After installing docker,

Step 1: check docker version, command for that "docker --version"

Step 2: command "docker info"

-if the client and server details are shown, it means your docker engine is running.

-in DevOps we use Docker Buildx and Docker Compose

\*Difference between Image and Container:

- Image

-is set of instracutions have given

-it is just a template

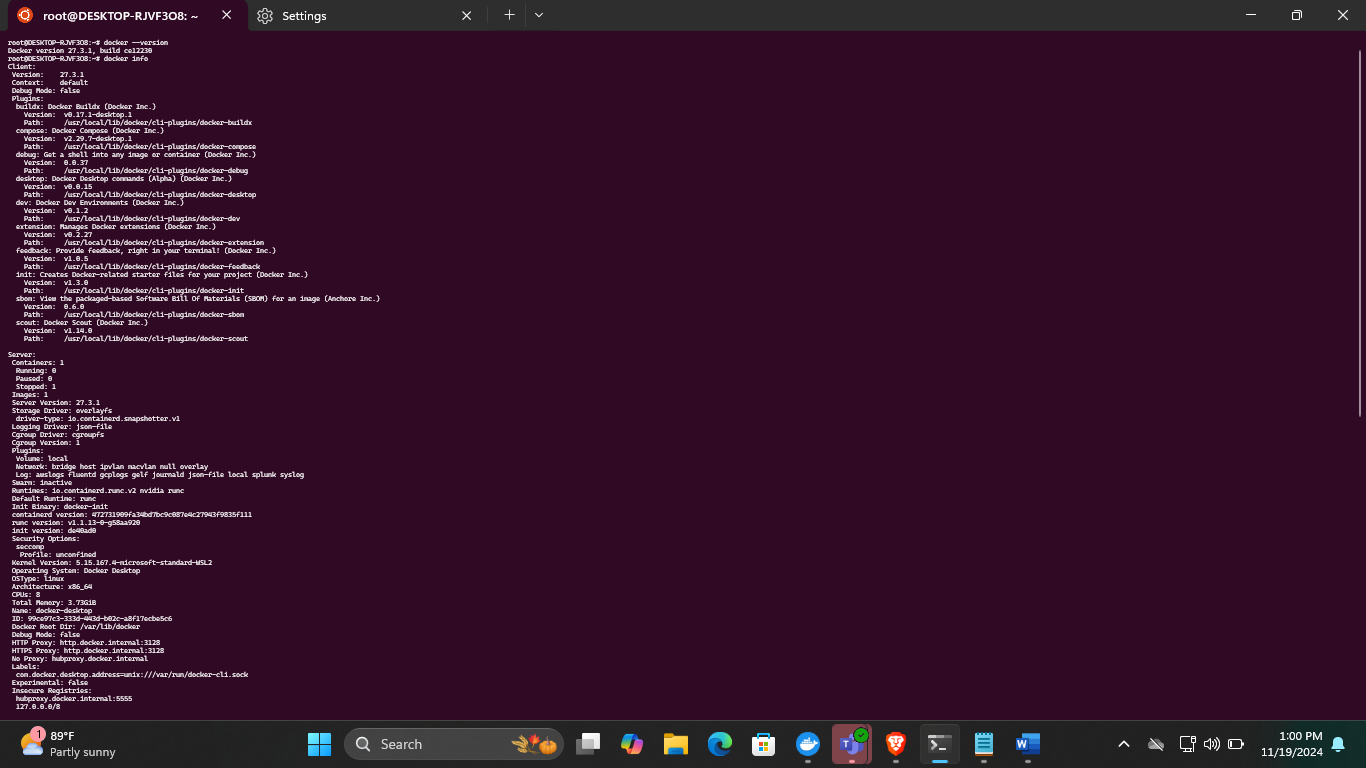
-in not running script

-can anyone take and use

-it has everything that needs to run a OS in it.

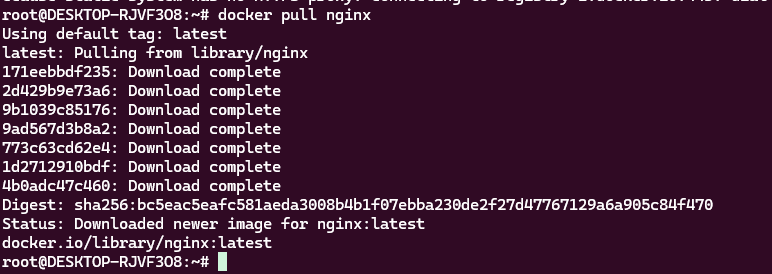
- Container

-When you start running that Image it becomes a container.



Step 3: command "docker pull nginx"

-when you give docker pull command to the nginx and not specify it's version, it always pull the lateset one.

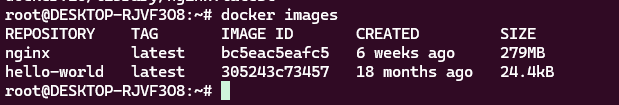


Step 3.1: when i want to pull a specific nginx image i use tag, so what is tag, let's take this command "docker pull nginx:1.26.2-alpine3.20-perl" as an example

":1.26.2-alpine3.20-perl" this a tag to the image that used in the above command.



Step 4: docker images



Step 5: docker ps (List all running containers)

  
  
docker ps -a -List all running containers including stopped ones  

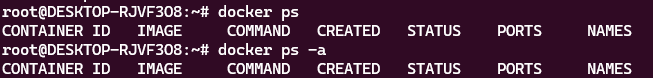

Step 6: command "docker stop <container>" - Stop a running container.

-before attempting remove container you need to stop the container first.

Step 7: command "docker rm <container>" -remove a stopped container.

-Check if it is removed.



Checking if it is removed  


Step 8: now i want to run nginx but it is in as a image. command to check that "docker images | grepnginx"

- A Docker image is a read-only template containing a set of instructions for creating a container that can run on the Docker platform, where did it get it from?

- it pulled that nginx image from library/nginx.

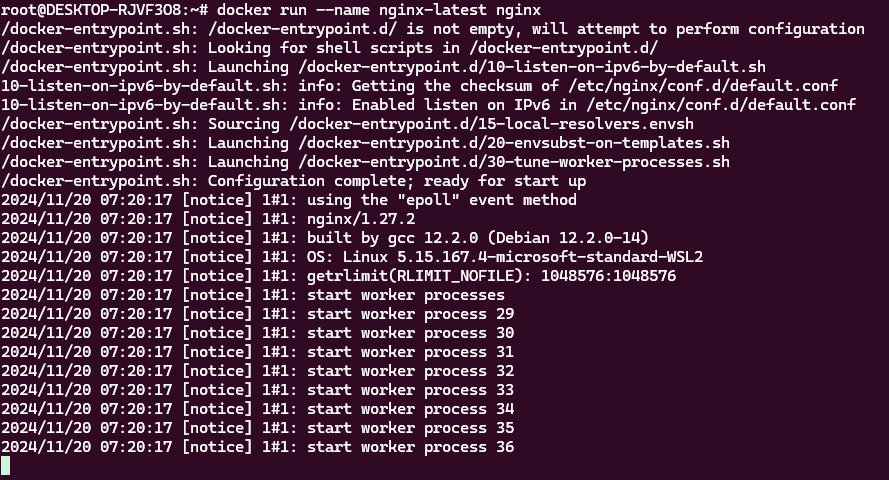
- website - docker.io/library/nginx:latest



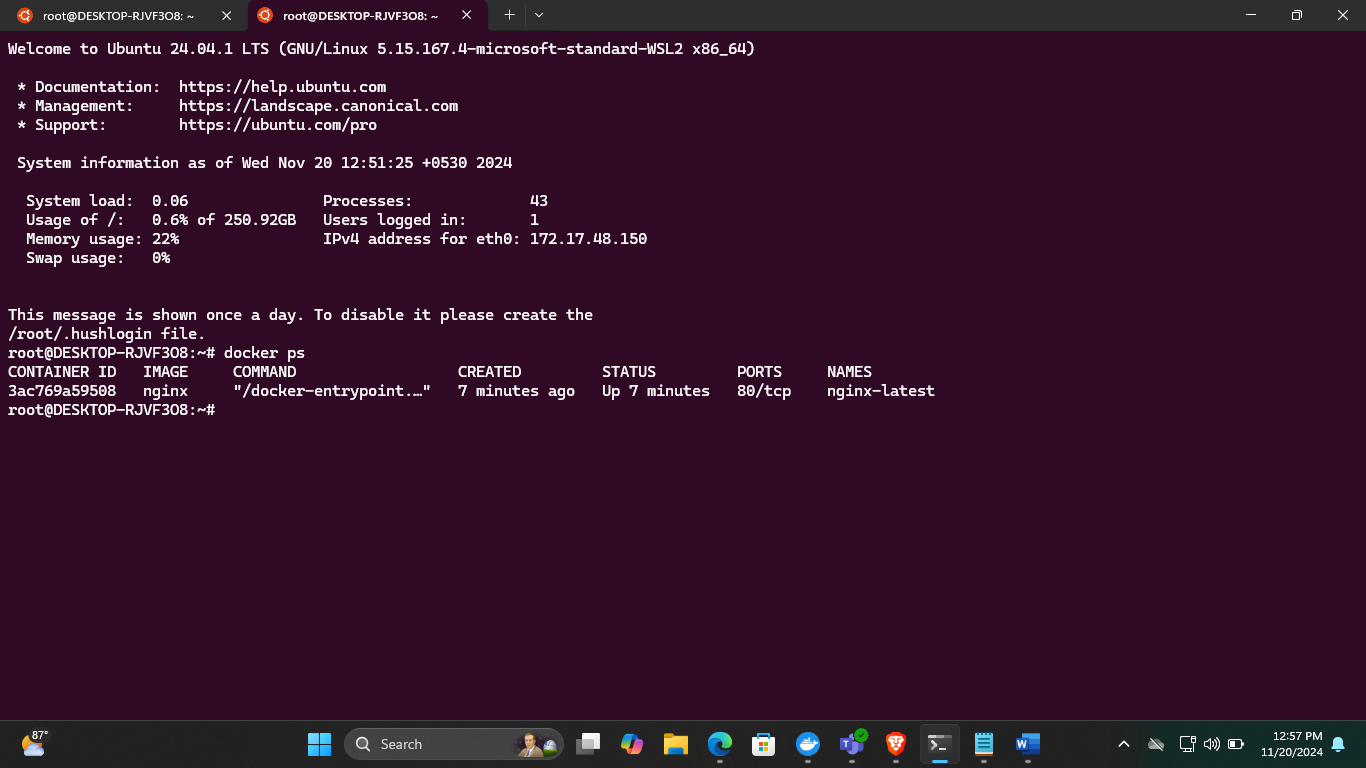
Step 9: how to run nginx container

-command "docker run --name <container name> <image name>"

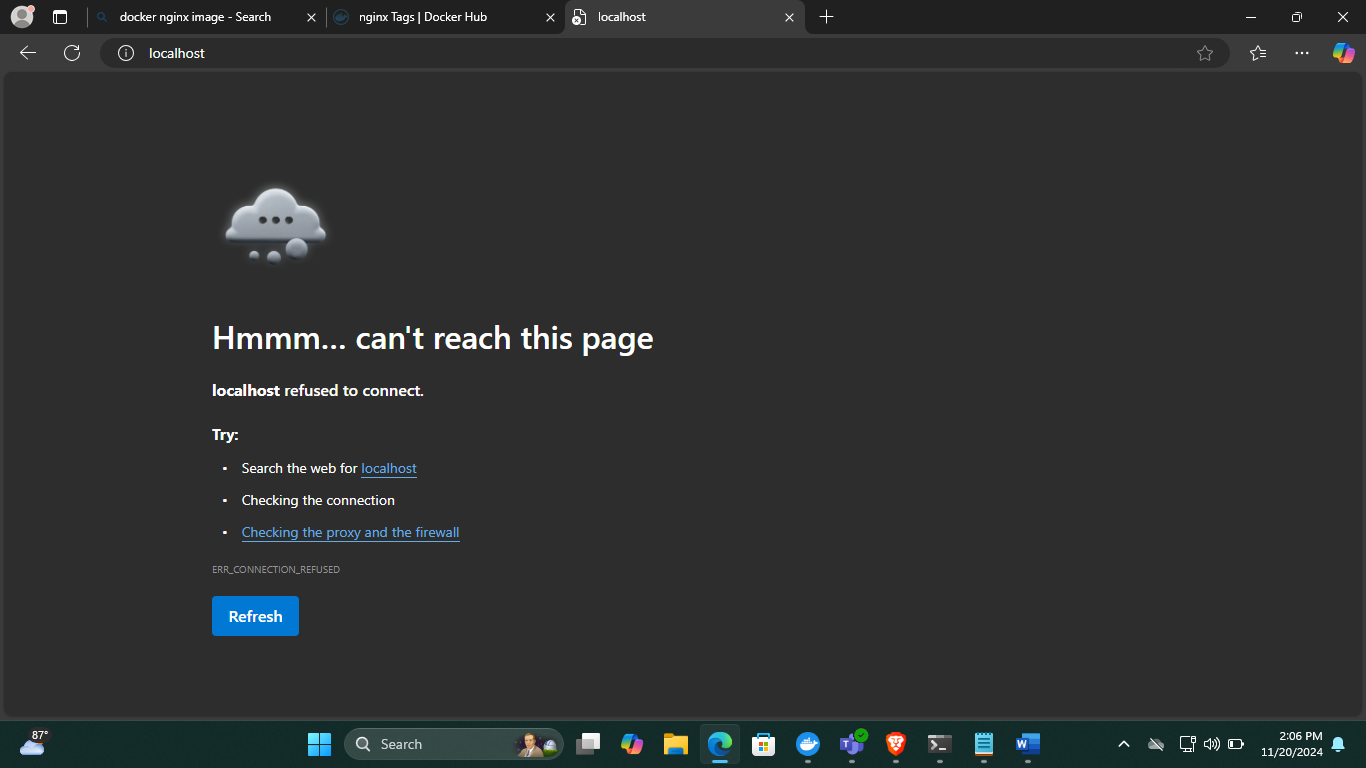
-this command will run the nginx container, in the terminal we can't write next command after running the container, so i am opening another tab to see the running container for that i am giving this command "docker ps"



Step 9.1: open another tab to see running container

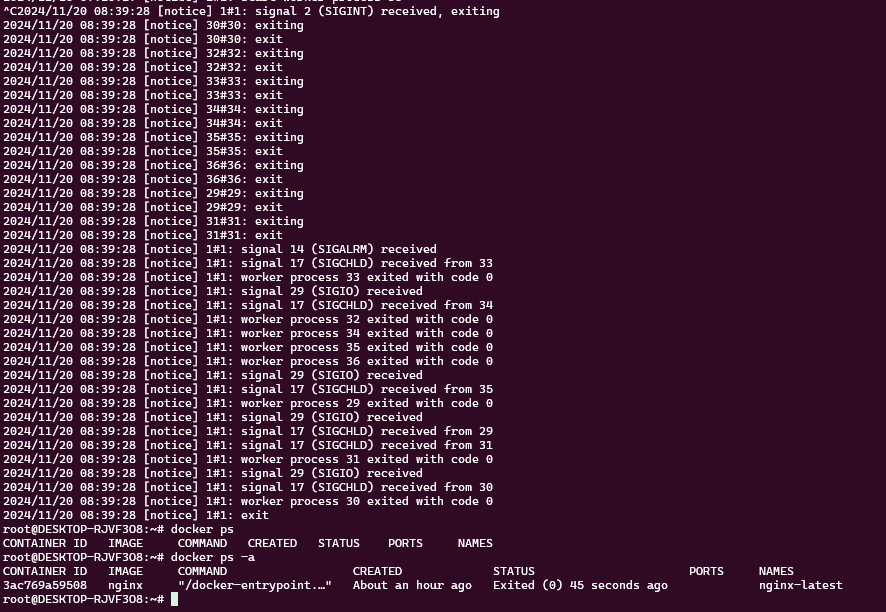


Step 10: check if it is running on my machine, to do that i go to web browser and search for localhost, and the browser said can't reach this page. what does that mean? it means nginx container has not move to my machine to run as localhost, it is currently running on its network (docker network).



Step 11: command, ctr+c to exit

-command "docker ps -a" to check the exited container



Step 12: detach mode:

-The detach option on the docker command line indicates that the docker client (docker) will make a request to the server (dockerd), and then the client will exit while that request continues on the server. Part of the confusion may be that docker looks like a single process, where in reality it is a client/server application where the client is just a thin frontend on a REST API to send every command to the server.

With a "docker container run --detach", this means the container will be created, the server will respond with a container id if successful, and the container will continue to run on the server while you are free to run other commands. This is often used for a server (e.g. nginx) you want to start in the background while you continue to run other commands. Note that you can still configure a container with the -interactive and -tty options (often abbreviated -it) and later run a docker container attach to connect to an already running container. (Note, until you attach to the container running with -itd, any attempt by the container to read from stdin would hang, instead of seeing an end of input that often triggers an immediate exit if you just passed -d.)

The opposite of detached mode is foreground mode. That is the default mode, when -d option is not used. In this mode, the console you are using to execute docker run will be attached to standard input, output and error. That means your console is attached to the container's process.

In detached mode, you can follow the standard output of your docker container with docker logs -f <container\_ID>.

Why do we need --detach mode?

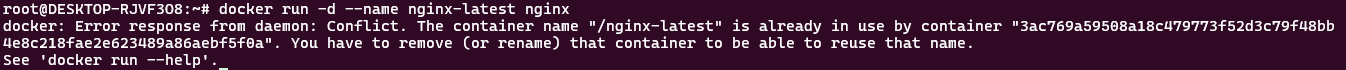
Server software is generally run in detached containers because it is rare that the software depends on an attached terminal.

Running detached containers is a perfect fit for programs that sit quietly in the background.

Note:

Generally, this type of program is called a daemon, or a service. A daemon generally interacts with other programs (or humans over a network) or some other communication channel. When you launch a daemon or other program in a container that you want to run in the background, remember to use either the --detach flag or its short form, -d.

when i run this command,"docker run -d --name nginx-latest nginx", it says, " The container name "/nginx-latest" is already in use by container "3ac769a59508a18c479773f52d3c79f48bb4e8c218fae2e623489a86aebf5f0a". You have to remove (or rename) that container to be able to reuse that name."

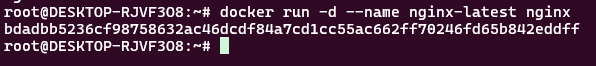


Step 13: to remove that container, it has to be already exited or stopped.

- command "docker rm <container name>"

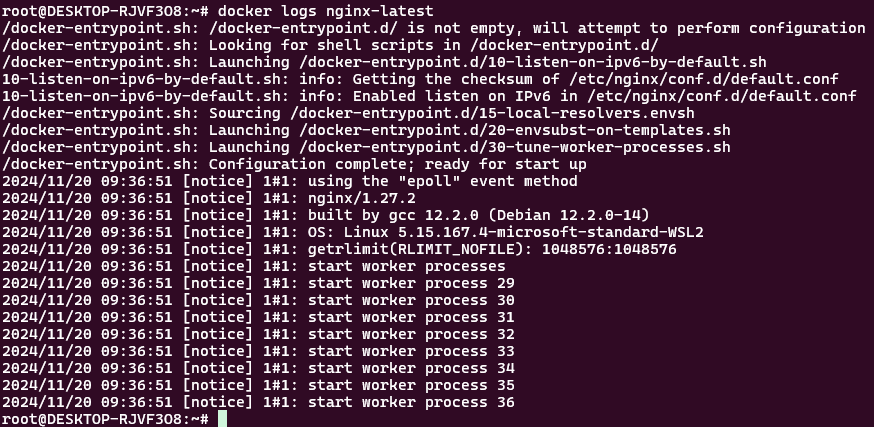


Step 14: now run detach command, "docker run -d --name <container name> <image name>"



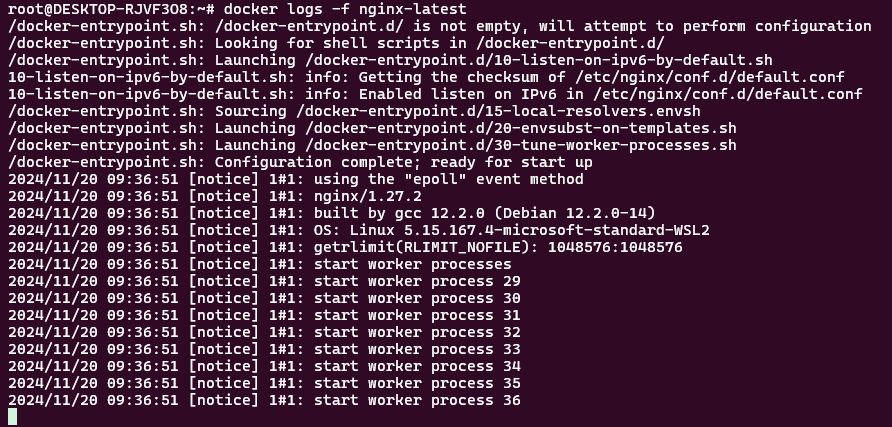
Step 15: to check whether it is running perfectly or it has erro we use log command

- command, "docker logs <container name>"

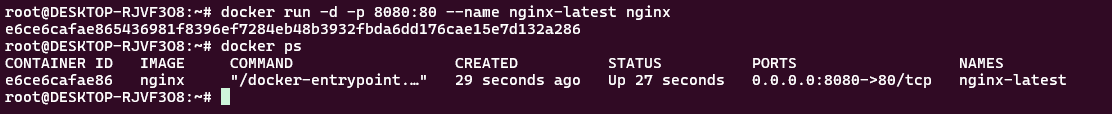


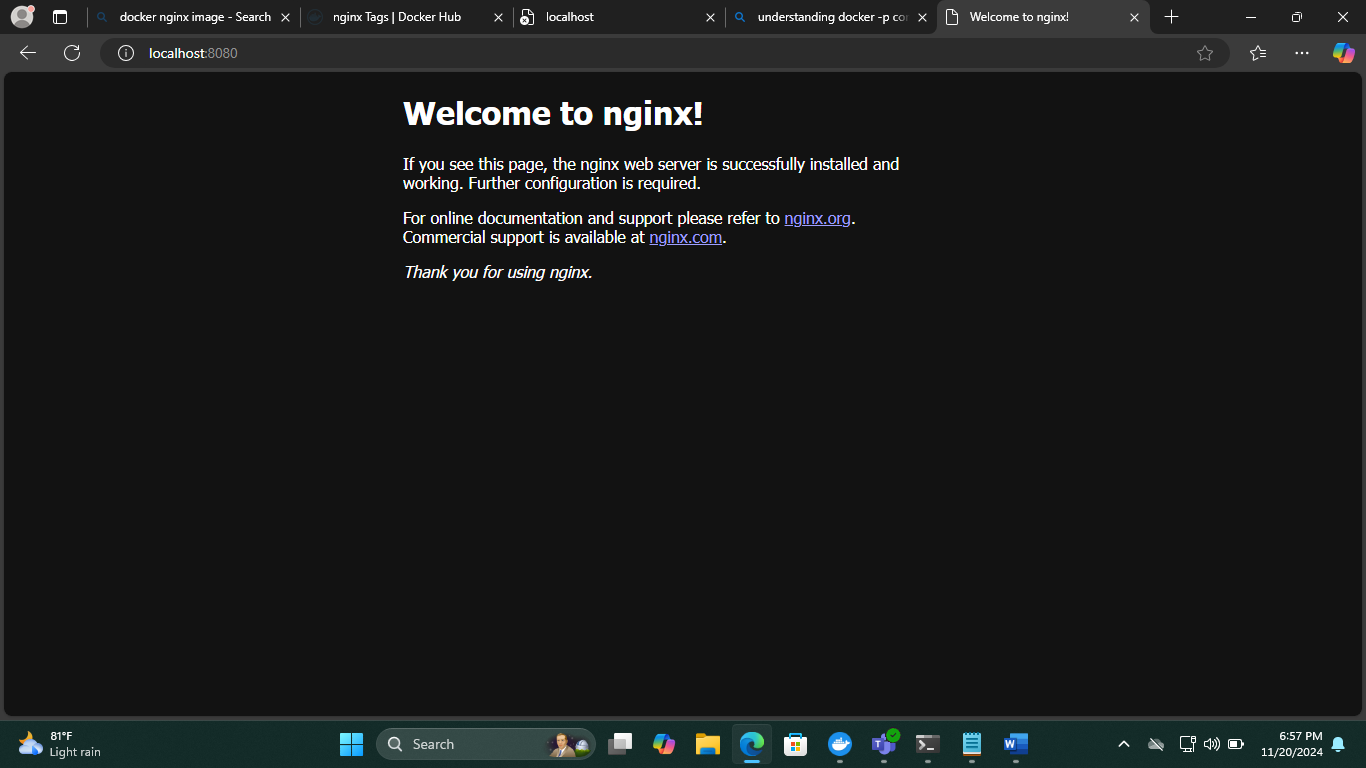
Step 16: command docker logs -f <container name>

-it shows every logs that keep entering, to be simply say it follows every log

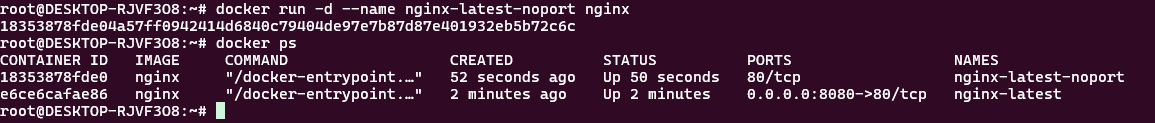


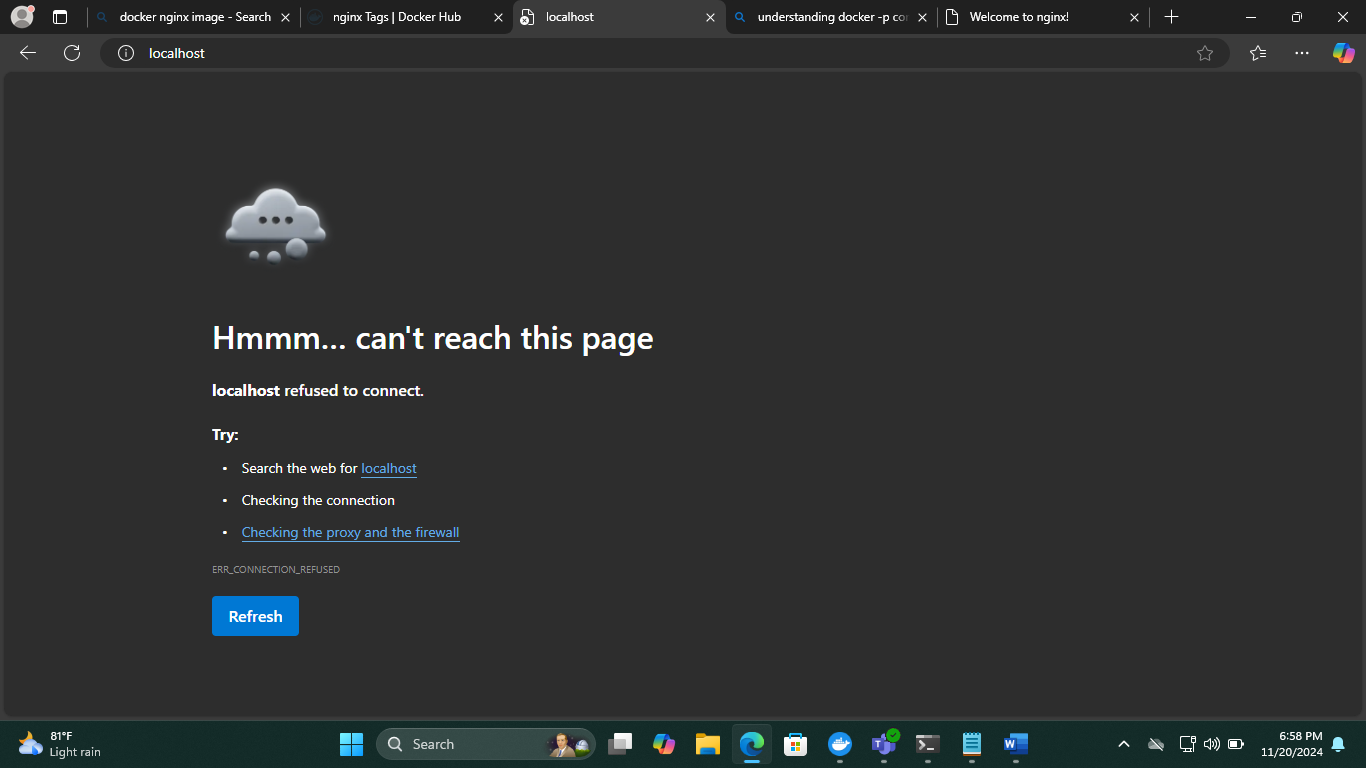
Step 17: -p (publish a container's port(s) to the host), command "docker run -d -p 8080:80 --name <container name> <image name> "



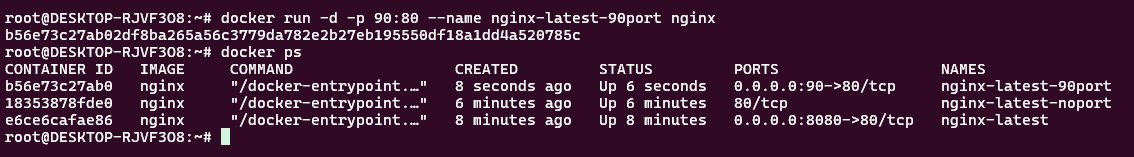


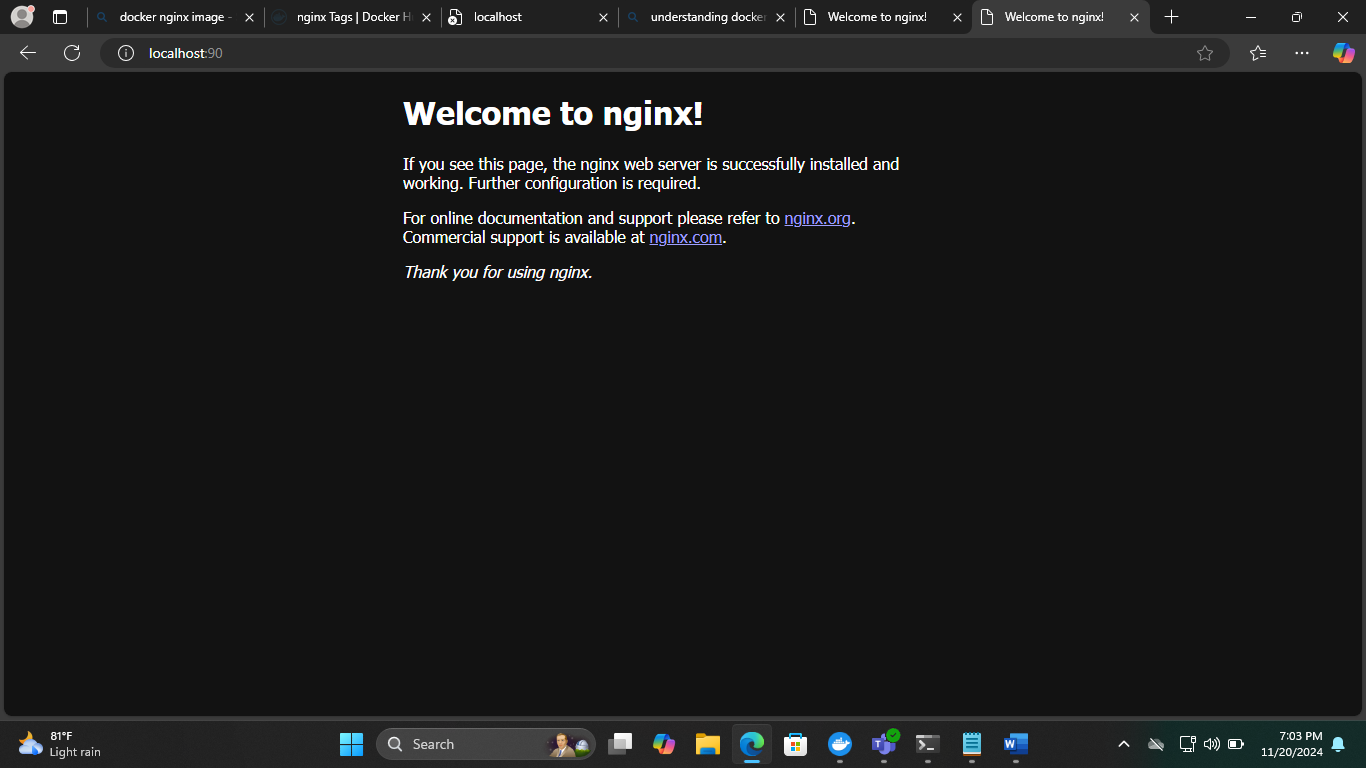
17.1





17.2



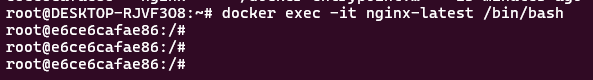


Step 18: entering into machine (login in a bash mode) command, "docker exec -it <container name> /bin/bash"

- exec (executive inside)

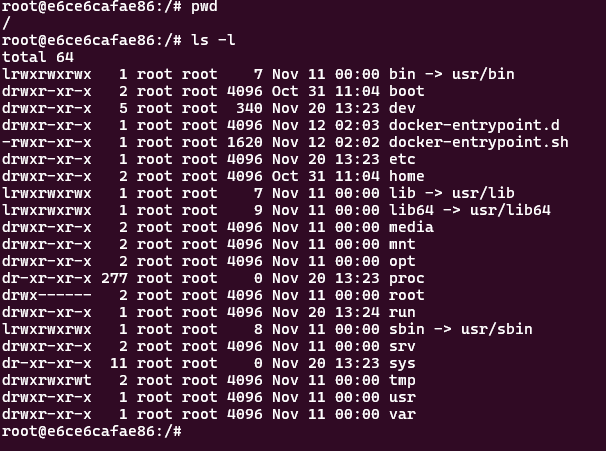
- i - intteractive

- t - terminal

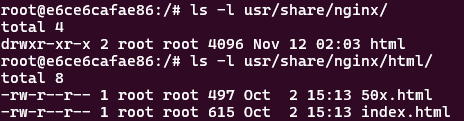


Step 19: once enter into machine, use command "pwd" and "ls -l" to check

- The pwd command, which stands for print working directory, is a fundamental and frequently used command in Linux. It displays the absolute path of the current working directory, starting from the root directory. This will print the full path of your current location in the file system.



Step 20: look for a html file in nginx, command, "ls -l usr/share/nginx/html/"



Step 21: Get inside that file, command, "cd ls -l usr/share/nginx/html/"



Step 22: cat the index.html file

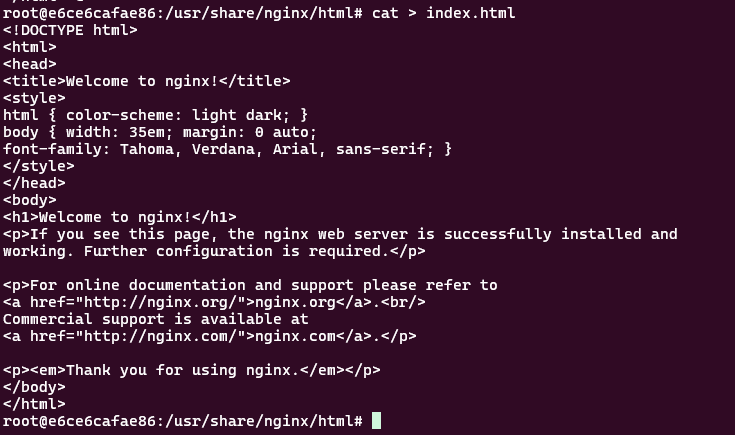
- copy the html file and put it in visual studio code and do some change in the html file to show "welcome to Jumisa"



Step 23: update the file, command "cat > index.html”, use ctrl+d command to save the file

- tried refreshing the web page, but nothing has changed so need to restart the container.

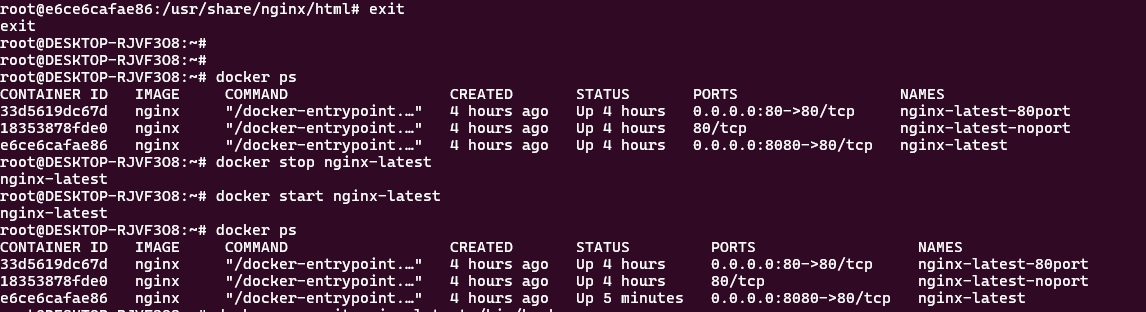
- before doing that use "exit" command to exit from the mechine.



Step 24: now retart the container to check whether it's updated or not

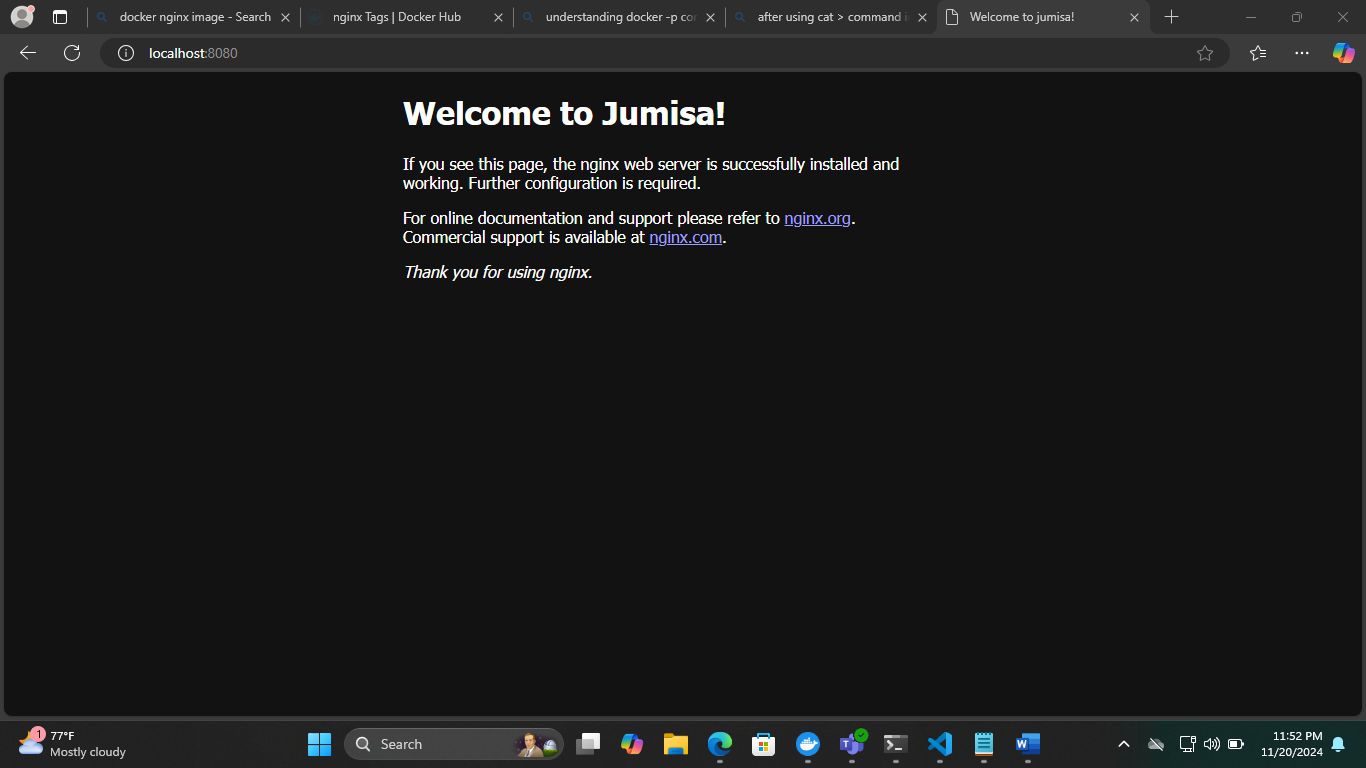
- docker stop nginx-latest

- docker start nginx-latest

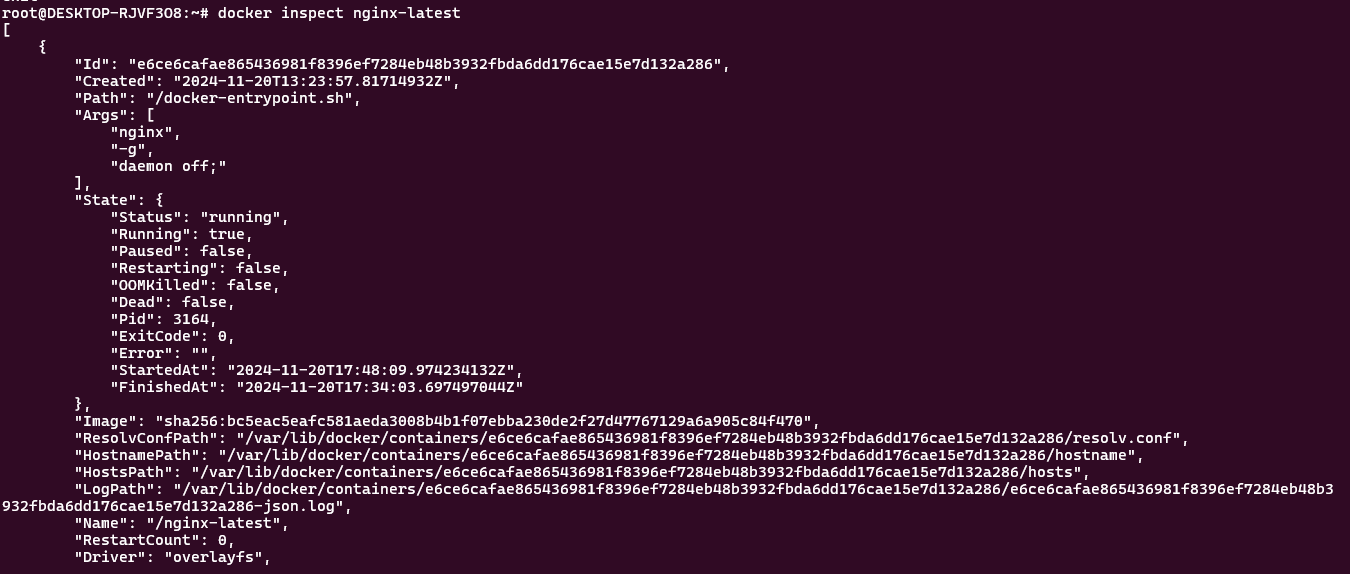


Step 25: now go back to the web page check it if it's updated.

- it's working!



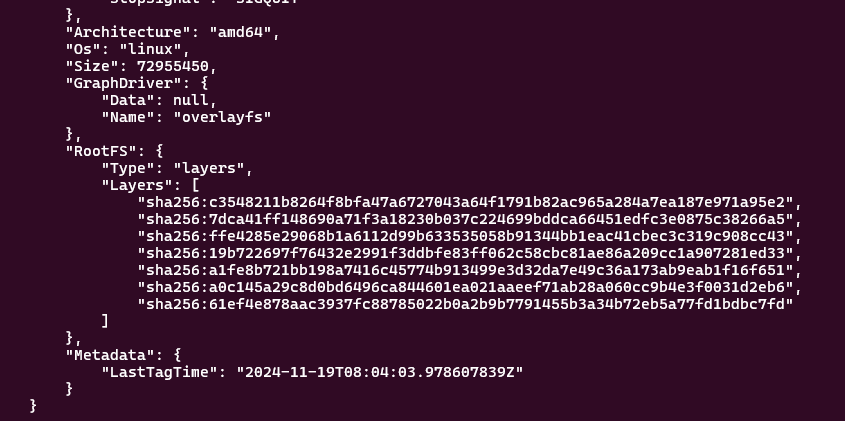
Step 26: inspect - get detailed information about container, command "docker inspect <container name>"







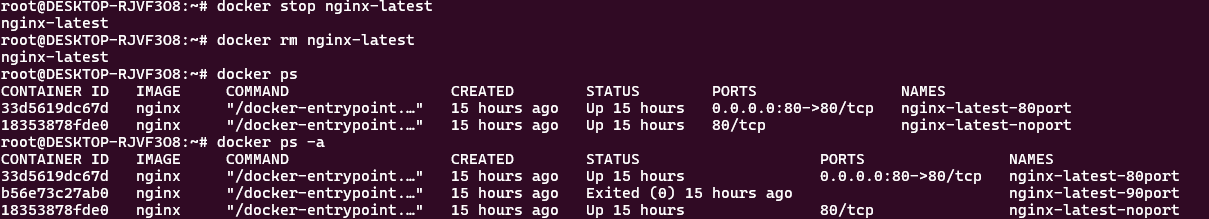
Step 27: Given task - to check what architecture - it will be inside image, command "docker inspect <image name>"



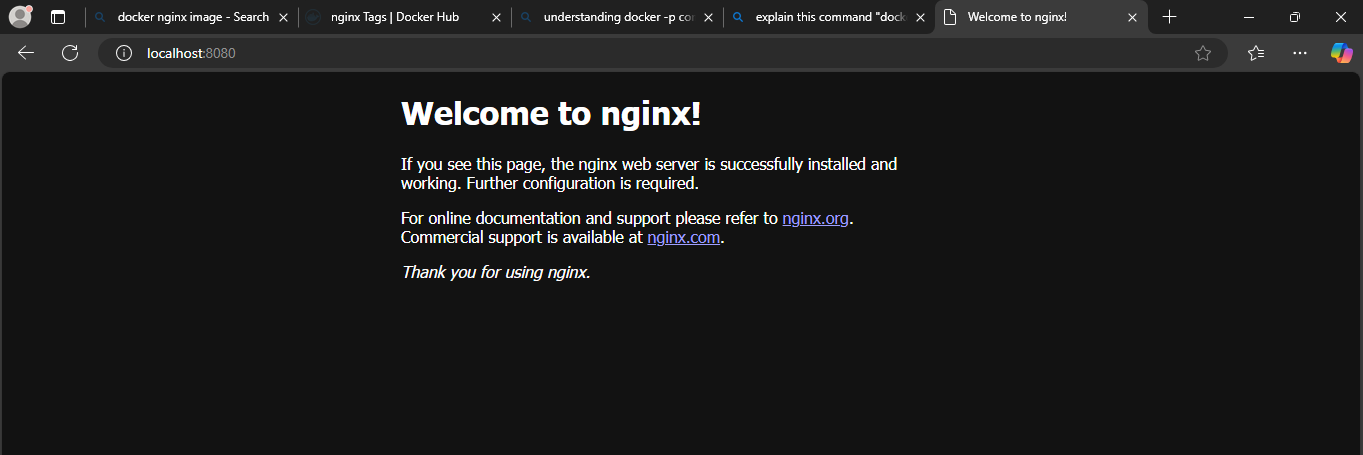
Step 28: restart command "docker restart <container name>"



Step 29: once i removed the container which i have made an update (changed nginx to Jumisa) later i run that container i will get the same old html file not the one i have made changes. so how to store and retain that? and for that we need a volume.

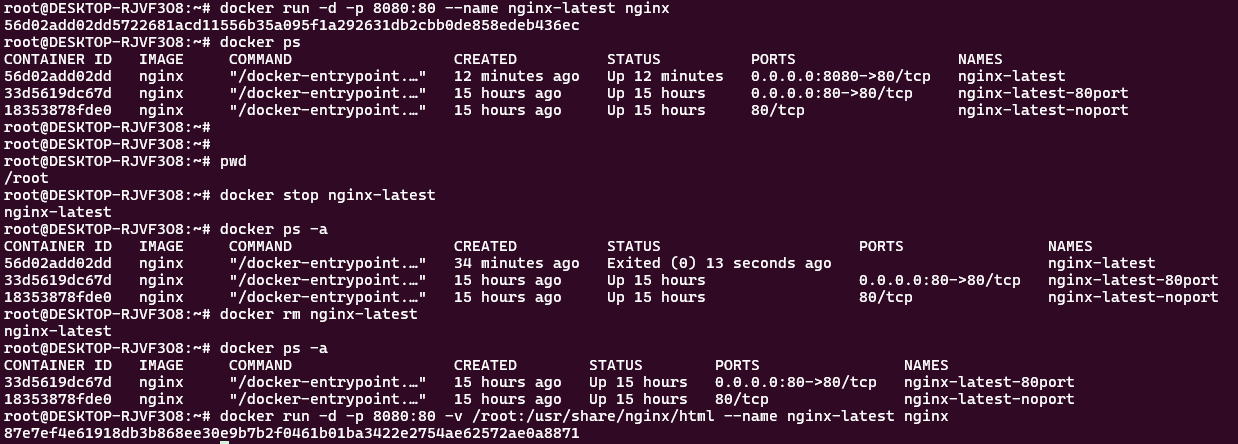


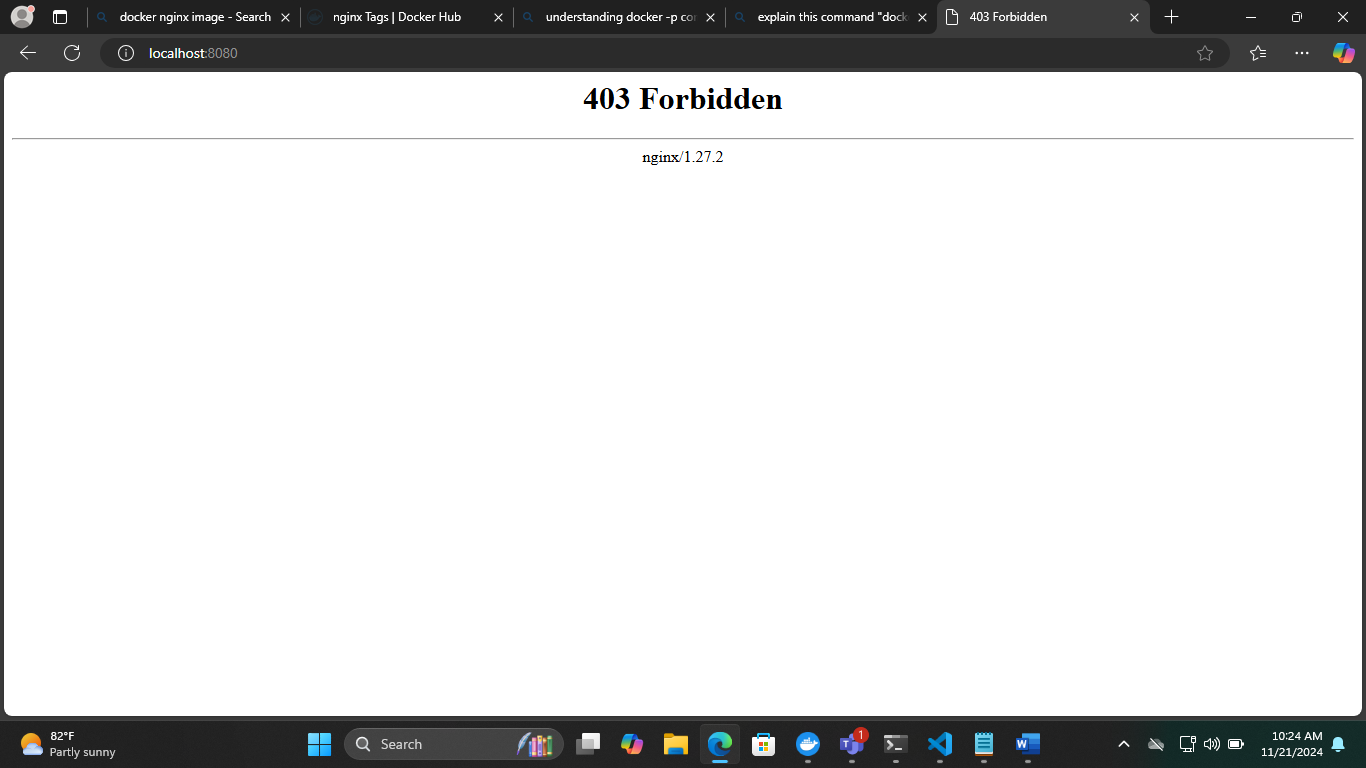




Step 30: Volume, command "docker run -d -p 8080:80 -v <root path>:<html file path> --name <container name> <image name>"

- refreshed the webpage it says "403 forbidden"

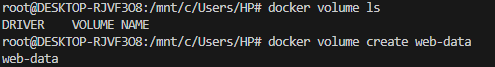




**Docker Volume**docker volume ls - list out all the docker volume

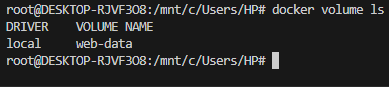
Step 1: docker volume create <volume name>

- to create a docker volume



Step 2: docker volume ls

- check if the volume is created



Step 3: docker run --rm -it -v web-date:/mnt alpine sh -

--rm:

This flag instructs Docker to automatically remove the container once it exits. This is useful for one-off commands or temporary containers.

-it:

This flag combination specifies two options:

-i: Interactive mode. Keeps STDIN open, allowing you to interact with the container.

-t: Allocates a pseudo-TTY. This is necessary for interactive sessions, providing a terminal interface.

-v web-date:/mnt:

This flag mounts a local directory or volume as a volume inside the container. Here's how it works:

web-date: This is the volume on my host machine. It is a named volume.

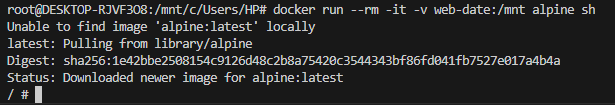
/mnt: This is the destination directory inside the container where the host volume will be mounted.

alpine:

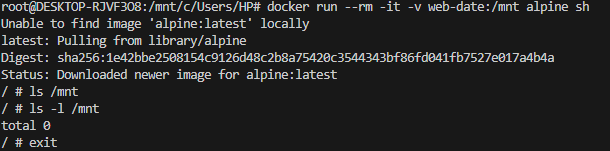
This specifies the Docker image to use for the container. Alpine Linux is a lightweight Linux distribution often used for Docker containers due to its small size.

sh:

This is the command to execute inside the container. sh launches a shell, providing you with a command-line interface to interact with the container's environment.



Used ls and ls -l command to list out what’s inside /mnt



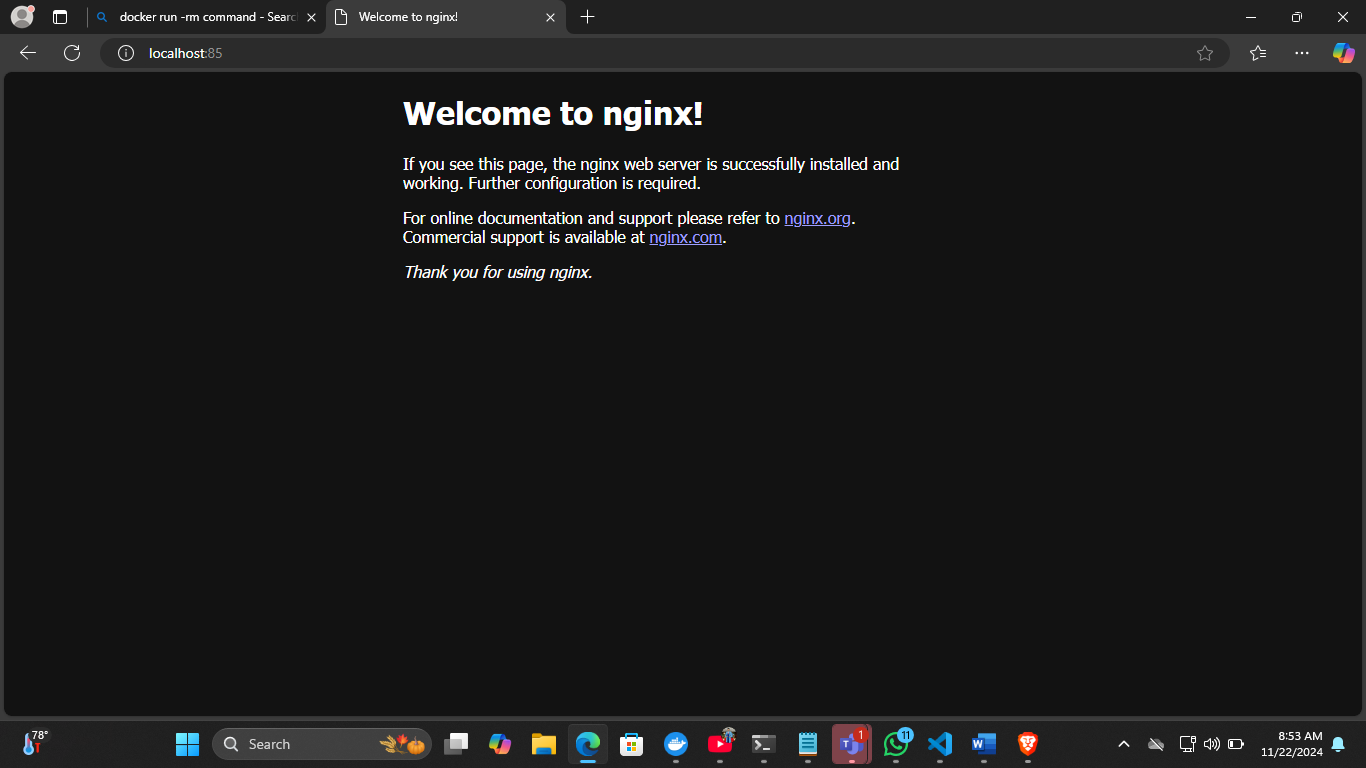
Step 4: docker run -d -v web-data:/usr/share/nginx/html -p 85:80 --name app nginx

- attaching it to the nginx

- check it in the browser if it's work



Check it in the browser



Step 5: entering into that container

- docker exec -it <container name> /bin/bash



Step 6: entering into the file

- cd /usr/share/nginx/html

- ls

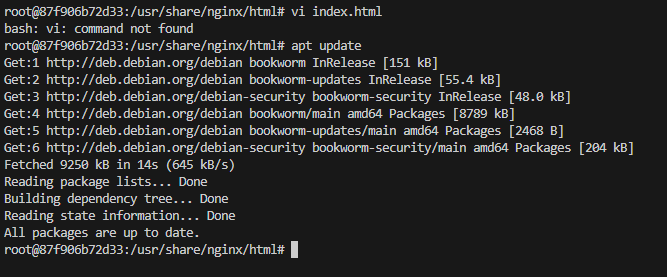


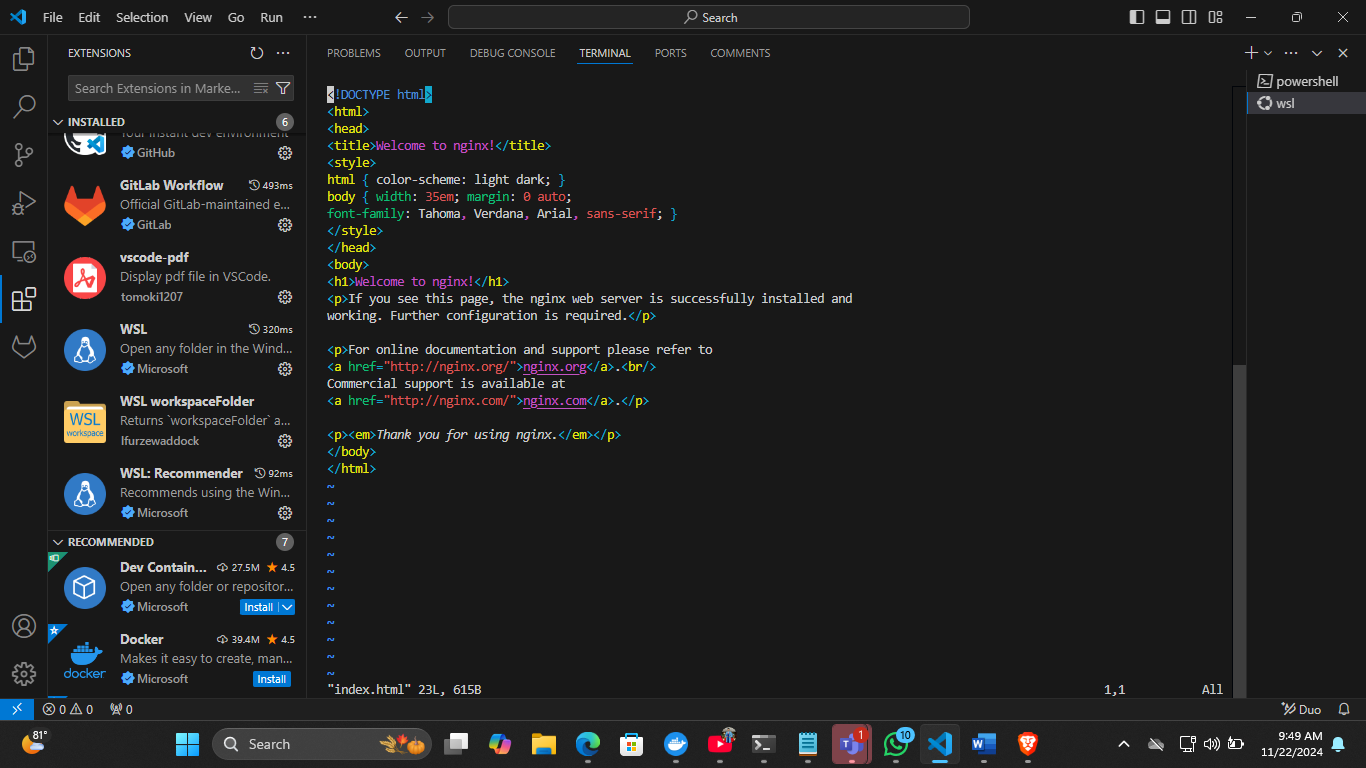
Step 7: edit html file

- vi index.html

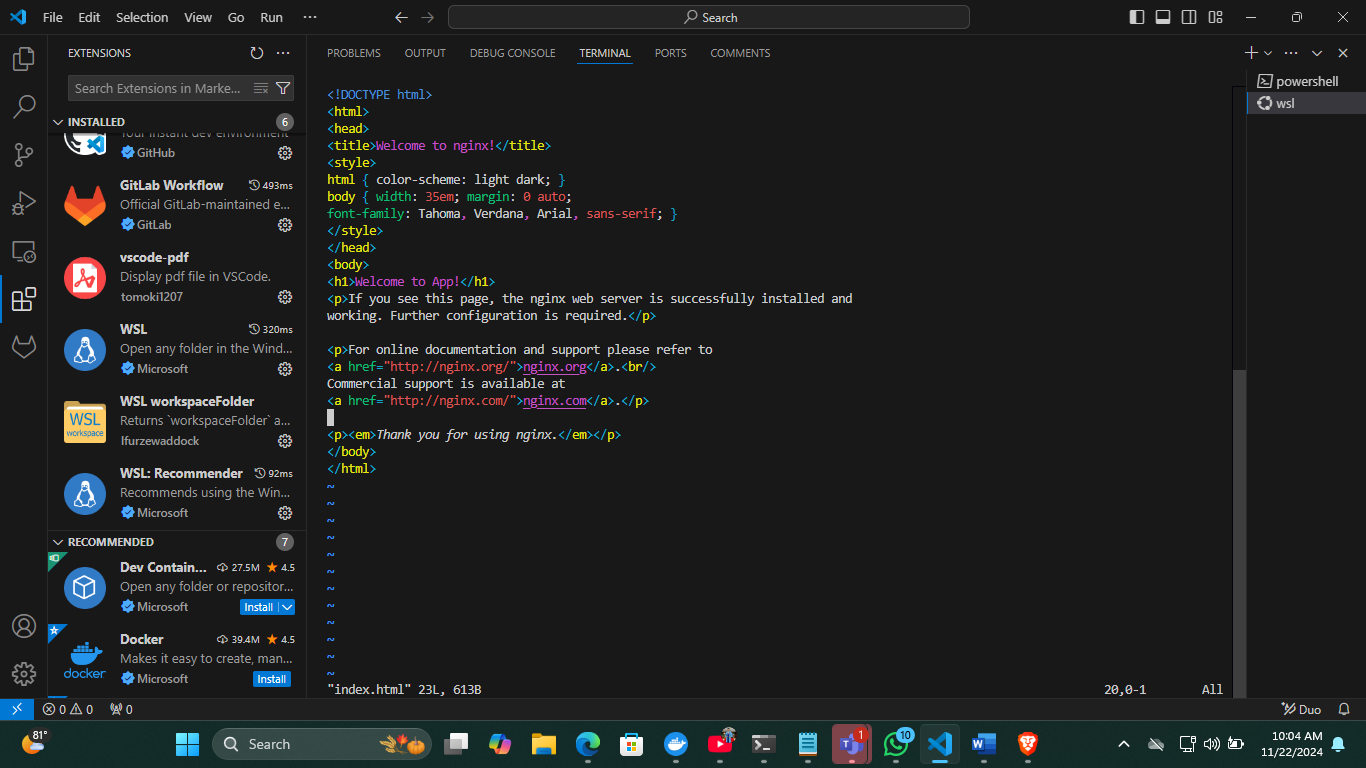
- it says vi command not found so updating using "apt update" command.

- apt install vim -y





Step 8: edit html file, "vi <file name>"



**i**: Insert text before the cursor

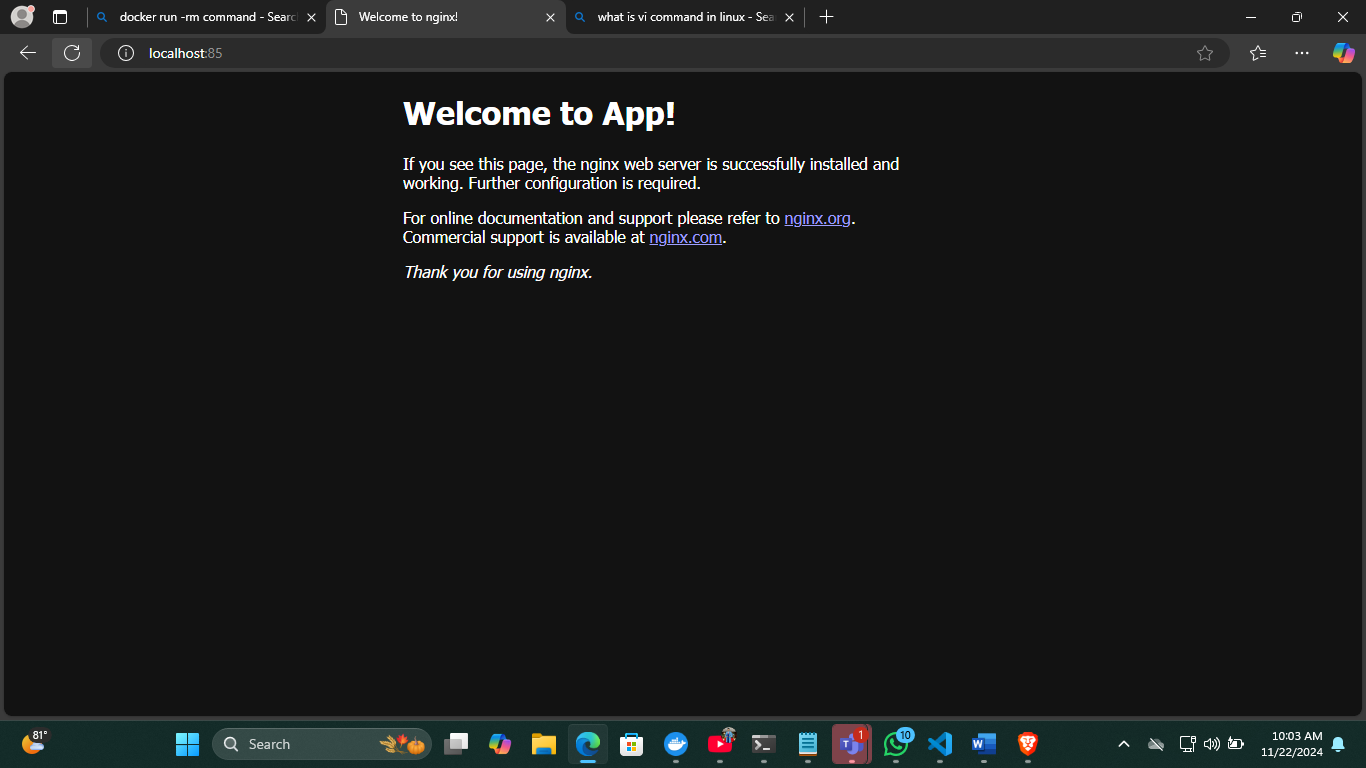
To save your changes and exit Vim:

Press Esc to enter Normal mode.

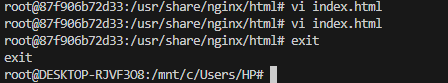
Type :w to save the file.

Type :q to quit Vim.

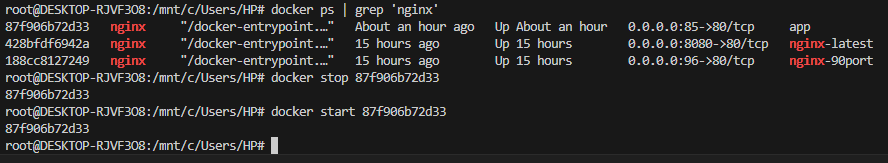
To save and quit simultaneously, type :wq.



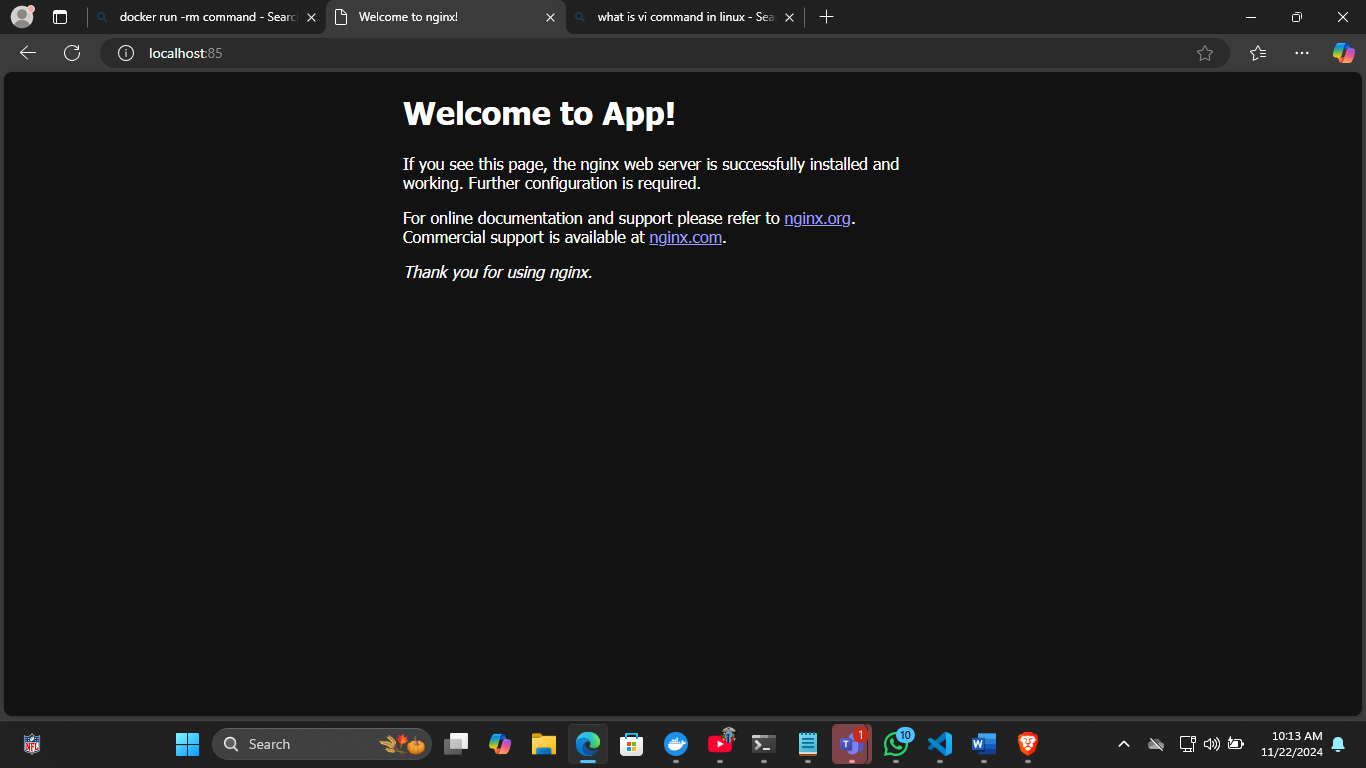
Step 9: exit



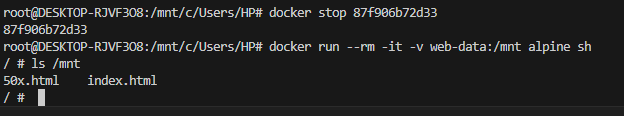
Step 10: check if volume works by stop and start container.



Successfully working!

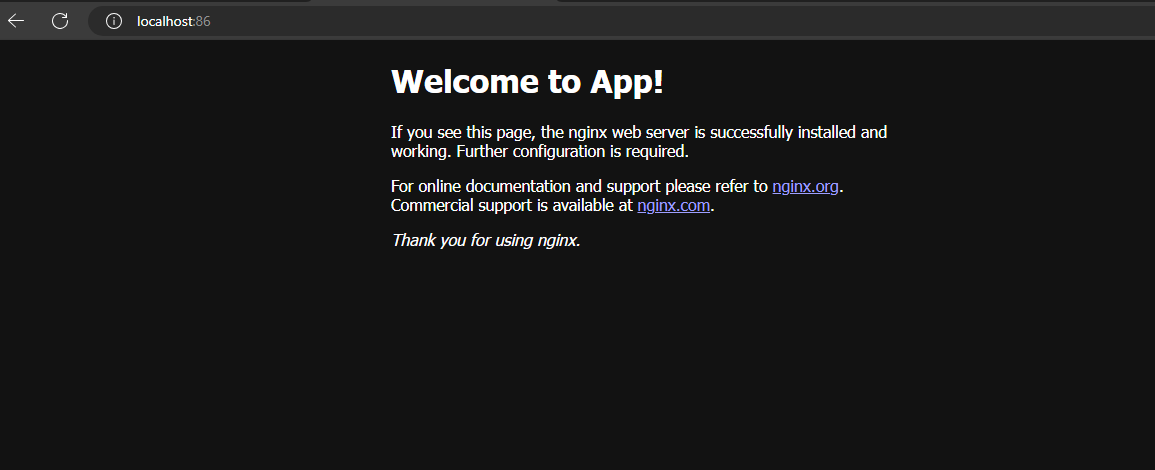


Step 11:



Step 12: adding volume to another container



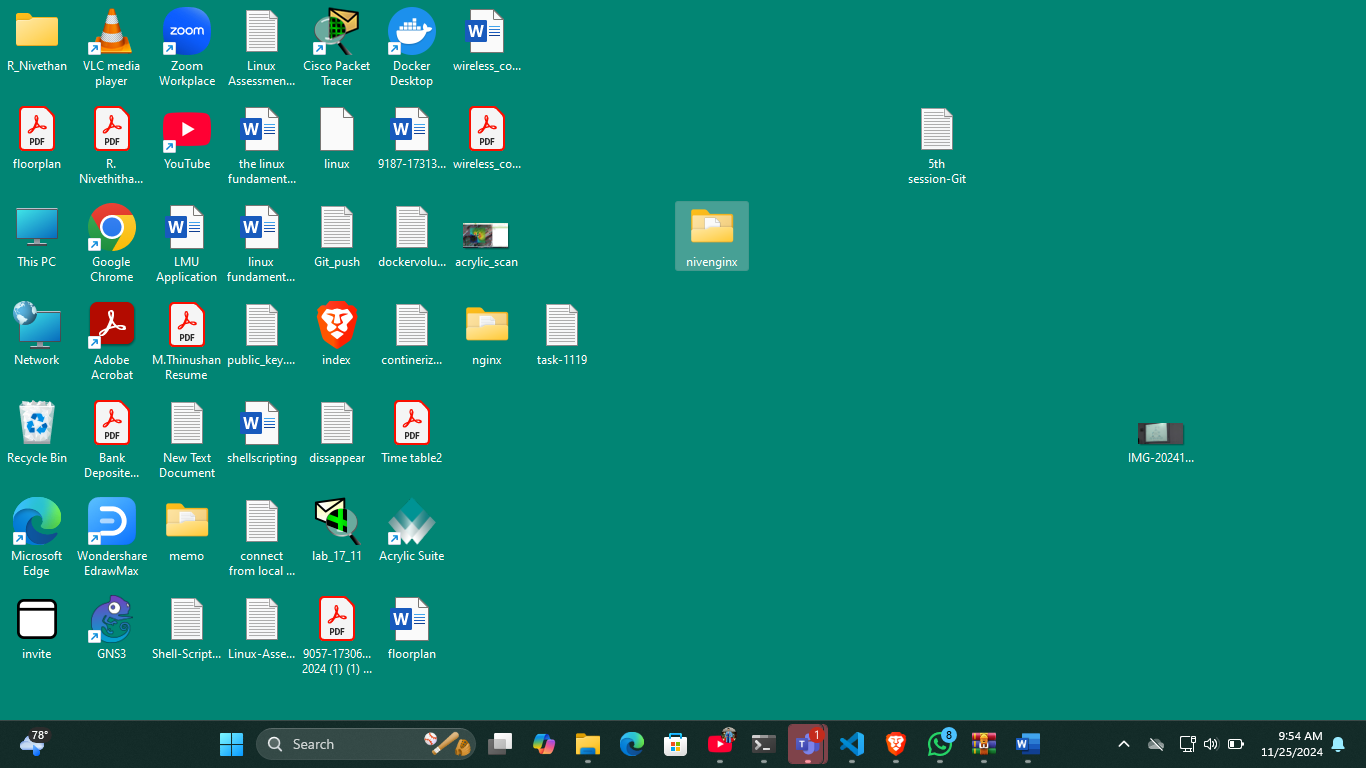


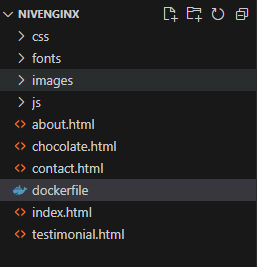
Step 13: remove the volume, command "docker volume rm <volume name>"



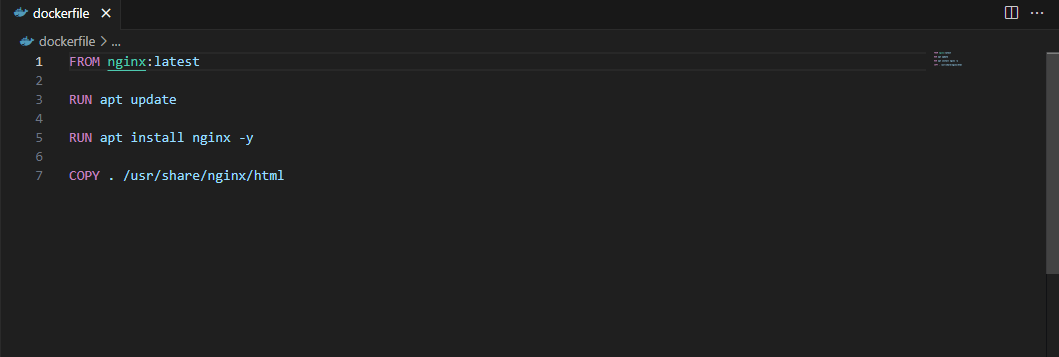
**Creating docker file**

**Step 1:** Create a folder

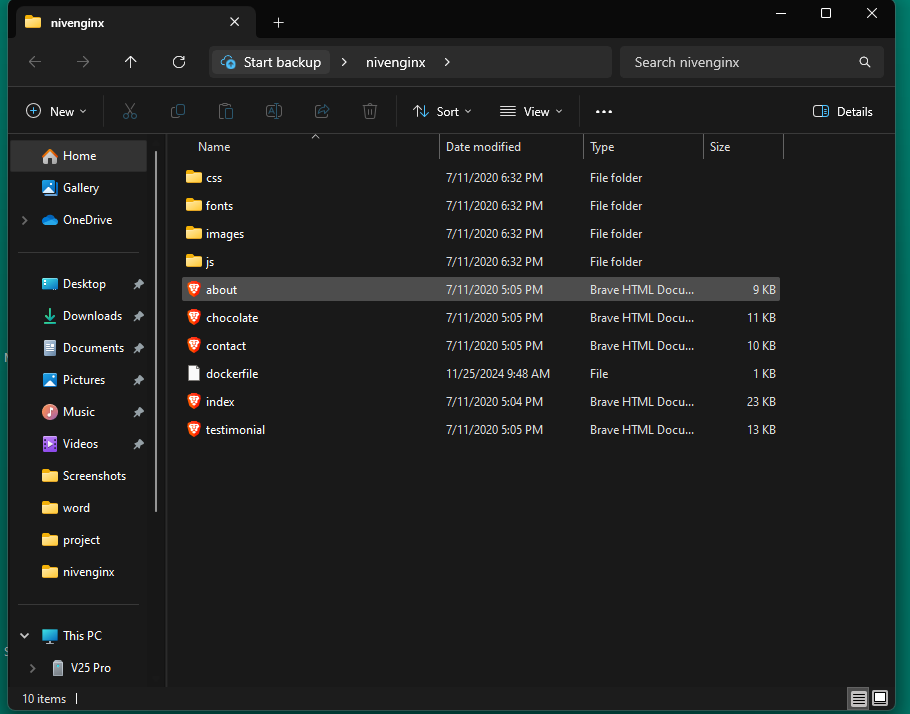
  
  
Step 2: create a docker file



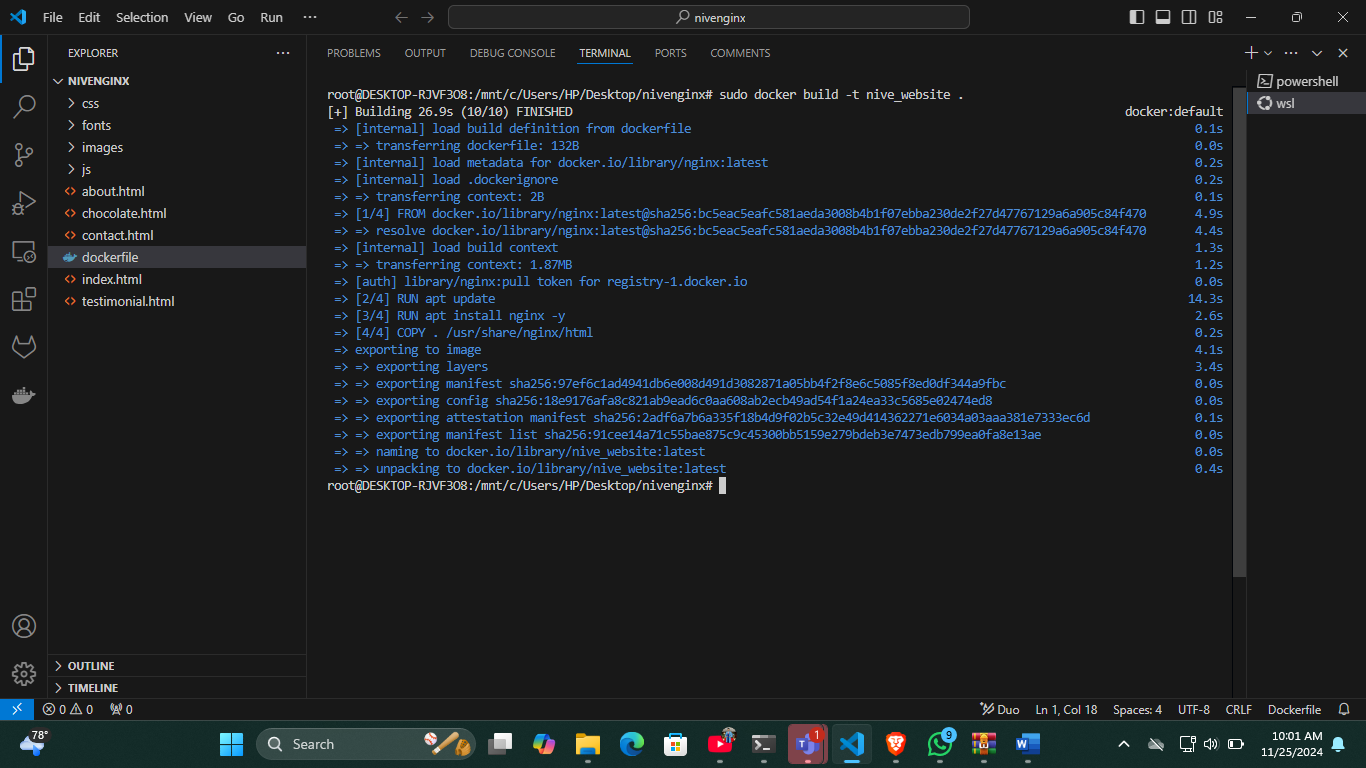
Step 3: Edit the docker file



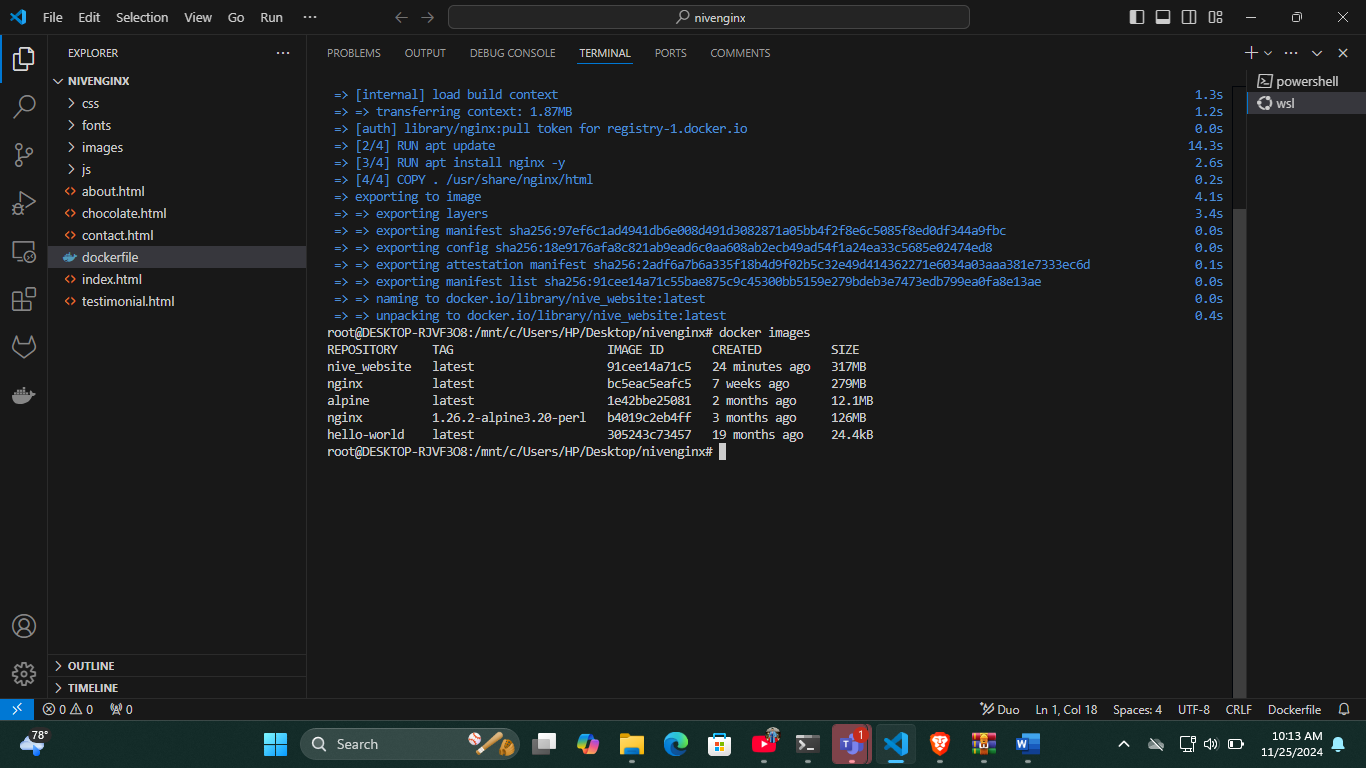
Step 4: create custom html file and save it to the folder where the docker file in.



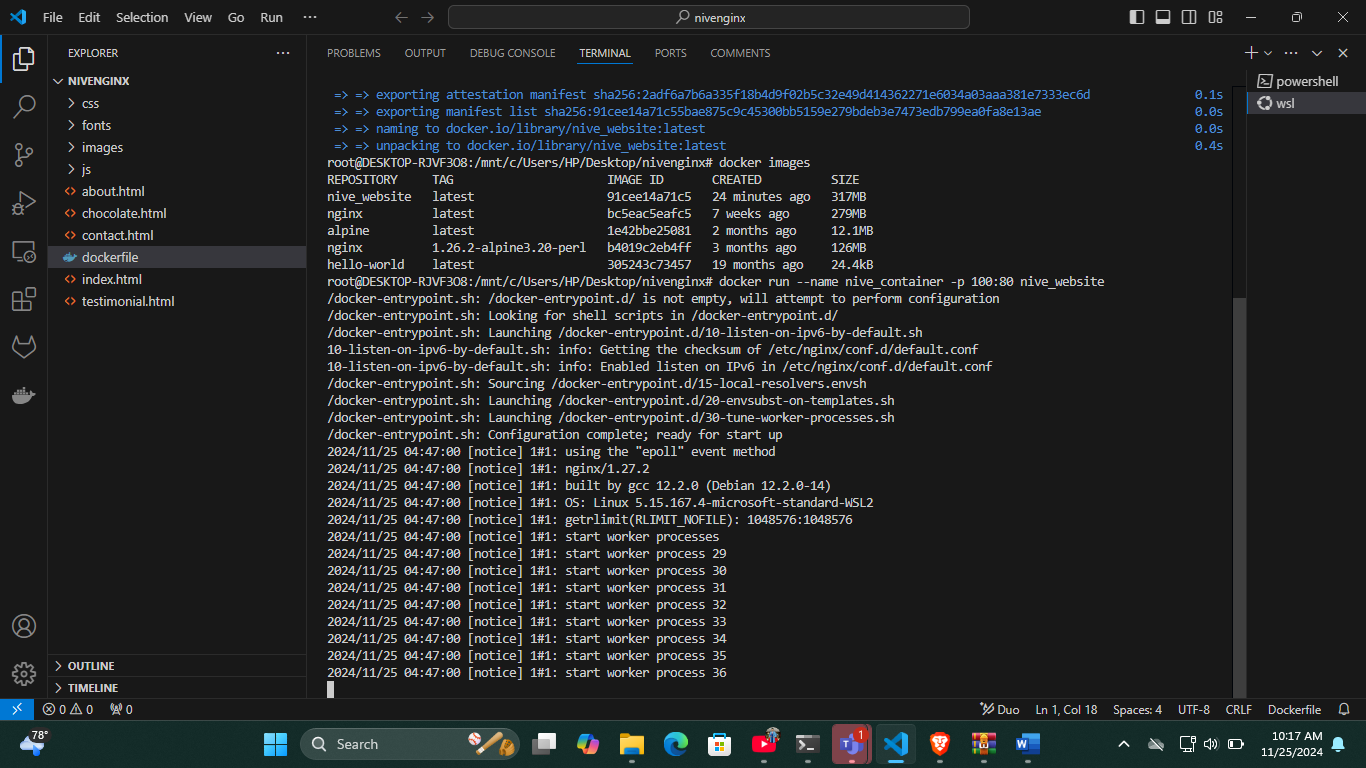
Step 5: open the terminal and run build command



Step 6: Check docker images



Step 7: run the container



Step 8: Check it in the browser

