Lab 0

1. How do you run a Node.js application using Docker, using the official Node.js image?

First create a "application.js" file in the current directory, then pull the official Nodejs image using this command:

```
Docker image pull node:latest
```

2. What command would you use to mount a local directory into a Node.js Docker container?

To map current directory"/" to "/app" directory in container, I would use:

```
Docker run -v /:/app -d node
```

3. What is the command to copy files from your local machine into a running Python Docker container and vice versa?

First run a python container:

```
Docker container run -it python:latest
```

Then copy content:

```
Docker cp /myLocalFolder container_id:/myPythonFolder


Docker cp container_id:/myPythonFolder /myLocalFolder
```

4. How can you execute a Python script inside a running Docker container?

```
docker exec <container_id> python /path/to/script.py
```

5. How do you start a Nginx Docker container and expose it on port 80?

First create the container then:

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

6. What command can you use to inspect the Nginx container's logs?

```
Docker logs container_name
```

7. How can you list all Docker images on your system?

```
Docker images -a
```

8. What is the purpose of the docker ps command, and how can you see all containers, including stopped ones?

The docker ps command lists all running containers, and you can include -a to see all containers including stopped ones.

9. How can you stop a running Docker container gracefully?

Using `docker stop container_id`

10. What command would you use to remove all stopped containers from your system?

Use `docker container prune`

11. How can you view detailed information about a Docker image, including its layers?

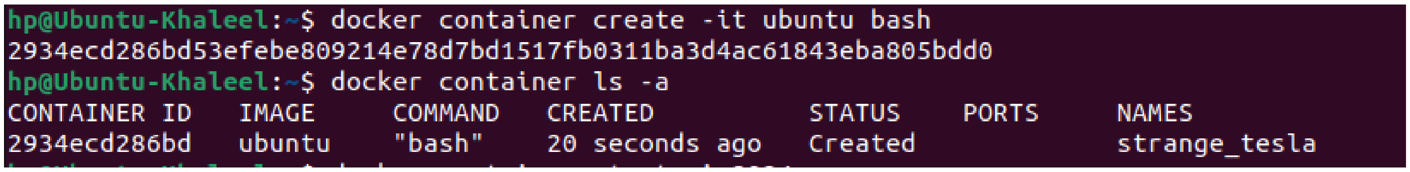
Using `docker inspect image_id`

12. What is the difference between a Docker image and a Docker container, and how do you create a container from an image?

An image is like the template we use to build a container which is the actual thing that holds a running program and isolates it from the host machine.

To create a container from an image of ubuntu for example you can use:

```

Docker container create ubuntu

```



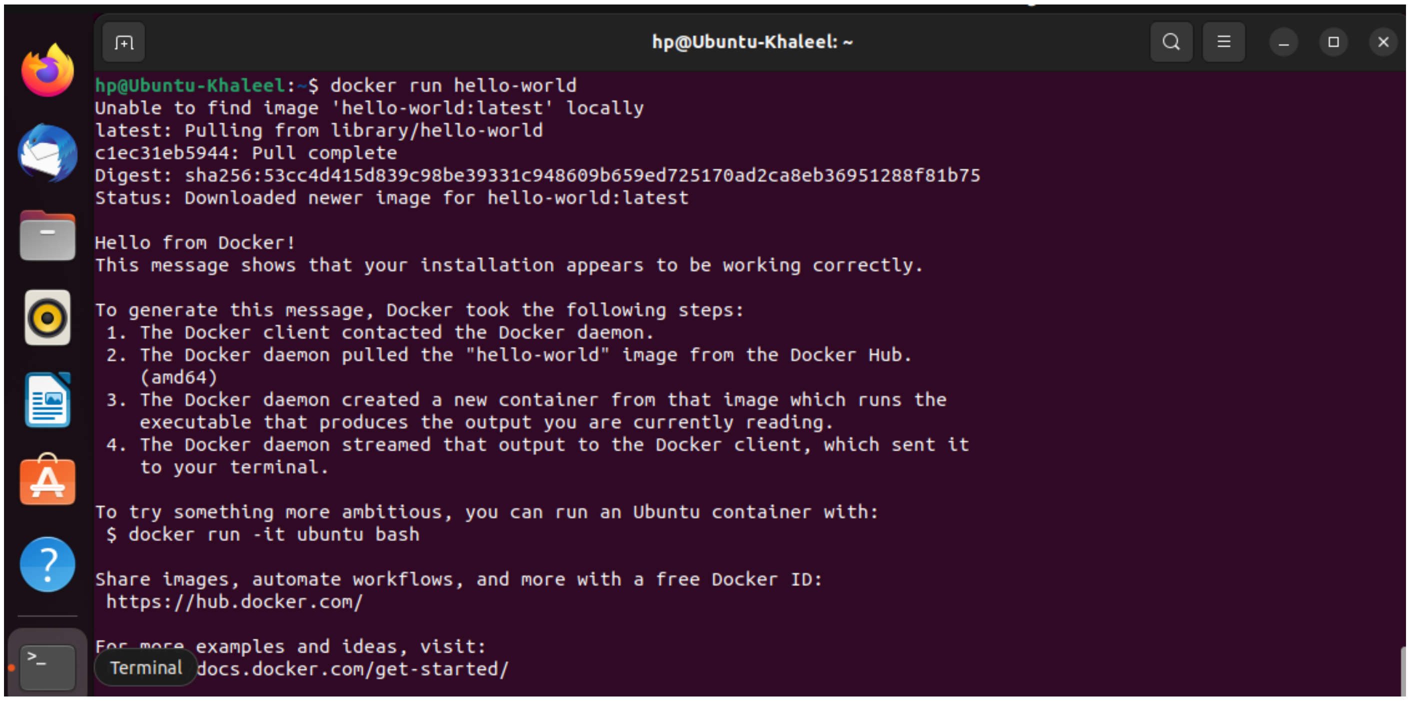
```
hp@Ubuntu-Khaleel:~$ docker container create -it ubuntu bash
2934ecd286bd53efebe809214e78d7bd1517fb0311ba3d4ac61843eba805bdd0
hp@Ubuntu-Khaleel:~$ docker container ls -a
CONTAINER ID   IMAGE     COMMAND   CREATED          STATUS    PORTS     NAMES
2934ecd286bd   ubuntu    "bash"    20 seconds ago   Created             strange_tesla
```

Lab 1

problem 1

- Run the container hello-world

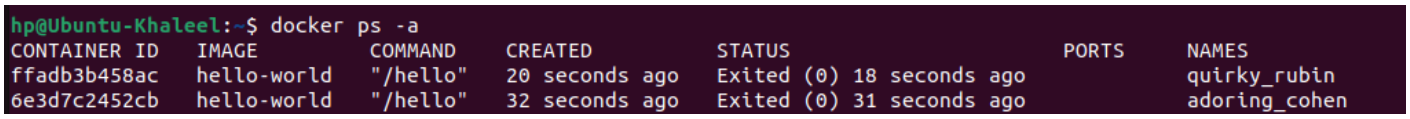
```
docker run hello-world
```



```
hp@Ubuntu-Khaleel:~$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:53cc4d415d839c98be39331c948609b659ed725170ad2ca8eb36951288f81b75
Status: Downloaded newer image for hello-world:latest

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/
```

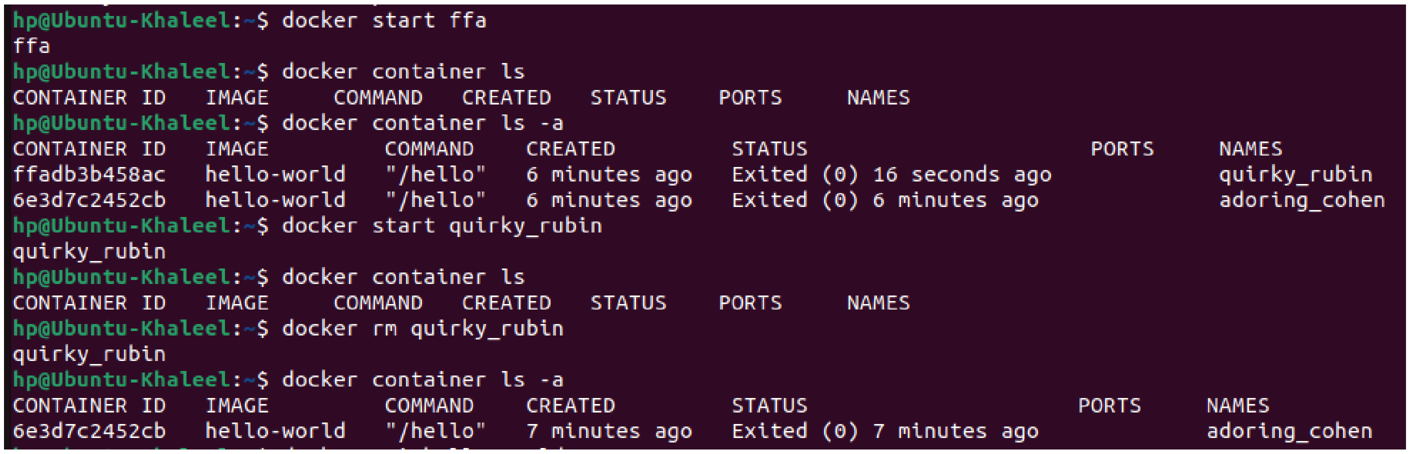
- Check the container status

```
docker ps -a
```


```
hp@Ubuntu-Khaleel:~$ docker ps -a
CONTAINER ID   IMAGE         COMMAND    CREATED          STATUS                      PORTS     NAMES
ffadb3b458ac   hello-world   "/hello"   20 seconds ago   Exited (0) 18 seconds ago             quirky_rubin
6e3d7c2452cb   hello-world   "/hello"   32 seconds ago   Exited (0) 31 seconds ago             adoring_cohen
```

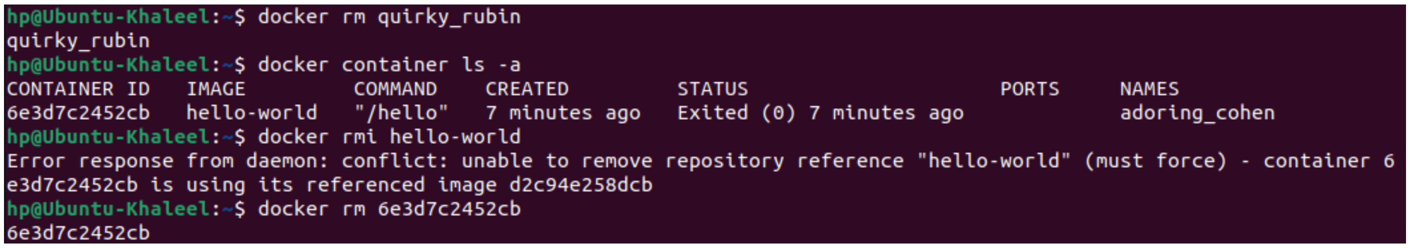
- Start the stopped container

`docker start <container_id_or_name>`


```
hp@Ubuntu-Khaleel:~$ docker start ffa
ffa
hp@Ubuntu-Khaleel:~$ docker container ls
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
hp@Ubuntu-Khaleel:~$ docker container ls -a
CONTAINER ID   IMAGE         COMMAND    CREATED         STATUS                      PORTS     NAMES
ffadb3b458ac   hello-world   "/hello"   6 minutes ago   Exited (0) 16 seconds ago             quirky_rubin
6e3d7c2452cb   hello-world   "/hello"   6 minutes ago   Exited (0) 6 minutes ago              adoring_cohen
hp@Ubuntu-Khaleel:~$ docker start quirky_rubin
quirky_rubin
hp@Ubuntu-Khaleel:~$ docker container ls
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
hp@Ubuntu-Khaleel:~$ docker rm quirky_rubin
quirky_rubin
hp@Ubuntu-Khaleel:~$ docker container ls -a
CONTAINER ID   IMAGE         COMMAND    CREATED         STATUS                     PORTS     NAMES
6e3d7c2452cb   hello-world   "/hello"   7 minutes ago   Exited (0) 7 minutes ago             adoring_cohen
```

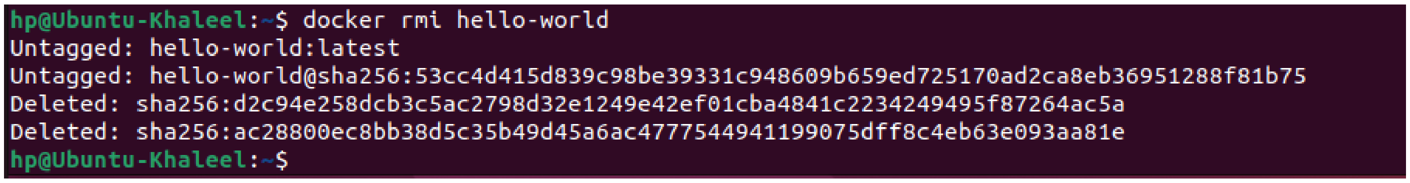
- Remove the container

`docker rm <container_id_or_name>`


```
hp@Ubuntu-Khaleel:~$ docker rm quirky_rubin
quirky_rubin
hp@Ubuntu-Khaleel:~$ docker container ls -a
CONTAINER ID   IMAGE         COMMAND    CREATED         STATUS                     PORTS     NAMES
6e3d7c2452cb   hello-world   "/hello"   7 minutes ago   Exited (0) 7 minutes ago             adoring_cohen
hp@Ubuntu-Khaleel:~$ docker rmi hello-world
Error response from daemon: conflict: unable to remove repository reference "hello-world" (must force) - container 6
e3d7c2452cb is using its referenced image d2c94e258dcb
hp@Ubuntu-Khaleel:~$ docker rm 6e3d7c2452cb
6e3d7c2452cb
```

- Remove the image

`docker rmi hello-world`


```
hp@Ubuntu-Khaleel:~$ docker rmi hello-world
Untagged: hello-world:latest
Untagged: hello-world@sha256:53cc4d415d839c98be39331c948609b659ed725170ad2ca8eb36951288f81b75
Deleted: sha256:d2c94e258dcb3c5ac2798d32e1249e42ef01cba4841c2234249495f87264ac5a
Deleted: sha256:ac28800ec8bb38d5c35b49d45a6ac4777544941199075dff8c4eb63e093aa81e
hp@Ubuntu-Khaleel:~$
```

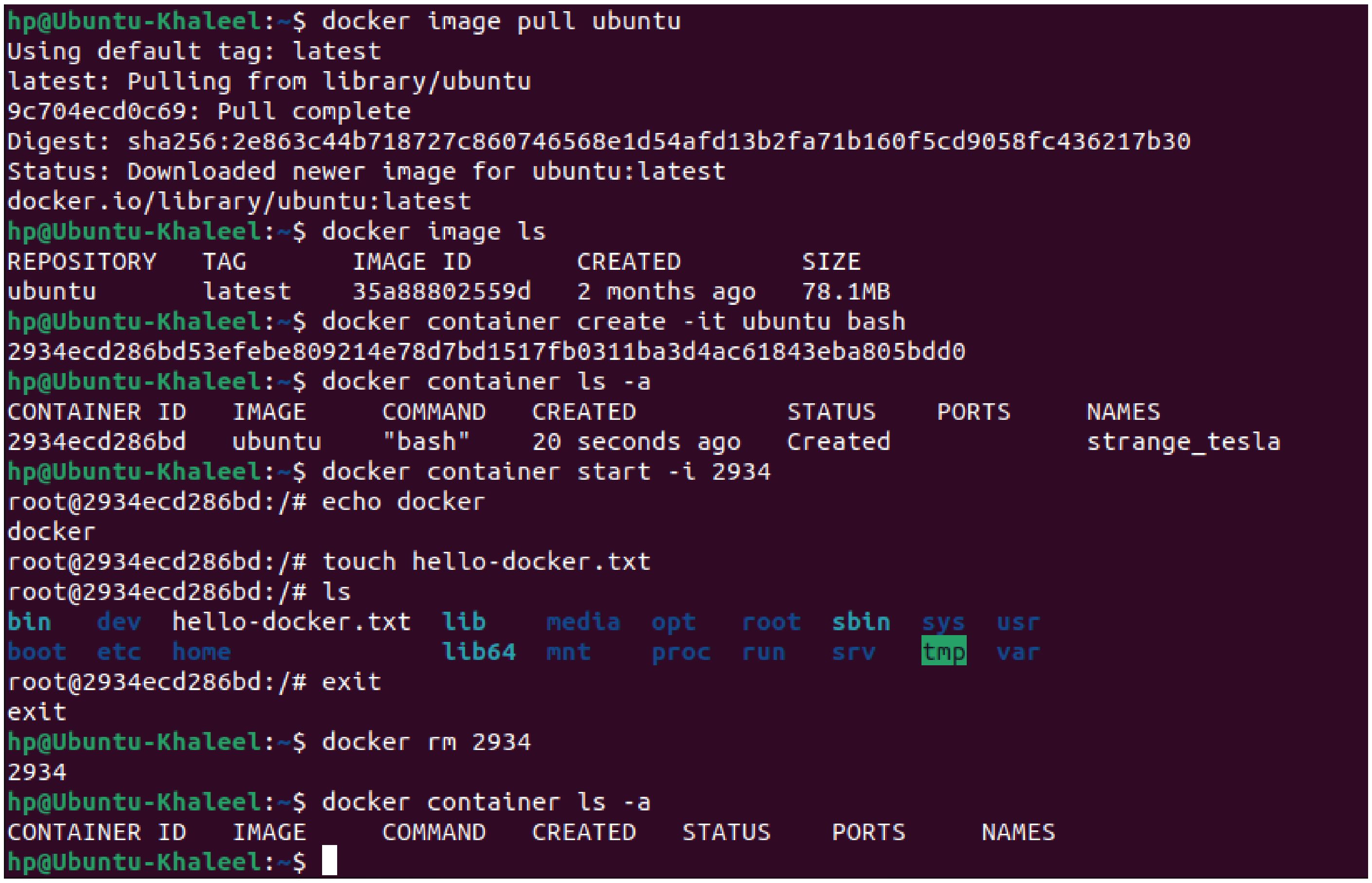
Problem 2

- Run container centos or ubuntu in an interactive mode

- Run the following command in the container “echo docker”

- Open a bash shell in the container and touch a file named hello-docker

- Stop the container and remove it. Write your comment about the file hello-docker



```
hp@Ubuntu-Khaleel:~$ docker image pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
9c704ecd0c69: Pull complete
Digest: sha256:2e863c44b718727c860746568e1d54afd13b2fa71b160f5cd9058fc436217b30
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
hp@Ubuntu-Khaleel:~$ docker image ls
REPOSITORY   TAG       IMAGE ID       CREATED        SIZE
ubuntu       latest    35a88802559d   2 months ago   78.1MB
hp@Ubuntu-Khaleel:~$ docker container create -it ubuntu bash
2934ecd286bd53efebe809214e78d7bd1517fb0311ba3d4ac61843eba805bdd0
hp@Ubuntu-Khaleel:~$ docker container ls -a
CONTAINER ID   IMAGE     COMMAND   CREATED          STATUS    PORTS     NAMES
2934ecd286bd   ubuntu    "bash"    20 seconds ago   Created             strange_tesla
hp@Ubuntu-Khaleel:~$ docker container start -i 2934
root@2934ecd286bd:/# echo docker
docker
root@2934ecd286bd:/# touch hello-docker.txt
root@2934ecd286bd:/# ls
bin   dev  hello-docker.txt  lib    media  opt   root  sbin  sys  usr
boot  etc  home              lib64  mnt    proc  run   srv   tmp  var
root@2934ecd286bd:/# exit
exit
hp@Ubuntu-Khaleel:~$ docker rm 2934
2934
hp@Ubuntu-Khaleel:~$ docker container ls -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
hp@Ubuntu-Khaleel:~$
```

Problem 3

Run a container nginx with name nginx and attach a volume to the container

- Volume for containing static html file

- Remove the container

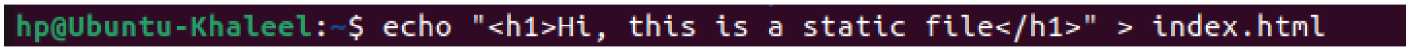
- Run a new container with the following:

- Attach the volume that was attached to the previous container

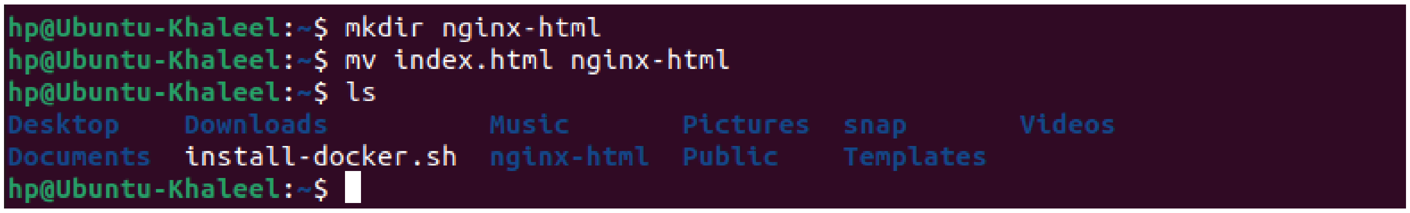
- Map port 80 to port 9898 on you host machine

- Access the html files from your browser

First, Create a Volume with a static html file

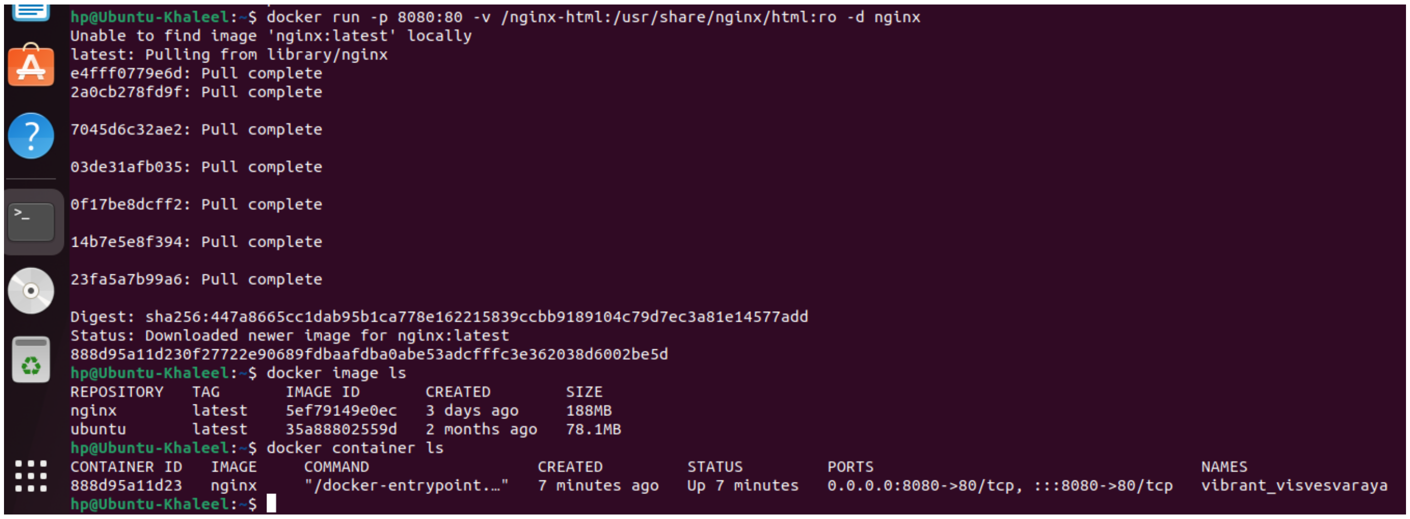


```
hp@Ubuntu-Khaleel:~$ echo "<h1>Hi, this is a static file</h1>" > index.html
```



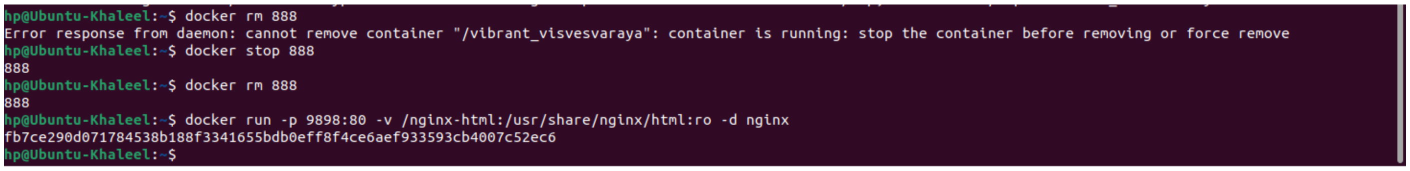
```
hp@Ubuntu-Khaleel:~$ mkdir nginx-html
hp@Ubuntu-Khaleel:~$ mv index.html nginx-html
hp@Ubuntu-Khaleel:~$ ls
Desktop    Downloads          Music       Pictures  snap       Videos
Documents  install-docker.sh  nginx-html  Public    Templates
hp@Ubuntu-Khaleel:~$
```

Then, running an Nginx container and attaching that volume to it



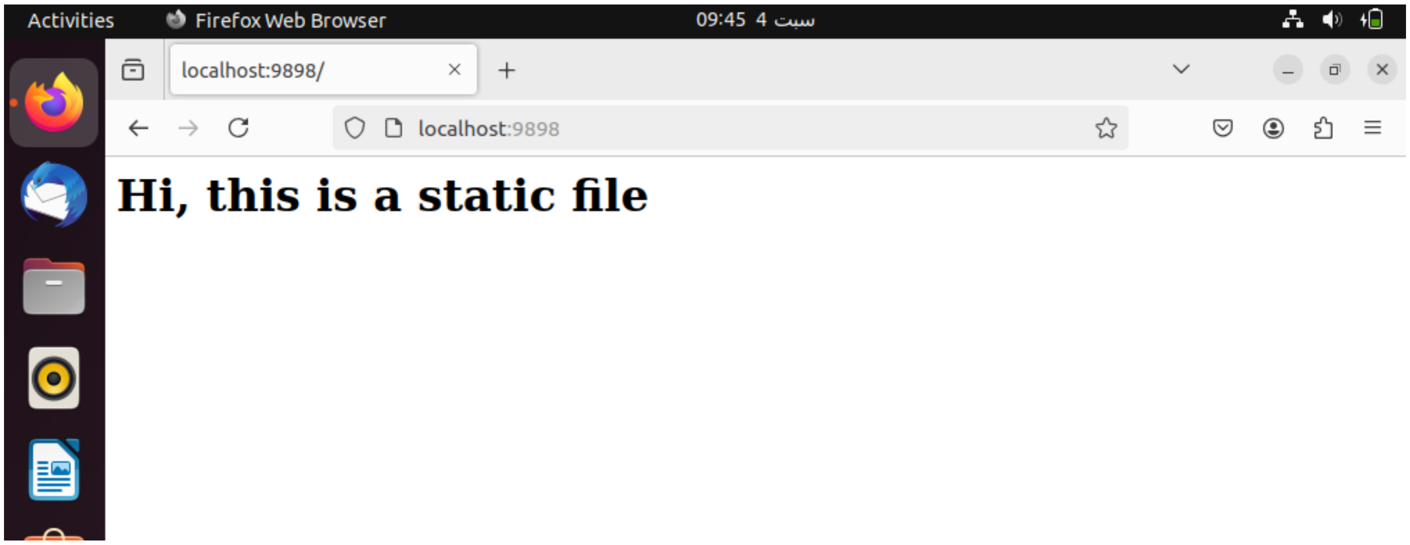
```
hp@Ubuntu-Khaleel:~$ docker run -p 8080:80 -v /nginx-html:/usr/share/nginx/html:ro -d nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
e4fff0779e6d: Pull complete
2a0cb278fd9f: Pull complete

7045d6c32ae2: Pull complete

03de31afb035: Pull complete

0f17be8dcff2: Pull complete

14b7e5e8f394: Pull complete

23fa5a7b99a6: Pull complete

Digest: sha256:447a8665cc1dab95b1ca778e162215839ccbb9189104c79d7ec3a81e14577add
Status: Downloaded newer image for nginx:latest
888d95a11d230f27722e90689fdbaafdba0abe53adcfffc3e362038d6002be5d
hp@Ubuntu-Khaleel:~$ docker image ls
REPOSITORY   TAG       IMAGE ID       CREATED        SIZE
nginx        latest    5ef79149e0ec   3 days ago     188MB
ubuntu       latest    35a88802559d   2 months ago   78.1MB
hp@Ubuntu-Khaleel:~$ docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS                                   NAMES
888d95a11d23   nginx     "/docker-entrypoint.…"   7 minutes ago   Up 7 minutes   0.0.0.0:8080->80/tcp, :::8080->80/tcp   vibrant_visvesvaraya
hp@Ubuntu-Khaleel:~$
```

Now delete that container, and create a new one but on port 9898



```
hp@Ubuntu-Khaleel:~$ docker rm 888
Error response from daemon: cannot remove container "/vibrant_visvesvaraya": container is running: stop the container before removing or force remove
hp@Ubuntu-Khaleel:~$ docker stop 888
888
hp@Ubuntu-Khaleel:~$ docker rm 888
888
hp@Ubuntu-Khaleel:~$ docker run -p 9898:80 -v /nginx-html:/usr/share/nginx/html:ro -d nginx
fb7ce290d071784538b188f3341655bdb0eff8f4ce6aef933593cb4007c52ec6
hp@Ubuntu-Khaleel:~$
```

Finally run the html file from your browser



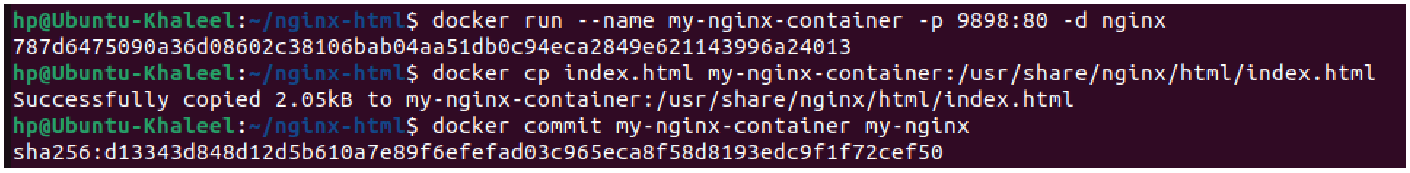
Hi, this is a static file

Problem 4

- Run the image nginx again without attaching any volumes

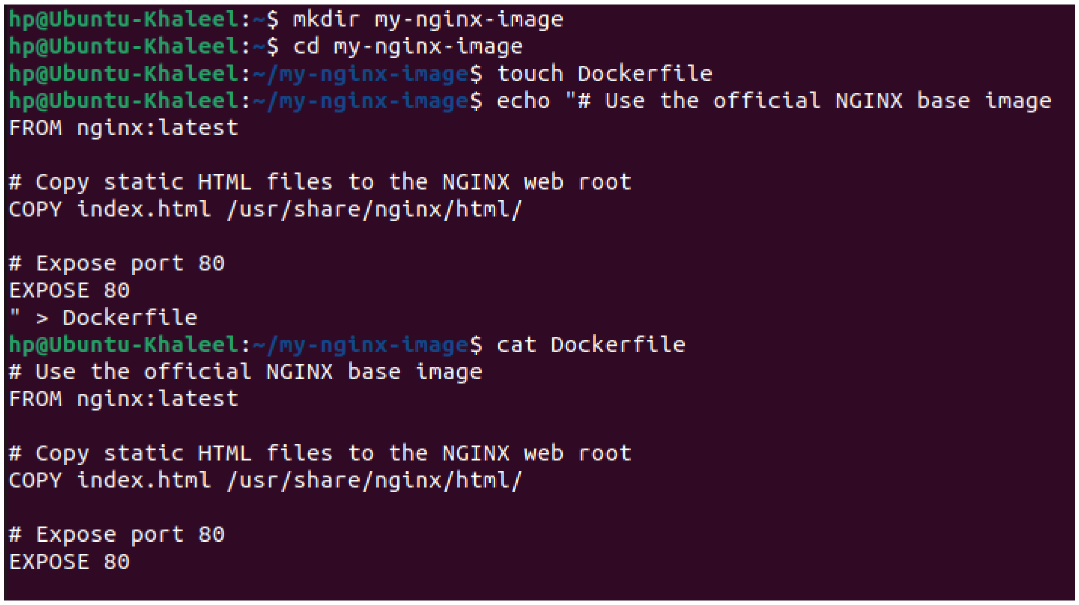
- Add html static files to the container and make sure they are accessible

- Commit the container with image name my nginx

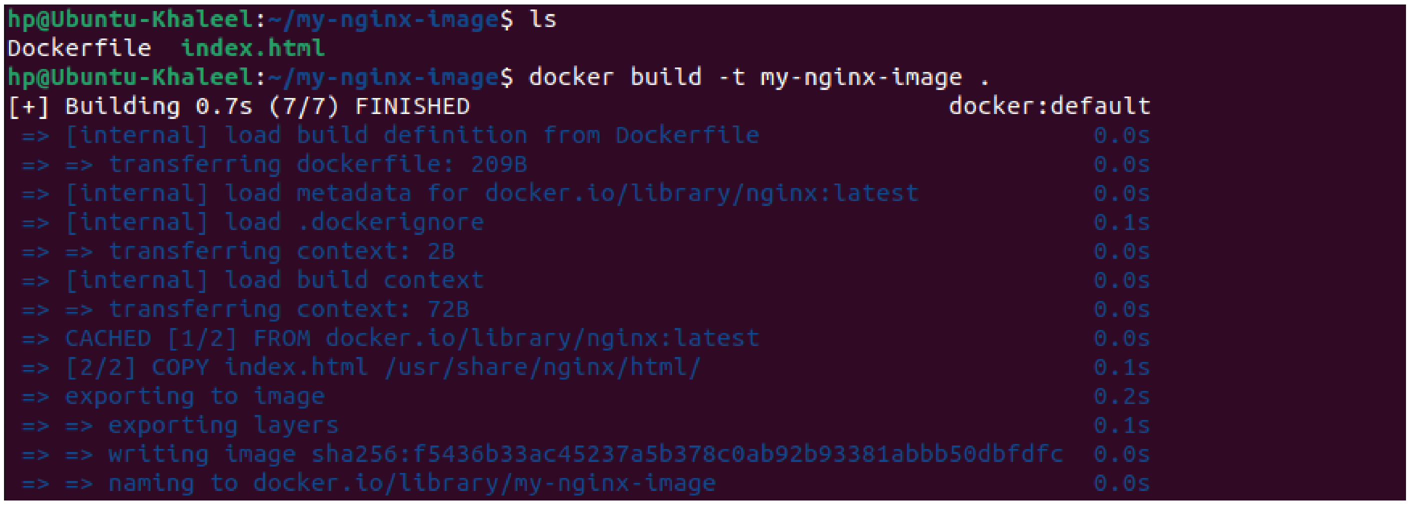


```
hp@Ubuntu-Khaleel:~/nginx-html$ docker run --name my-nginx-container -p 9898:80 -d nginx
787d6475090a36d08602c38106bab04aa51db0c94eca2849e621143996a24013
hp@Ubuntu-Khaleel:~/nginx-html$ docker cp index.html my-nginx-container:/usr/share/nginx/html/index.html
Successfully copied 2.05kB to my-nginx-container:/usr/share/nginx/html/index.html
hp@Ubuntu-Khaleel:~/nginx-html$ docker commit my-nginx-container my-nginx
sha256:d13343d848d12d5b610a7e89f6efefad03c965eca8f58d8193edc9f1f72cef50
```

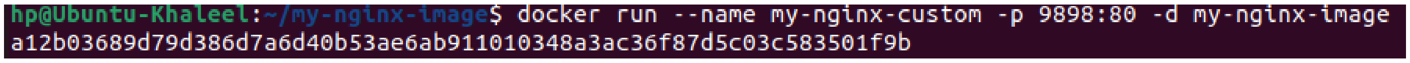
- Create a dockerfile for ngnix and build the image from this dockerfile



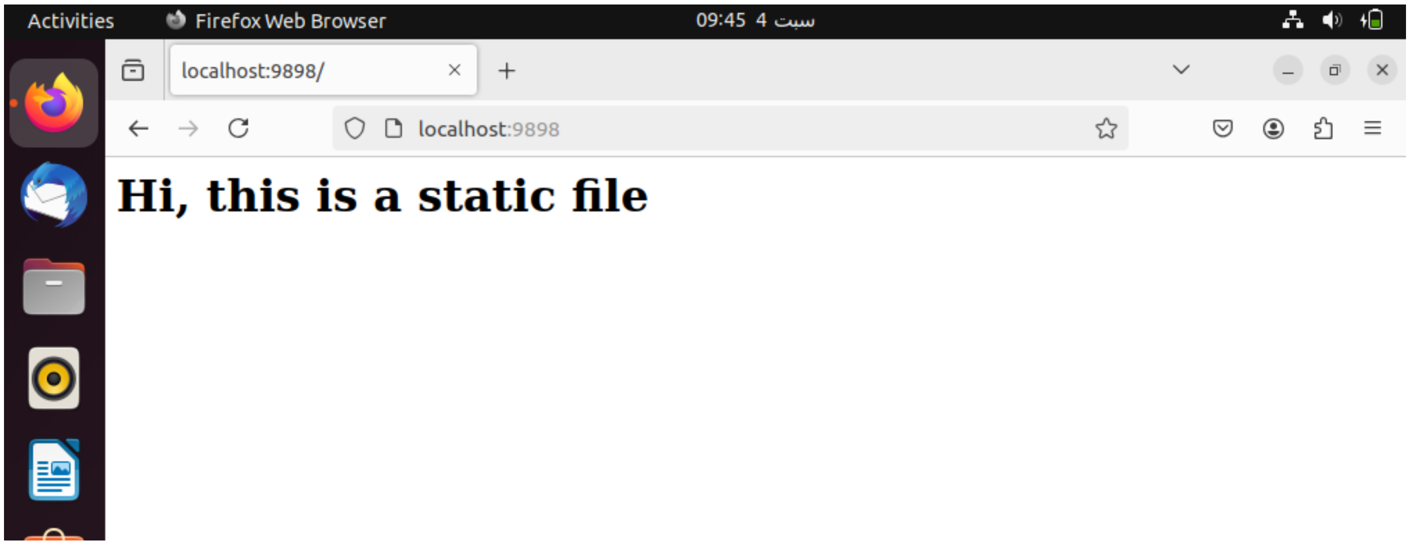
```
hp@Ubuntu-Khaleel:~$ mkdir my-nginx-image
hp@Ubuntu-Khaleel:~$ cd my-nginx-image
hp@Ubuntu-Khaleel:~/my-nginx-image$ touch Dockerfile
hp@Ubuntu-Khaleel:~/my-nginx-image$ echo "# Use the official NGINX base image
FROM nginx:latest

# Copy static HTML files to the NGINX web root
COPY index.html /usr/share/nginx/html/

# Expose port 80
EXPOSE 80
" > Dockerfile
hp@Ubuntu-Khaleel:~/my-nginx-image$ cat Dockerfile
# Use the official NGINX base image
FROM nginx:latest

# Copy static HTML files to the NGINX web root
COPY index.html /usr/share/nginx/html/

# Expose port 80
EXPOSE 80
```



```
hp@Ubuntu-Khaleel:~/my-nginx-image$ ls
Dockerfile  index.html
hp@Ubuntu-Khaleel:~/my-nginx-image$ docker build -t my-nginx-image .
[+] Building 0.7s (7/7) FINISHED                                 docker:default
 => [internal] load build definition from Dockerfile                      0.0s
 => => transferring dockerfile: 209B                                      0.0s
 => [internal] load metadata for docker.io/library/nginx:latest           0.0s
 => [internal] load .dockerignore                                         0.1s
 => => transferring context: 2B                                           0.0s
 => [internal] load build context                                         0.0s
 => => transferring context: 72B                                          0.0s
 => CACHED [1/2] FROM docker.io/library/nginx:latest                      0.0s
 => [2/2] COPY index.html /usr/share/nginx/html/                          0.1s
 => exporting to image                                                    0.2s
 => => exporting layers                                                   0.1s
 => => writing image sha256:f5436b33ac45237a5b378c0ab92b93381abbb50dbfdfc  0.0s
 => => naming to docker.io/library/my-nginx-image                         0.0s
```



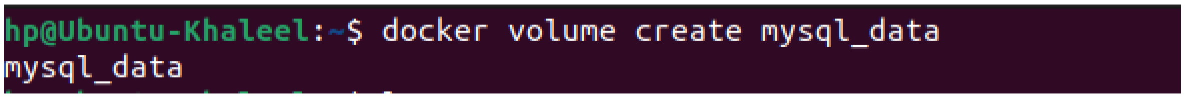
```
hp@Ubuntu-Khaleel:~/my-nginx-image$ docker run --name my-nginx-custom -p 9898:80 -d my-nginx-image
a12b03689d79d386d7a6d40b53ae6ab911010348a3ac36f87d5c03c583501f9b
```



Hi, this is a static file

Problem 5

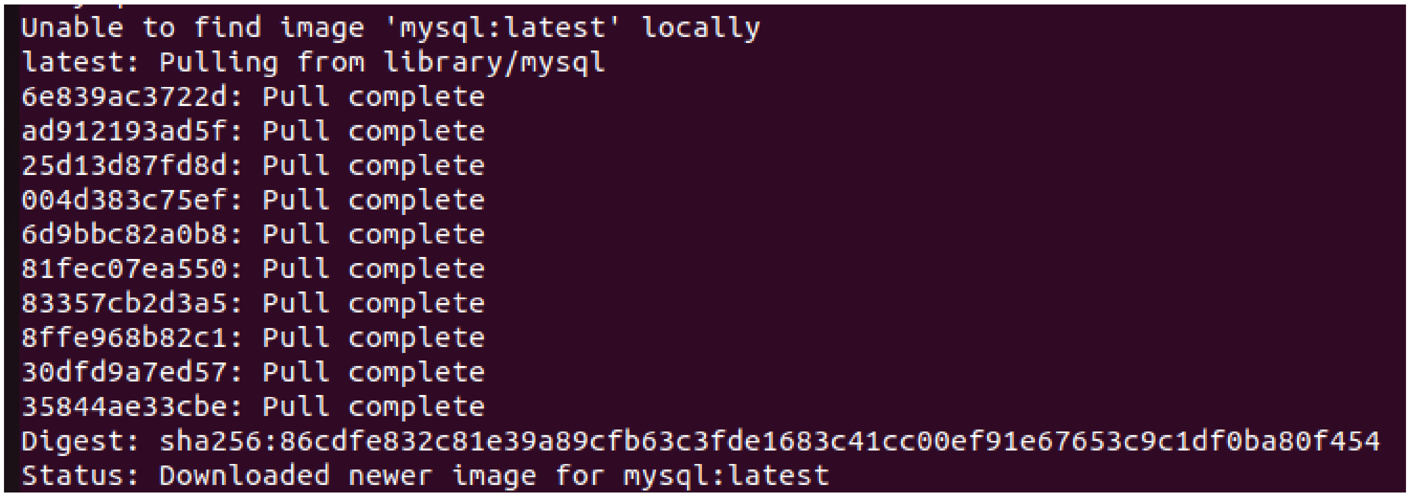
- Create a volume called mysql_data



```
hp@Ubuntu-Khaleel:~$ docker volume create mysql_data
mysql_data
```

- deploy a MySQL database called app-database.

- use the mysql latest image

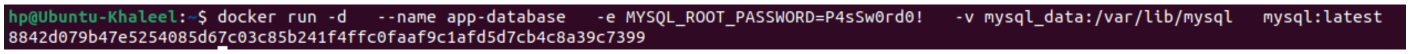


```
Unable to find image 'mysql:latest' locally
latest: Pulling from library/mysql
6e839ac3722d: Pull complete
ad912193ad5f: Pull complete
25d13d87fd8d: Pull complete
004d383c75ef: Pull complete
6d9bbc82a0b8: Pull complete
81fec07ea550: Pull complete
83357cb2d3a5: Pull complete
8ffe968b82c1: Pull complete
30dfd9a7ed57: Pull complete
35844ae33cbe: Pull complete
Digest: sha256:86cdfe832c81e39a89cfb63c3fde1683c41cc00ef91e67653c9c1df0ba80f454
Status: Downloaded newer image for mysql:latest
```

- use the -e flag to set MYSQL_ROOT_PASSWORD to P4sSw0rd0!.

- mount the mysql_data volume to /var/lib/mysql.

- the container should run in the background.



```
hp@Ubuntu-Khaleel:~$ docker run -d   --name app-database   -e MYSQL_ROOT_PASSWORD=P4sSw0rd0!   -v mysql_data:/var/lib/mysql   mysql:latest
8842d079b47e5254085d67c03c85b241f4ffc0faaf9c1afd5d7cb4c8a39c7399
```

Lab 2

Problem 1

From ubuntu image:

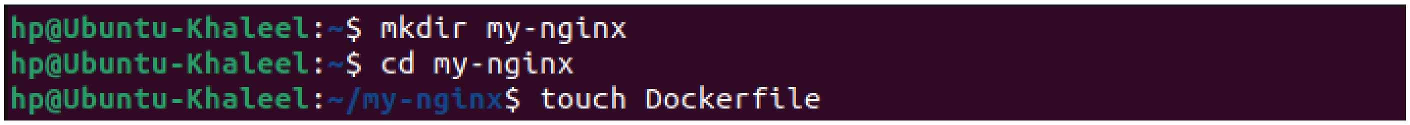
- Install nginx

- index.html one as file

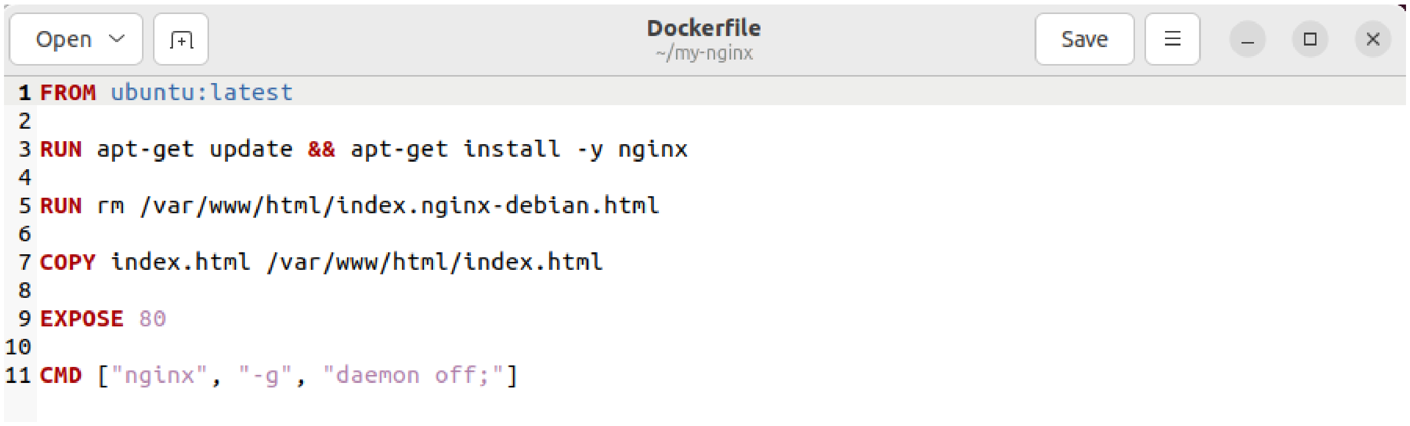
- Expose

- Start

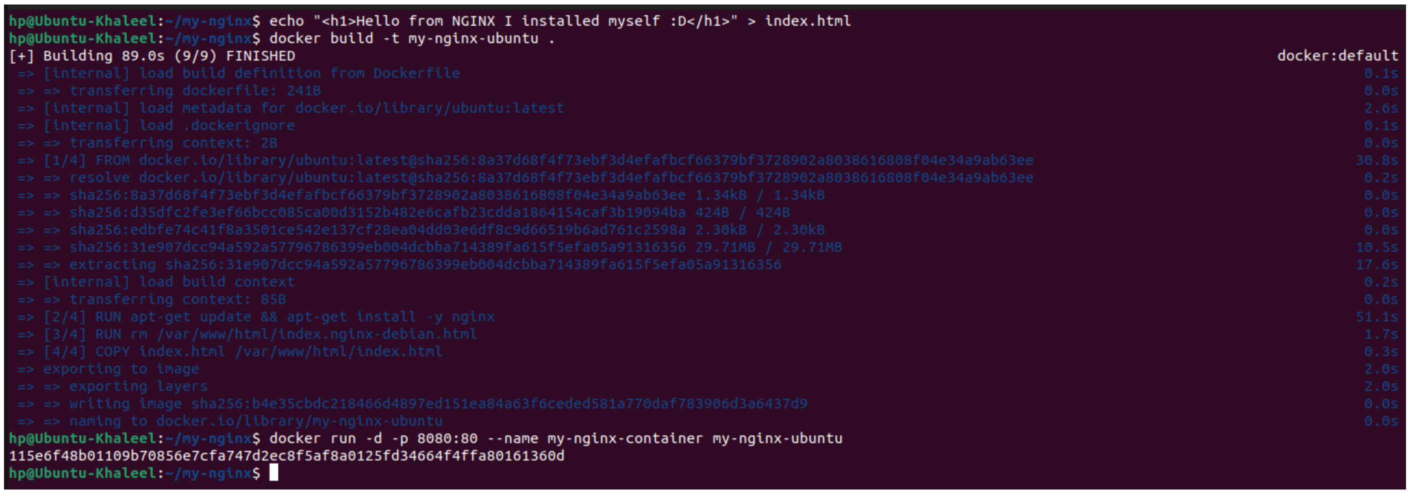
- Port mapping



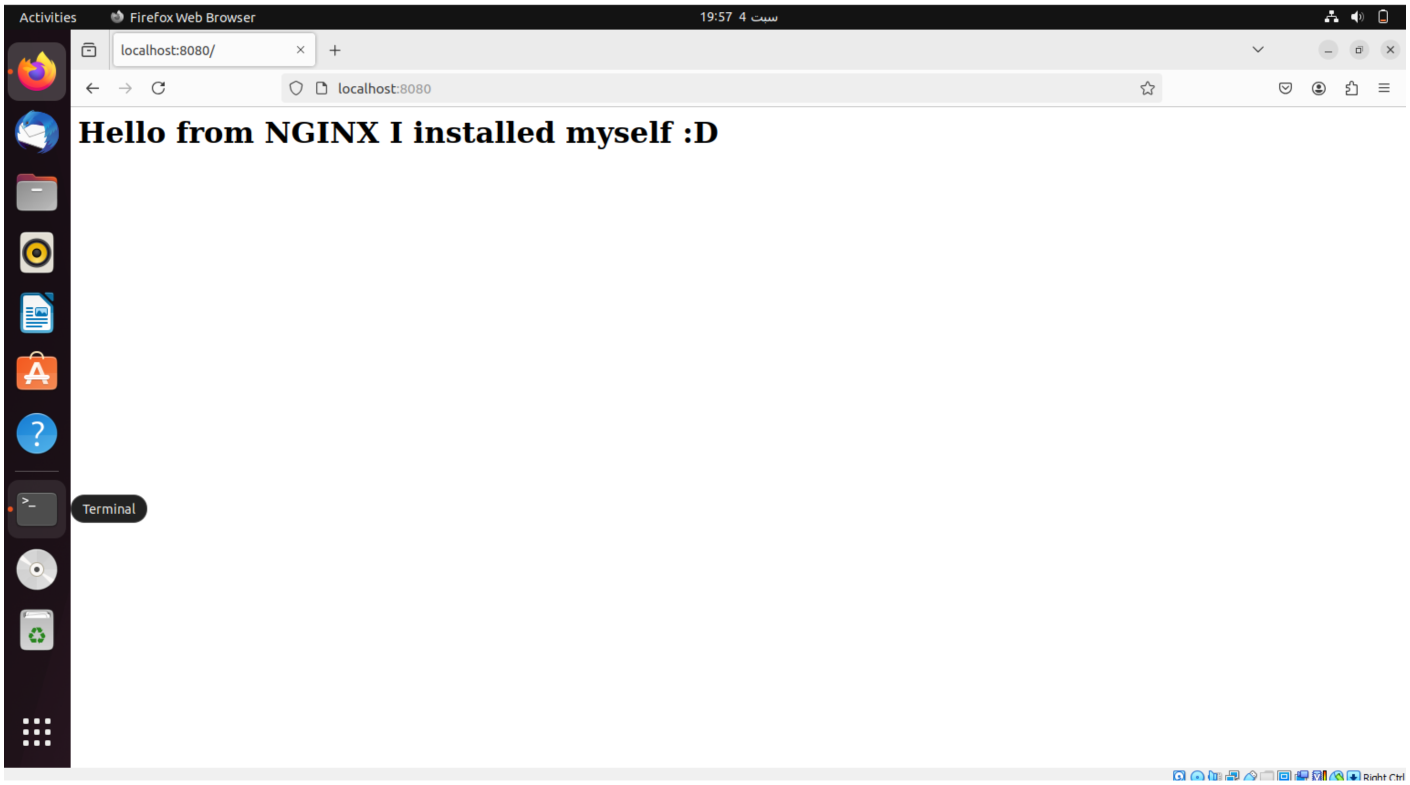
```
hp@Ubuntu-Khaleel:~$ mkdir my-nginx
hp@Ubuntu-Khaleel:~$ cd my-nginx
hp@Ubuntu-Khaleel:~/my-nginx$ touch Dockerfile
```



```
FROM ubuntu:latest

RUN apt-get update && apt-get install -y nginx

RUN rm /var/www/html/index.nginx-debian.html

COPY index.html /var/www/html/index.html

EXPOSE 80

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



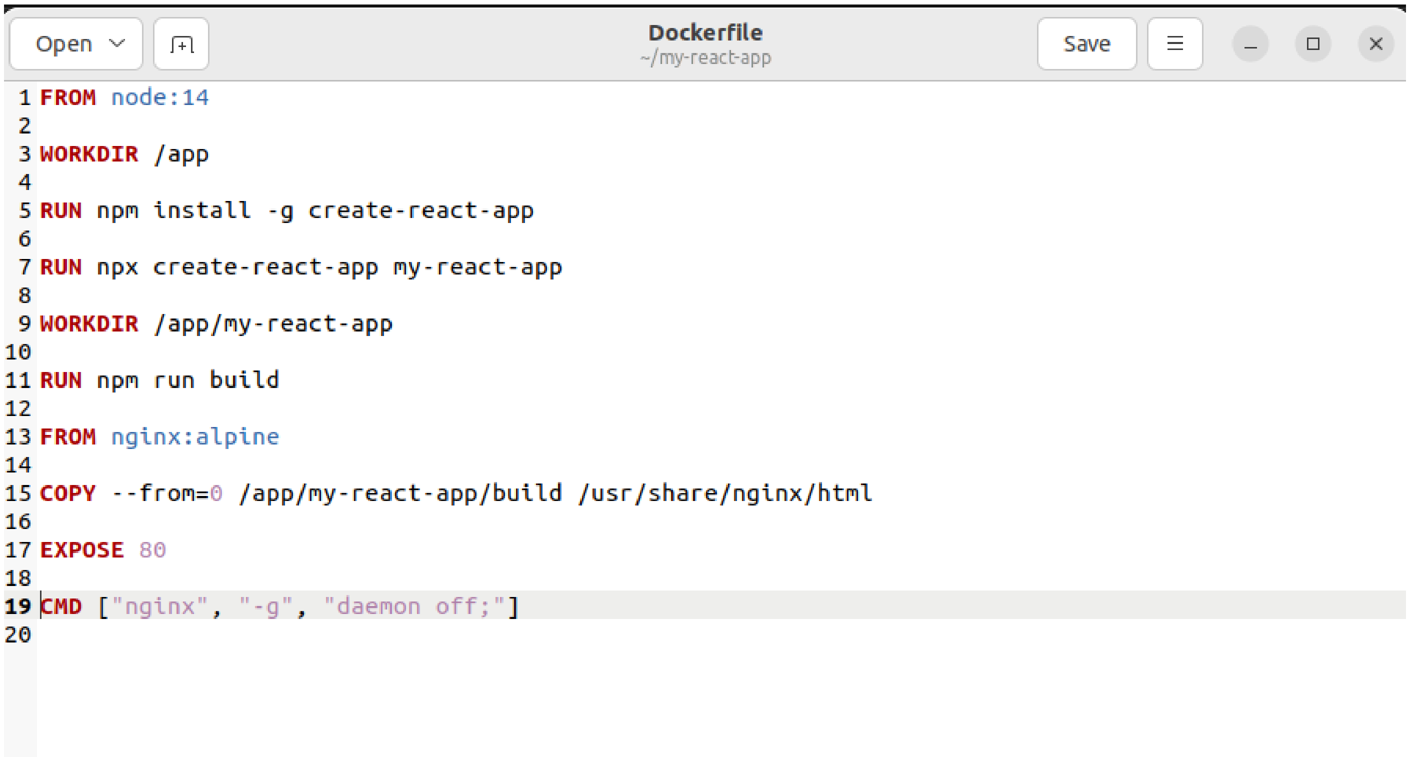
```
hp@Ubuntu-Khaleel:~/my-nginx$ echo "<h1>Hello from NGINX I installed myself :D</h1>" > index.html
hp@Ubuntu-Khaleel:~/my-nginx$ docker build -t my-nginx-ubuntu .
[+] Building 89.0s (9/9) FINISHED                                                                              docker:default
 => [internal] load build definition from Dockerfile                                                                      0.1s
 => => transferring dockerfile: 241B                                                                                      0.0s
 => [internal] load metadata for docker.io/library/ubuntu:latest                                                          2.6s
 => [internal] load .dockerignore                                                                                         0.1s
 => => transferring context: 2B                                                                                           0.0s
 => [1/4] FROM docker.io/library/ubuntu:latest@sha256:8a37d68f4f73ebf3d4efafbcf66379bf3728902a8038616808f04e34a9ab63ee   30.8s
 => => resolve docker.io/library/ubuntu:latest@sha256:8a37d68f4f73ebf3d4efafbcf66379bf3728902a8038616808f04e34a9ab63ee    0.2s
 => => sha256:8a37d68f4f73ebf3d4efafbcf66379bf3728902a8038616808f04e34a9ab63ee 1.34kB / 1.34kB                            0.0s
 => => sha256:d35dfc2fe3ef66bcc085ca00d3152b482e6cafb23cdda1864154caf3b19094ba 424B / 424B                                0.0s
 => => sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a 2.30kB / 2.30kB                            0.0s
 => => sha256:31e907dcc94a592a57796786399eb004dcbba714389fa615f5efa05a91316356 29.71MB / 29.71MB                         10.5s
 => => extracting sha256:31e907dcc94a592a57796786399eb004dcbba714389fa615f5efa05a91316356                                17.6s
 => [internal] load build context                                                                                         0.2s
 => => transferring context: 85B                                                                                          0.0s
 => [2/4] RUN apt-get update && apt-get install -y nginx                                                                 51.1s
 => [3/4] RUN rm /var/www/html/index.nginx-debian.html                                                                    1.7s
 => [4/4] COPY index.html /var/www/html/index.html                                                                        0.3s
 => exporting to image                                                                                                    2.0s
 => => exporting layers                                                                                                   2.0s
 => => writing image sha256:b4e35cbdc218466d4897ed151ea84a63f6ceded581a770daf783906d3a6437d9                              0.0s
 => => naming to docker.io/library/my-nginx-ubuntu                                                                        0.0s
hp@Ubuntu-Khaleel:~/my-nginx$ docker run -d -p 8080:80 --name my-nginx-container my-nginx-ubuntu
115e6f48b01109b70856e7cfa747d2ec8f5af8a0125fd34664f4ffa80161360d
hp@Ubuntu-Khaleel:~/my-nginx$ 
```



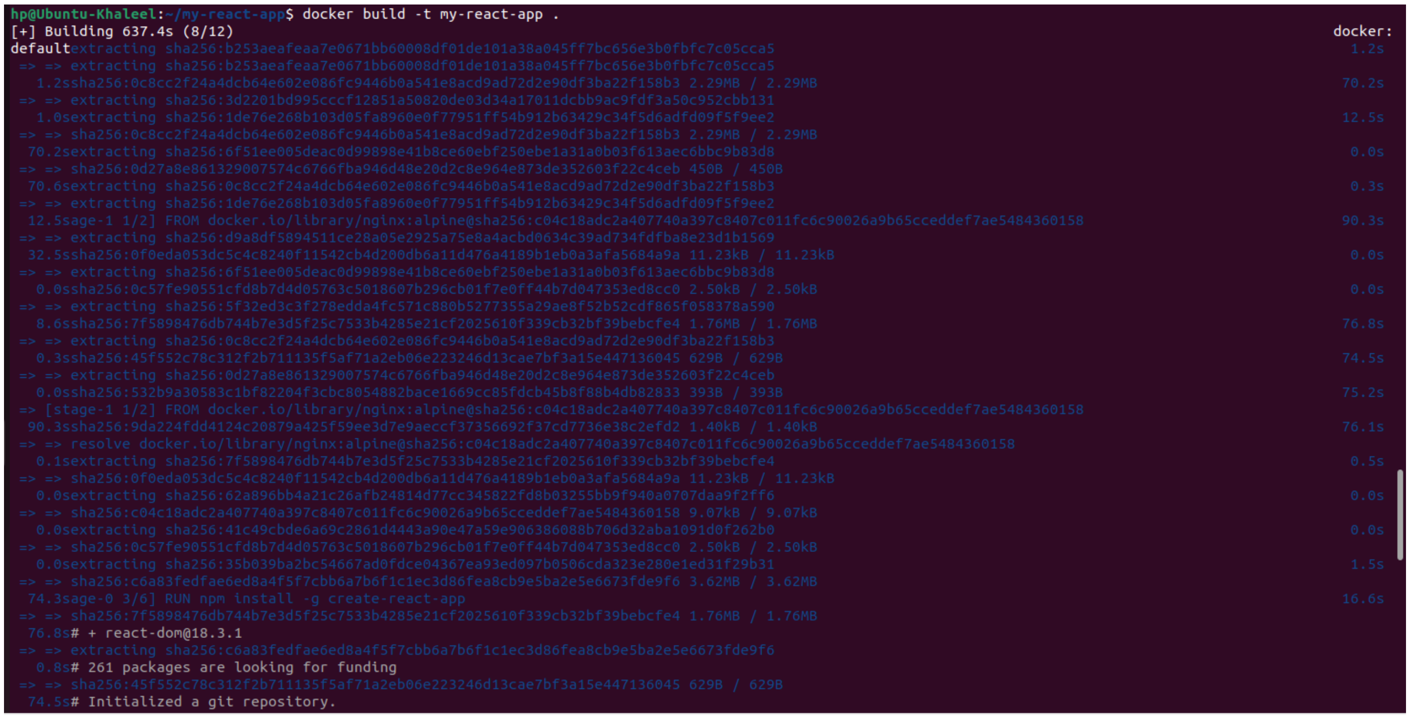
Hello from NGINX I installed myself :D

Problem 2

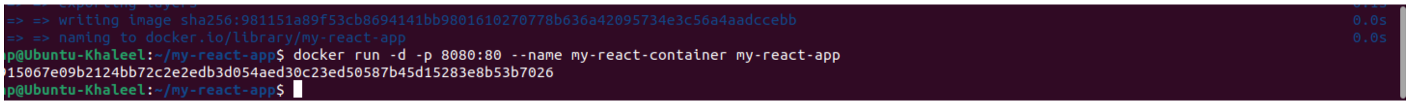
- Create react app docker container "using Dockerfile"



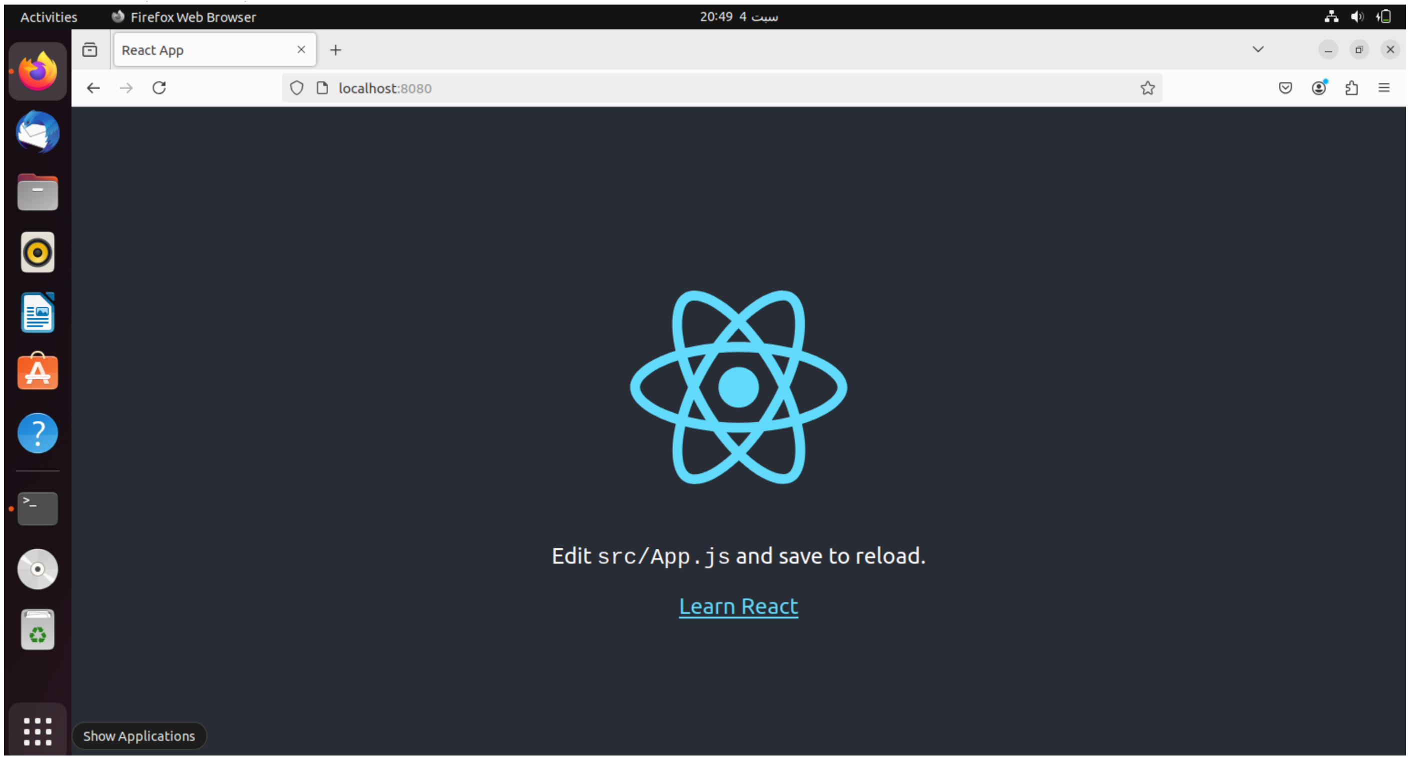
```
FROM node:14

WORKDIR /app

RUN npm install -g create-react-app

RUN npx create-react-app my-react-app

WORKDIR /app/my-react-app

RUN npm run build

FROM nginx:alpine

COPY --from=0 /app/my-react-app/build /usr/share/nginx/html

EXPOSE 80

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

```



```
hp@Ubuntu-Khaleel:~/my-react-app$ docker build -t my-react-app .
[+] Building 637.4s (8/12)                                                                                          docker:
defaultextracting sha256:b253aeafeaa7e0671bb60008df01de101a38a045ff7bc656e3b0fbfc7c05cca5                             1.2s
 => => extracting sha256:b253aeafeaa7e0671bb60008df01de101a38a045ff7bc656e3b0fbfc7c05cca5
   1.2ssha256:0c8cc2f24a4dcb64e602e086fc9446b0a541e8acd9ad72d2e90df3ba22f158b3 2.29MB / 2.29MB                       70.2s
 => => extracting sha256:3d2201bd995cccf12851a50820de03d34a17011dcbb9ac9fdf3a50c952cbb131
   1.0sextracting sha256:1de76e268b103d05fa8960e0f77951ff54b912b63429c34f5d6adfd09f5f9ee2                            12.5s
 => => sha256:0c8cc2f24a4dcb64e602e086fc9446b0a541e8acd9ad72d2e90df3ba22f158b3 2.29MB / 2.29MB
  70.2sextracting sha256:6f51ee005deac0d99898e41b8ce60ebf250ebe1a31a0b03f613aec6bbc9b83d8                             0.0s
 => => sha256:0d27a8e861329007574c6766fba946d48e20d2c8e964e873de352603f22c4ceb 450B / 450B
  70.6sextracting sha256:0c8cc2f24a4dcb64e602e086fc9446b0a541e8acd9ad72d2e90df3ba22f158b3                             0.3s
 => => extracting sha256:1de76e268b103d05fa8960e0f77951ff54b912b63429c34f5d6adfd09f5f9ee2
  12.5sage-1 1/2] FROM docker.io/library/nginx:alpine@sha256:c04c18adc2a407740a397c8407c011fc6c90026a9b65cceddef7ae5484360158      90.3s
 => => extracting sha256:d9a8df5894511ce28a05e2925a75e8a4acbd0634c39ad734fdfba8e23d1b1569
  32.5ssha256:0f0eda053dc5c4c8240f11542cb4d200db6a11d476a4189b1eb0a3afa5684a9a 11.23kB / 11.23kB                      0.0s
 => => extracting sha256:6f51ee005deac0d99898e41b8ce60ebf250ebe1a31a0b03f613aec6bbc9b83d8
   0.0ssha256:0c57fe90551cfd8b7d4d05763c5018607b296cb01f7e0ff44b7d047353ed8cc0 2.50kB / 2.50kB                        0.0s
 => => extracting sha256:5f32ed3c3f278edda4fc571c880b5277355a29ae8f52b52cdf865f058378a590
   8.6ssha256:7f5898476db744b7e3d5f25c7533b4285e21cf2025610f339cb32bf39bebcfe4 1.76MB / 1.76MB                       76.8s
 => => extracting sha256:0c8cc2f24a4dcb64e602e086fc9446b0a541e8acd9ad72d2e90df3ba22f158b3
   0.3ssha256:45f552c78c312f2b711135f5af71a2eb06e223246d13cae7bf3a15e447136045 629B / 629B                           74.5s
 => => extracting sha256:0d27a8e861329007574c6766fba946d48e20d2c8e964e873de352603f22c4ceb
   0.0ssha256:532b9a30583c1bf82204f3cbc8054882bace1669cc85fdcb45b8f88b4db82833 393B / 393B                           75.2s
 => [stage-1 1/2] FROM docker.io/library/nginx:alpine@sha256:c04c18adc2a407740a397c8407c011fc6c90026a9b65cceddef7ae5484360158
  90.3ssha256:9da224fdd4124c20879a425f59ee3d7e9aeccf37356692f37cd7736e38c2efd2 1.40kB / 1.40kB                       76.1s
 => => resolve docker.io/library/nginx:alpine@sha256:c04c18adc2a407740a397c8407c011fc6c90026a9b65cceddef7ae5484360158
   0.1sextracting sha256:7f5898476db744b7e3d5f25c7533b4285e21cf2025610f339cb32bf39bebcfe4                             0.5s
 => => sha256:0f0eda053dc5c4c8240f11542cb4d200db6a11d476a4189b1eb0a3afa5684a9a 11.23kB / 11.23kB
   0.0sextracting sha256:62a896bb4a21c26afb24814d77cc345822fd8b03255bb9f940a0707daa9f2ff6                             0.0s
 => => sha256:c04c18adc2a407740a397c8407c011fc6c90026a9b65cceddef7ae5484360158 9.07kB / 9.07kB
   0.0sextracting sha256:41c49cbde6a69c2861d4443a90e47a59e906386088b706d32aba1091d0f262b0                             0.0s
 => => sha256:0c57fe90551cfd8b7d4d05763c5018607b296cb01f7e0ff44b7d047353ed8cc0 2.50kB / 2.50kB
   0.0sextracting sha256:35b039ba2bc54667ad0fdce04367ea93ed097b0506cda323e280e1ed31f29b31                             1.5s
 => => sha256:c6a83fedfae6ed8a4f5f7cbb6a7b6f1c1ec3d86fea8cb9e5ba2e5e6673fde9f6 3.62MB / 3.62MB
  74.3sage-0 3/6] RUN npm install -g create-react-app                                                                16.6s
 => => sha256:7f5898476db744b7e3d5f25c7533b4285e21cf2025610f339cb32bf39bebcfe4 1.76MB / 1.76MB
  76.8s# + react-dom@18.3.1
 => => extracting sha256:c6a83fedfae6ed8a4f5f7cbb6a7b6f1c1ec3d86fea8cb9e5ba2e5e6673fde9f6
   0.8s# 261 packages are looking for funding
 => => sha256:45f552c78c312f2b711135f5af71a2eb06e223246d13cae7bf3a15e447136045 629B / 629B
  74.5s# Initialized a git repository.
```



```
=> => writing image sha256:981151a89f53cb8694141bb9801610270778b636a42095734e3c56a4aadccebb                          0.0s
=> => naming to docker.io/library/my-react-app                                                                       0.0s
p@Ubuntu-Khaleel:~/my-react-app$ docker run -d -p 8080:80 --name my-react-container my-react-app
15067e09b2124bb72c2e2edb3d054aed30c23ed50587b45d15283e8b53b7026
p@Ubuntu-Khaleel:~/my-react-app$
```



Edit src/App.js and save to reload.

Learn React

Problem 3

- Create flask app to count number of visits to browser:

- Create new directory called flask

```
mkdir flask

cd flask
```

- then add app.py file

```
cat <<EOF > app.py

from flask import Flask

from redis import Redis

app = Flask(__name__)

redis = Redis(host='redis', port=6379)

@app.route('/')

def hello():

 count = redis.incr('hits')

 return f'Hello! This page has been visited {count} times.'

if __name__ == "__main__":

app.run(host="0.0.0.0", port=5000)

EOF
```

- and requirements.txt files

```
cat <<EOF > requirements.txt

flask

redis

EOF
```

- Create Dockerfile for the python app

```
cat <<EOF > Dockerfile
```

```
FROM python:3.8-slim

WORKDIR /app

COPY requirements.txt requirements.txt

RUN pip install -r requirements.txt

COPY . .

CMD ["python", "app.py"]

EOF
```

- Create docker-compose for the app and use Redis as temp DB.

```
cat <<EOF > docker-compose.yml

version: '3'

services:

  web:

    build: .

    ports:

      - "5000:5000"

    depends_on:

      - redis

  redis:

    image: "redis:alpine"

EOF
```

- Build and run with Docker Compose

```
docker-compose up --build
```