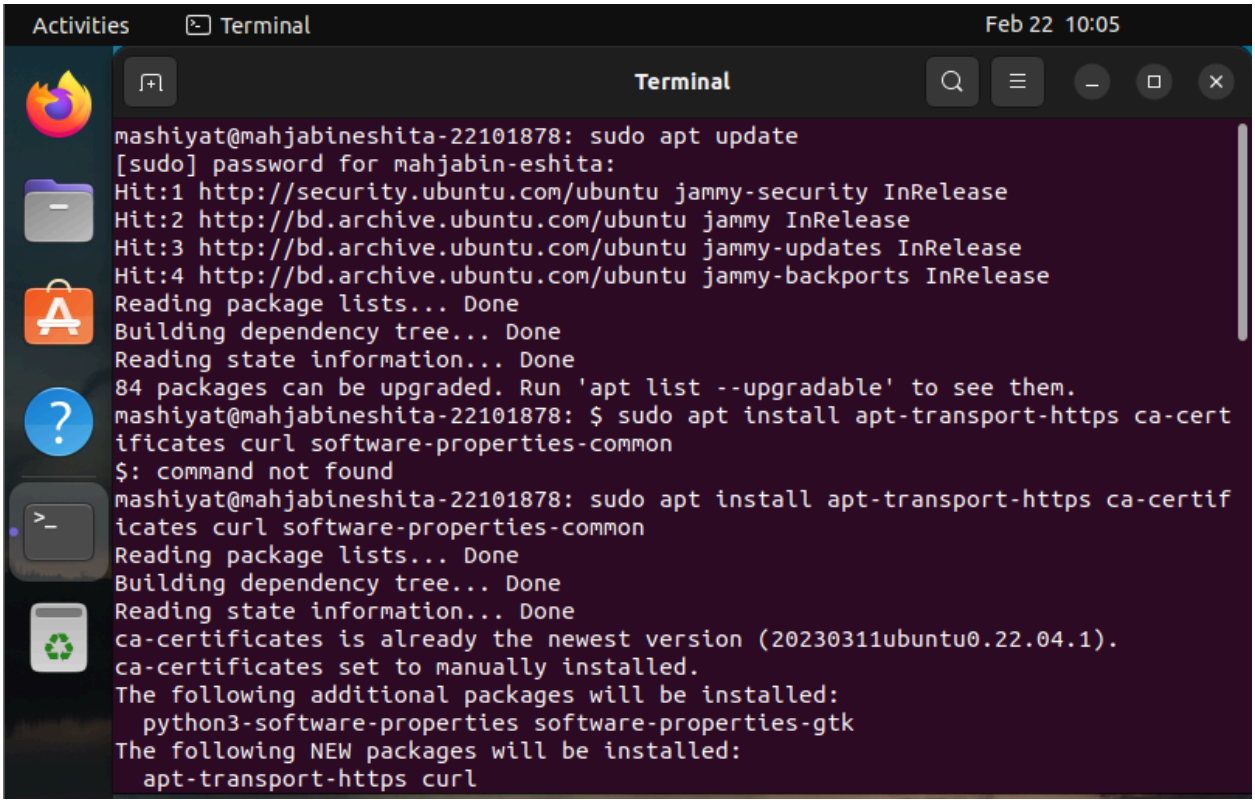
1. **Install docker**

I used **sudo apt update** for installing the updates that is a prerequisite

Then I used the command **$ sudo apt install apt-transport-https ca-certificates curl software-properties-common**



I used these commands for further process.

Then I used **curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -** for adding GPG key for official Docker repository of the system.

```
mashiyat@mahjabineshita-22101878: curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
```

Then I am adding Docker repository to APT sources using this command :

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"

I fixed some problems here that I faced then I did my further work.

```
mashiyat@mahjabineshita-22101878: sudo nano /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-jammy.list
mashiyat@mahjabineshita-22101878: sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -
cs) stable"
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu jammy stable'
Description:
Archive for codename: jammy components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Found existing deb entry in /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-jammy.list
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-jammy.list
Found existing deb-src entry in /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-jammy.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-jammy.list
Get:1 https://download.docker.com/linux/ubuntu jammy InRelease [48.8 kB]
Hit:2 http://security.ubuntu.com/ubuntu jammy-security InRelease
Get:3 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages [26.7 kB]
Hit:4 http://bd.archive.ubuntu.com/ubuntu jammy InRelease
Hit:5 http://bd.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:6 http://bd.archive.ubuntu.com/ubuntu jammy-backports InRelease
Fetched 75.5 kB in 1s (65.7 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/jammy/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg
), see the DEPRECATION section in apt-key(8) for details.
mashiyat@mahjabineshita-22101878:
```

I am confirming that I am installing from Docker repo than using Ubuntu repository using this command: **apt-cache policy docker-ce**

```
mashiyat@mahjabineshita-22101878: apt-cache policy docker-ce
docker-ce:
  Installed: (none)
  Candidate: 5:25.0.3-1~ubuntu.22.04~jammy
  Version table:
     5:25.0.3-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
     5:25.0.2-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
     5:25.0.1-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
     5:25.0.0-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
     5:24.0.9-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
     5:24.0.8-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
     5:24.0.7-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
     5:24.0.6-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
     5:24.0.5-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
     5:24.0.4-1~ubuntu.22.04~jammy 500
```

I am using **sudo apt install docker-ce** for installing docker.

```
mashiyat@mahjabineshita-22101878: sudo apt install docker-ce
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras
  docker-compose-plugin git git-man liberror-perl pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite git-daemon-run | git-daemon-sysvinit
  git-doc git-email git-gui gitk gitweb git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli
  docker-ce-rootless-extras docker-compose-plugin git git-man liberror-perl
  pigz slirp4netns
0 upgraded, 11 newly installed, 0 to remove and 81 not upgraded.
Need to get 121 MB of archives.
After this operation, 441 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 https://download.docker.com/linux/ubuntu jammy/stable amd64 containerd.io
amd64 1.6.28-1 [29.6 MB]
Get:2 http://bd.archive.ubuntu.com/ubuntu jammy/universe amd64 pigz amd64 2.6-1
[63.6 kB]
Get:3 https://download.docker.com/linux/ubuntu jammy/stable amd64 docker-buildx-
```

Now I am logging in to my docker using **sudo docker login**

```
mashiyat@mahjabineshita-22101878: sudo docker login
Log in with your Docker ID or email address to push and pull images from Docker
Hub. If you don't have a Docker ID, head over to https://hub.docker.com/ to crea
te one.
You can log in with your password or a Personal Access Token (PAT). Using a limi
ted-scope PAT grants better security and is required for organizations using SSO
. Learn more at https://docs.docker.com/go/access-tokens/

Username:
```

2. Show outputs of basic Docker commands

I created a docker Id for this assignment then I used the further commands

```
Username: jadedmashiyat24
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
mashiyat@mahjabineshita-22101878:
```

Using this command: sudo docker pull hello-world

Here, docker pull hello-world will download the hello-world image with the latest tag from

Docker Hub to your local system.

```
mashiyat@mahjabineshita-22101878: sudo docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:d000bc569937abbe195e20322a0bde6b2922d805332fd6d8a68b19f524b7d21d
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
mashiyat@mahjabineshita-22101878:
```

Then used **sudo docker images** that will help me to know all the images that I have currently

```
mashiyat@mahjabineshita-22101878: sudo docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hello-world   latest    d2c94e258dcb   9 months ago   13.3kB
mashiyat@mahjabineshita-22101878:
```

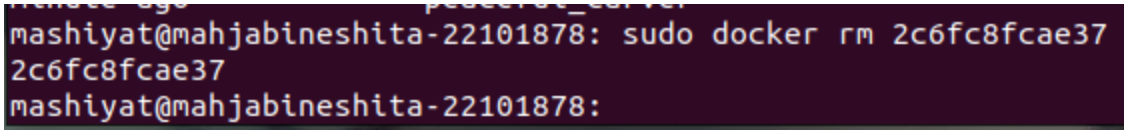
Then I used sudo docker run hello-world

```
mashiyat@mahjabineshita-22101878: sudo docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

mashiyat@mahjabineshita-22101878:
```

The sudo docker ps -a this command will show all the containers both stopped and running

```
mashiyat@mahjabineshita-22101878: sudo docker ps -a
CONTAINER ID   IMAGE         COMMAND    CREATED              STATUS
            PORTS     NAMES
2c6fc8fcae37   hello-world   "/hello"   About a minute ago   Exited (0) About a
minute ago             peaceful_carver
mashiyat@mahjabineshita-22101878:
```

Removing the docker container using, sudo docker rm docker id



3. Create a Docker image using Dockerfile:

Here, I have used

mkdir docker

ls

/home/mahjabin-eshita/docker/

ls

Touch Dockerfile

sudo gedit Dockerfile

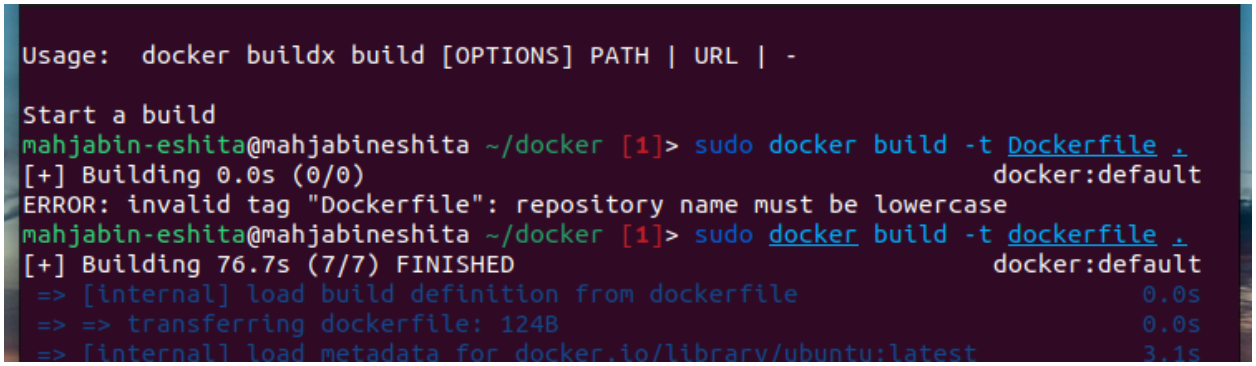
sudo docker build -t Dockerfile [got error here]

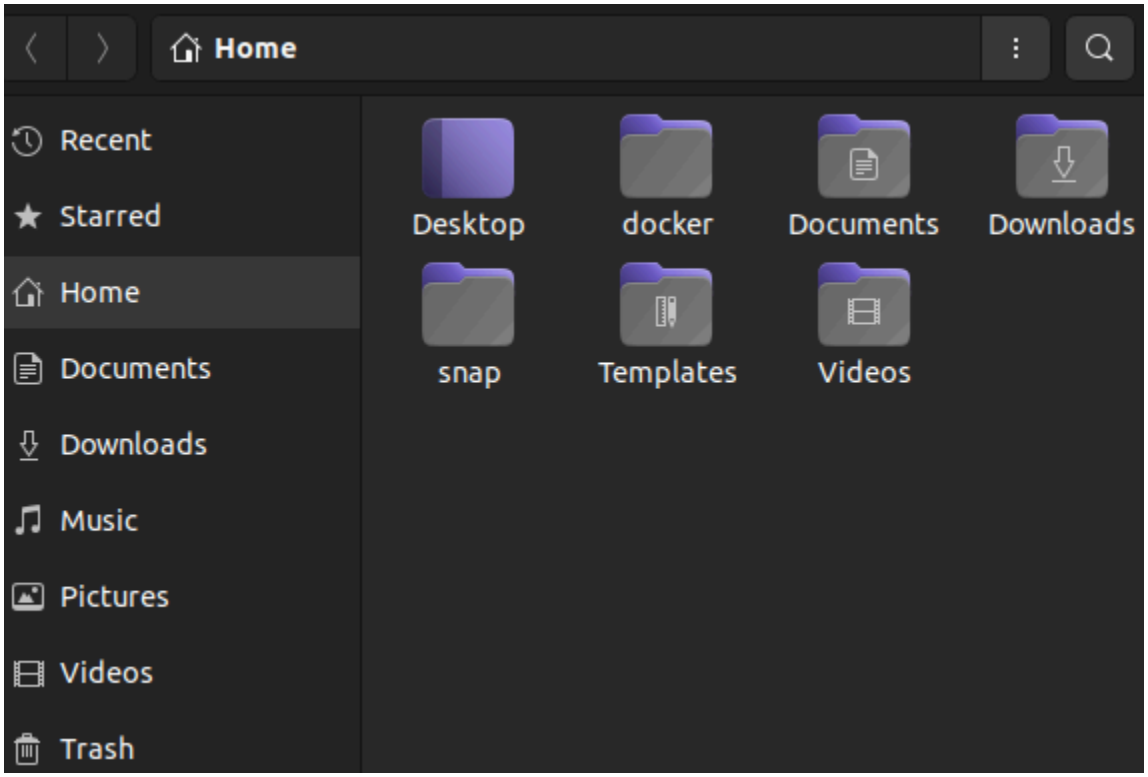
sudo docker build -t dockerfile .

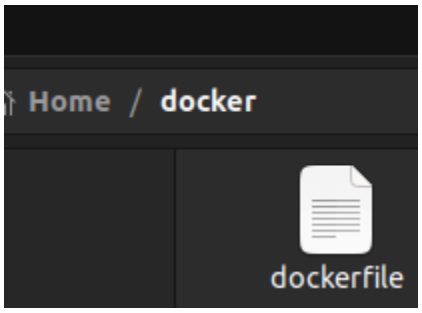
Then the image got created on the file

```
FROM ubuntu:latest
RUN apt-get update && apt-get install -y
software-properties-common
```

```
mahjabin-eshita@mahjabineshita ~> /home/mahjabin-eshita/docker/
mahjabin-eshita@mahjabineshita ~/docker> ls
mahjabin-eshita@mahjabineshita ~/docker> touch Dockerfile
mahjabin-eshita@mahjabineshita ~/docker> sudo gedit Dockerfile

(gedit:5068): dconf-WARNING **: 08:29:12.948: failed to commit changes to dconf:
Failed to execute child process “dbus-launch” (No such file or directory)

(gedit:5068): dconf-WARNING **: 08:29:12.950: failed to commit changes to dconf:
Failed to execute child process “dbus-launch” (No such file or directory)
```







4. Run a container as a single task, show outputs, and show the status of all containers (using docker ps -a)

Used command:

sudo docker build -t hello-eshita .

sudo docker run hello-eshita

sudo docker ps-a

```
mahjabin-eshita@mahjabineshita ~/docker [1]> sudo docker build -t hello-eshita .
[+] Building 0.1s (6/6) FINISHED                                  docker:default
 => [internal] load build definition from dockerfile                       0.0s
 => => transferring dockerfile: 124B                                       0.0s
 => [internal] load metadata for docker.io/library/ubuntu:latest           0.0s
 => [internal] load .dockerignore                                          0.0s
 => => transferring context: 2B                                            0.0s
 => [1/2] FROM docker.io/library/ubuntu:latest                             0.0s
 => CACHED [2/2] RUN apt-get update && apt-get install -y software-proper  0.0s
 => exporting to image                                                     0.0s
 => => exporting layers                                                    0.0s
 => => writing image sha256:d3fb283f462a1c8ccc86656006d4825a276a04498cfb3  0.0s
 => => naming to docker.io/library/hello-eshita                            0.0s
mahjabin-eshita@mahjabineshita ~/docker> sudo docker run hello-eshita
mahjabin-eshita@mahjabineshita ~/docker> sudo docker ps -a
CONTAINER ID   IMAGE          COMMAND       CREATED          STATUS
          PORTS     NAMES
Trash  4eef1d8f   hello-eshita   "/bin/bash"   31 seconds ago   Exited (0) 31 secon
ds ago             great_saha
385f203f16a2   dockerfile     "/bin/bash"   4 minutes ago    Exited (0) 4 minute
s ago              brave_cerf
68aa34e03fa5   dockerfile     "/bin/bash"   6 minutes ago    Exited (0) 6 minute
s ago              inspiring_cartwright
mahjabin-eshita@mahjabineshita ~/docker>
```

5. Run a container in iterative mode and install different packages in the container. Show each step.

Command:

sudo docker run -it hello-eshita /bin/bash

```
mahjabin-eshita@mahjabineshita ~/docker> docker run -it hello-eshita /bin/bash
docker: permission denied while trying to connect to the Docker daemon socket at
 unix:///var/run/docker.sock: Post "http://%2Fvar%2Frun%2Fdocker.sock/v1.24/cont
ainers/create": dial unix /var/run/docker.sock: connect: permission denied.
See 'docker run --help'.
mahjabin-eshita@mahjabineshita ~/docker [126]> sudo docker run -it hello-eshita
/bin/bash
root@655b1f096c41:/#
```

Here, I have installed **Spotify** on the container

```
root@655b1f096c41:/# apt-get install spotify
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package spotify
```

6. Run a database container in the background. Then show logs of the running container. After, access the container in interactive mode. show some SQL queries inside the container.

I used the command below for running the MySQL container in the background:

sudo docker run -d –name some-mysql -e MYSQL_ROOT_PASSWORD=my-secret-pw mysql:latest

```
mashiyat@mahjabineshita-22101878: sudo docker run -d --name some-mysql -e MYSQL_
ROOT_PASSWORD=my-secret-pw mysql:latest
[sudo] password for mahjabin-eshita:
Unable to find image 'mysql:latest' locally
latest: Pulling from library/mysql
81badc5f380f: Pull complete
c490e5dd1a9d: Pull complete
87aeb61f1478: Pull complete
1cacbea6ceda: Pull complete
1e72891ace67: Pull complete
42b720363d36: Pull complete
6b3b50f9990a: Pull complete
3811d52cfa61: Pull complete
05bc7a0277d8: Pull complete
cc0abd25a274: Pull complete
Digest: sha256:ff5ab9cdce0b4c59704b4e2a09deed5ab8467be795e0ea20228b8528f53fcf82
Status: Downloaded newer image for mysql:latest
cdef7d03a7a9fbda6be551a682e41c8fb6e220402ac8aee448b595f2465df31b
mashiyat@mahjabineshita-22101878:
```

Then I used **sudo docker logs cdef7d03a7a9** to see the logs of the running container

docker exec -it cdef7d03a7a9 bash

mysql -u root -p -e 'SHOW DATABASES;'

```
bash-4.4# my secret pw
bash: my: command not found
                -u root -p -e 'SHOW DATABASES;'
ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: Y
ES)
bash-4.4# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 8.3.0 MySQL Community Server - GPL

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SHOW satabases;
```

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.00 sec)

mysql>
```

```
mysql> CREATE table cloud;
ERROR 1046 (3D000): No database selected
mysql> CREATE database cloud;
Query OK, 1 row affected (0.04 sec)

mysql> CREATE table cloud;
ERROR 1046 (3D000): No database selected
mysql> SHOW databases;
+--------------------+
| Database           |
+--------------------+
| cloud              |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql>
```

Creating a new database named cloud

I created a table to show that MySQL works

```
mysql> INSERT INTO clouds (ID, Number, Name)
    -> VALUES
    -> (221018, 01, 'LUNA LOVEGOOD'),
    -> (223012, 02, 'PIKACHU'),
    -> (782962, 03, 'TOM');
Query OK, 3 rows affected (0.04 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM clouds;
+--------+--------+---------------+
| ID     | Number | Name          |
+--------+--------+---------------+
| 221018 |      1 | LUNA LOVEGOOD |
| 223012 |      2 | PIKACHU       |
| 782962 |      3 | TOM           |
+--------+--------+---------------+
```

7. Push your own image into the Docker public registry/Hub.

I logged in to my docker for the task

I Used:

sudo dockerlogin

/home/mahjabin-eshita//docker/ : my directory path which represents location

Docker build -t eshita . : This command is used to build a Docker image from a Dockerfile and a "context". The build process can refer to any of the files in the context.

sudo docker run -p 4000:80 eshita : I am using eshita image to run the docker, the -p flag I am using to map a network port inside the container (80) to a port on the host machine (4000).

sudo docker push jadedmashiyat24/eshita:latest : I am using this command to push the Docker image named eshita with the latest tag to the Docker Hub registry under the jadedmashiyat24 account.

sudo docker tag eshita jadedmashiyat24/eshita:latest

I am using this command to tag the eshita image with a new tag (jadedmashiyat24/eshita:latest).

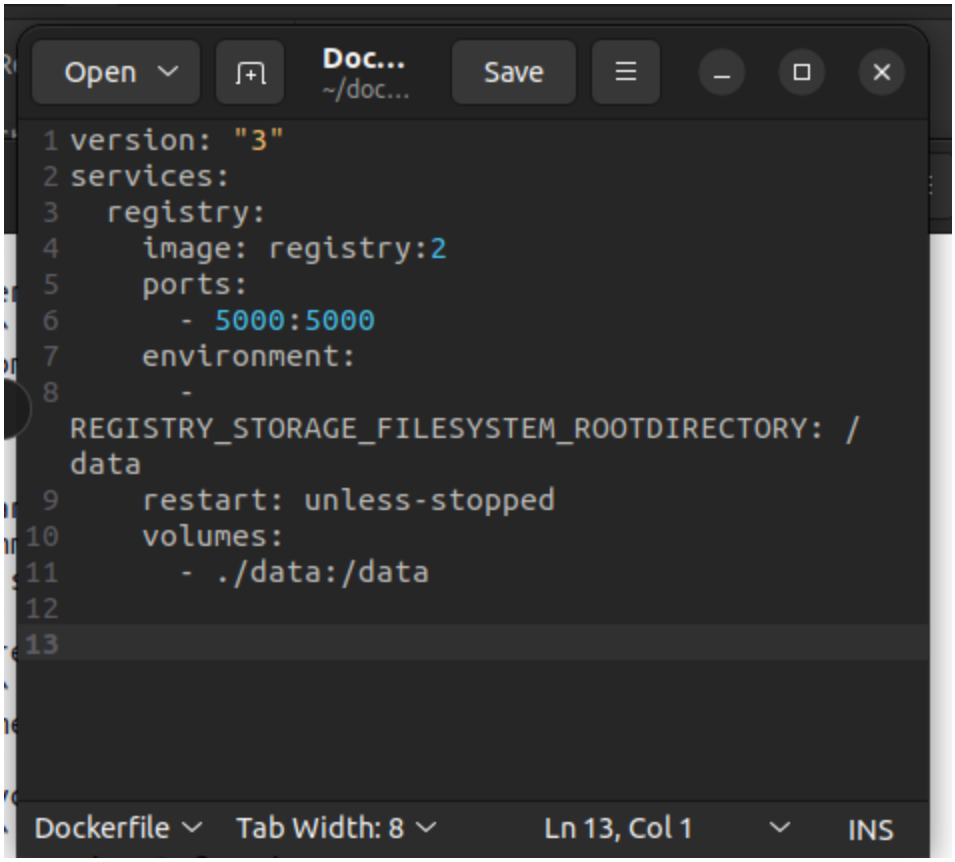
sudo docker push jadedmashiyat24/eshita:latest

I am using this command to push the Docker image with the new tag to the Docker Hub registry.

```
mashiyat@mahjabineshita-22101878: sudo docker build -t eshita .
[+] Building 0.0s (1/1) FINISHED                          docker:default
 => [internal] load build definition from Dockerfile                0.0s
 => => transferring dockerfile: 2B                                  0.0s
ERROR: failed to solve: failed to read dockerfile: open Dockerfile: no such file or directory
mashiyat@mahjabineshita-22101878: fish
Welcome to fish, the friendly interactive shell
Type help for instructions on how to use fish
mahjabin-eshita@mahjabineshita ~> /home/mahjabin-eshita//docker/
mahjabin-eshita@mahjabineshita ~/docker> docker build -t eshita .
ERROR: permission denied while trying to connect to the Docker daemon socket at unix:///var/ru
n%2Fdocker.sock/_ping": dial unix /var/run/docker.sock: connect: permission denied
mahjabin-eshita@mahjabineshita ~/docker [1]> sudo docker build -t eshita .
[+] Building 0.1s (6/6) FINISHED                          docker:default
 => [internal] load build definition from Dockerfile                0.0s
 => => transferring dockerfile: 124B                                0.0s
 => [internal] load metadata for docker.io/library/ubuntu:latest    0.0s
 => [internal] load .dockerignore                                   0.0s
 => => transferring context: 2B                                     0.0s
 => [1/2] FROM docker.io/library/ubuntu:latest                      0.0s
 => CACHED [2/2] RUN apt-get update && apt-get install -y softwa    0.0s
 => exporting to image                                              0.0s
 => => exporting layers                                             0.0s
 => => writing image sha256:d3fb283f462a1c8ccc86656006d4825a276a    0.0s
 => => naming to docker.io/library/eshita                           0.0s
mahjabin-eshita@mahjabineshita ~/docker> sudo docker run -p 4000:80 eshita
mahjabin-eshita@mahjabineshita ~/docker> sudo docker push jadedmashiyat24/eshita:latest
The push refers to repository [docker.io/jadedmashiyat24/eshita]
An image does not exist locally with the tag: jadedmashiyat24/eshita
mahjabin-eshita@mahjabineshita ~/docker [1]> sudo docker tag eshita jadedmashiyat24/eshita:lat
mahjabin-eshita@mahjabineshita ~/docker> sudo docker push jadedmashiyat24/eshita:latest
The push refers to repository [docker.io/jadedmashiyat24/eshita]
631cc566ce3d: Pushed
d101c9453715: Mounted from library/ubuntu
latest: digest: sha256:97d4dc4db7d05954e092e56d6425800fe9544422664047b9af239774aae7aab5 size:
mahjabin-eshita@mahjabineshita ~/docker>
```

8. How to make your own private registry? Show steps.

Here, I created the docker-compose file. Where I wrote the lines in the files first.



I used to pull version 2 of the Docker registry image from Docker Hub by using this command:
docker pull registry:2

```
mahjabin-eshita@mahjabineshita ~/docker> sudo docker pull registry:2
2: Pulling from library/registry
Digest: sha256:f4e1b878d4bc40a1f65532d68c94dcfbab56aa8cba1f00e355a206e7f6cc9111
Status: Downloaded newer image for registry:2
docker.io/library/registry:2
mahjabin-eshita@mahjabineshita ~/docker>
```

I used this command to run the registry command:
docker run -d -p 5000:5000 --restart=always --name myregistry registry:2
Before pushing the image I needed to tag it with the address of my registry using the command:
docker tag myimage localhost:5000/myimage

```
mahjabin-eshita@mahjabineshita ~/docker> sudo docker run -d -p 5000:500 --restart=always --name myregistry registry:2
1c0186644290c719166261e440f73aae8a8b5e8ae4ab6fad081f411a7143c944
mahjabin-eshita@mahjabineshita ~/docker> docker tag ubuntu localhost:5000/ubuntu
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post "http://%2Fvar%2Frun%2Fdo
cker.sock/v1.24/images/ubuntu/tag?repo=localhost%3A5000%2Fubuntu&tag=latest": dial unix /var/run/docker.sock: connect: permission de
nied
mahjabin-eshita@mahjabineshita ~/docker [1]> sudo docker tag ubuntu localhost:5000/ubuntu
mahjabin-eshita@mahjabineshita ~/docker> sudo docker push localhost:5000/ubuntu
Using default tag: latest
The push refers to repository [localhost:5000/ubuntu]
Get "http://localhost:5000/v2/": read tcp 127.0.0.1:37872->127.0.0.1:5000: read: connection reset by peer
mahjabin-eshita@mahjabineshita ~/docker [1]>
```

9. Create a small website or app with minimal functionality (Could be a simple HTML website that has a button that opens a static image/file) inside a Docker container. Then run the application (inside the container) in the background of your HOST machine in any port. Browse the website from your host machine.

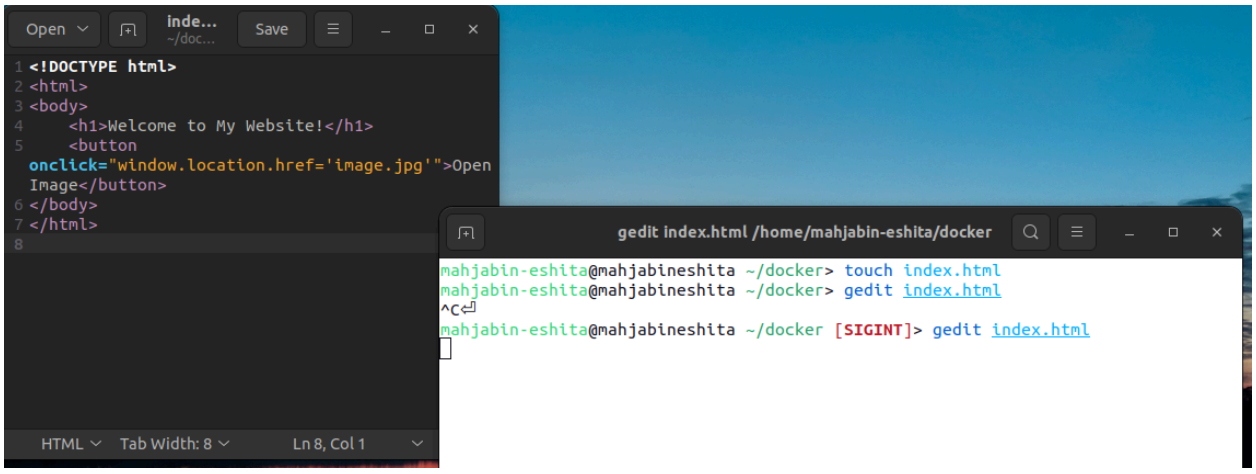
Creating an HTML file

<! DOCTYPE html>
<html>
<body>
<h1>Welcome to My Website!</h1>
<button>
onclick = "window.location.herf= 'image.jpg'" > Open Image</button>
</body>
</html>

Then I used **index.html**

Then used

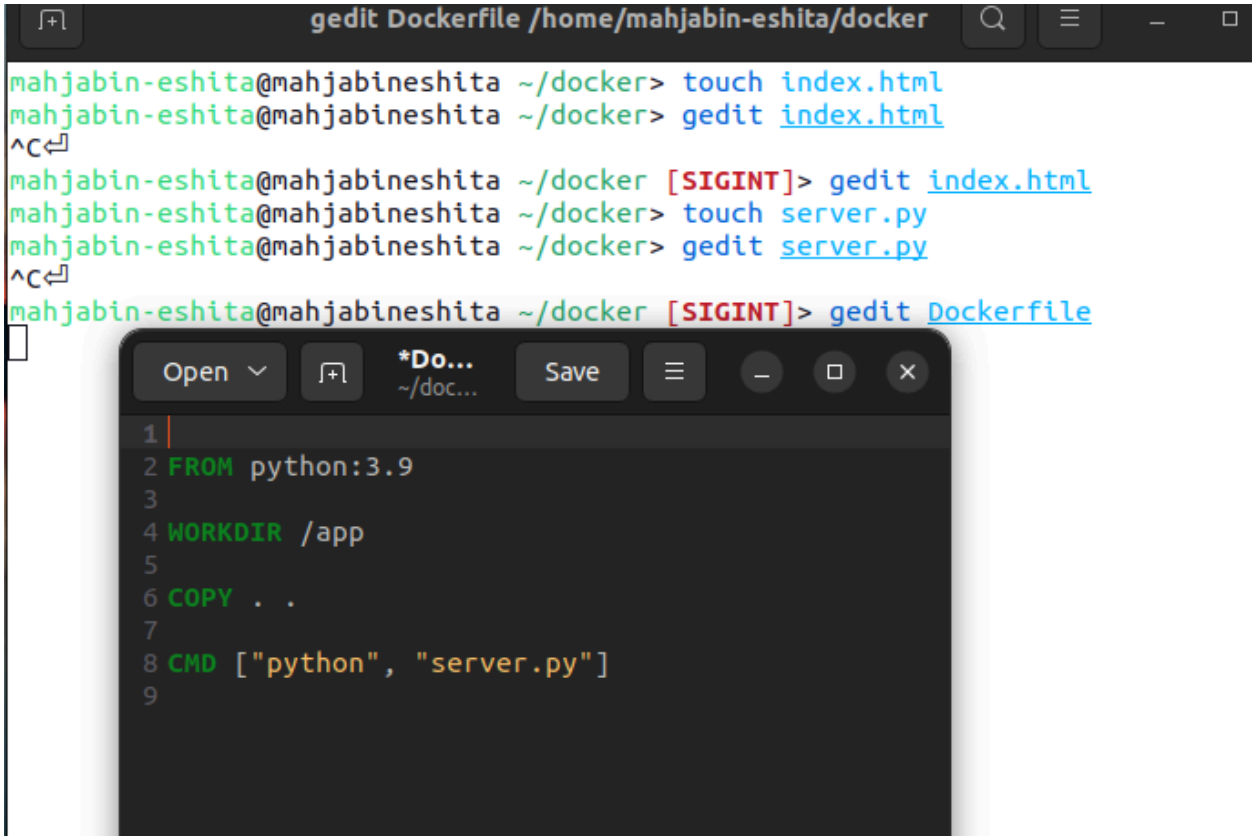
touch index.html
gedit index.html



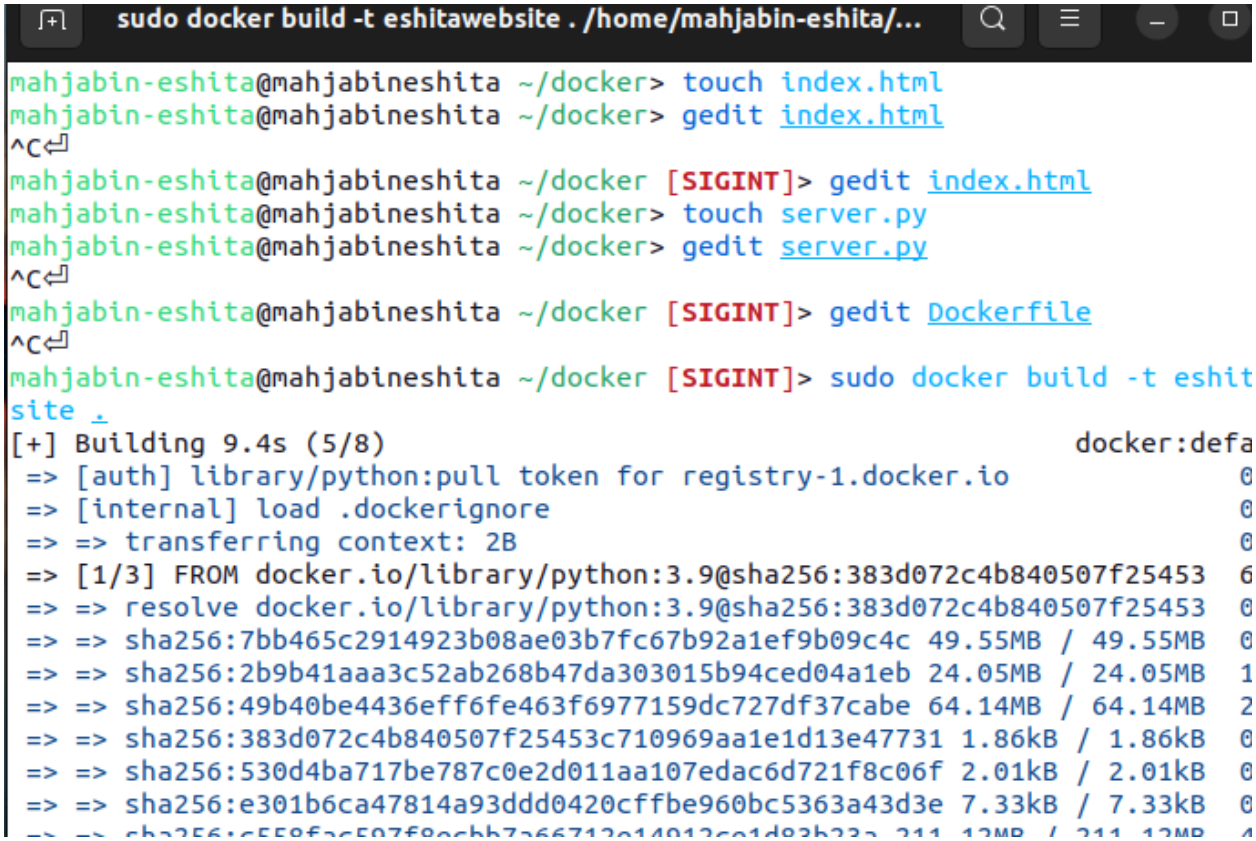
Creating a python server

Editing the docker file

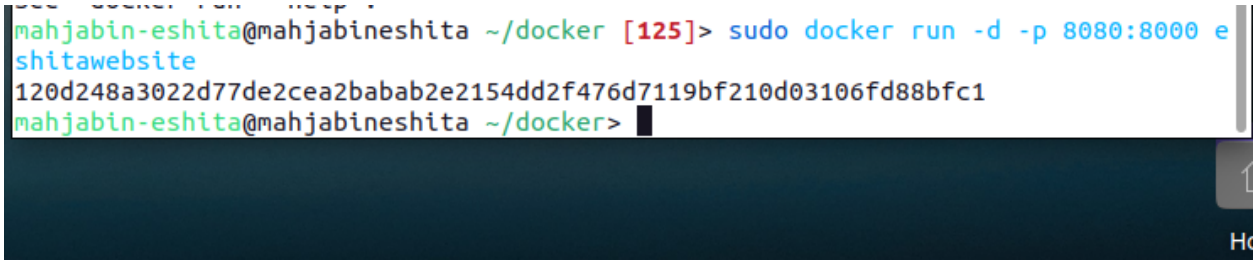
FROM python:3.9
WORKDIR /app
COPY . .
CMD [“python”, “server.py”]


gedit Dockerfile /home/mahjabin-eshita/docker
mahjabin-eshita@mahjabineshita ~/docker> touch index.html
mahjabin-eshita@mahjabineshita ~/docker> gedit index.html
^C⏎
mahjabin-eshita@mahjabineshita ~/docker [SIGINT]> gedit index.html
mahjabin-eshita@mahjabineshita ~/docker> touch server.py
mahjabin-eshita@mahjabineshita ~/docker> gedit server.py
^C⏎
mahjabin-eshita@mahjabineshita ~/docker [SIGINT]> gedit Dockerfile
Open
*Do...
~/doc...
Save
1
2 FROM python:3.9
3
4 WORKDIR /app
5
6 COPY . .
7
8 CMD ["python", "server.py"]
9

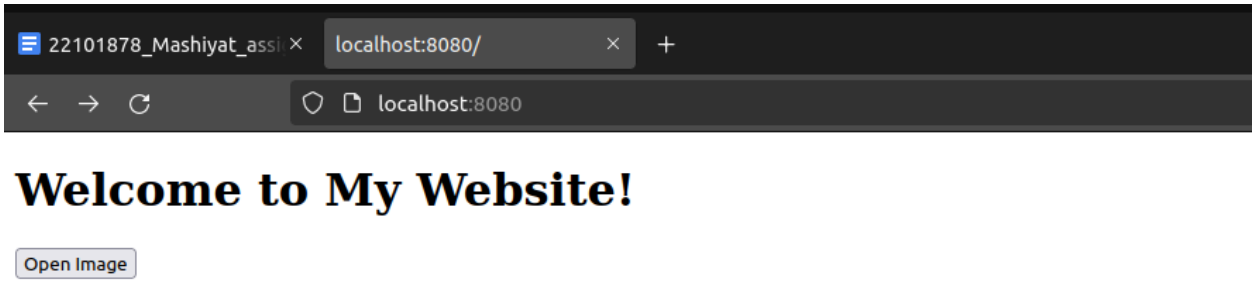
Building the docker image **using -sudo docker buid -t eshitawebsite .**



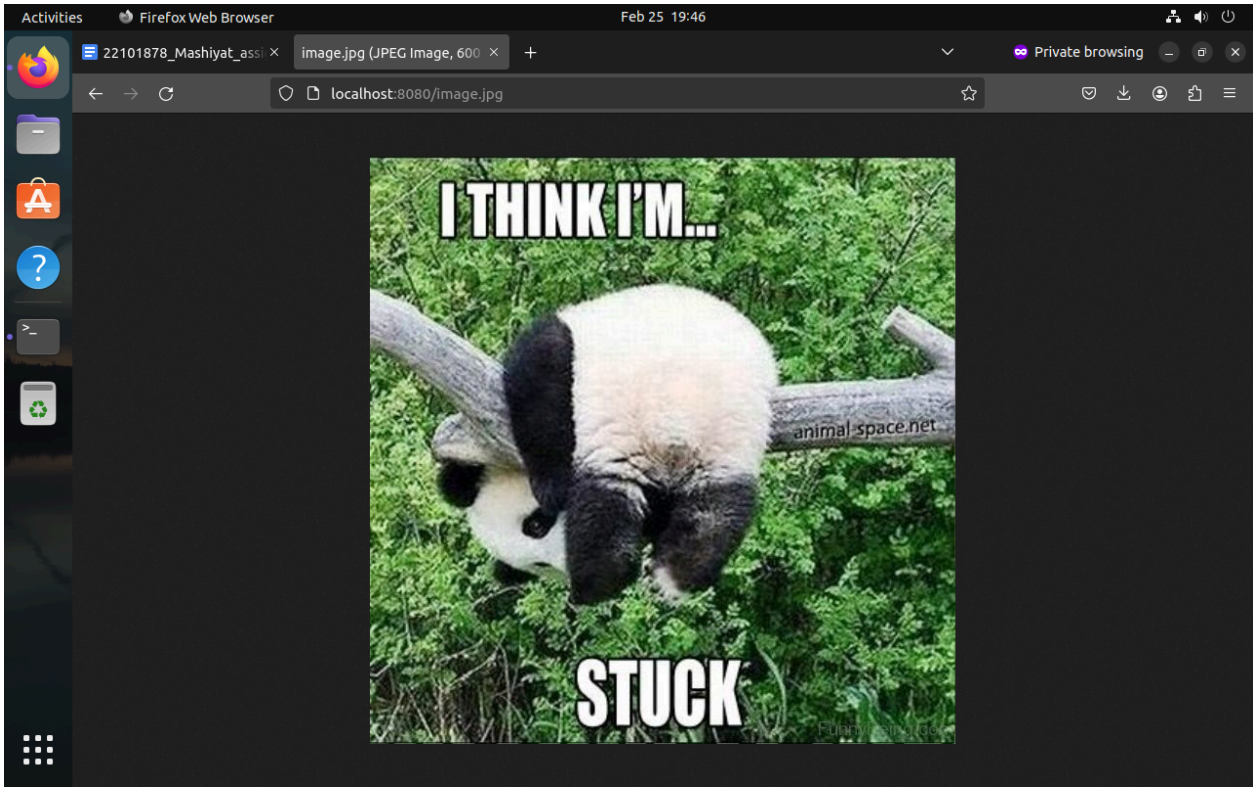
Running the docker container using **sudo docker run -d -p 8080:8000 eshitawebsite**



Running the website. I connected my local host and here, you can see the details of my website. Then I got my static website.



Here is the image.



10. Migrate the new container having the application into another machine. Again run the container and browse the URL. It should work.

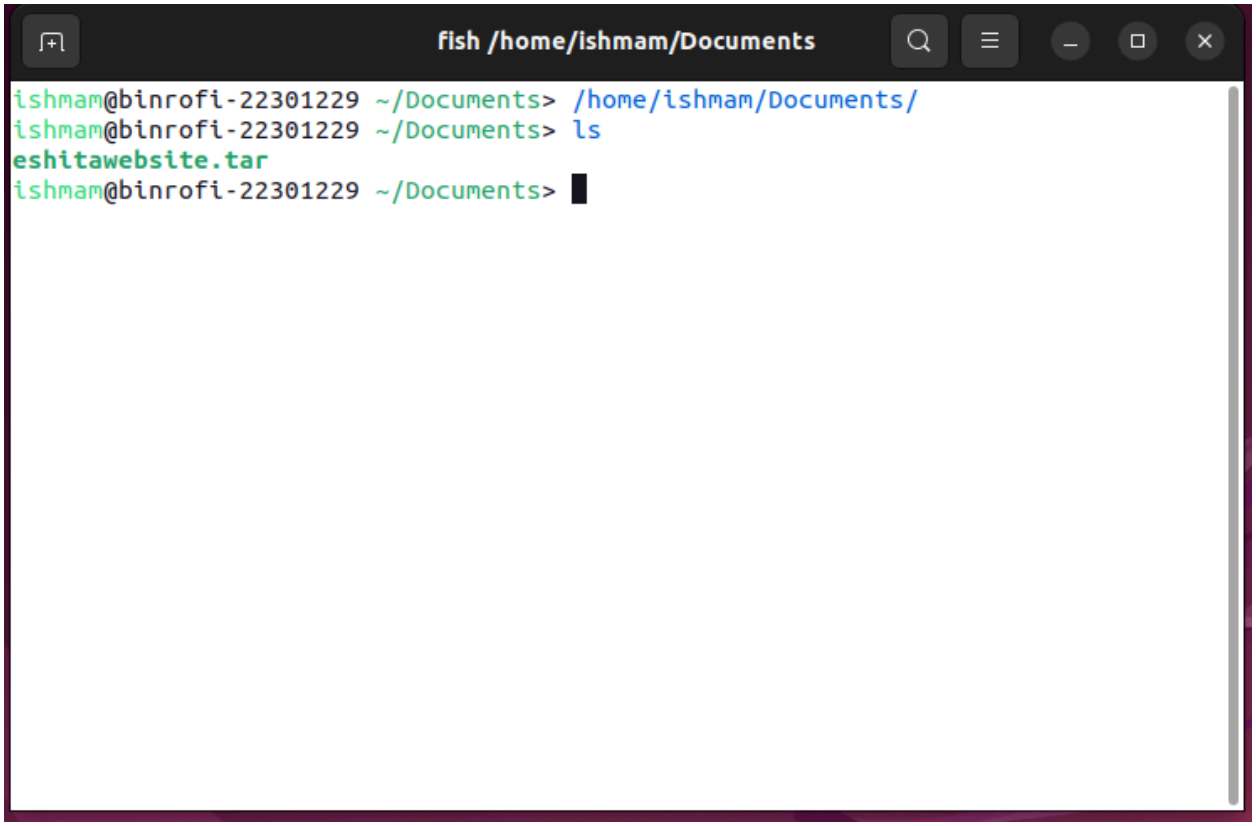
Saving the docker image as tar file and giving it permission to access the file

sudo docker save -o eshitawebsite.tar.eshitawebsite
ls
Dockerfile eshitawebsite.tar image.jpg index.htmml server.py
Sudo chmod 777 eshitawebsite.tar
ls-l

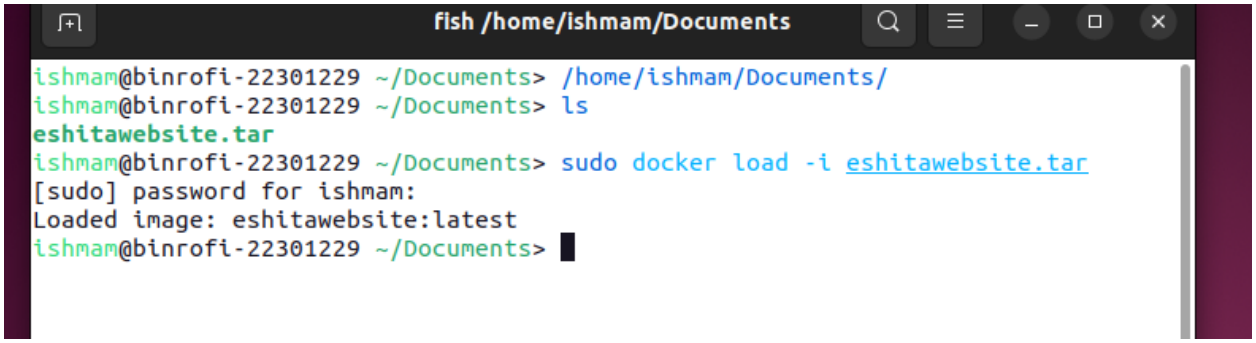
Made the permission

```
mahjabin-eshita@mahjabineshita ~/docker> sudo docker save -o eshitawebsite.tar e
shitawebsite
mahjabin-eshita@mahjabineshita ~/docker> ls
Dockerfile  eshitawebsite.tar  image.jpg  index.html  server.py
mahjabin-eshita@mahjabineshita ~/docker> fish
Welcome to fish, the friendly interactive shell
Type help for instructions on how to use fish
mahjabin-eshita@mahjabineshita ~/docker> sudo chmod 777  eshitawebsite.tar
mahjabin-eshita@mahjabineshita ~/docker> ls -l
total 995692
-rw-rw-r-- 1 mahjabin-eshita mahjabin-eshita         71 Feb 25 19:43 Dockerfile
-rwxrwxrwx 1 root            root            1019479040 Feb 25 19:47 eshitawebsi
te.tar
-rw-rw-r-- 1 mahjabin-eshita mahjabin-eshita      89448 Feb 25 19:39 image.jpg
-rw-rw-r-- 1 mahjabin-eshita mahjabin-eshita        158 Feb 25 19:38 index.html
-rw-rw-r-- 1 mahjabin-eshita mahjabin-eshita        222 Feb 25 19:41 server.py
mahjabin-eshita@mahjabineshita ~/docker>
```

Now moving the file to another device
I will be using my friend Ishmam's pc.



Load the docker on his pc



Running the docker

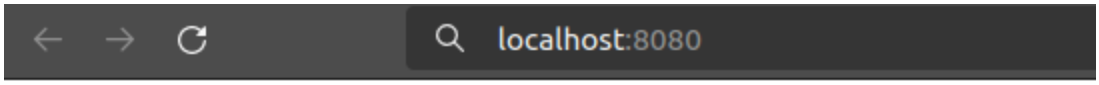
sudo docker load -i eshitawebsite.tar

Then I ran the docker file using

sudo docker run -d -p 8080:80 eshitawebsite

```
ishmam@binrofi-22301229 ~/Documents [125]> sudo docker run -d -p 8080:80 eshitawebsite
1a1ad5ece1557f04f2393c674b354b178631f1e764d64fbc735beb17088ed4a8
ishmam@binrofi-22301229 ~/Documents>
```

It is now running on his pc too



Welcome to My Website!

Open Image