**Explore Docker commands for content management**

**Install Docker in instance**

**Create Ubuntu Machine on AWS**

Connect using git bash

Go to Root Account

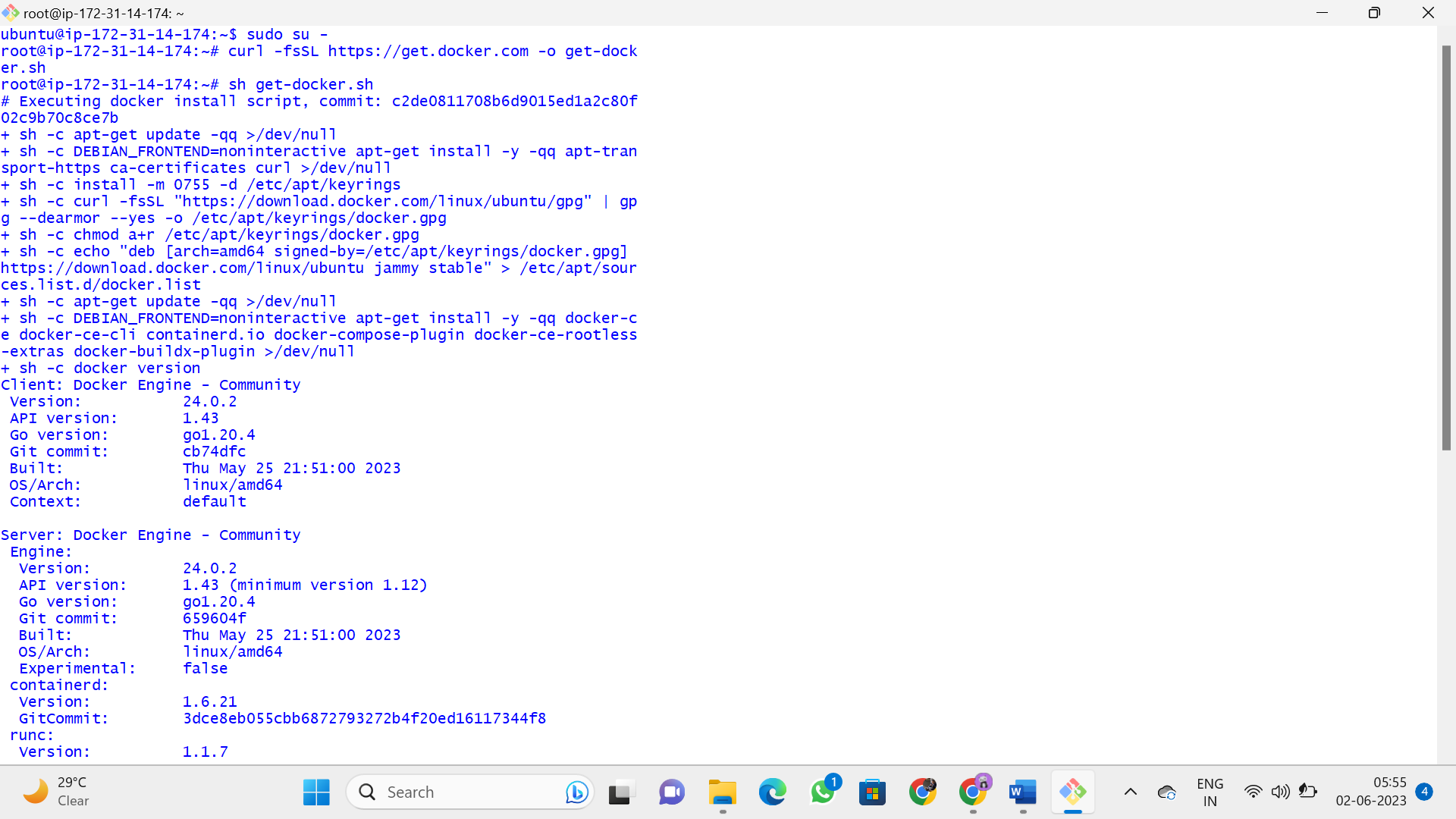
**$ sudo su –**

Open browser (<https://get.docker.com/>) go to that link

Copy the commands and paste it in terminal

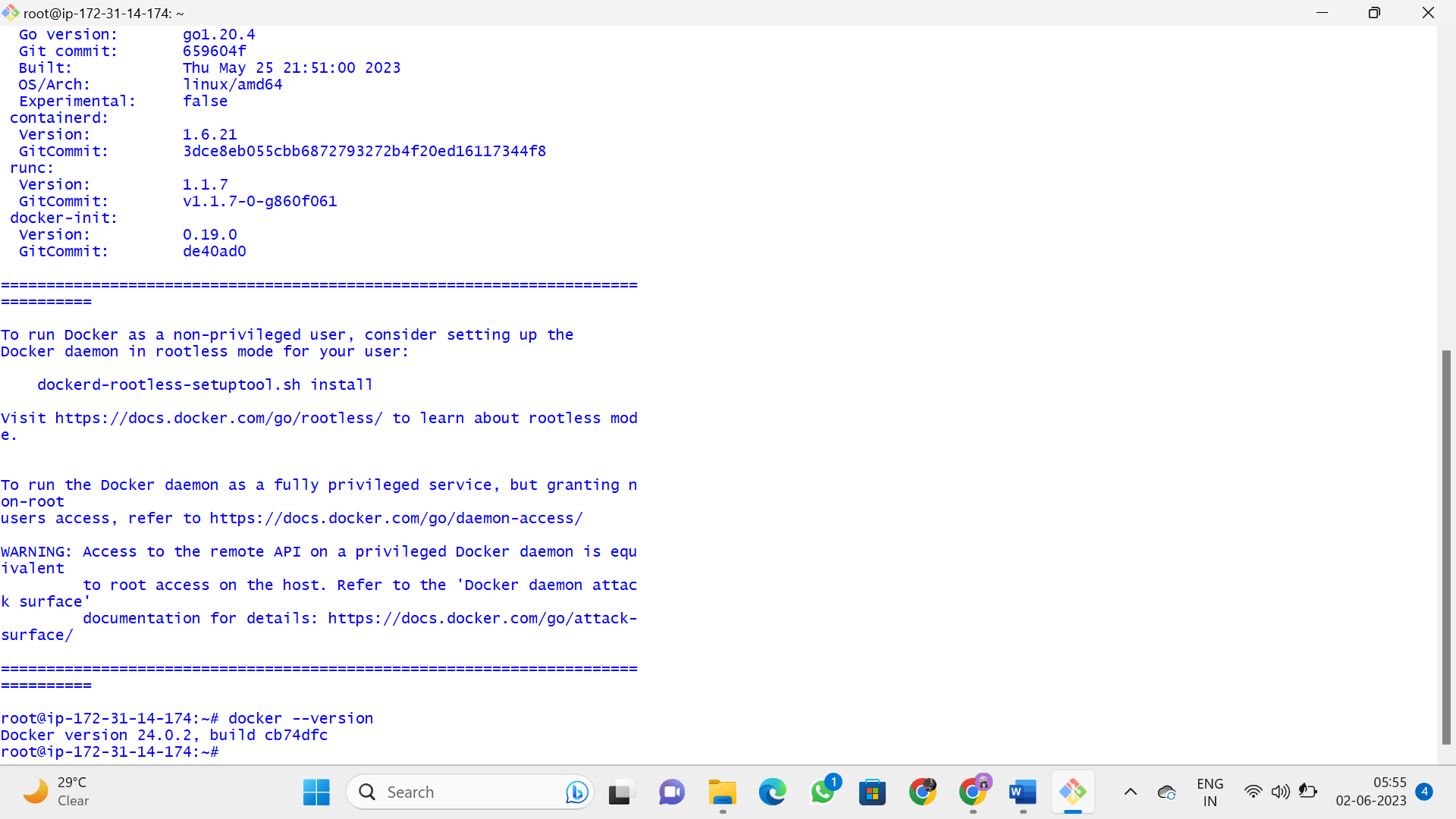
**# curl -fsSL https://get.docker.com -o get-docker.sh ( this will download shell script in the machine)**

**# sh get-docker.sh ( This will execute the shell script, which will install docker )**



How to check the docker is installed or not

**# docker –version**



We should be comfortable with four terms

1) **Docker Images**

Combinations of binaries / libraries which are necessary for one software application.

2) **Docker Containers**

When image is installed and in comes into running condition, it is called container.

3) **Docker Host**

Machine on which docker is installed, is called as Docker host.

4) **Docker Client**

Terminal used to run docker run commands ( Git bash )

On linux machine, git bash will work like docker client.

**Docker Commands**

**--------------------**

**Working on Images**

**-------------------------**

**1 To download a docker image**

# docker pull image\_name

**2 To see the list of docker images**

# docker image ls

(or)

# docker images

**3 To delete a docker image from docker host**

# docker rmi image\_name/image\_id

**4) To upload a docker image into docker hub**

# docker push image\_name

**5) To tag an image**

# docker tag image\_name ipaddress\_of\_local\_registry:5000/image\_name

**6) To build an image from a customised container**

# docker commit container\_name/container\_id new\_image\_name

**7) To create an image from docker file**

# docker build -t new\_image\_name

**8) To search for a docker image**

# docker search image\_name

**9) To delete all images that are not attached to containers**

# docker system prune -a

**Working on containers**

-----------------------------

10) **To see the list of all running continers**

# docker container ls

**11) To see the list of running and stopped containers**

# docker ps -a

**12) To start a container**

# docker start container\_name/container\_id

**13) To stop a running container**

# docker stop container\_name/container\_id

14) **To restart a running container**

# docker restart container\_name/container\_id

To restart after 10 seconds

# docker restart -t 10 container\_name/container\_id

**15) To delete a stopped container**

# docker rm container\_name/container\_id

**16) To delete a running container**

# docker rm -f container\_name/container id

**17) To stop all running containers**

# docker stop $(docker ps -aq)

**18) To restart all containers**

# docker restart $(docker ps -aq)

**19) To remove all stopped containers**

# docker rm $(docker ps -aq)

**20) To remove all contianers(running and stopped)**

# docker rm -f $(docker ps -aq)

**21) To see the logs generated by a container**

# docker logs container\_name/container\_id

**22) To see the ports used by a container**

# docker port container\_name/container\_id

**23) To get detailed info about a container**

# docker inspect container\_name/container\_id

**24) To go into the shell of a running contianer which is moved into background**

# docker attach container\_name/container id

**25) To execute anycommand in a container**

# docker exec -it container\_name/container\_id command

Eg: To launch the bash shell in a contianer

# docker exec -it container\_name/container\_id bash

**26) To create a container from a docker image ( imp )**

docker run image\_name

**Run command options**

-it for opening an interactive terminal in a container

--name Used for giving a name to a container

-d Used for running the container in detached mode as a background process

-e Used for passing environment varaibles to the container

-p Used for port mapping between port of container with the dockerhost port.

-P Used for automatic port mapping ie, it will map the internal port of the container

with some port on host machine.

This host port will be some number greater than 30000

-v Used for attaching a volume to the container

--volume-from Used for sharing volume between containers

--network Used to run the contianer on a specific network

--link Used for linking the container for creating a multi container architecture

--memory Used to specify the maximum amount of ram that the container can use .

EXAMPLE :

sudo su -

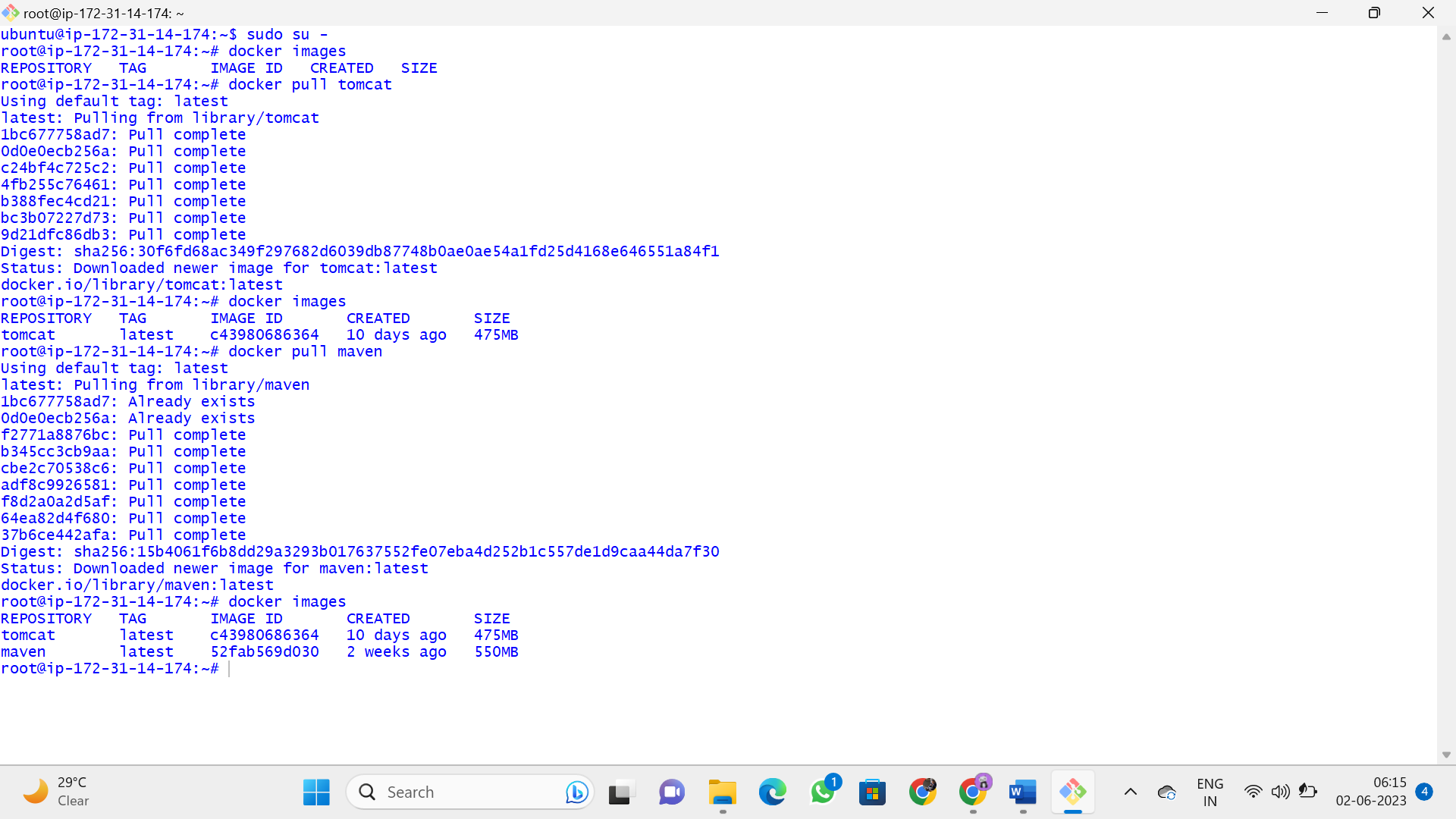
To download tomcat image

# docker pull tomcat

# docker images

# docker pull maven

If you do not specify the version, by default, we get latest version



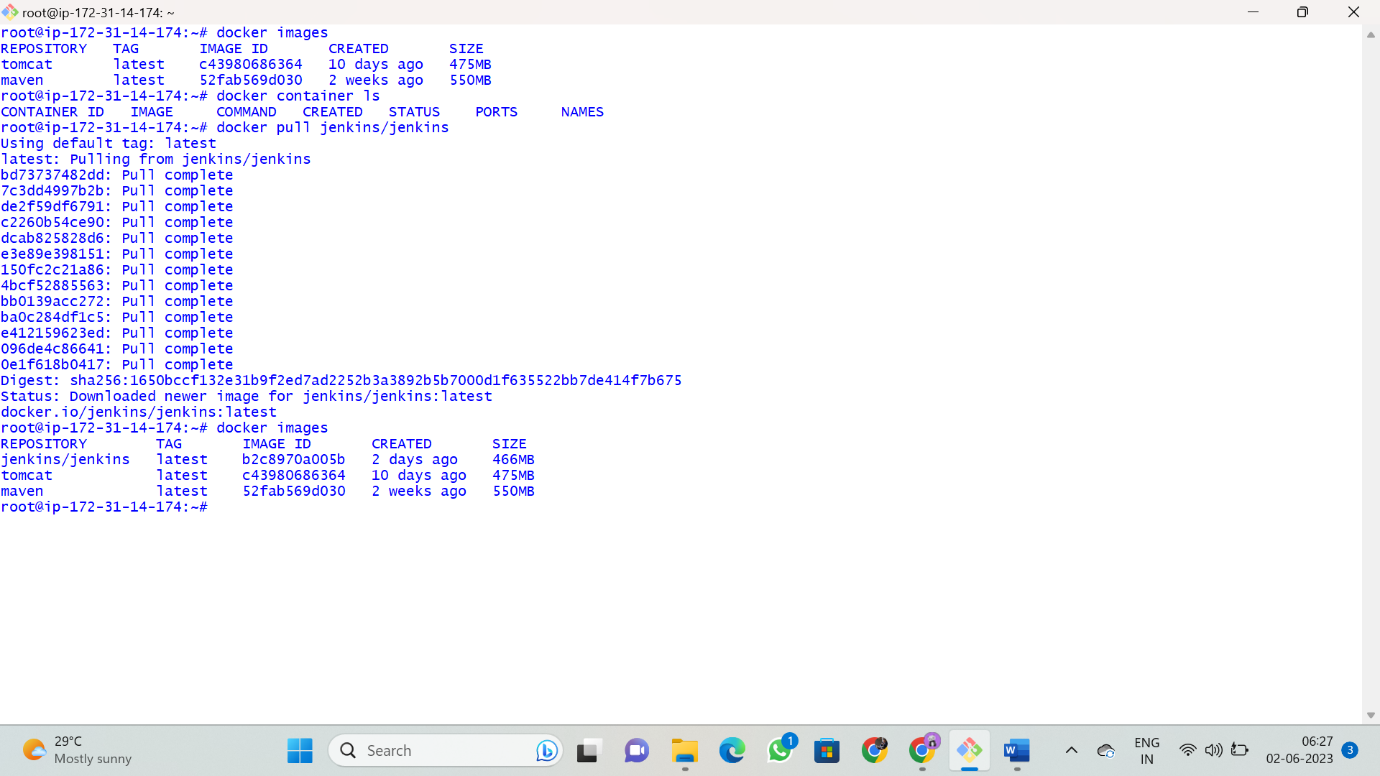
TO create a container from an image

# docker pull jenkins/Jenkins

# docker images

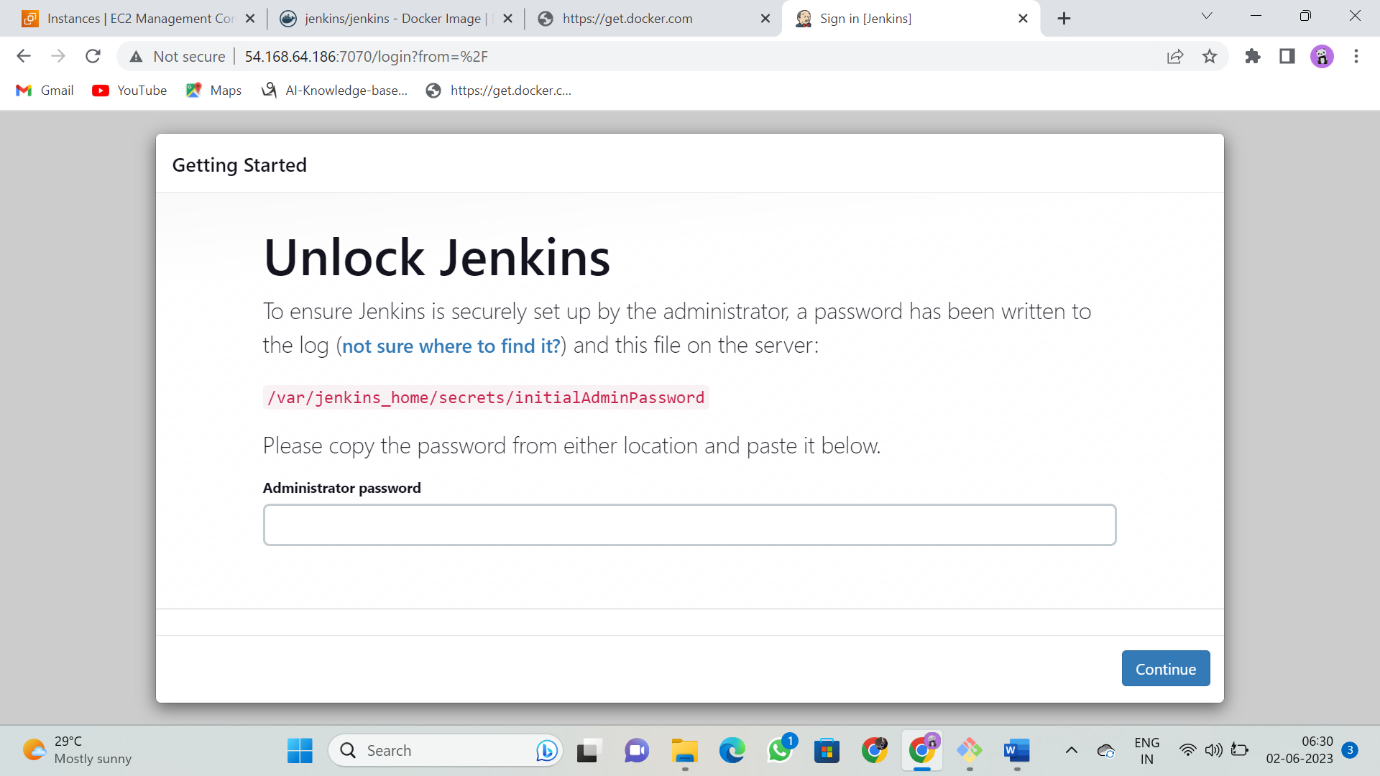
# docker container ls

# docker run --name myjenkins -p 7070:8080 -d jenkins/Jenkins



To check for jenkins ( Open browser )

http:// 54.168.64.186:7070



EXAMPLE :

**To start mysql as container, open interactive terminal in it, create a sample table.**

# docker run --name mydb -d -e MYSQL\_ROOT\_PASSWORD=ajay007 mysql:5

# docker container ls

**I want to open bash terminal of mysql**

# docker exec -it mydb bash

**To connect to mysql database**

# mysql -u root -p

**enter the password, we get mysql prompt**

**TO see list of databases**

> show databases;

**TO switch to a databse**

> use db\_name

> use mysql

**TO create emp tables and dept tables**

https://justinsomnia.org/2009/04/the-emp-and-dept-tables-for-mysql/

**EXPERIMENT 7:**

**Develop a simple containerized application using Docker**

Multi container architecture using docker

------------------------------------------

This can be done in 2 ways

1) --link

2) docker-compose

**1)—link**

Create LAMP Architecture using docker

L -- linux

A -- apache tomcat

M -- mysql

P -- php

( Linux os we already have )

Lets remove all the docker containers

# docker rm -f $(docker ps -aq)

# docker container ls ( we have no containers now )

1) TO start mysql as container

# docker run --name mydb -d -e MYSQL\_ROOT\_PASSWORD=0070 mysql:5

2) TO start tomcat as container

# docker run --name apache -d -p 6060:8080 --link mydb:mysql tomcat

TO see the list of containers

# docker container ls

To check if tomcat is linked with mysql

# docker inspect apache ( apache is the name of the container )

3) TO start php as container

# docker run --name php -d --link apache:tomcat --link mydb:mysql php

++++++++++++++++++++

**2) docker-compose**

Docker compose

* This is a feature of docker using which we can create multicontainer architecture using yaml files.
* This yaml file contains information about the containers that we want to launch and how they have to be linked with each other.
* Yaml is a file format.
* It is not a scripting language.
* Yaml will store the data in key value pairs
* Lefthand side – Key
* Righthand side – Value
* Yaml file is space indented.

**Installing Docker compose**

-----------------------

* + 1. Open <https://docs.docker.com/compose/install/>
    2. Go to linux section

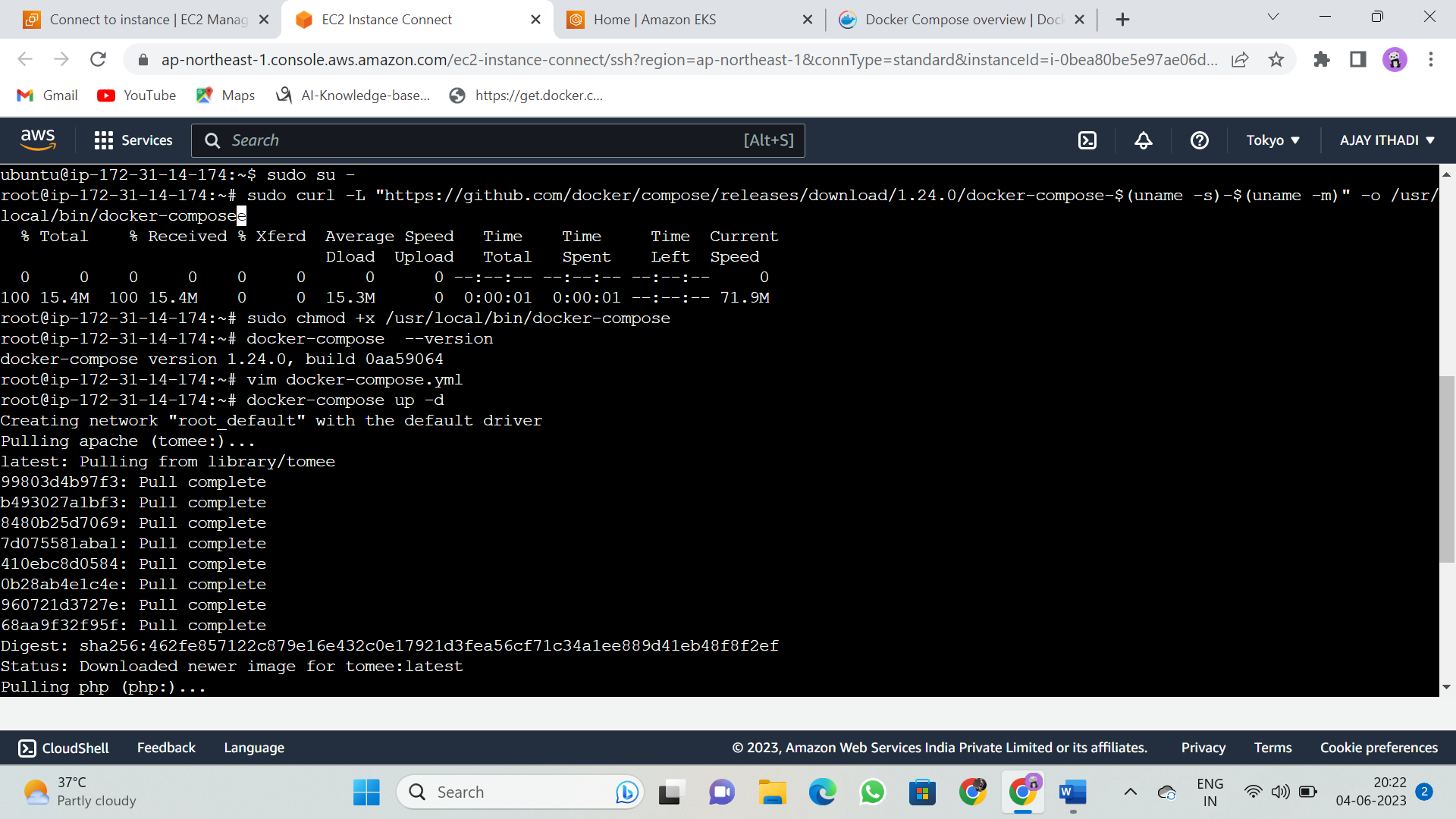
Copy and paste the below two commands

# sudo curl -L "https://github.com/docker/compose/releases/download/1.24.0/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

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

How to check docker compose is installed or not?

# docker-compose --version



# docker pull httpd

**Create a docker compose file for setting up LAMP architecture**

# vim docker-compose.yml

---

version: '3'

services:

mydb:

image: mysql:5

environment:

MYSQL\_ROOT\_PASSWORD: ajay007

apache:

image: httpd

ports:

- 6060:8080

links:

- mydb:mysql

php:

image: php

links:

- mydb:mysql

- apache:tomcat

...

# docker-compose up -d

To see the list of the containers

# docker container ls

( Observation - we are unable to see the php container)

# docker ps -a

