**DOCKER**

DOCKER

* Docker is a platform that uses containerization to package applications with all their dependencies, ensuring they run consistently across different environments.
* Docker is a containerization tool.
* Traditionally for developing a application we are using hardware and then os and supported files in it and last preparing applications.
* In this traditional method for one hardware usage we can develop only one application.
* So to overcome it we came with virtualization.

VIRTUALIZATION

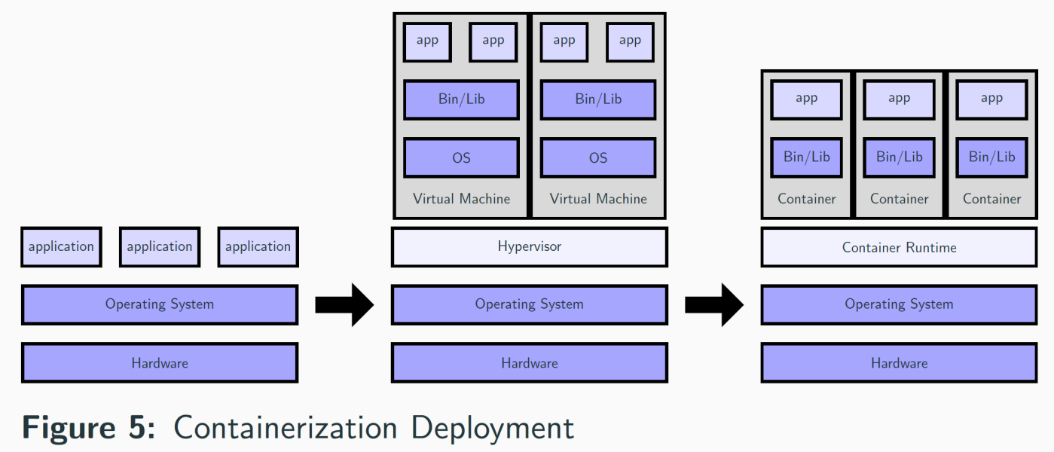
* Virtual machine is an third party we can utilise multiple os like windows, Linux, ubuntu, red hat simultaneously.
* It is virtualizing the end-end hardware in our laptop.
* It means we are using one hardware to develop multiple applications like the process is

1. Usage of one hardware
2. Here we are introducing **hypervisor** which will helps to create many applications.
3. Then os🡪runtime🡪applications.

* Docker will overcome the drawbacks of vmware.
* Here this step can be repeated no of times so that we can develop many applications.
* The disadvantage of this traditional method is **wasting of resource.**
* The advantage in virtualization is reducing the wastage of resource we are using multiple os to develop multiple applications.
* There is also one disadvantage in virtualization like for one application we are using one os so here also there is a resource wasting.
* So, to overcome this we are using docker.
* Docker is a server level virtualization tool.

CONTAINERIZATION

* The process in this docker is we are having one hardware and then we are introducing host os rather than multiple os for multiple applications and then we are using **docker demon** which is like a service and using libraries and creating applications.
* In the virtualization there is a problem facing like the developer will develop the application in one os and it is send to an tester who is using another os here will raise problem like the application is not opening from my side.
* So, here comes the topic of **containerization** that if we are storing our application in a container and sending to the tester it will work.
* Containerization is started because to reduce the resource wastage.
* The another advantage of containerization is auto scaling and auto healing.
* The advantage refers to when any container is crashed or down the containerization will help to create a new container in 1 minute because it is having os with 40mb 60mb 80mb.
* But in the virtualization it will take some time due to its os 8-12 GB, 4mb.



What dockers will do?

* Docker is a server level virtualization tool.
* Docker is created in cloud.
* In dockers the developers will develop the code and shipped to the container after building and application will be available to the end user once we deploy this containers onto web server.
* The containers consists of 1 server, 1os, 1process.
* We can create n no of the containers on hardware of the system.
* For one image we are having one container.
* For one project we are having any images and containers.
* Docker is a client server architecture.
* Developers and operation teams can also write the docker file.

Creating the docker image.

* There are 2 types in creating the docker image.

1. From docker file.
2. Using docker commands.

* Docker images will be stored in containers.
* Along with the containers images will be stored in docker hub.
* Docker image will be pulled into multiple environments.

Docker hub:

* Used to take the backup of both images and containers.
* From here we will start all the environments.

Installation of docker

* Open AWS account and login.
* Create a instance named docker and start the instance and connect it with the git bash.
* Now install the docker by using the command **yum install docker -y** or

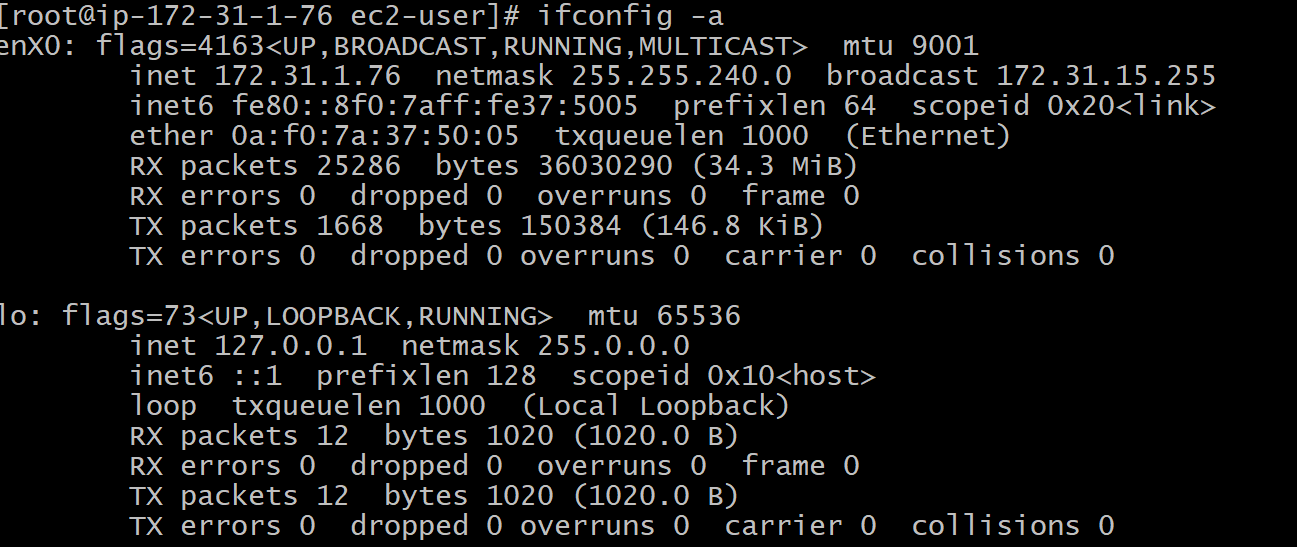
**Yum install -y docker.**

* By the above command installation will be successful and complete.
* Now create a docker hub account
* Create a two repositories in docker hub like dev and QA and also prod.

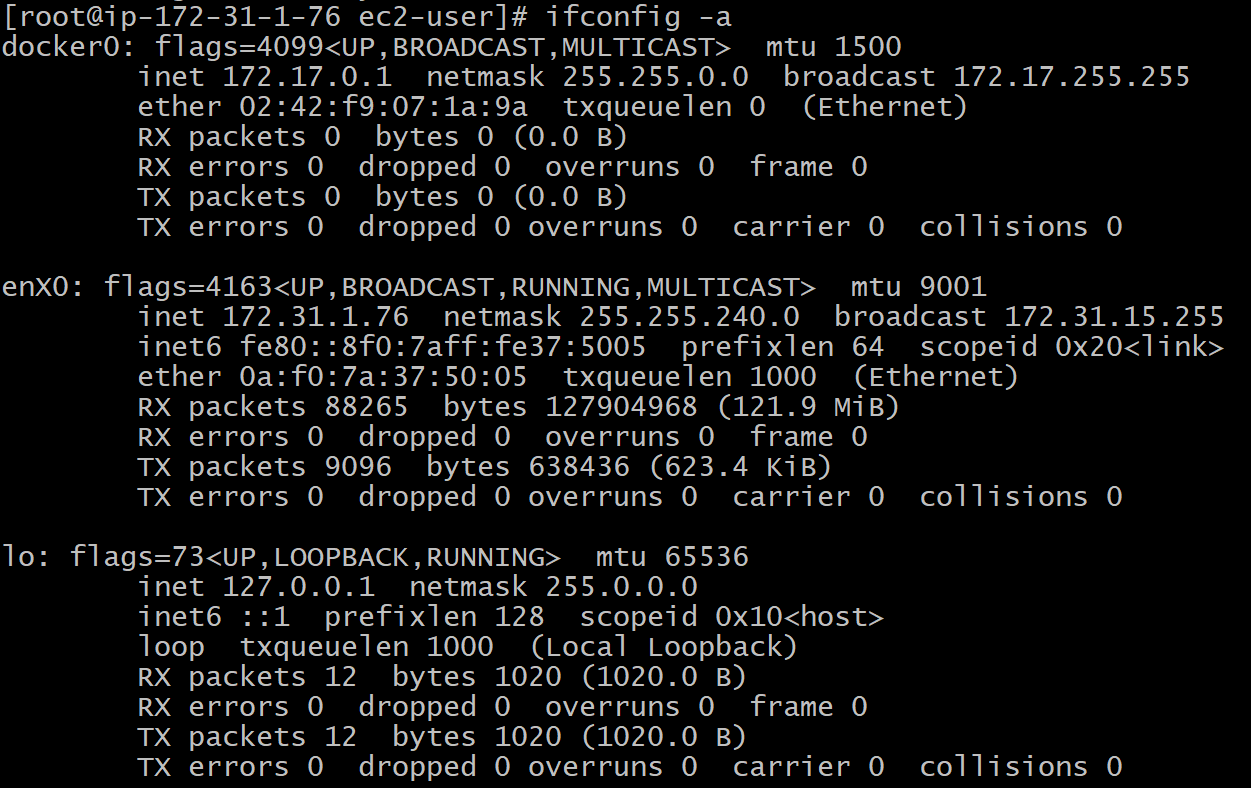
**Start using the commands**.

* Open git bash and after connecting with the Linux type Sudo su .
* Now use the command **ifconfig -a**
* By using the above command we can see the before the installation of docker if we give that command it will only 2 files present in it like enxo and lo.
* But after installation of docker it will display the docker file along with those 2 files.

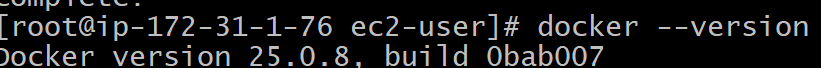
**Before docker installation**

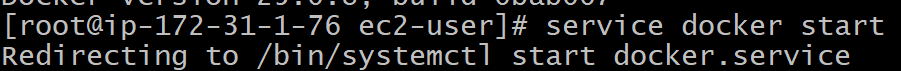


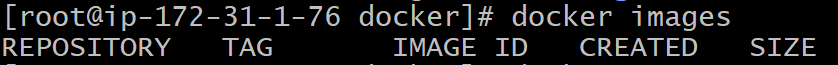
After docker installation



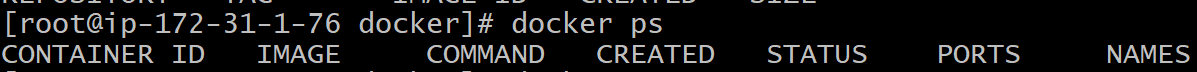
* For checking the version of docker we use **docker –version**



* The main important thing to be noticed is the docker root directory is **/var/lib/docker.**
* By the way first we should start our docker to run all the commands by using **service docker start.**
* To check is there any images present in the docker we use command like **docker images** and **docker images ls.**
* In this we can see the parameters involved in docker image.



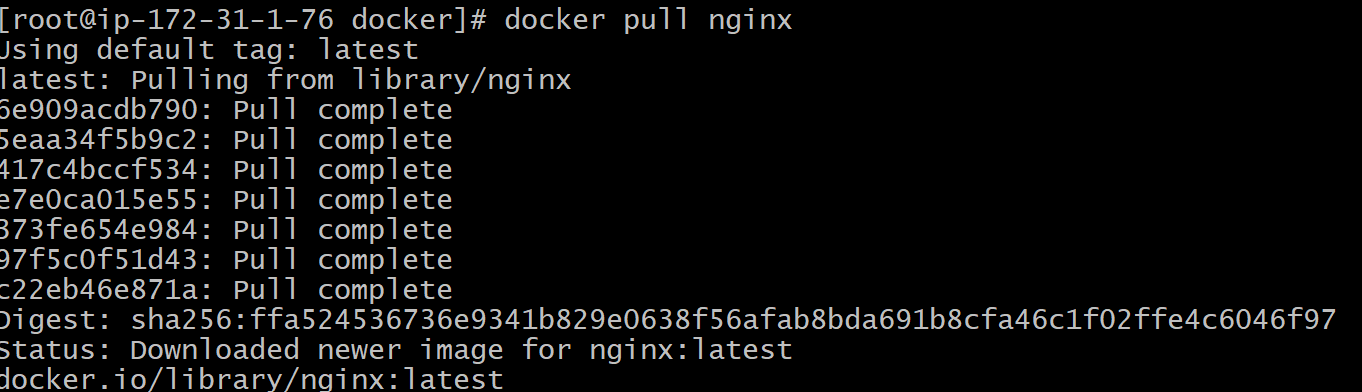
* **Docker ps** is used to know about the containers which are in running stage present.

****

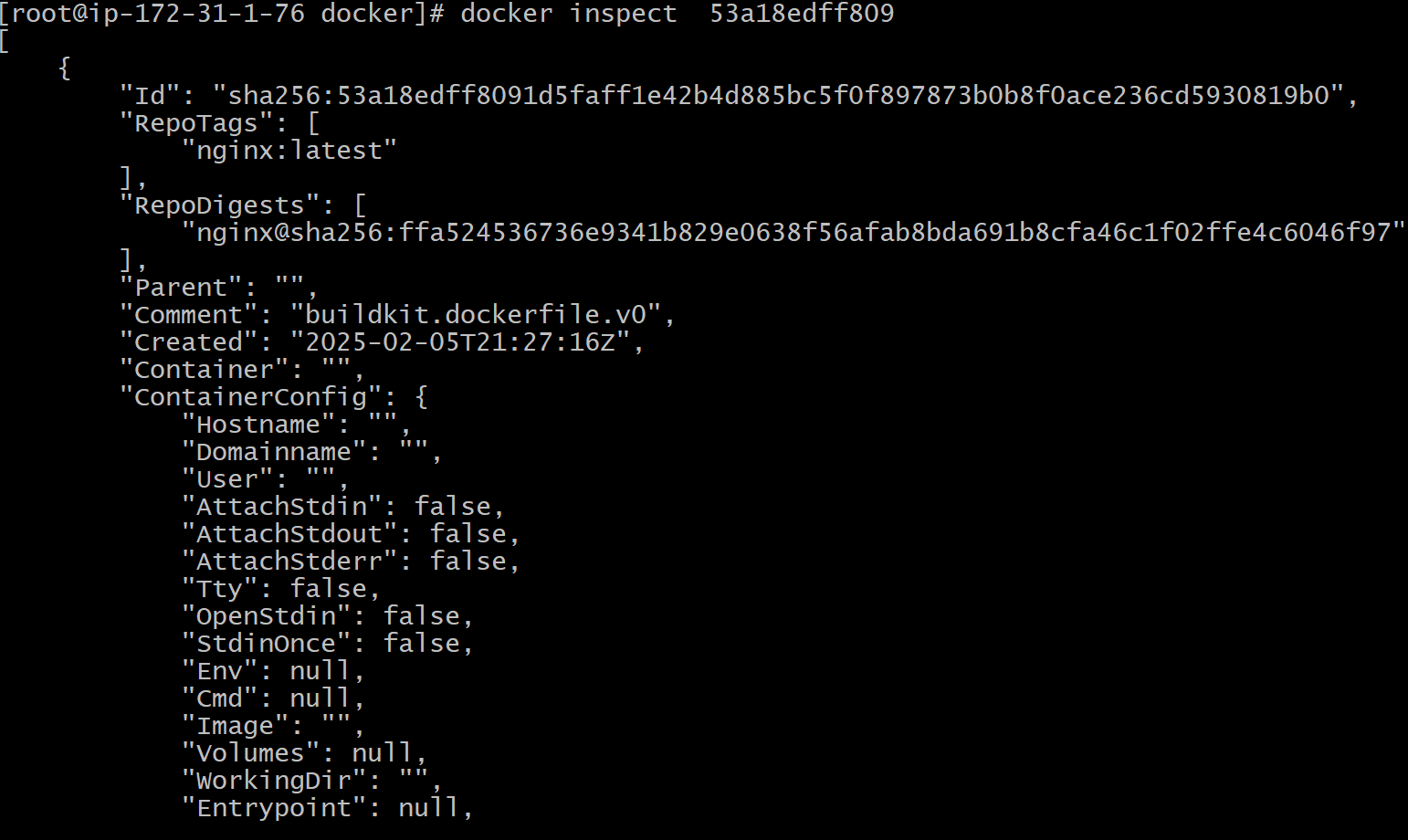
* **Docker ps -a** is used to know about the current running containers and all the containers present.
* To pull the image there are having 2 types

1. Pulling docker image from online ( official website docker hub using commands ).
2. By writing the docker file.

* Now pulling an image from the online.
* To pull an image called nginx which is a webserver.
* **1.Docker pull nginx** is used to pull the image.

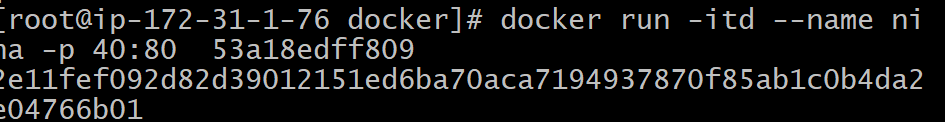
****

* The important thing is in the image id it will consider the first 12 numbers or characters in it.
* To know the particular total information about the image or container we use the command like **docker inspect image id.**

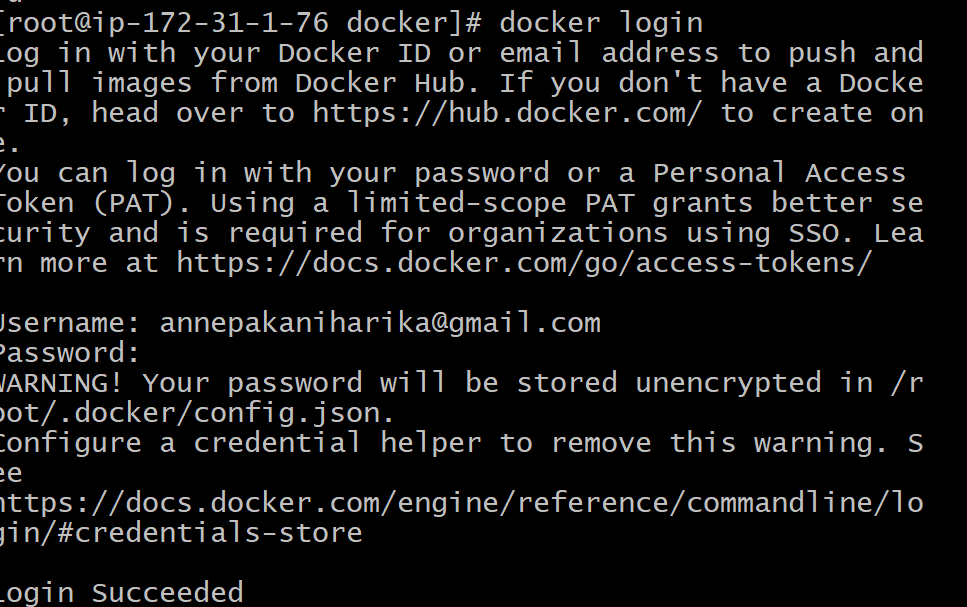
****

**To create a container**

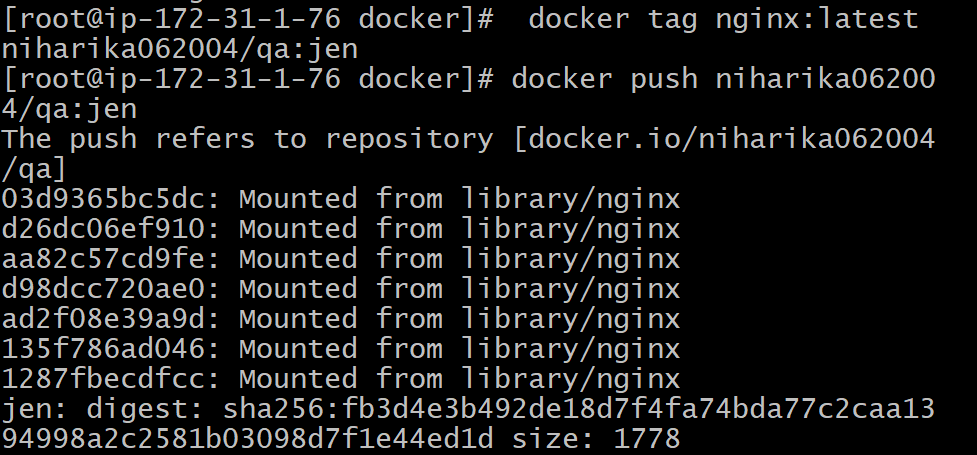
* The syntax is docker run – itd – name niha -p <cid:apn> image id
* Itd: interactive detached mode
* Apn: port number



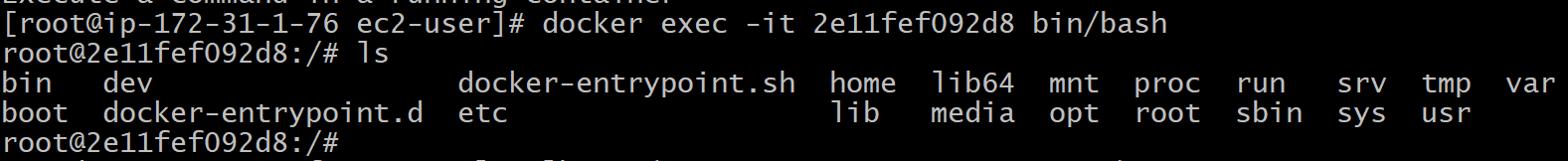
* To login into the docker we use **docker login**



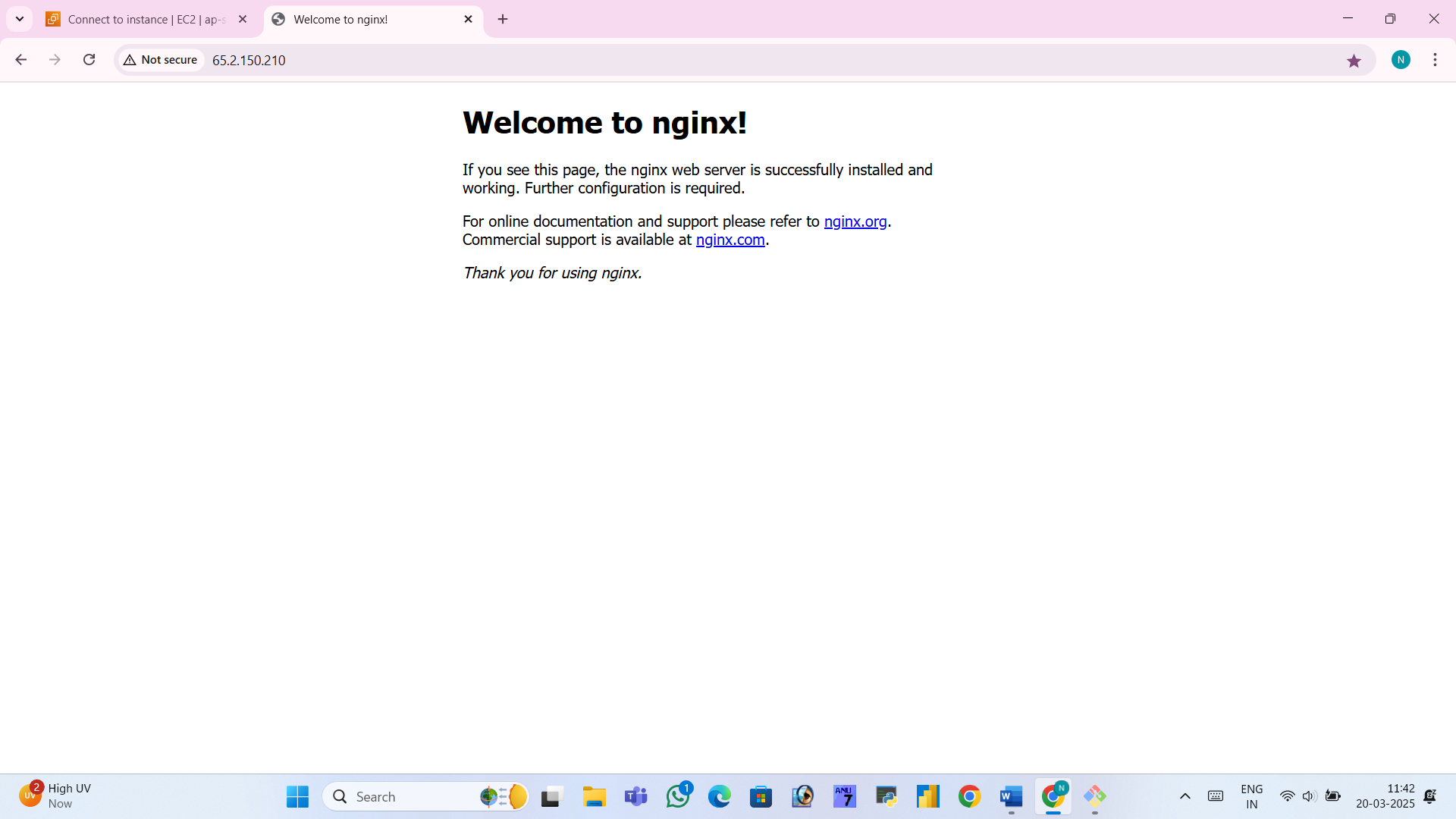
* Next we need to tag the image that to enter into a repository we use docker tag nginx:latest niharika062004/qa:jen
* For pushing we use docker push username/qa:jen



* The command docker exec -it containerid /bin/bash



* After this step let us see copy the ip4 address of that instance and add port number to that ip4 address and paste in the google browser then we can see the page of the nginx.



2**.ubuntu**

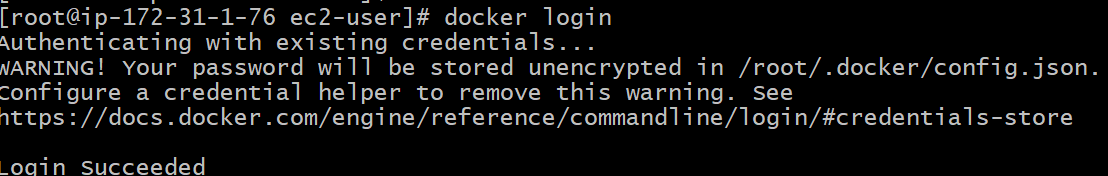
* The first thing start with the instance and connect to the git bash.
* Now switch to the root user by using command **sudo su** .

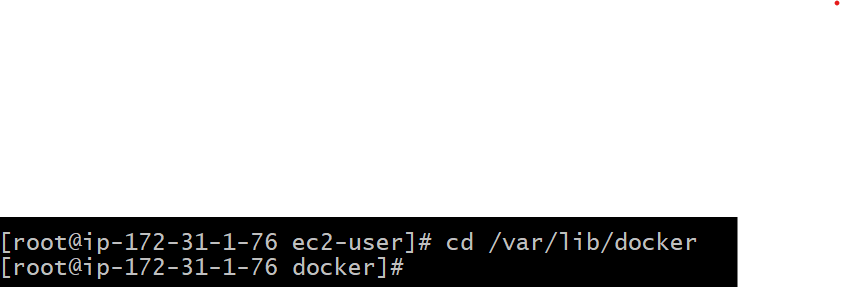


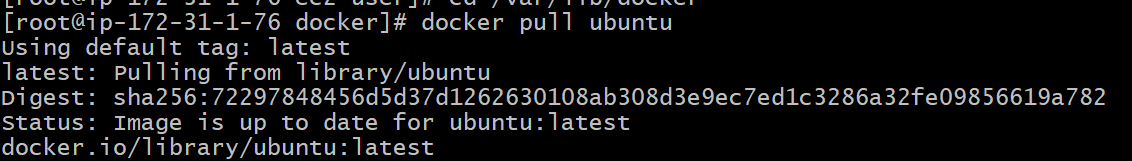
* Now we need to start the docker so that we use the command like **service docker start.**



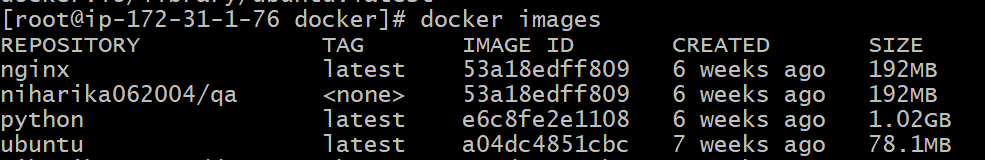
* Now we need to login into the docker by using command **docker login .**



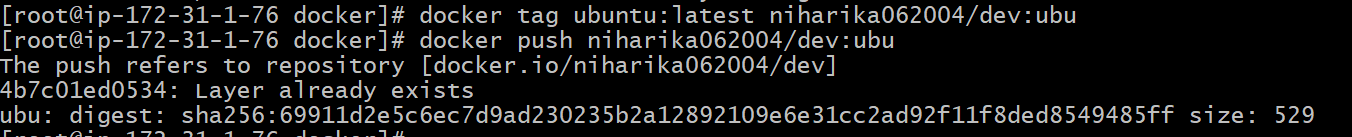
* We need to change our directory to the docker by using command
* **cd /var/lib/docker**
* Then we are pulling the ubuntu using **docker pull ubuntu.**



* Ubuntu got pulled successfully.
* Now let us make the conformation by using the **docker images command** we can see whether ubuntu image is created or not.



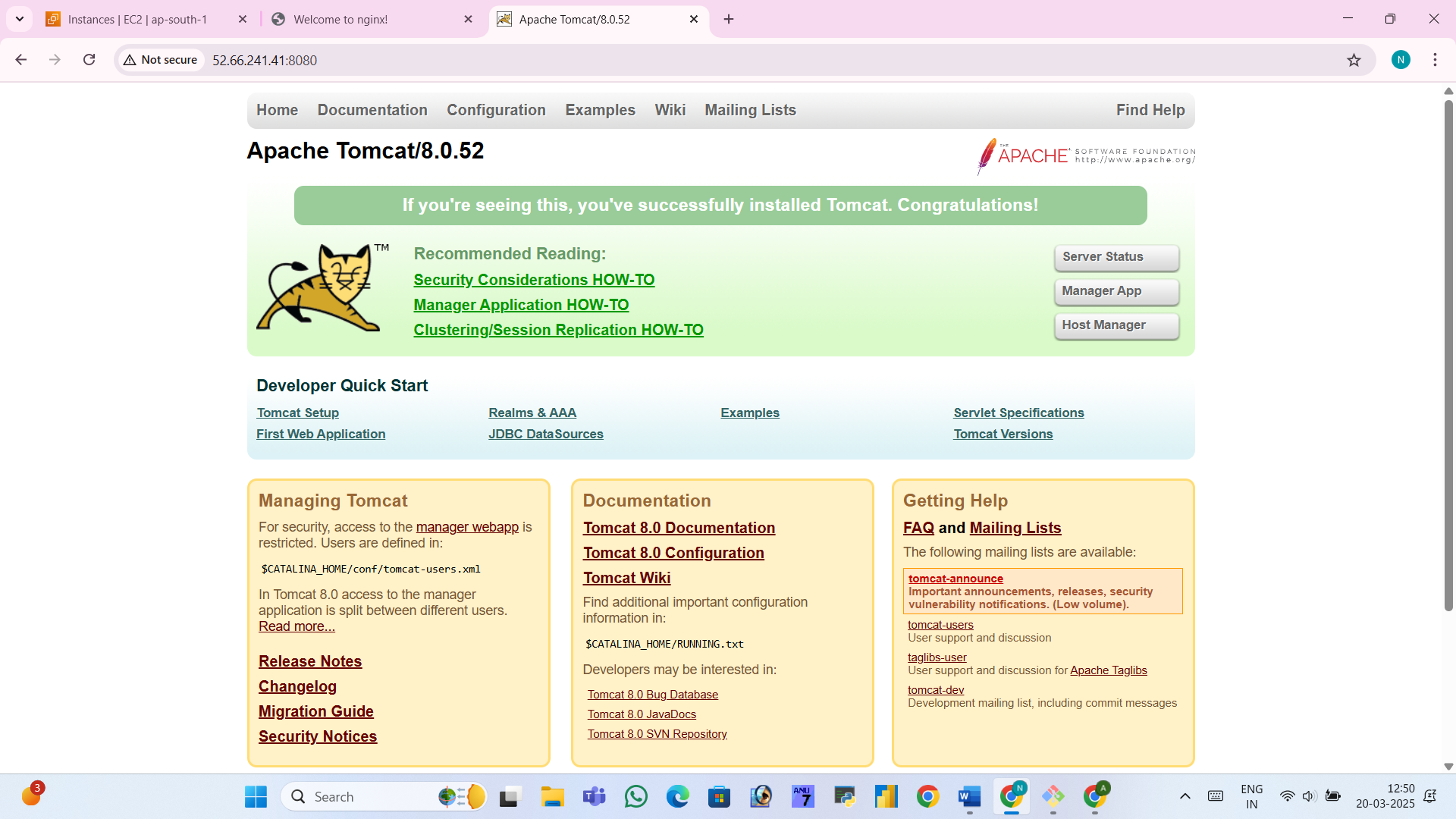
* Here we can see the ubuntu with tag latest and its image id and size.
* Now we are tagging the image using the command and also pushing it.
* Docker tag ubuntu:latest niharika062004/dev:ubu
* Docker push niharika062004/dev:ubu



* Now we have to run it by using
* Docker run -it –name c1 -p 8080:80 ubuntu bash
* Apt update && apt install -y apache2
* Service apache2 start
* For getting out of it press ctrl+p+q
* Docker ps
* The above all commands are used to install the apache2 because ubuntu is an os we cannot see it in the webserver , for that purpose we are install apache2 in the ubuntu and starting that apache2 and checking the container is running or not by using the docker ps.
* Again now we going to the instances and coping the instance ip4 address and as we assigned port number to ubuntu is 8080:80 we should give the ip4 address in the google browser and give port number as 8080 then the page .

3**.Tomcat**

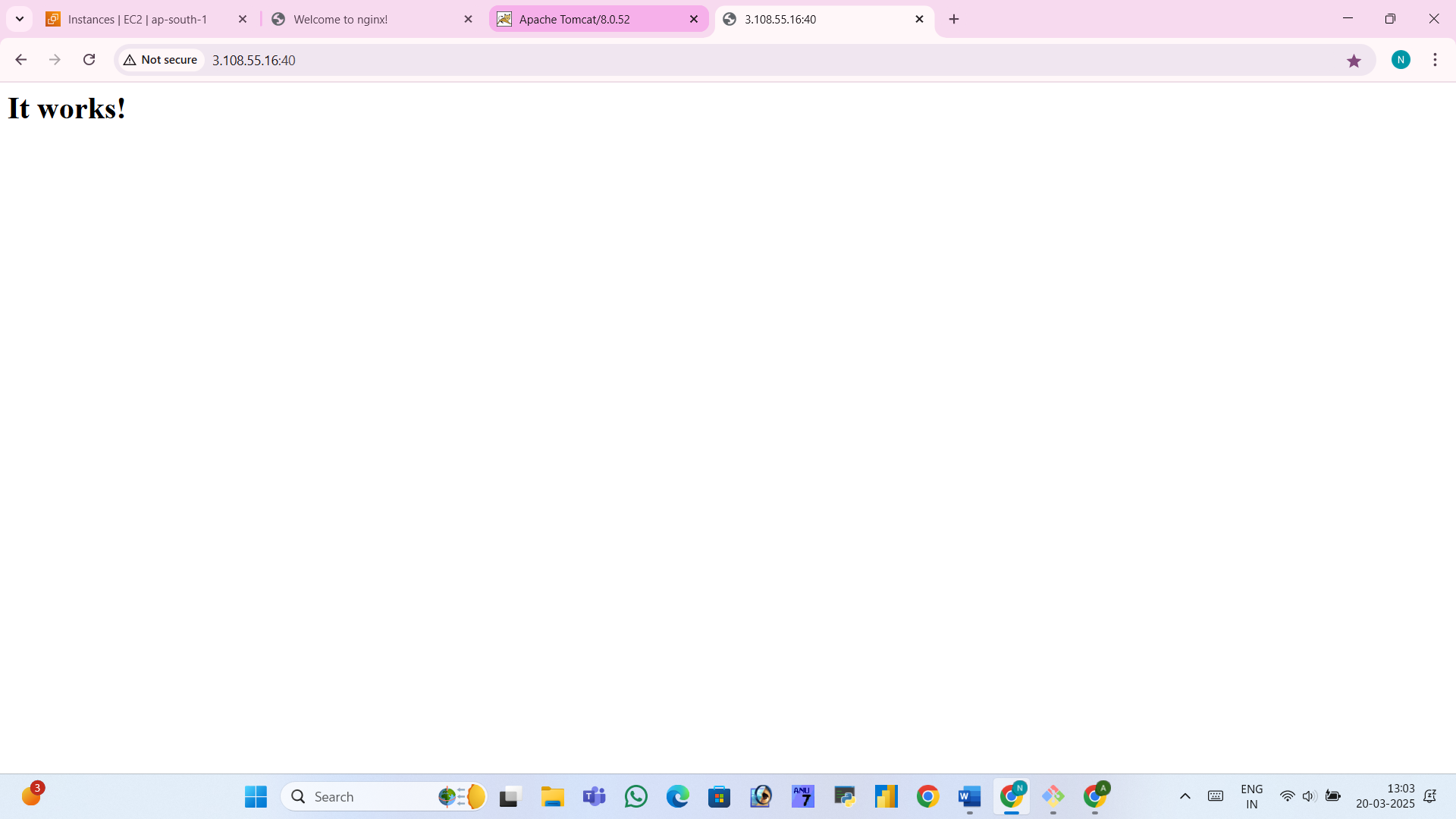
* By using commands above first we need to start the instance and connect to the git hub .
* Now pulling the image of tomcat so that we use command like
* **Docker pull tomcat:8.0.52**
* Checking whether the image is present or not by using the **docker images.**
* By using the **docker ps** command we can know whether the tomcat container is running or not.
* Now we should run by using **docker run -d –name tomcat1 -p 8080:8080 tomcat:8.0.52**



* The above image is an output of tomcat webserver while pasting it on the google browser using the ip4 address and port number like 8080.

4.HTTDP

* Now pull the httdp using the command **docker pull httdp.**
* Check whether the image is present or not by using the **docker images.**
* By using the **docker ps** command we can know whether the httdp container is running or not.
* Now we should run by using docker run -itd --name httpd2 -p 40:80 httpd.
* Again now we going to the instances and coping the instance ip4 address and as we assigned port number to ubuntu is 40:80 we should give the ip4 address in the google browser and give port number as 40 then the page will look like below.



5.**Jenkins**

* We need to pull the Jenkins by using the command **docker pull jenkins/jenkins**
* now we need to run and port the Jenkins
* **docker run -d --name jenkins \**
* **-p 8080:8080 -p 50000:50000 \**
* **-v jenkins\_home:/var/jenkins\_home \**
* **jenkins/jenkins:lts**
* **-**d → Runs the container in detached mode.
* --name jenkins → Names the container jenkins.
* -p 8080:8080 → Maps port 8080 of the host to Jenkins.
* -p 50000:50000 → Maps the port for agent communication.
* -v jenkins\_home:/var/jenkins\_home → Stores Jenkins data persistently.
* To verify whether it is running or not we use **docker ps.**
* Jenkin will ask for the admin password at the page so we can use the password for which we have used docker login
* docker exec jenkins cat /var/jenkins\_home/secrets/initialAdminPassword.
* Again now we going to the instances and coping the instance ip4 address and as we assigned port number to jenkins is 8080:8080 we should give the ip4 address in the google browser and give port number as 8080 then the page .

