainer000000000000000000000001.regtrans-ms  
  NTUSER.DAT{a2332f18-cdbf-11ec-8680-002248483d79}.TMContainer000000000000000000000002.regtrans-ms  
root@84ec6e9c56a2:/app# |
```

Experiment - 6

Q1. Check for an available network in the system. If there's one, create two container and try to ping in between them using their IPs.

Now create a custom bridge network and connect two containers to it. Then try to ping in between the containers using their names.

Check for available networks in the system.

```
C:\Users\ACER>docker login
Authenticating with existing credentials...
Login Succeeded

Logging in with your password grants your terminal complete access to your account.
For better security, log in with a limited-privilege personal access token. Learn more at https://docs.docker.com/go/access-tokens/

C:\Users\ACER>docker network ls
NETWORK ID     NAME      DRIVER      SCOPE
995910bfb590   bridge    bridge      local
ca7f5a87ad46   host      host       local
fbe7403d706c   none      null       local
```

Now create two containers.

```
C:\Users\ACER>docker run -dt --name=nettry1 ubuntu
ddaea9b3d4a2423c9c3ec7072975981e2d6d7571d3de6f4f59c2a9f9e990b428

C:\Users\ACER>docker run -dt --name=nettry2 ubuntu
363feb7e66715f4387d1f30d426871f26b8a9acb5426c18a73e87def7fc0ac3
```

Inspect them and note their IPs.

```
"Networks": {
    "bridge": {
        "IPAMConfig": null,
        "Links": null,
        "Aliases": null,
        "NetworkID": "995910bfb590c543a6f6fa8eb493bb827fb5b8e60cc79f201d541aa6c7a1778e",
        "EndpointID": "e484f10f97db160c7ad580ab65cb492594a78697d0691daee01946df2bdefa0d",
        "Gateway": "172.17.0.1",
        "IPAddress": "172.17.0.2",
        "IPPrefixLen": 16,
        "IPv6Gateway": "",
        "GlobalIPv6Address": "",
        "GlobalIPv6PrefixLen": 0,
        "MacAddress": "02:42:ac:11:00:02",
        "DriverOpts": null
    }
}
```

Now get into the bash of nettry1

```
C:\Users\ACER>docker exec -it nettry1 bash
root@aab40aa98524:/# apt-get install -y iputils-ping
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package iputils-ping
root@aab40aa98524:/# ping 172.17.0.3
bash: ping: command not found
root@aab40aa98524:/# apt-get install -y iputils-ping
Reading package lists... Done
```

Install the ping command.

```
root@aab40aa98524:/# apt-get install -y iputils-ping
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libcap2-bin libpam-cap
```

Now ping the second container using its IP.

```
Setting up iputils-ping (3.20211213-1) ...
root@aab40aa98524:/# ping 172.17.0.3
PING 172.17.0.3 (172.17.0.3) 56(84) bytes of data.
64 bytes from 172.17.0.3: icmp_seq=1 ttl=64 time=16.7 ms
64 bytes from 172.17.0.3: icmp_seq=2 ttl=64 time=0.109 ms
64 bytes from 172.17.0.3: icmp_seq=3 ttl=64 time=0.110 ms
64 bytes from 172.17.0.3: icmp_seq=4 ttl=64 time=0.142 ms
64 bytes from 172.17.0.3: icmp_seq=5 ttl=64 time=0.115 ms
^C
--- 172.17.0.3 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4133ms
rtt min/avg/max/mdev = 0.109/3.439/16.723/6.641 ms
root@aab40aa98524:/# |
```

Now get into the bash of nettry2

```
C:\Users\ACER>docker exec -it nettry2 bash
root@ecae6239671c:/# apt-get update
Get:1 http://archive.ubuntu.com/ubuntu jammy InRelease [270 kB]
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:5 http://security.ubuntu.com/ubuntu jammy/universe amd64 Packages [998 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy/main amd64 Packages [1792 kB]
Get:7 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 Packages [44.0 kB]
Get:8 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [1014 kB]
```

Install the ping command.

```
root@ecae6239671c:/# apt-get install -y iputils-ping
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libcap2-bin libpam-cap
The following NEW packages will be installed:
  libcap2-bin libpam-cap
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
```

Now ping the first container using its IP.

```
root@ecae6239671c:/# ping 172.17.0.2
PING 172.17.0.2 (172.17.0.2) 56(84) bytes of data.
64 bytes from 172.17.0.2: icmp_seq=1 ttl=64 time=0.141 ms
64 bytes from 172.17.0.2: icmp_seq=2 ttl=64 time=0.070 ms
64 bytes from 172.17.0.2: icmp_seq=3 ttl=64 time=0.094 ms
64 bytes from 172.17.0.2: icmp_seq=4 ttl=64 time=0.129 ms
64 bytes from 172.17.0.2: icmp_seq=5 ttl=64 time=0.111 ms
^C
--- 172.17.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4194ms
rtt min/avg/max/mdev = 0.070/0.109/0.141/0.025 ms
```

Now, try removing the default bridge.

```
C:\Users\ACER>docker network rm 995910bfb590
Error response from daemon: bridge is a pre-defined network and cannot be removed
```

Create a User-defined bridge.

```
C:\Users\ACER>docker network create --driver=bridge my-net
a8c97309c206434f4b4d17cf3d796b45ef1afeb9d414b2ba542c99e97c4e943c

C:\Users\ACER>docker network ls
NETWORK ID      NAME      DRIVER      SCOPE
995910bfb590   bridge    bridge      local
ca7f5a87ad46   host      host       local
a8c97309c206   my-net    bridge      local
fbe7403d706c   none      null       local
```

Inspect the created network.

```
C:\Users\ACER>docker network inspect my-net
[
  {
    "Name": "my-net",
    "Id": "a8c97309c206434f4b4d17cf3d796b45ef1afeb9d414b2ba542c99e97c4e943c",
    "Created": "2023-09-19T08:03:55.498924813Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "172.18.0.0/16",
          "Gateway": "172.18.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {},
    "Options": {},
    "Labels": {}
  }
]
```

Create two containers and connect them to the network.

```
C:\Users\ACER>docker run -dt --name=nettry1 ubuntu
f54909aff1f448dbad5ee52f324a4dea038431af6fba125f29147ae8dbeb27a4

C:\Users\ACER>docker run -dt --name=nettry2 ubuntu
140d96fa9c747b009b38864b9219d4e67334069f3dd746952029d98d8c295625
```

```
C:\Users\ACER>docker network connect my-net nettry1
C:\Users\ACER>docker network connect my-net nettry2
Error response from daemon: No such container: nettry2
C:\Users\ACER>docker network connect my-net nettry2
```

Now again inspect the network.

```
C:\Users\ACER>docker network inspect my-net
[
    {
        "Name": "my-net",
        "Id": "a8c97309c206434f4b4d17cf3d796b45ef1afeb9d414b2ba542c99e97c4e943c",
        "Created": "2023-09-19T08:03:55.498924813Z",
        "Scope": "local",
    },
    "ConfigOnly": false,
    "Containers": {
        "140d96fa9c747b009b38864b9219d4e67334069f3dd746952029d98d8c295625": {
            "Name": "nettry2",
            "EndpointID": "8d86ee7684943abc72be2146034dd9b7fd71849c4ba9f519501860c9f40407c8",
            "MacAddress": "02:42:ac:12:00:03",
            "IPv4Address": "172.18.0.3/16",
            "IPv6Address": ""
        },
        "f54909aff1f448dbad5ee52f324a4dea038431af6fba125f29147ae8dbeb27a4": {
            "Name": "nettry1",
            "EndpointID": "37ade74218f4ed1c9dc976d8ab34a75d82b86993eb679bd8f5b187f9f9ff7d3d",
            "MacAddress": "02:42:ac:12:00:02",
            "IPv4Address": "172.18.0.2/16",
            "IPv6Address": ""
        }
    }
]
```

Now get into the bash of nettry1

```
C:\Users\ACER>docker exec -it nettry1 bash
root@f54909aff1f4:/# apt-get update
Get:1 http://archive.ubuntu.com/ubuntu jammy InRelease [270 kB]
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:4 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 Packages [109 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy/restricted amd64 Packages [164 kB]
```

Install the ping command.

```
Reading package lists... Done
root@f54909aff1f4:/# apt-get install -y iputils-ping
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libcap2-bin libpam-cap
The following NEW packages will be installed.
```

Now ping the second container using its name.

```
root@f54909aff1f4:/# ping nettry2
PING nettry2 (172.18.0.3) 56(84) bytes of data.
64 bytes from nettry2.my-net (172.18.0.3): icmp_seq=1 ttl=64 time=0.167 ms
64 bytes from nettry2.my-net (172.18.0.3): icmp_seq=2 ttl=64 time=0.121 ms
64 bytes from nettry2.my-net (172.18.0.3): icmp_seq=3 ttl=64 time=0.153 ms
^C
--- nettry2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.121/0.147/0.167/0.019 ms
root@f54909aff1f4:/# exit
```

Now get into the bash of nettry2

```
C:\Users\ACER>docker exec -it nettry2 bash
root@140d96fa9c74:/# apt-get update
Get:1 http://archive.ubuntu.com/ubuntu jammy InRelease [270 kB]
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [110 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy-backports InRelease [110 kB]
Install the ping command.
```

```
root@140d96fa9c74:/# apt-get install -y iputils-ping
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libcap2-bin libpam-cap
The following NEW packages will be installed:
  iputils-ping libcap2-bin libpam-cap
0 upgraded, 3 newly installed, 0 to remove and 6 not upgraded.
Need to get 76.8 kB of archives.
After this operation, 200 kB of additional disk space will be used.
```

Now ping the first container using its name.

```
Setting up iputils-ping (3.20211210-1) ...
root@140d96fa9c74:/# ping nettry1
PING nettry1 (172.18.0.2) 56(84) bytes of data.
64 bytes from nettry1.my-net (172.18.0.2): icmp_seq=1 ttl=64 time=0.102 ms
64 bytes from nettry1.my-net (172.18.0.2): icmp_seq=2 ttl=64 time=0.159 ms
64 bytes from nettry1.my-net (172.18.0.2): icmp_seq=3 ttl=64 time=0.129 ms
^C
--- nettry1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2027ms
rtt min/avg/max/mdev = 0.102/0.130/0.159/0.023 ms
root@140d96fa9c74:/# exit
```

Disconnect the two containers and remove the user-defined network

```
C:\Users\ACER>docker network disconnect my-net nettry1
C:\Users\ACER>docker network disconnect my-net nettry2
C:\Users\ACER>docker network rm my-net
```

Experiment - 7

Q1: Compose a web program using Dockercompose.

Create a directory named compositest.

```
PS C:\Users\ACER> mkdir compositest

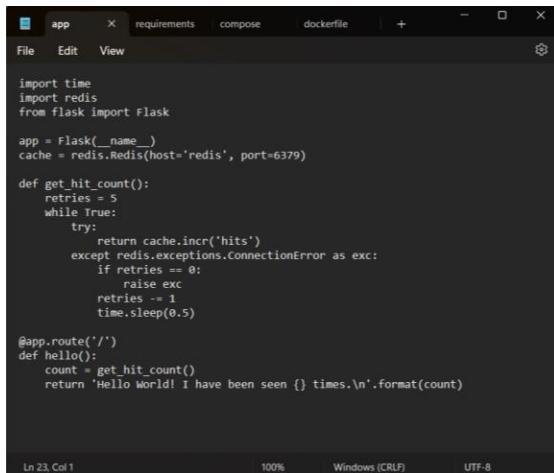
Directory: C:\Users\ACER

Mode                LastWriteTime         Length Name
----                -----         -----    Name
d----        9/19/2023   1:45 PM           0    compositest

PS C:\Users\ACER> cd compositest
```

In this directory, create the following file:

app.py



```
app app requirements compose dockerfile | + - x
File Edit View
import time
import redis
from flask import Flask

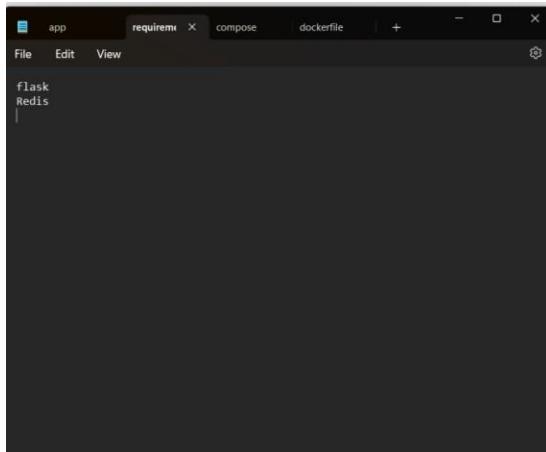
app = Flask(__name__)
cache = redis.Redis(host='redis', port=6379)

def get_hit_count():
    retries = 5
    while True:
        try:
            return cache.incr('hits')
        except redis.exceptions.ConnectionError as exc:
            if retries == 0:
                raise exc
            retries -= 1
            time.sleep(0.5)

@app.route('/')
def hello():
    count = get_hit_count()
    return 'Hello World! I have been seen {} times.\n'.format(count)

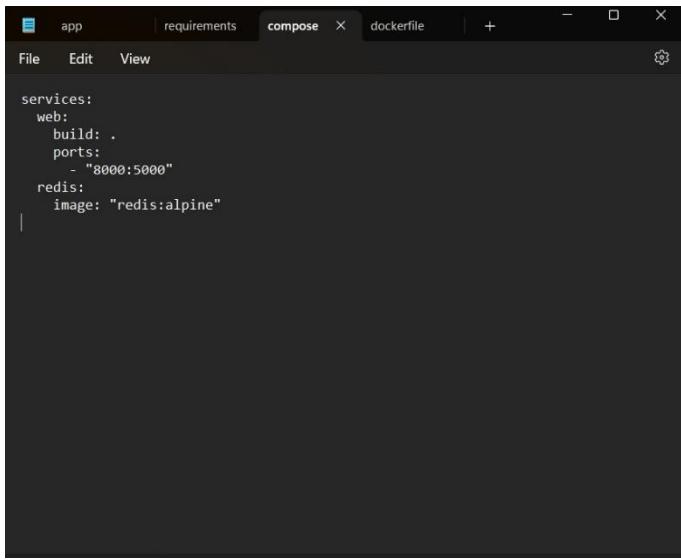
Ln 23, Col 1          100%      Windows (CRLF)      UTF-8
```

requirements.txt



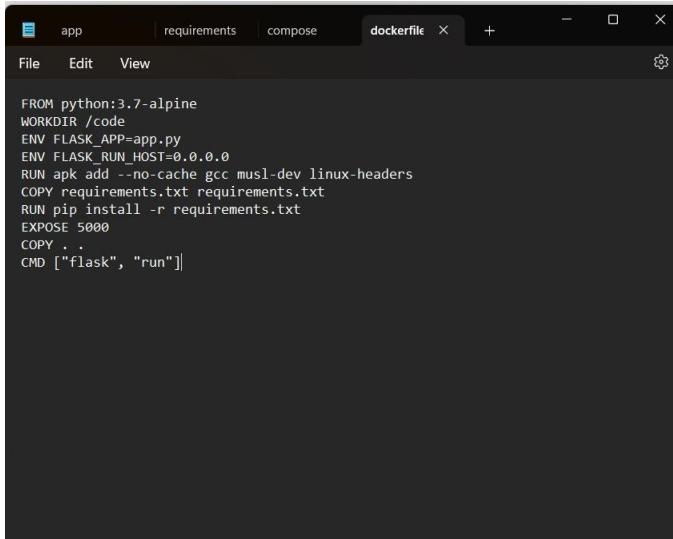
```
app requirements compose dockerfile | + - x
File Edit View
flask
Redis
|
```

compose.YAML



```
app requirements compose dockerfile
File Edit View
services:
  web:
    build: .
    ports:
      - "8000:5000"
  redis:
    image: "redis:alpine"
```

dockerfile



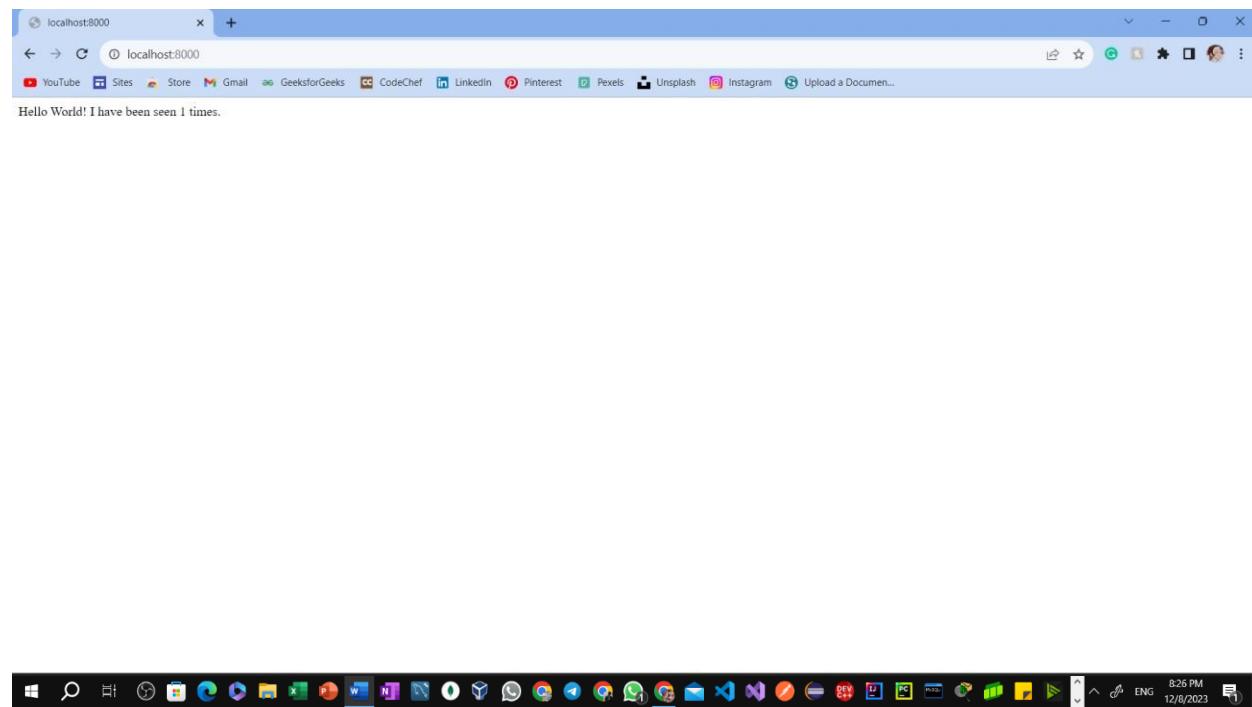
```
FROM python:3.7-alpine
WORKDIR /code
ENV FLASK_APP=app.py
ENV FLASK_RUN_HOST=0.0.0.0
RUN apk add --no-cache gcc musl-dev linux-headers
COPY requirements.txt requirements.txt
RUN pip install -r requirements.txt
EXPOSE 5000
COPY . .
CMD ["flask", "run"]
```

Now run this entire set up using compose up.

```
Command Prompt x Windows PowerShell x + - x
=> ==> transferring context: 1.07kB
=> CACHED [web 2/6] WORKDIR /code
=> CACHED [web 3/6] RUN apk add --no-cache gcc musl-dev linux-headers
=> ERROR [web 4/6] COPY requirements.txt requirements.txt
=====
> [web 4/6] COPY requirements.txt requirements.txt:
=====

failed to solve: failed to compute cache key: failed to calculate checksum of ref 7398cbfa-6c28-4aa3-b6d3-20dfdf7f1565:t56da9y9qqju4jgffnowecaki: "/requirements.txt": not found
PS C:\Users\ACER\composetest> docker compose up
[+] Building 97.9s (11/11) FINISHED
  => [web internal] load .dockerrigore
  => ==> transferring context: 2B
  => [web internal] load build definition from dockerfile
  => ==> transferring dockerfile: 299B
  => [web internal] load metadata for docker.io/library/python:3.7-alpine
  => [web internal] load build context
  => ==> transferring context: 182B
  => [web 1/6] FROM docker.io/library/python:3.7-alpine@sha256:9d9b05fc8acd85a9fc0dalda1a8e90f76b88bd36fab8f57c4c7ef027fbcc9
  => => resolve docker.io/library/python:3.7-alpine@sha256:9d9b05fc8acd85a9fc0dalda1a8e90f76b88bd36fab8f57c4c7ef027fbcc9
  => sha256:9d9b05fc8acd85a9fc0dalda1a8e90f76b88bd36fab8f57c4c7ef027fbcc9 1.65kB / 1.65kB
  => sha256:0c844a18f7b7177cd835de934a7801ba34708117321906a2f657cf7f67c24 1.37kB / 1.37kB
  => sha256:1b091eba55bd12af5d3b9a1fc5d9fe95751ffbf789c2ce5e530e6a66f535d1f8 6.87kB / 6.87kB
  => sha256:66e1d5e78e12aa8a632b88bf2a6eb60b4aff9ee8a6ca52e27f51f7b1be 622.31kB / 622.31kB
  => sha256:66bebda5a2f787efebba7f72fc4fa9d6cea1ab46679fc6292458d4d43722f2c 18.94MB / 18.94MB
  => sha256:f26771d857d7bfe2ced24f357e8391127f30ec6960d0c1240babf76a8391466 243B / 243B
  => sha256:9125851493e0d48767e4af3b4a1ada72287081af779634df29f30ac552056 2.85MB / 2.85MB
  =>> extracting sha256:66e1d5e78e12aa8a632b88bf2a6eb60b4aff9ee8a6ca52e27f51f7b1be
  =>> extracting sha256:66bebda5a2f787efebba7f72fc4fa9d6cea1ab46679fc6292458d4d43722f2c
  =>> extracting sha256:f26771d857d7bfe2cd624f357e8391127f30ec6960d0c1240babf76a8391466
  =>> extracting sha256:9125851493e0d48767e4aff3b4a1ada72287081af779634df29f30ac552056
  => [web 2/6] WORKDIR /code
  => [web 3/6] RUN apk add --no-cache gcc musl-dev linux-headers
  => [web 4/6] COPY requirements.txt requirements.txt
  => [web 5/6] RUN pip install -r requirements.txt
  => [web 6/6] COPY . .
  => [web] exporting to image
  =>> exporting layers
  =>> writing image sha256:6b564d9ce16f57784cc0bcf24508c8bb9562f5492f3fedad61b7a14170f80b6b
  =>> naming to docker.io/library/composetest-web
```

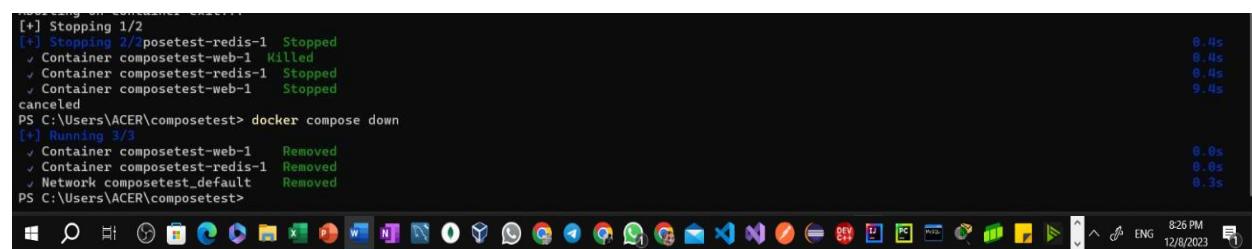
Now the browser go to the localhost:8000



The number increases every time we refresh the page.



Stop and remove this setup using compose down



Experiment - 8 - Demonstration of Docker Swarm

The screenshot shows the AWS CloudWatch Log Stream interface. At the top, there is a green success message: "Successfully initiated launch of instances (i-0f33af4070d5d798c, i-0f780df5bc7808216, i-03a582522dd6d82d0)". Below this, there is a "Launch log" button. The main area is titled "Next Steps" and contains several cards with links to other AWS services: "Create billing and free tier usage alerts", "Connect to your instance", "Connect an RDS database", "Create EBS snapshot policy", "Manage detailed monitoring", "Create Load Balancer", "Create AWS budget", and "Manage CloudWatch alarms". At the bottom, there are links for "CloudShell", "Feedback", and copyright information: "© 2023, Amazon Web Services India Private Limited or its affiliates. Privacy Terms Cookie preferences".

```
ubuntu@ip-172-31-87-243:~$ apt-get update
Reading package lists... Done
E: Could not open lock file /var/lib/apt/lists/lock - open (13: Permission denied)
E: Unable to lock directory /var/lib/apt/lists/
W: Problem unlinking the file /var/cache/apt/pkgcache.bin - RemoveCaches (13: Permission denied)
W: Problem unlinking the file /var/cache/apt/srcpkgcache.bin - RemoveCaches (13: Permission denied)
ubuntu@ip-172-31-87-243:~$ sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-en [5652 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 c-n-f Metadata [286 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [217 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse Translation-en [112 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 c-n-f Metadata [8372 B]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1016 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [228 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [15.6 kB]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [905 kB]
```

```

etting up dnsmasq-base (2.86-1.lubuntu0.3) ...
etting up runc (1.1.7-0ubuntu1~22.04.1) ...
etting up dns-root-data (2021011101) ...
etting up bridge-utils (1.7-1lubuntu3) ...
etting up pigz (2.6-1) ...
etting up containerd (1.7.2-0ubuntu1~22.04.1) ...
reated symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /lib/systemd/system/containerd.service.
etting up ubuntu-fan (0.12.16) ...
reated symlink /etc/systemd/system/multi-user.target.wants/ubuntu-fan.service → /lib/systemd/system/ubuntu-fan.service.
etting up docker.io (24.0.5-0ubuntu1~22.04.1) ...
dding group `docker' (GID 122) ...
one.
reated symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
reated symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
rocessing triggers for dbus (1.12.20-2ubuntu4.1) ...
rocessing triggers for man-db (2.10.2-1) ...
canning processes...
canning linux images...

unning kernel seems to be up-to-date.

o services need to be restarted.

o containers need to be restarted.

o user sessions are running outdated binaries.

o VM guests are running outdated hypervisor (qemu) binaries on this host.

```

The screenshot shows the AWS CloudWatch Metrics interface. A line chart displays the 'CPUUtilization' metric for an instance over a 24-hour period. The Y-axis represents CPU utilization from 0% to 100%, and the X-axis represents time from 00:00 to 23:00 UTC. The chart shows a general upward trend in CPU usage throughout the day. A tooltip at the top right of the chart area highlights a peak utilization of 100% at approximately 10:00 UTC.

```

hitendrasioda/nginx-ubuntu: ~ Console Home | Console Home + 
us-east-1.console.aws.amazon.com/console/home?nc2=h_ct&region=us-east-1&src=header-signin#
Mail YouTube Axis SBI YES Amazon Flipkart Drive IRCTC GitHub SAP PY PW Chat GPT AWS GPT 4 Strivers A2Z > | All Bookmarks
aws Services Search [Alt+S]
Setting up runc (1.1.7~Ubuntu-0.22.04.1) ...
Setting up dns-root-data (2021011101) ...
Setting up bridge-utils (1.7~Ubuntu3) ...
Setting up pigz (2.6-1) ...
Setting up containerd (1.7.2~Ubuntu1~22.04.1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /lib/systemd/system/containerd.service.
Setting up ubuntu-fan (0.12.16) ...
Created symlink /etc/systemd/system/multi-user.target.wants/ubuntu-fan.service → /lib/systemd/system/ubuntu-fan.service.
Setting up docker.io (24.0.5~Ubuntu1~22.04.1) ...
Adding group 'docker' (GID 122) ...
Done.
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
processing triggers for dbus (1.12.20~Ubuntu4.1) ...
Processing triggers for man-db (2.10.2-1) ...
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-172-31-85-89:~$ sudo docker swarm join --token SWMTKN-1-414qkzaazacc4rzwfhsyei6xbx1sg38c71v303n9ky1x66opjq-5tr4g0mlbjhhcscun829rhja2 34.201.46.100:2377
This node joined a swarm as a worker.
ubuntu@ip-172-31-85-89:~$ []

i-04492b987fab5de15 (Server 3)
PublicIPs: 35.171.162.208 PrivateIPs: 172.31.85.89

```

CloudShell Feedback © 2025, Amazon Web Services India Private Limited or its affiliates. Privacy Terms Cookie preferences

```

hitendrasioda/nginx-ubuntu: ~ Console Home | Console Home + 
us-east-1.console.aws.amazon.com/console/home?nc2=h_ct&region=us-east-1&src=header-signin#
Mail YouTube Axis SBI YES Amazon Flipkart Drive IRCTC GitHub SAP PY PW Chat GPT AWS GPT 4 Strivers A2Z > | All Bookmarks
aws Services Search [Alt+S]
Setting up ubuntu-fan (0.12.16) ...
Created symlink /etc/systemd/system/multi-user.target.wants/ubuntu-fan.service → /lib/systemd/system/ubuntu-fan.service.
Setting up docker.io (24.0.5~Ubuntu1~22.04.1) ...
Adding group 'docker' (GID 122) ...
Done.
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Processing triggers for dbus (1.12.20~Ubuntu4.1) ...
Processing triggers for man-db (2.10.2-1) ...
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-172-31-91-73:~$ [[200-docker swarm join --token SWMTKN-1-414qkzaazacc4rzwfhsyei6xbx1sg38c71v303n9ky1x66opjq-5tr4g0mlbjhhcscun829rhja2 34.201.46.100:2377-
docker: command not found
ubuntu@ip-172-31-91-73:~$ sudo docker swarm join --token SWMTKN-1-414qkzaazacc4rzwfhsyei6xbx1sg38c71v303n9ky1x66opjq-5tr4g0mlbjhhcscun829rhja2 34.201.46.100:2377a
Error response from daemon: could not find local IP address: dial udp: address udp/2377a: unknown port
ubuntu@ip-172-31-91-73:~$ sudo docker swarm join --token SWMTKN-1-414qkzaazacc4rzwfhsyei6xbx1sg38c71v303n9ky1x66opjq-5tr4g0mlbjhhcscun829rhja2 34.201.46.100:2377a
Error response from daemon: could not find local IP address: dial udp: address udp/2377a: unknown port
ubuntu@ip-172-31-91-73:~$ sudo docker swarm join --token SWMTKN-1-414qkzaazacc4rzwfhsyei6xbx1sg38c71v303n9ky1x66opjq-5tr4g0mlbjhhcscun829rhja2 34.201.46.100:2377
This node joined a swarm as a worker.
ubuntu@ip-172-31-91-73:~$ []

i-0c3ebb4b15bacbedd (Server 2)
PublicIPs: 100.26.206.199 PrivateIPs: 172.31.91.73

```

```

ubuntu@ip-172-31-87-243:~$ sudo docker service create --replicas=2 --name=HelloWorld alpine ping docker.com
t2l1nk0so2df15j172t8xxvi
overall progress: 2 out of 2 tasks
1/2: running [=====>]
2/2: running [=====>]
verify: service converged
ubuntu@ip-172-31-87-243:~$ docker service ls
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.24/services?status=true"
al unix /var/run/docker.sock: connect: permission denied
ubuntu@ip-172-31-87-243:~$ sudo docker service ls
ID           NAME      MODE      REPLICAS  IMAGE          PORTS
t2l1nk0so2d  HelloWorld replicated  2/2    alpine:latest
ubuntu@ip-172-31-87-243:~$ sudo docker service inspect helloworld
[]
Status: Error: no such service: helloworld, Code: 1
ubuntu@ip-172-31-87-243:~$ sudo docker service inspect HelloWorld
[
  {
    "ID": "t2l1nk0so2df15j172t8xxvi",
    "Version": {
      "Index": 21
    },
    "CreatedAt": "2023-10-03T08:47:12.647346611Z",
    "UpdatedAt": "2023-10-03T08:47:12.647346611Z",
    "Spec": {
      "TaskTemplate": {
        "ContainerSpec": {
          "Image": "alpine:latest",
          "Labels": {
            "com.docker.swarm-mode": "true"
          }
        }
      }
    }
  }
]

```

```
ubuntu@ip-172-31-87-243:~$ sudo docker service inspect --pretty HelloWorld
ID: t2ltink0so2df15j172t8xxvi
Name: HelloWorld
Service Mode: Replicated
Replicas: 2
Placement:
UpdateConfig:
  Parallelism: 1
  On failure: pause
  Monitoring Period: 5s
  Max failure ratio: 0
  Update order: stop-first
RollbackConfig:
  Parallelism: 1
  On failure: pause
  Monitoring Period: 5s
  Max failure ratio: 0
  Rollback order: stop-first
ContainerSpec:
  Image: alpine:latest@sha256:eece025e432126ce23f223450a0326fbebde39cdf496a85d8c016293fc851978
  Args: ping docker.com
  Init: false
Resources:
Endpoint Mode: vip
```

```
rm Remove one or more containers
rmi Remove one or more images
save Save one or more images to a tar archive (streamed to STDOUT by default)
start Start one or more stopped containers
stats Display a live stream of container(s) resource usage statistics
stop Stop one or more running containers
tag Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
top Display the running processes of a container
unpause Unpause all processes within one or more containers
update Update configuration of one or more containers
wait Block until one or more containers stop, then print their exit codes

Global Options:
  --config string          Location of client config files (default "/root/.docker")
  -c, --context string     Name of the context to use to connect to the daemon (overrides DOCKER_HOST env var and default context set with "docker context use")
  -D, --debug              Enable debug mode
  -H, --host list           Daemon socket to connect to
  -l, --log-level string   Set the logging level ("debug", "info", "warn", "error", "fatal") (default "info")
  --tls                     Use TLS; implied by --tlsv1
  --tlscacert string       Trust certs signed only by this CA (default "/root/.docker/ca.pem")
  --tlscert string          Path to TLS certificate file (default "/root/.docker/cert.pem")
  --tlskey string           Path to TLS key file (default "/root/.docker/key.pem")
  --tlsv1                   Use TLS and verify the remote
  -v, --version             Print version information and quit

Run 'docker COMMAND --help' for more information on a command.

Or more help on how to use Docker, head to https://docs.docker.com/go/guides/
```

```
ubuntu@ip-172-31-87-243:~$ sudo docker service scale HelloWorld=5
HelloWorld scaled to 5
overall progress: 5 out of 5 tasks
1/5: running  [=====>]
2/5: running  [=====>]
3/5: running  [=====>]
4/5: running  [=====>]
5/5: running  [=====>]
verify: Service converged
ubuntu@ip-172-31-87-243:~$ █
```

```
ubuntu@ip-172-31-87-243:~$ sudo docker service ls
ID          NAME      MODE      REPLICAS      IMAGE      PORTS
t2ltink0so2d  HelloWorld  replicated  5/5        alpine:latest
ubuntu@ip-172-31-87-243:~$ █
```

```

mountu@ip-172-31-87-243:~$ sudo docker service ls
ID           NAME      MODE      REPLICAS  IMAGE
t2l1nkoos2d  HelloWorld replicated 5/5    alpine:latest
ubuntu@ip-172-31-87-243:~$ sudo docker node promote 100.26.206.199
Error response from daemon: node 100.26.206.199 not found
ubuntu@ip-172-31-87-243:~$ sudo docker ps
CONTAINER ID   IMAGE      COMMAND     CREATED      STATUS      PORTS      NAMES
9d8821d8a3d9  alpine:latest "ping docker.com"  20 minutes ago  Up 20 minutes   HelloWorld.2.q31xeacz9hb9a5ot0q769j2n3
ubuntu@ip-172-31-87-243:~$ sudo docker node ls
ID           HOSTNAME      STATUS      AVAILABILITY  MANAGER STATUS  ENGINE VERSION
sbar78by8j9jwv46dlq773v6x  ip-172-31-85-89  Ready       Active        24.0.5
y0c7qhb9a17r7tyd3yhlwokbu * ip-172-31-87-243  Ready       Active        Leader        24.0.5
vy06g9dsbxdqn1r80dom1bnug ip-172-31-91-73  Ready       Active        24.0.5
ubuntu@ip-172-31-87-243:~$ sudo docker node promote ip-172-31-85-89
Node ip-172-31-85-89 promoted to a manager in the swarm.
ubuntu@ip-172-31-87-243:~$ sudo docker node ls
ID           HOSTNAME      STATUS      AVAILABILITY  MANAGER STATUS  ENGINE VERSION
sbar78by8j9jwv46dlq773v6x  ip-172-31-85-89  Ready       Active        Reachable      24.0.5
y0c7qhb9a17r7tyd3yhlwokbu * ip-172-31-87-243  Ready       Active        Leader        24.0.5
vy06g9dsbxdqn1r80dom1bnug ip-172-31-91-73  Ready       Active        24.0.5
ubuntu@ip-172-31-87-243:~$ []

```

```

0.0 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

*** System restart required ***
Last login: Tue Oct 17 08:01:37 2023 from 18.206.107.29
ubuntu@ip-172-31-85-89:~$ sudo docker node ls
ID           HOSTNAME      STATUS      AVAILABILITY  MANAGER STATUS  ENGINE VERSION
sbar78by8j9jwv46dlq773v6x * ip-172-31-85-89  Ready       Active        Leader        24.0.5
y0c7qhb9a17r7tyd3yhlwokbu ip-172-31-87-243  Down        Active        24.0.5
vy06g9dsbxdqn1r80dom1bnug ip-172-31-91-73  Down        Active        24.0.5
ubuntu@ip-172-31-85-89:~$ sudo docker node update --availability drain ^C
ubuntu@ip-172-31-85-89:~$ sudo docker node update --availability drain sbar78by8j9jwv46dlq773v6x
sbar78by8j9jwv46dlq773v6x
ubuntu@ip-172-31-85-89:~$ sudo docker node ls
ID           HOSTNAME      STATUS      AVAILABILITY  MANAGER STATUS  ENGINE VERSION
sbar78by8j9jwv46dlq773v6x * ip-172-31-85-89  Ready       Drain        Leader        24.0.5
y0c7qhb9a17r7tyd3yhlwokbu ip-172-31-87-243  Down        Active        24.0.5
vy06g9dsbxdqn1r80dom1bnug ip-172-31-91-73  Down        Active        24.0.5
ubuntu@ip-172-31-85-89:~$ []

```