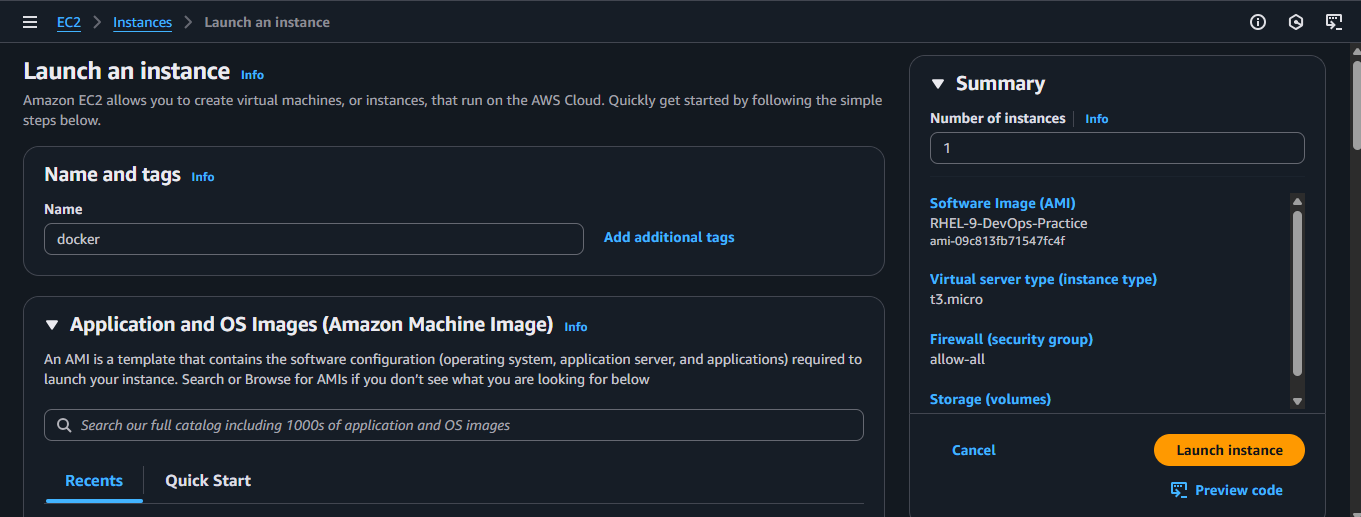
**Session – 51**

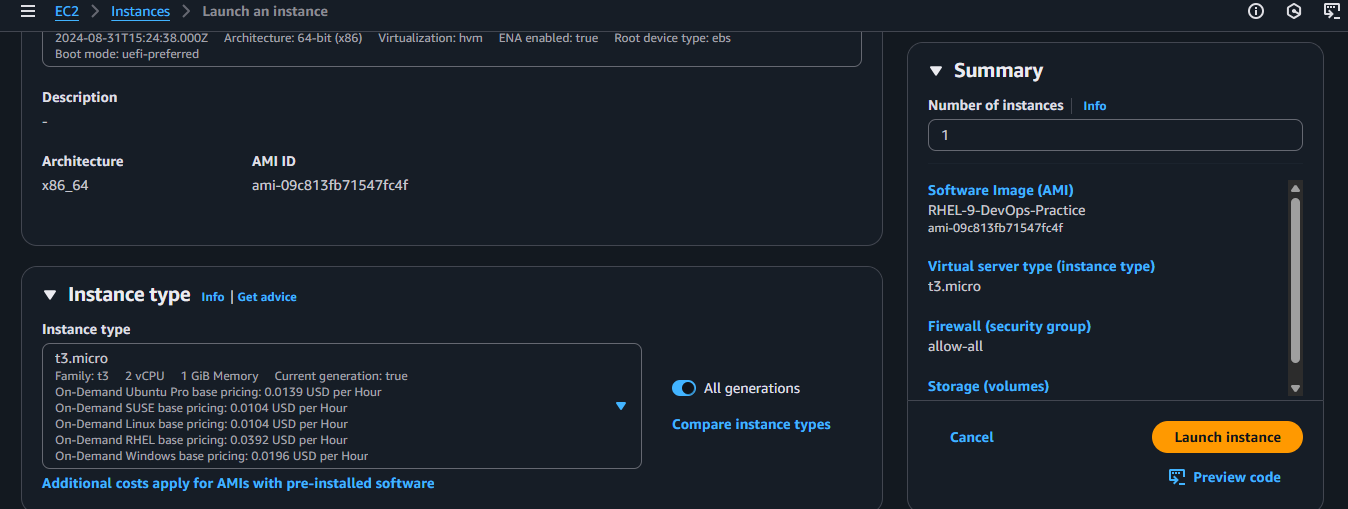
**----------------------**

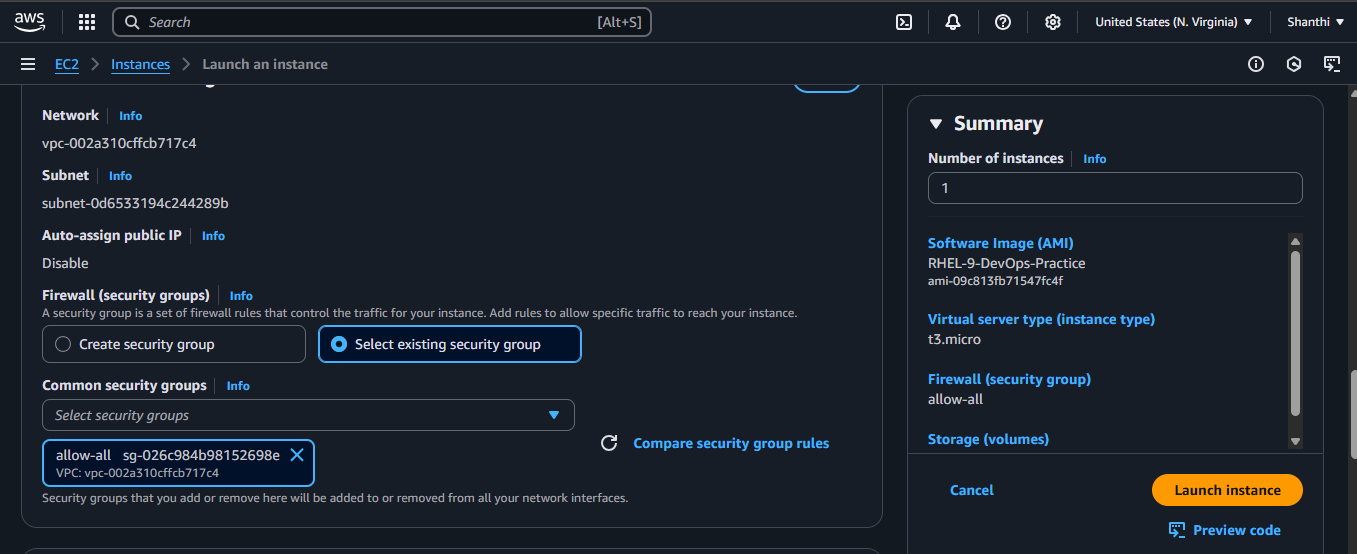
**Docker commands**

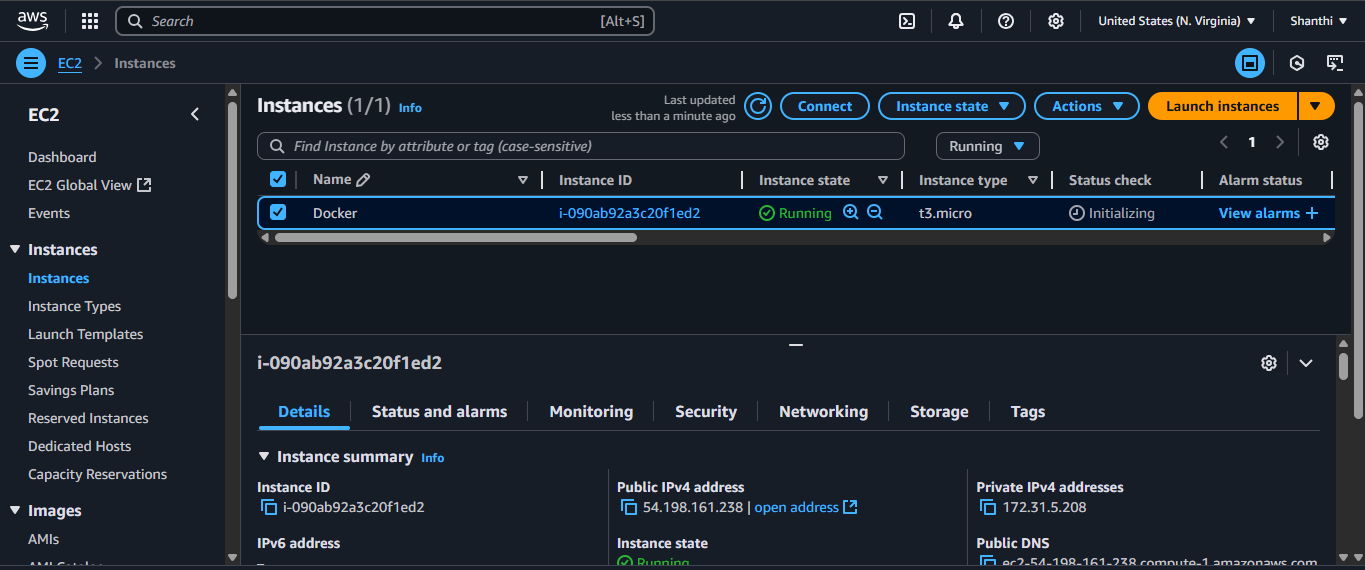
**How to create docker Images.**

Create instance through terraform for speed.









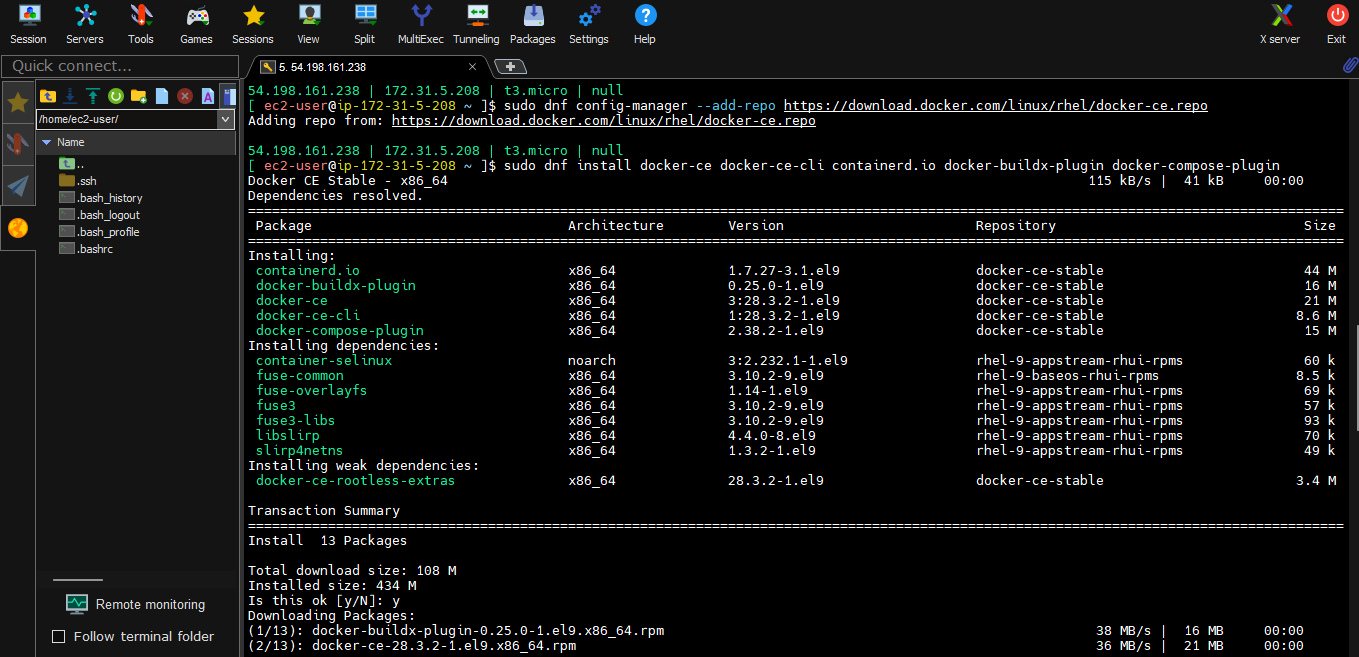
Create docker instance and connect in Mobaxterm.

sudo dnf -y install dnf-plugins-core



sudo dnf config-manager --add-repo <https://download.docker.com/linux/rhel/docker-ce.repo>

sudo dnf install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

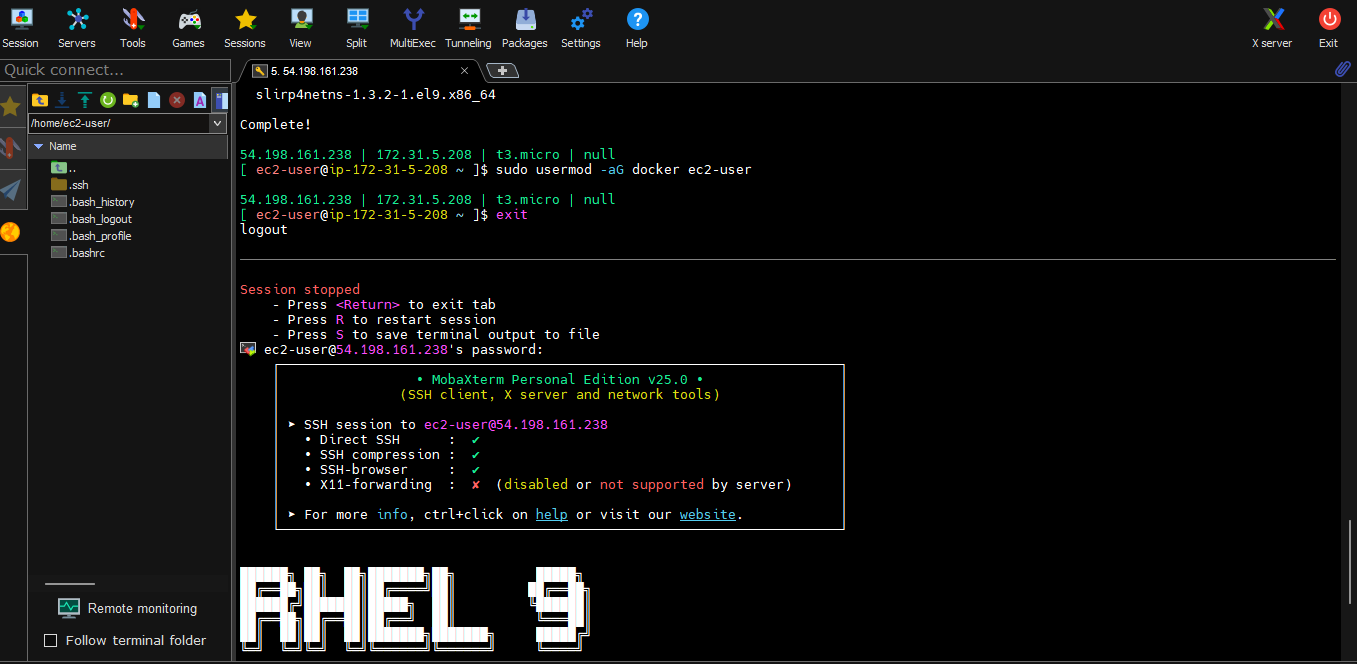


add your normal user to docker group – it will not have access in Normal user that’s why we are changing to group user, Use below commands

We are into docker command that’s why we get access

**sudo usermod -aG docker ec2-user**

Exit and start the server again

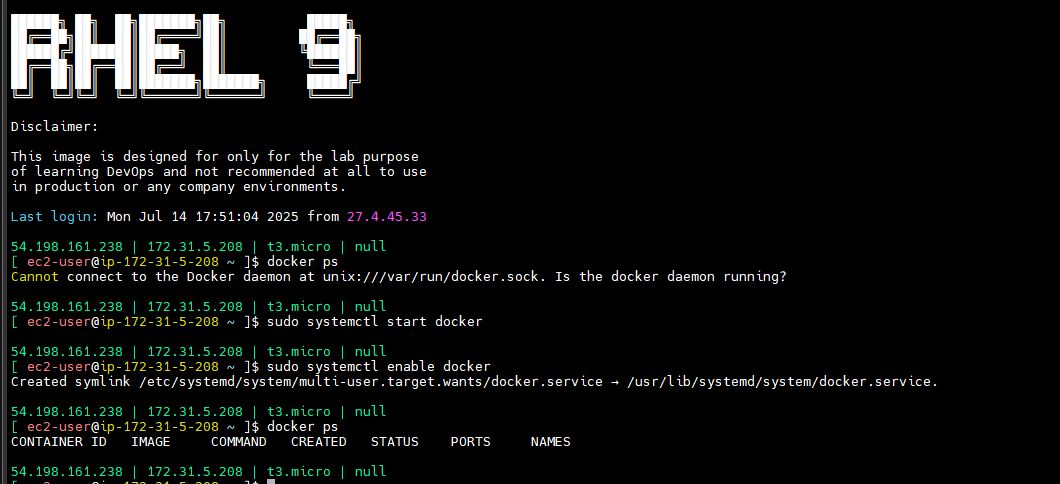


**docker ps**

**sudo systemctl start docker**

**sudo systemctl enable docker**

**docker ps**



**docker ps** - running containers

Image - generally ami (amazon machine image)

In amazon machine image what we are doing

One ami is there = os + configuration (all conf)

AMI = OS + CONFIGRATIONS (system packages + run time + app code + app libraries)

Configurations means what in generally

System Packages - our needed packages will download.

App run time(node js, python,java) whatever we need will download.

Application code and application libraries.

Same exactly here also same. Where it will comes into OS.

Image - Bare min os + remaining all same

Image = Bare min os + system packages = app runtime + app code + app libraries

How much Bare min? Less than on MP3 song.

So that is portable.

Less cost ( already informed in last class gone through that one).

When will run computer?

If you run AMI it is called as server, server is running instance of AMI.

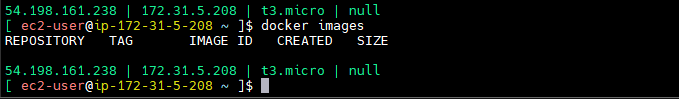
Similarly if ran AMI It will get container.

Container is running instance of image.

If running one image that is container

If run **docker ps** you will get docker containers.

**docker images** -



If need to download any image

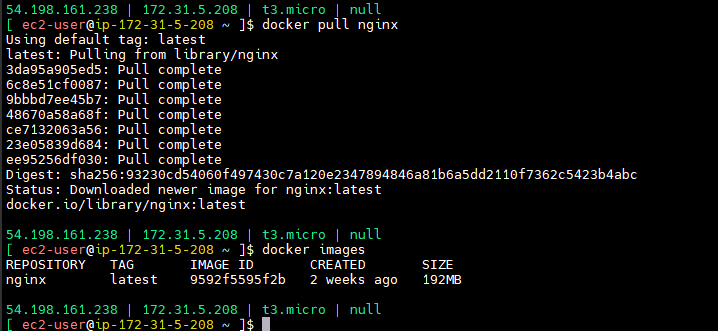
docker pull <image-name>:<tag/version> --> get the image

docker pull nginx - I’m not giving any tag or not giving version. If I given like this means -- pull latest image simple data .

**docker pull nginx**

After ran nginx see images

**docker images**



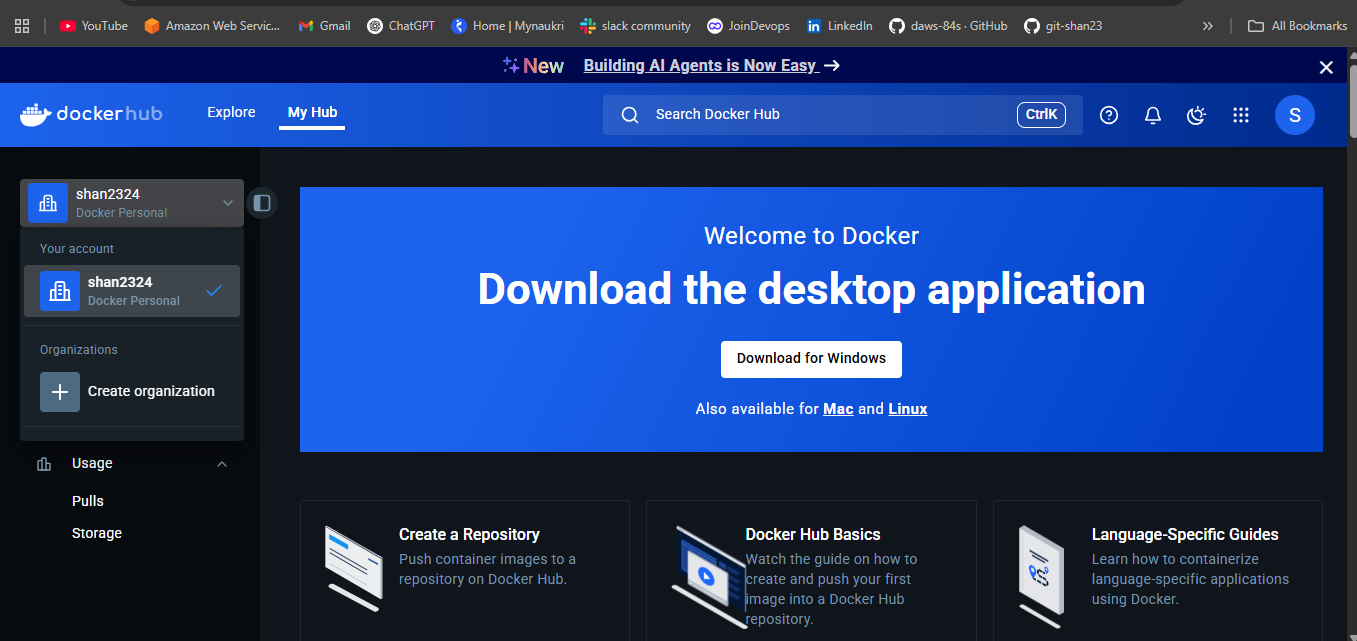
From where it’s pulling this one - there is a central registry have central hub.

**Docker hub** -- search in google

We have m-repos same like that here have a docker registory.

Sign up this one same like github only

Sign up and signin in docker hub.

after login will get this one.

You can search it here images for example nginx -- this is official images - will get all official images

See which version you have.

Total image details also there.(which versions how to use image).

From image I have pull container.

**docker create nginx** - container

**docker ps** - it’s not showing

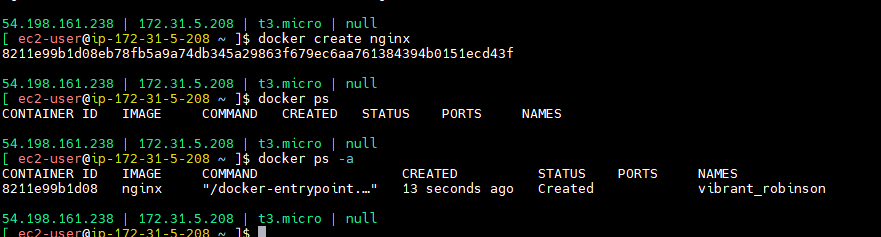
**docker ps -a** - it’s showing

A - all containers ps - running containers only.

docker create nginx

docker ps

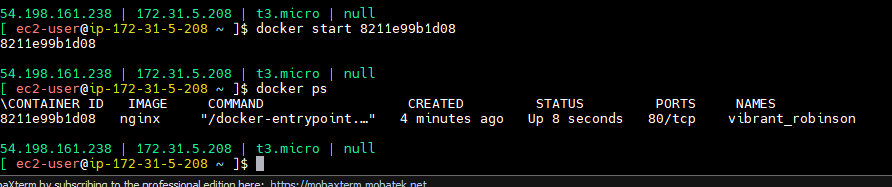
docker ps -a



**docker ps -a** - all containers including all status.

**docker start 8211e99b1d08**

**docker ps**

docker start <container ID>

docker rm <container-ID>

can’t remove running container. So you remove forcely

docker rm -f <container-ID)

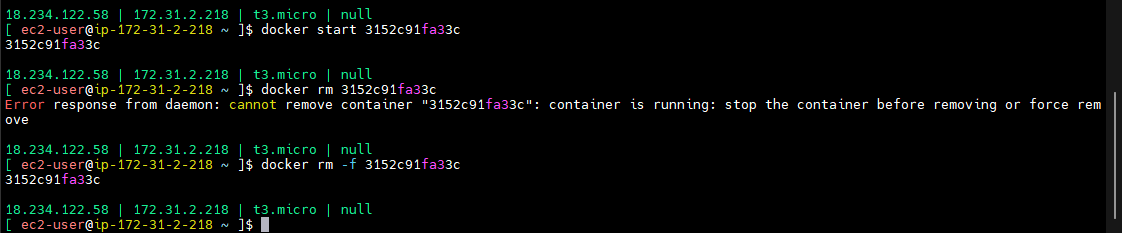
docker ps -a = see here it was removed.if you stop will come here.

**docker start 3152c91fa33c**

**docker rm 3152c91fa33c**

Error response from daemon: cannot remove container "3152c91fa33c": container is running: stop the container before removing or force remove

**docker rm -f 3152c91fa33c**



First we are pulling images from there we created one container then we started. Without these steps.

**docker images**

docker images this name is there if you want to delete that image

**docker rmi nginx** (docker remove image)

**docker images** (docker images also will gone totally)

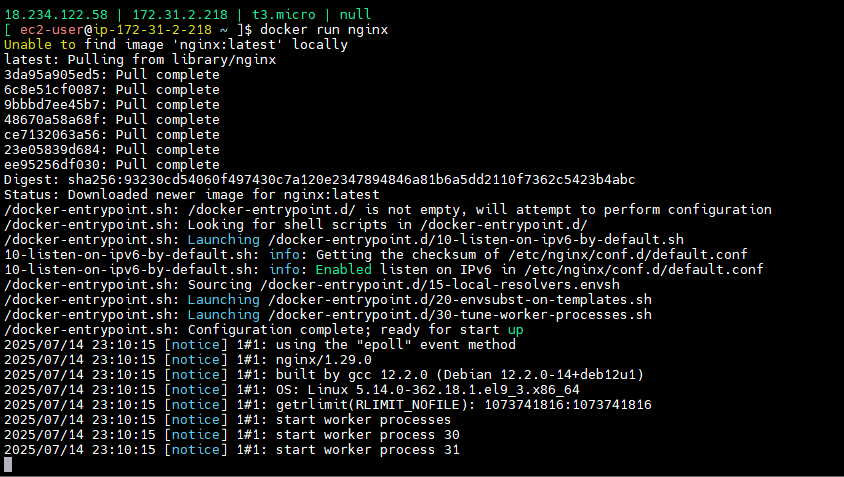


For removeimg comtainer - rm

For removeing image rmi

Now no need that many steps (pull,create,start). you may use

**docker run nginx**

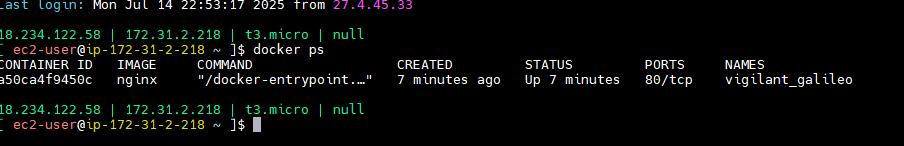


This command will pull the image then create a container and then start the container.

docker run <image>:<tag> -> pull image + create container + start container

See now this server was blocked this one running in forehead. container is running but blocking the terminal. I can’t do anything. I have to open another session for testing or anything.

**docker ps** in seconed session



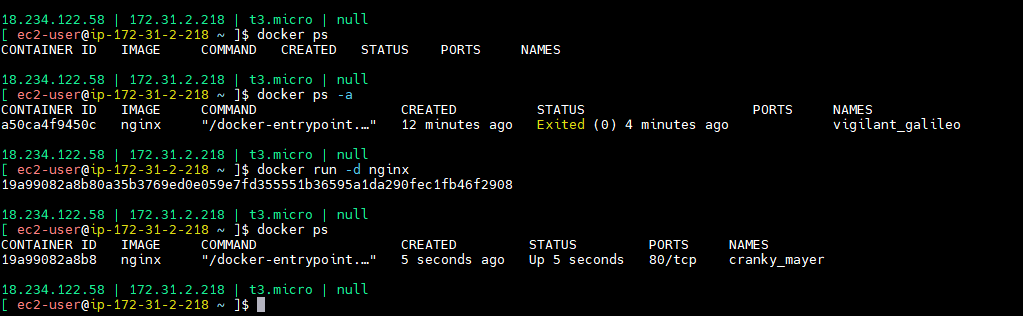
Ctr+ c - exit that underground process.

**docker ps -a** - container was exited. Because I killed the process I given ctrl + c

Now I want to move background

**docker run -d nginx**

**docker ps**



doocker run -d nginx - detach the screen

Here running nginx how can I access this one

**Diagram**

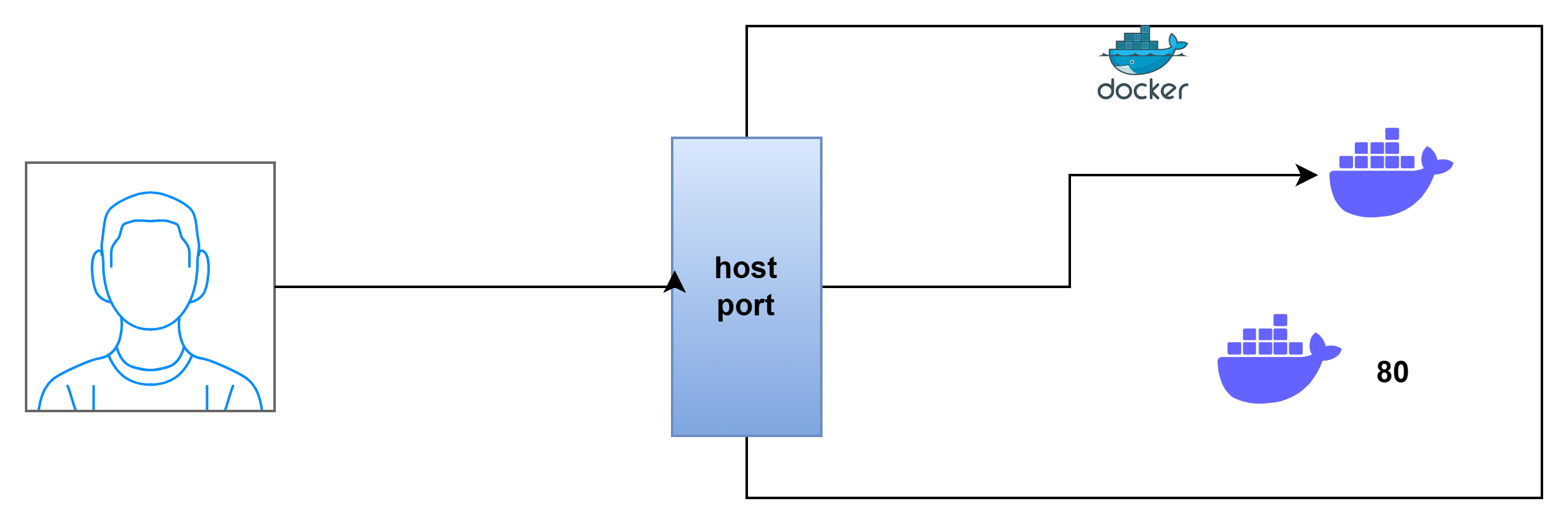
This is docker server ( docker installed server).

This container also like a mini server.

In one server how many ports have 0-65,535. In containerization also same ports we have.

How we will hit our server - through port only

I’m hiting in browser means which port that one - 80



Login through ssh means port no 22

While hit the server should use any port

We have user - user connecting to our server means he is using a port.

First will come to Host that port then it will forward to docker.

Nginx image also here open port 80.

Which port its opening no need to mind it. If we document we come to know.

This port no 80 to listner. 80 is container port. But request comes to host poster.

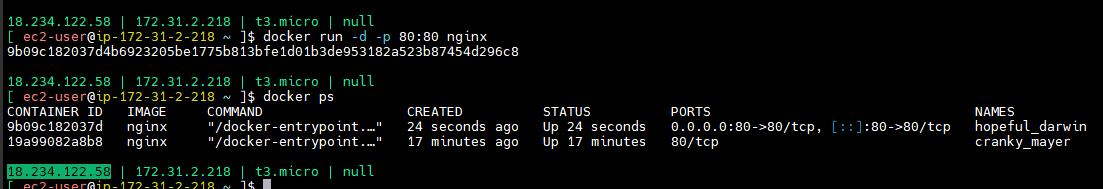
Then

**docker run -d -p 80:80 nginx**

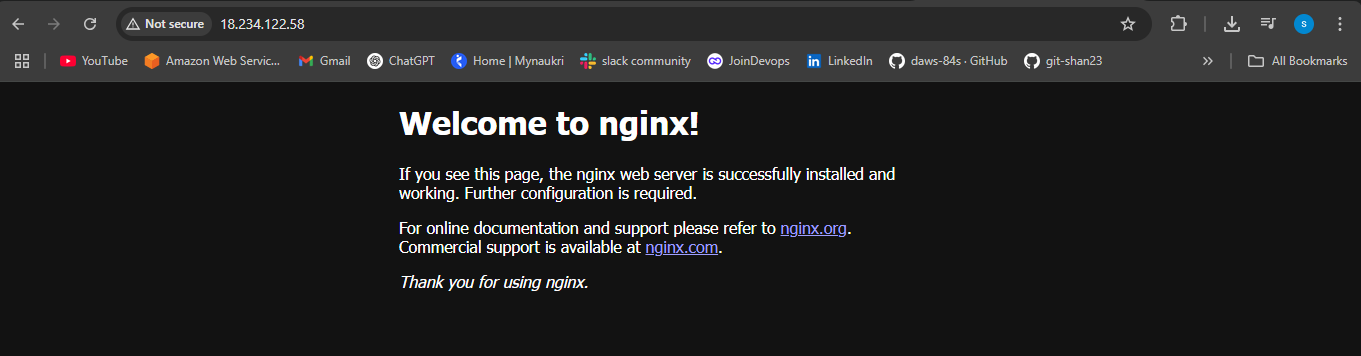
-d - detach port

-p - host port

Host port is Your wish but container port should be 80. because there it’s opening nginx port no 80.



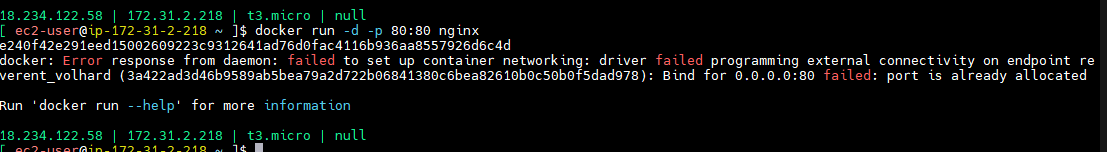
Now if I check this one in browser will get nginx welcome folder



Each time you run a Docker command, it creates a new container.

I can create one more container also now. Sheill I use port no 80 again.

**docker run -d -p 80:80 nginx**

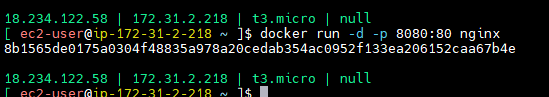


Host port you can use again and again. but port 80 is already in use on the host.

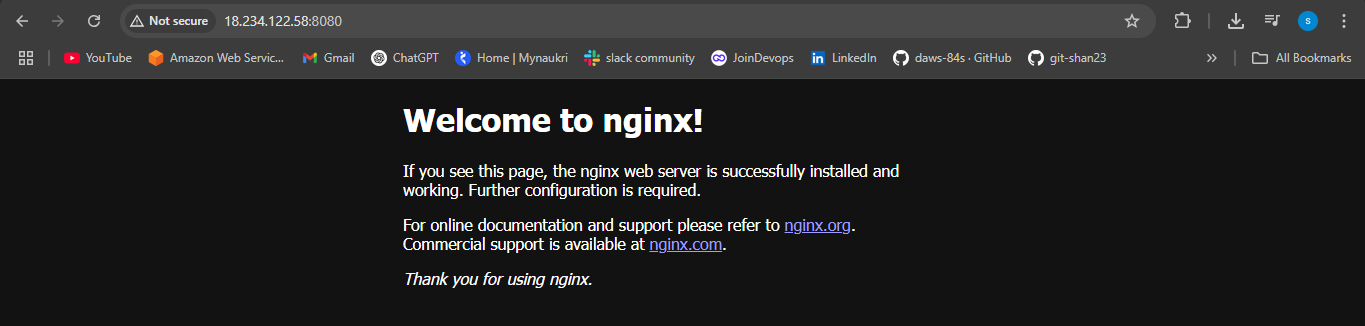
You case use Host port differntly.

Now if I given 8080 then this is a different container.

**docker run -d -p 8080:80 nginx**



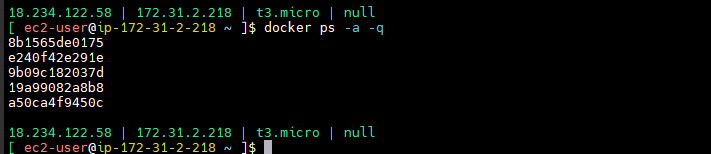
<http://18.234.122.58:8080/> in browser



You may create no.of containers in one server.

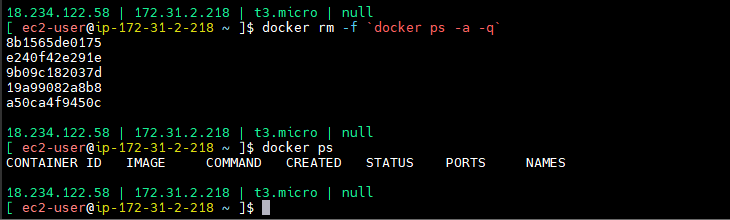
Host port alone we can reuse obviously.

**docker ps -a -q** ( all container id’s will get).



**docker rm -f `docker ps -a -q`** (at a time delete)

docker ps

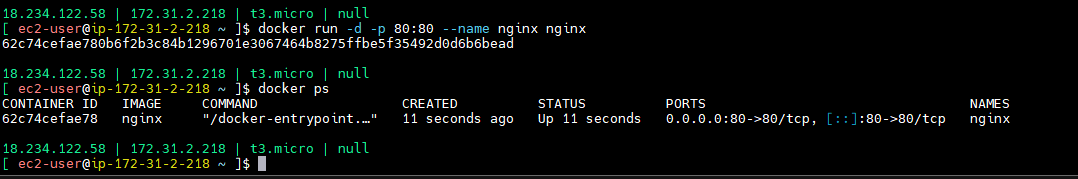


If you are not given name docker will select two dictionary words.

If you want give name

**docker run -d -p 80:80 --name nginx nginx**

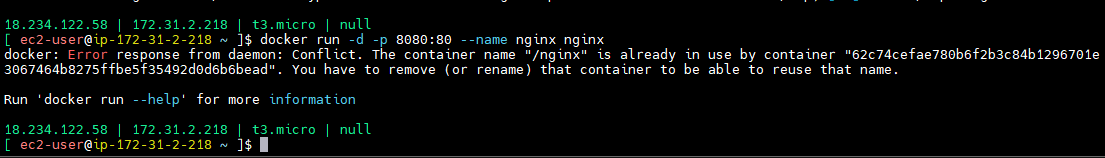
**docker ps**



Our name came nginx.

You can’t create one more if you change port also

**docker run -d -p 8080:80 --name nginx nginx**



Different name and different host name you should select. If you want create similarly.

Now you can use instead if container ID use container name.

Now container is running. I want to login in containers for checking.

Already I logged in server.How I can login in container.

**docker exec -it nginx bash**

It - interactive terminal



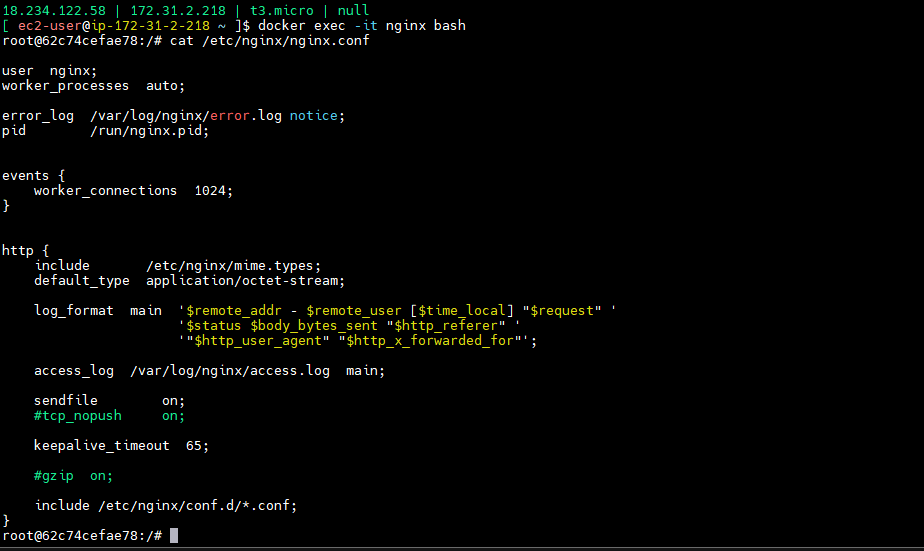
Now I’m inside container.

Now you can check the configurations.

**cat /etc/nginx/nginx.conf**

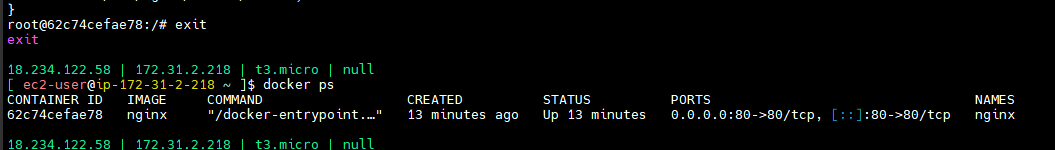
there in very bare min.

Then you can exit.



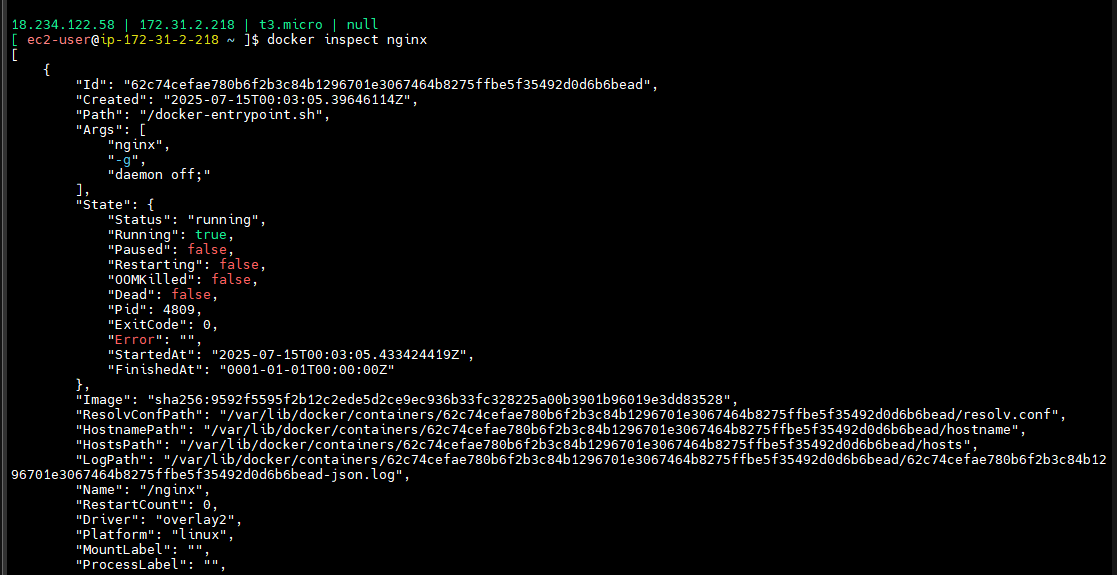
Containers was created.

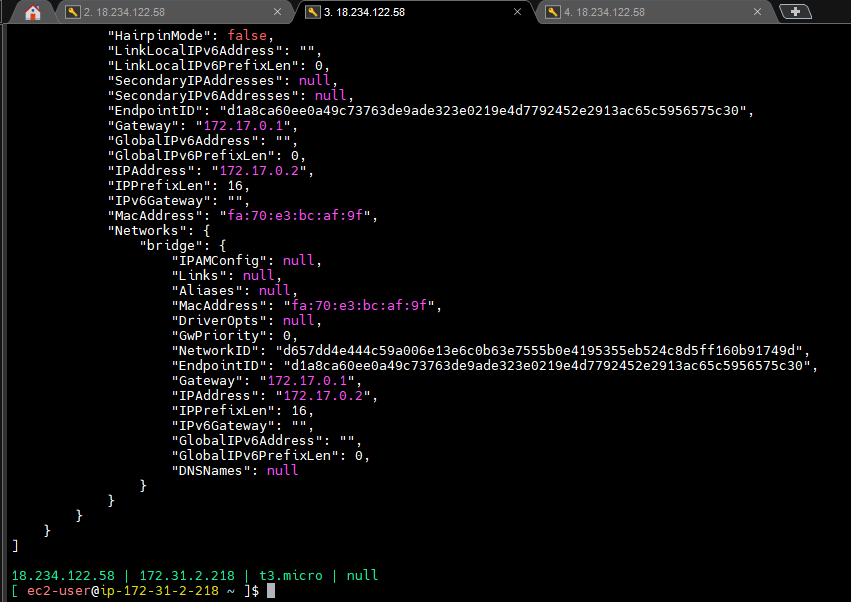
**docker ps**



What is container ip address.

**docker inspect nginx**



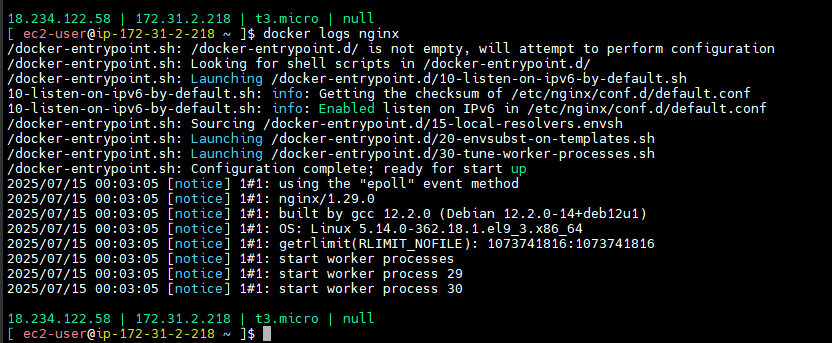


This is total information docker inspect.

docker inspect container-name/container-ID

How to check logs?

**docker logs nginx**



If you want full logs login in containers and see there.

This is all docker commands.

Who will allocate ip adders for containers we will see docker networking.

We pulled existing images. We can’t pill for our roboshop. We have create images.

How can you create custom images to your applications?

There is way called

Dockerfile -- a set of instructions to create customized images.

Now we have to write docker custom images.

Create a Dockerfile repository in Github.

/c/devops/daws-84s/repos/dockerfiles

How to create dockerfiles?

We have a documentation.

Dockerfile reference (search in Google) -- official document this one

This is very simple we have some instructions here that instructions we have to practice.

|  |  |
| --- | --- |
| **Instruction** | **Description** |
| [ADD](https://docs.docker.com/reference/dockerfile/" \l "add) | Add local or remote files and directories. |
| [ARG](https://docs.docker.com/reference/dockerfile/" \l "arg) | Use build-time variables. |
| [CMD](https://docs.docker.com/reference/dockerfile/" \l "cmd) | Specify default commands. |
| [COPY](https://docs.docker.com/reference/dockerfile/" \l "copy) | Copy files and directories. |
| [ENTRYPOINT](https://docs.docker.com/reference/dockerfile/" \l "entrypoint) | Specify default executable. |
| [ENV](https://docs.docker.com/reference/dockerfile/" \l "env) | Set environment variables. |
| [EXPOSE](https://docs.docker.com/reference/dockerfile/" \l "expose) | Describe which ports your application is listening on. |
| [FROM](https://docs.docker.com/reference/dockerfile/" \l "from) | Create a new build stage from a base image. |
| [HEALTHCHECK](https://docs.docker.com/reference/dockerfile/" \l "healthcheck) | Check a container's health on startup. |
| [LABEL](https://docs.docker.com/reference/dockerfile/" \l "label) | Add metadata to an image. |
| [MAINTAINER](https://docs.docker.com/reference/dockerfile/" \l "maintainer-deprecated) | Specify the author of an image. |
| [ONBUILD](https://docs.docker.com/reference/dockerfile/" \l "onbuild) | Specify instructions for when the image is used in a build. |
| [RUN](https://docs.docker.com/reference/dockerfile/" \l "run) | Execute build commands. |
| [SHELL](https://docs.docker.com/reference/dockerfile/" \l "shell) | Set the default shell of an image. |
| [STOPSIGNAL](https://docs.docker.com/reference/dockerfile/" \l "stopsignal) | Specify the system call signal for exiting a container. |
| [USER](https://docs.docker.com/reference/dockerfile/" \l "user) | Set user and group ID. |
| [VOLUME](https://docs.docker.com/reference/dockerfile/" \l "volume) | Create volume mounts. |
| [WORKDIR](https://docs.docker.com/reference/dockerfile/" \l "workdir) | Change working directory. |

Fiest we need OS.

Base image is important that is sever or container.

We can’t develop images. We have to existing images.

First Instruction **FROM base image**.

**FORM**

**========**

Instruction syntax is simple

Instruction name and then options

You can use ant image.

We was used redhat enterprise Linux in servers. But in images that is not free.

We can use **FORM almalinux :9** (this is also 100% equivalent redhot)

For my base image set into almalinux.

If you are not given version that will ktake new version.

**FROM/Dockerfile**

FROM almalinux:9

How to build the image

docker build -t from:v1 . --> what is dot represent here current directory has Dockerfile.

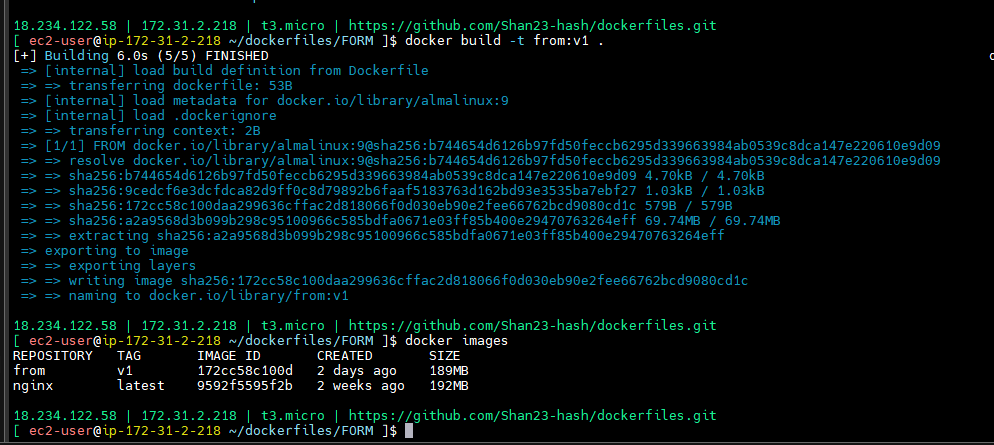
In current direcoty there in dockerfile -- read that one -- do build -- do it tag from:name tag:v1

Version is our wish.

Where tou have docker file there you should run.

**docker build -t from:v1 .**

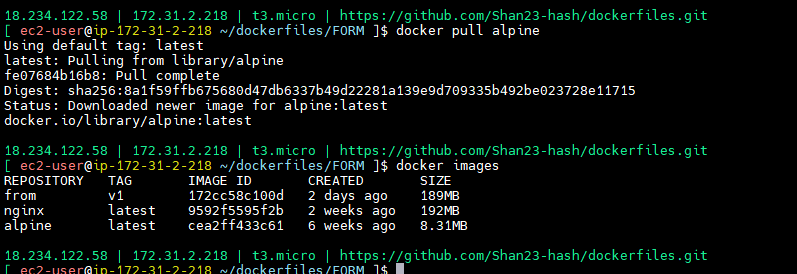
**docker images**



See size

**docker pull alpine**

**docker images**



Used electronic devices So into that OS embedded enterprise with using C language.

So this is the first insttuction

From represents the base OS it should be the first instruction in dockerfie.

**RUN INSTRUCTION**

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

I takes base image it is enough this..

On base images needed things may run,

RUN commands

RUN instructions configure the image like installing packages, doing some configurations, etc..

RUN executes at the time of image buidling

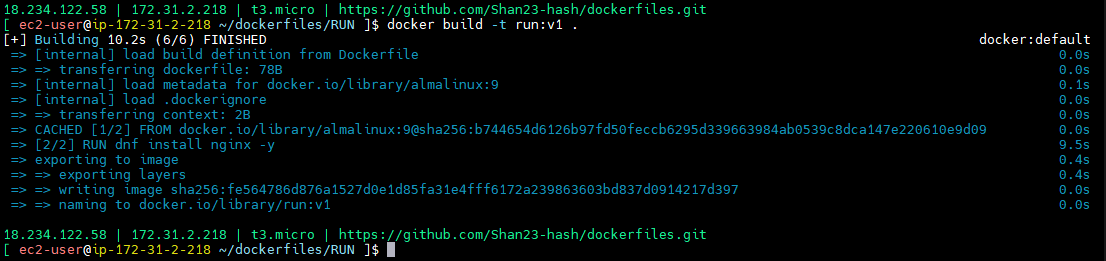
**RUN/Dockerfile**

FROM almalinux:9

RUN dnf install nginx -y

I should use run command

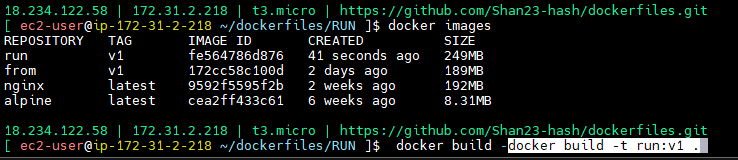
**docker build -t run:v1 .**



Now our Image size was increased

Run size is 249MB

**docker images**



On top almalinux we installed nginx that is creating obliviously memory.

**CMD**

**-------**

Finally I want run nginx

systemctl start nginx --> etc/systemd/system/nginx.service

In this service which conmmand will work.

Systemctl start nginx is there.

I will run that command

Nginx container run command (search in google)

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

**CMD/Dockerfile**

FROM run:v1

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

-g = set configuration directories.

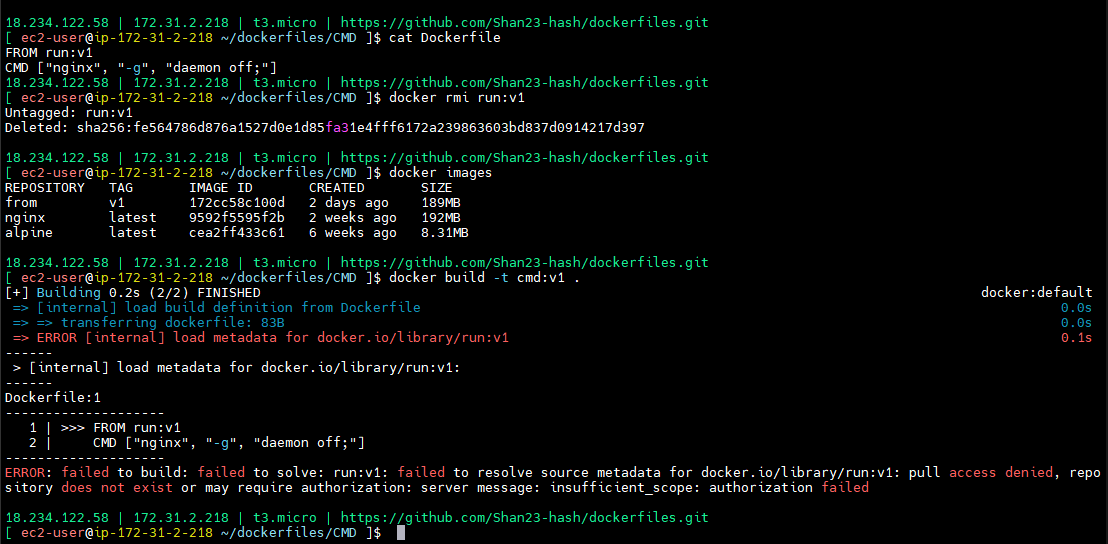
Daemon off - continuous running

**docker rmi run:v1**

**docker images**

**docker build -t cmd:v1 .**

Here we are given run:v1 but we removed that one that’s why error got.

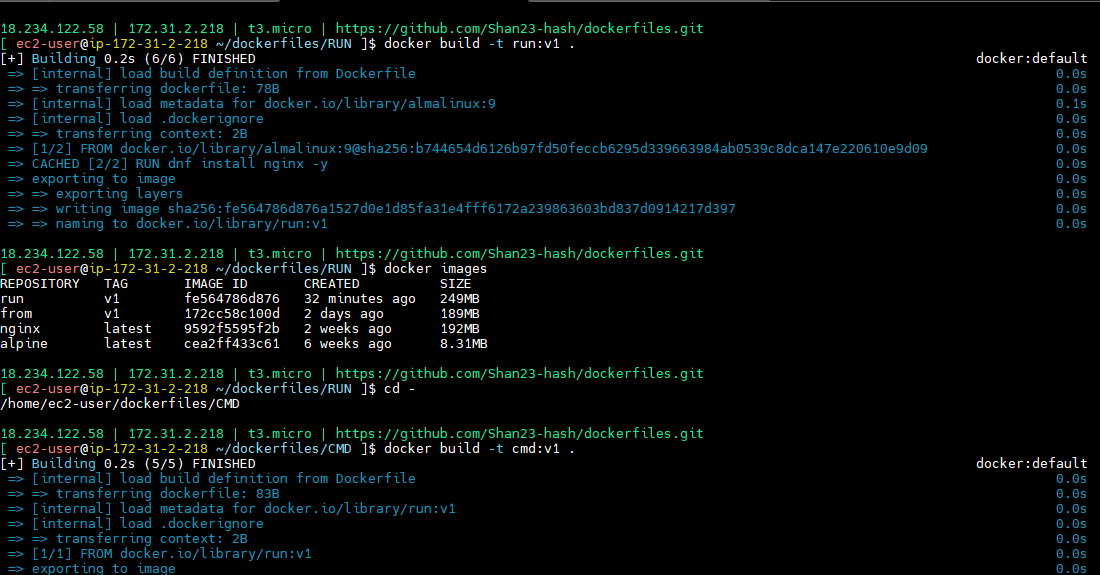


docker pull nginx -> first it checks locally, if it does not exist it checks in hub.

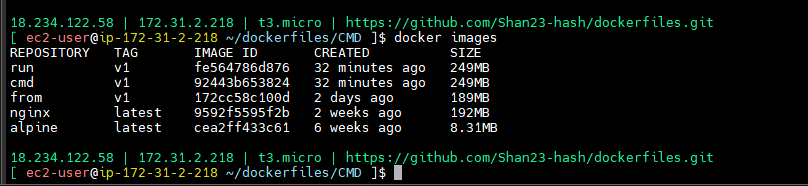
docker build -t run:v1 .

docker images

docker build -t cmd:v1 .



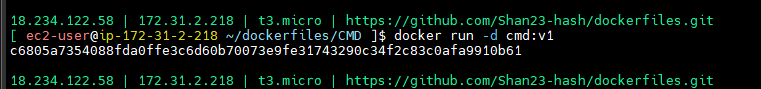
docker images



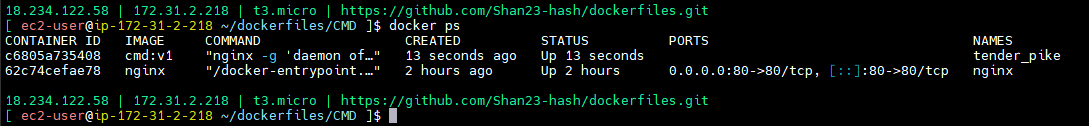
Cmd instruction was run or not

There instruction 1 by 1 is there

**docker run -d cmd:v1**



**docker ps**



Cmd container was running

**Imp que**

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

**CMD**

**=========**

**CMD executes at the time of container creation i.e at the time of docker run. there should be only one CMD instruction inside Dockerfile**

Can I have multiple cmd commands

No, cmd should be only one instruction, one only we will use docker file. If you given multple take latest one.

Run and cmd differnce

Run: a contaioner can have multiple run instructions

Cmd: Docker file will have on;y one cmd instruction.

In nginx we have download code.

Html directory path -- /usr/share/nginx.html

How to add code there

So than an instruction called container.

**COPY**

**=========**

**copies the code from local to container**

**COPY/Dockerfile**

FROM almalinux:9

RUN dnf install nginx -y

RUN rm -rf /usr/share/nginx/html/index.html

COPY qi/ /usr/share/nginx/html/

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

**COPY/index.html**

<h1>Hello, I am index file from container</h1>

--> Write docker file in ncopy dirctory

--> want to put code in same dirctory so I’m giving index.html

--> so this one I’m copy to local

--> usally copy here source and destination. Source : index.html

--> destination should be /usr/share/nginx/html/index.html

--> in this html we h+ave defau+lt html also right.

--> no of instructions also you write.

Daemon-off - run in foreground

So directly you rn means it may happen override or not.

**COPY/Dockerfile**

FROM almalinux:9

RUN dnf install nginx -y

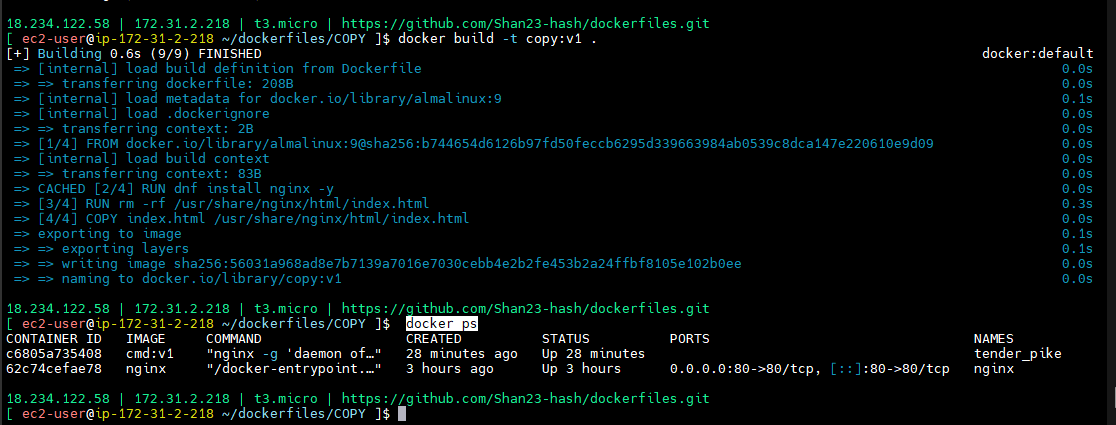
RUN rm -rf /usr/share/nginx/html/index.html

COPY index.html /usr/share/nginx/html/index.html

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

**docker build -t copy:v1 .**

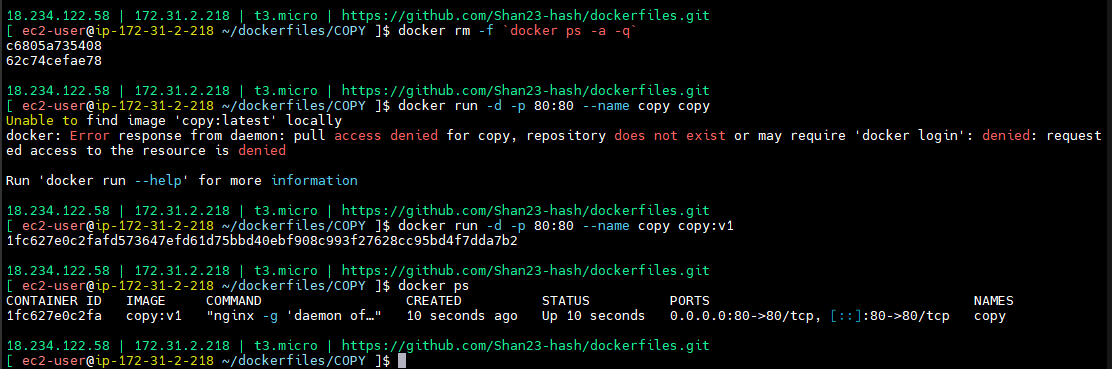
**docker ps**



**docker rm -f `docker ps -a -q`**

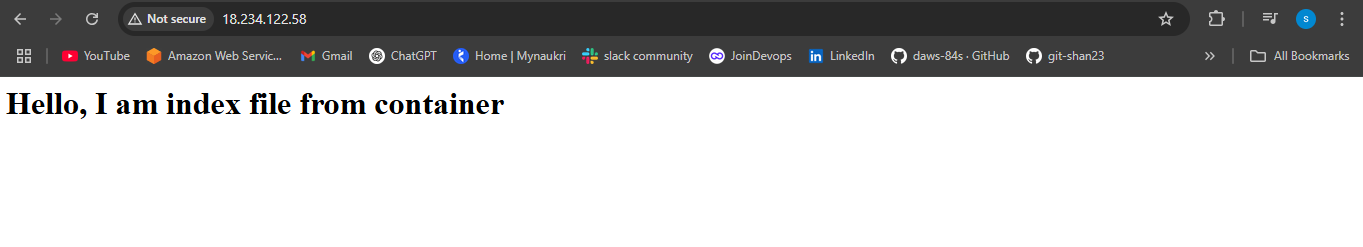
**docker run -d -p 80:80 --name copy copy:v1**

**docker ps**



Output:

<http://18.234.122.58/>



C:\devops\daws-84s\repos\dockerfiles\COPY\qi

qi - developer was build this one

qi / - in this directory list move to index.html

**COPY/Dockerfile**

FROM almalinux:9

RUN dnf install nginx -y

RUN rm -rf /usr/share/nginx/html/index.html

COPY qi/ /usr/share/nginx/html/

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

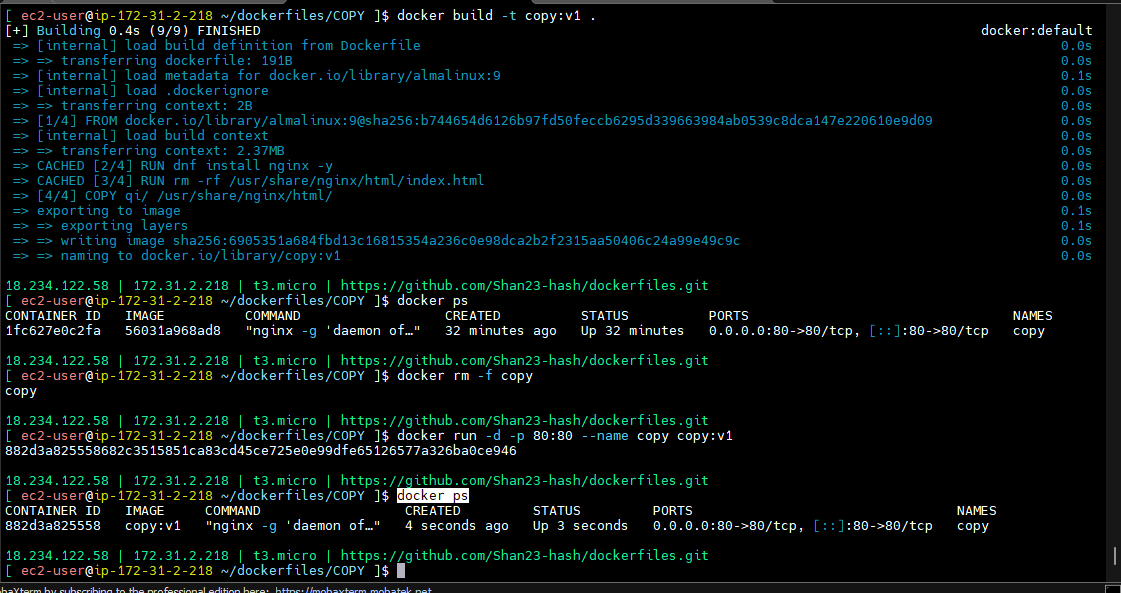
I did changes instruction in dockerfile. Shell I rebuild or not -- > yes should do

**docker build -t copy:v1 .**

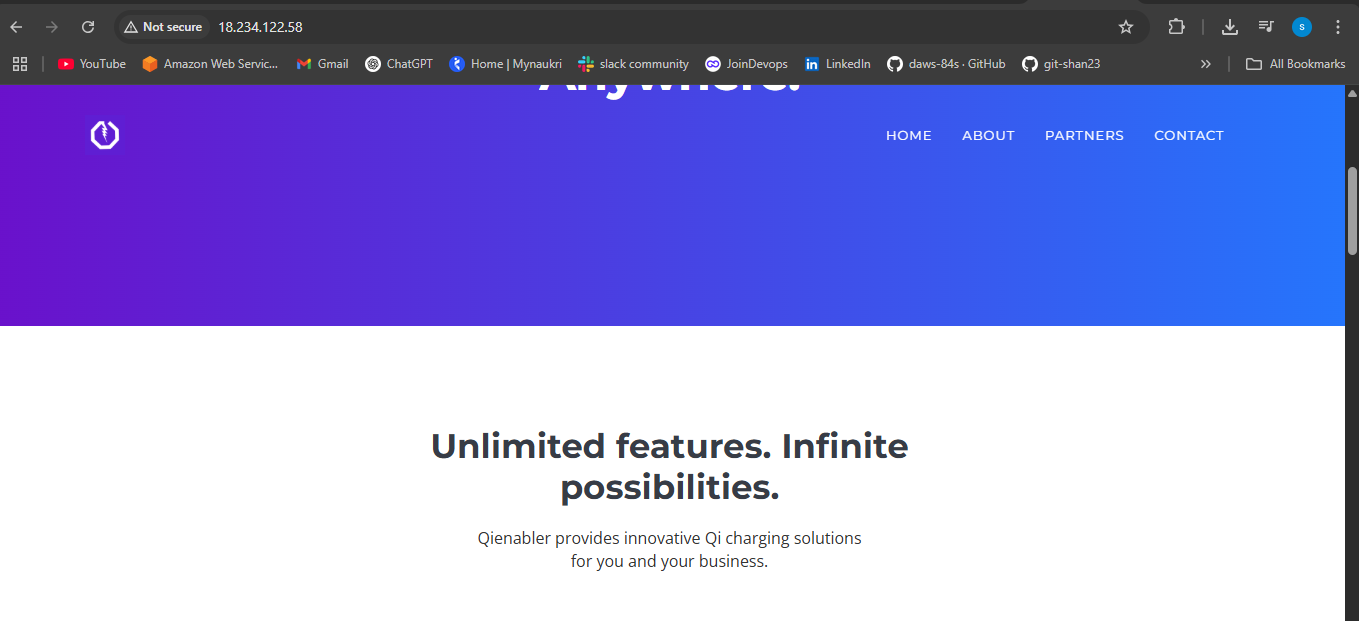
**docker rm -f copy**

**docker run -d -p 80:80 --name copy copy:v1**

**docker ps**



Output:



This is their website

In one server 10members static website will run very easily

In one t3.micro

Through run different different containers.

Port no will change.

8080,8081,8082 - maintain the websites is very easy

Local code copy to container.

Copy and add question also important

**ADD**

**=========**

COPY and ADD both copies the code from local to container. but it has two more advantages

1. it can directly fetch the file from internet

2. it can directly untar the file into container

**ADD/Dockerfile**

FROM almalinux:9

RUN dnf install nginx -y

RUN rm -rf /usr/share/nginx/html/index.html

ADD <https://raw.githubusercontent.com/Shan23-hash/Notes/refs/heads/main/session-50.txt> /usr/share/nginx/html/index.html

RUN chmod +r /usr/share/nginx/html/index.html

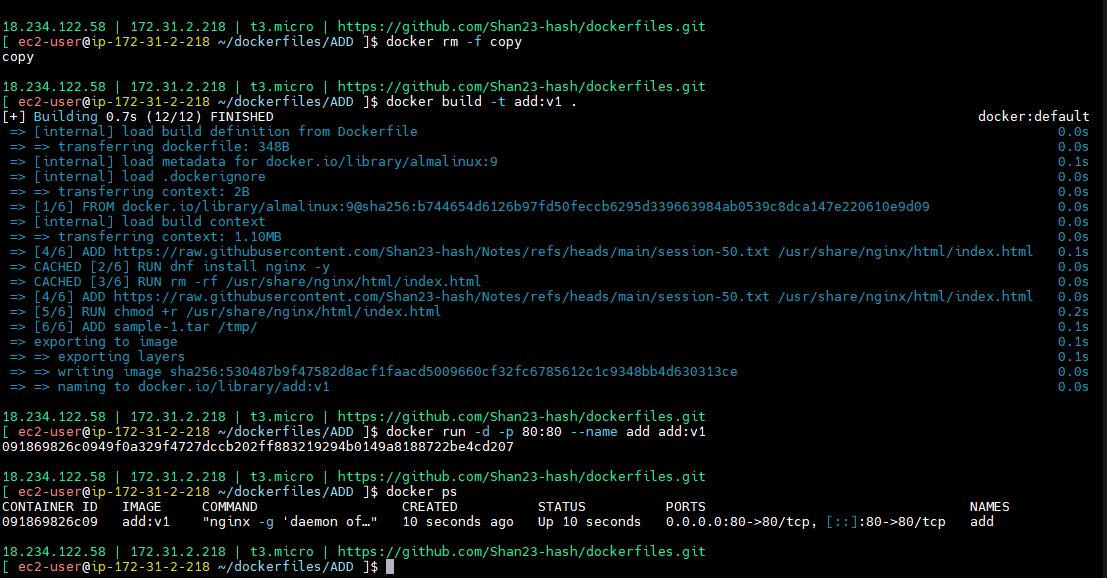
CMD ["nginx", "-g", "daemon off;"]

**docker rm -f copy**

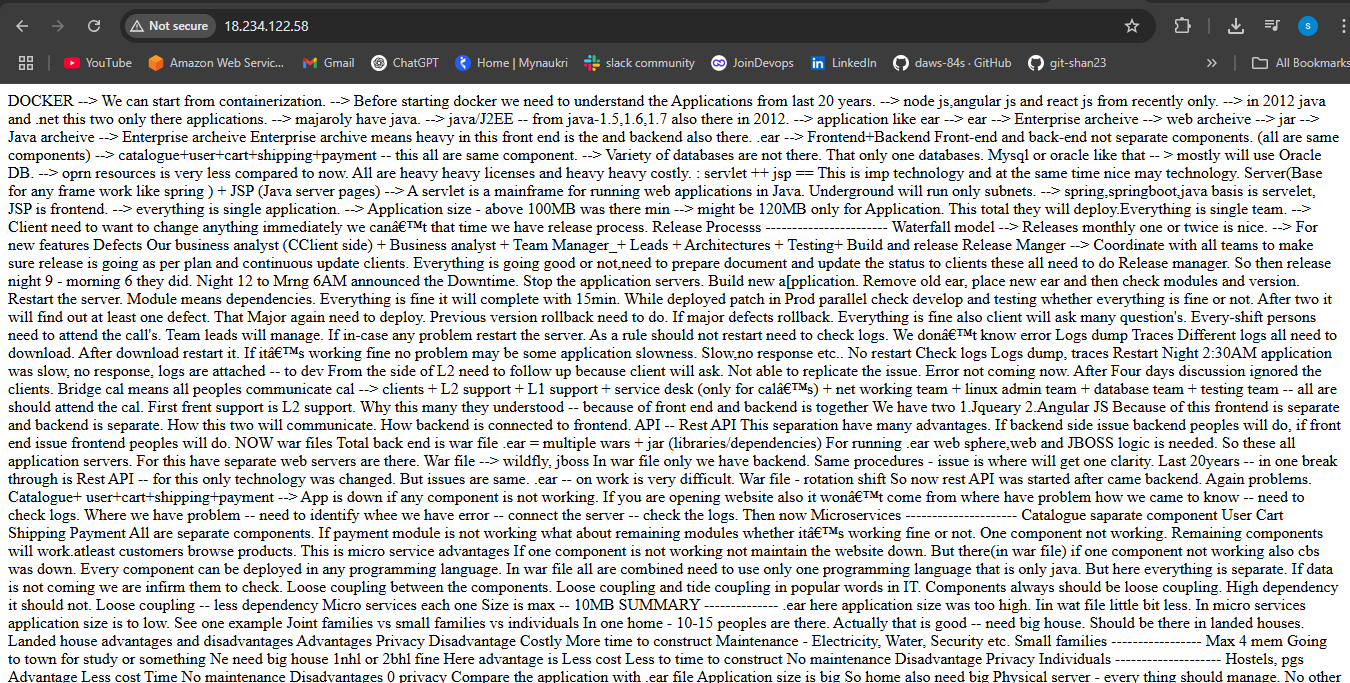
**docker build -t add:v1 .**

**docker run -d -p 80:80 --name add add:v1**

**docker ps**



<http://18.234.122.58/>

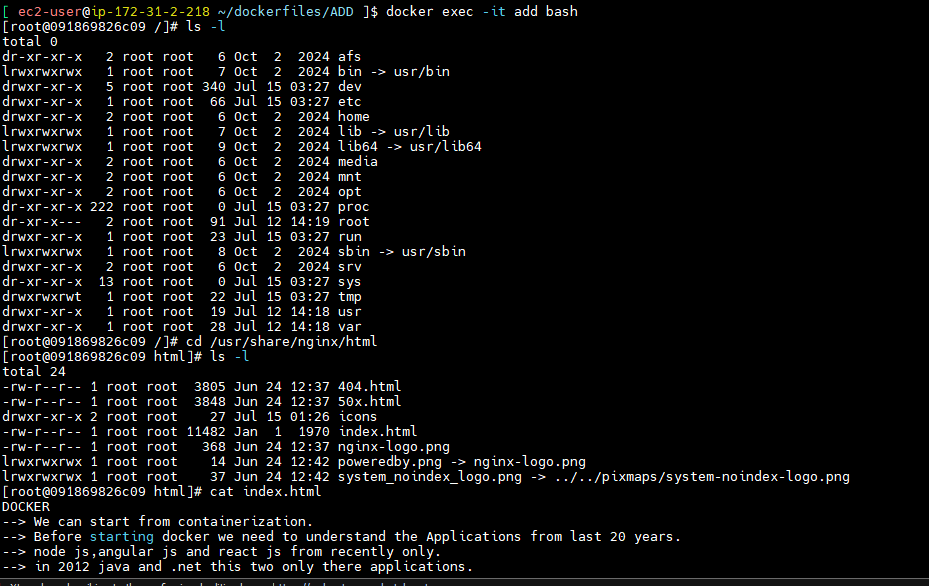


**docker exec -it add bash**

**cd /usr/share/nginx/html**

**ls -l**

**cat index.html**



In index.html output content only.

2. it can directly untar the file into container

Take any sample tar file

FROM almalinux:9

RUN dnf install nginx -y

RUN rm -rf /usr/share/nginx/html/index.html

ADD <https://raw.githubusercontent.com/Shan23-hash/Notes/refs/heads/main/session-50.txt> /usr/share/nginx/html/index.html

RUN chmod +r /usr/share/nginx/html/index.html

ADD sample-1.tar /tmp/

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

**docker rm -f add**

**docker build -t add:v1 .**

**docker run -d -p 80:80 --name add add:v1**

**docker exec -it add bash**

**cd /tmp**

**ls -l**

**cd sample-1**

**ls -l**

