DEPLOYING A STATIC HTML WEBSITE USING DOCKER AND NGINX

1. Create EC2 instance and use the below commands to install docker.

- Setting up the Docker repository on the ubuntu system

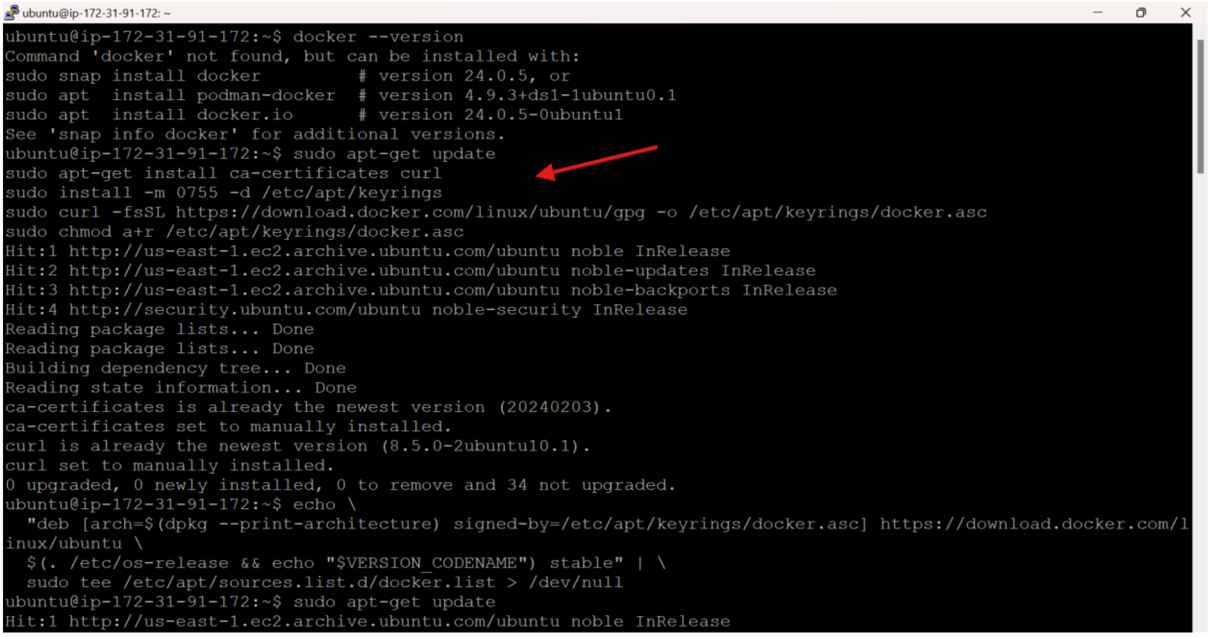
$ sudo apt-get update

$ sudo apt-get install ca-certificates curl

$ sudo install -m 0755 -d /etc/apt/keyrings

$ sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc

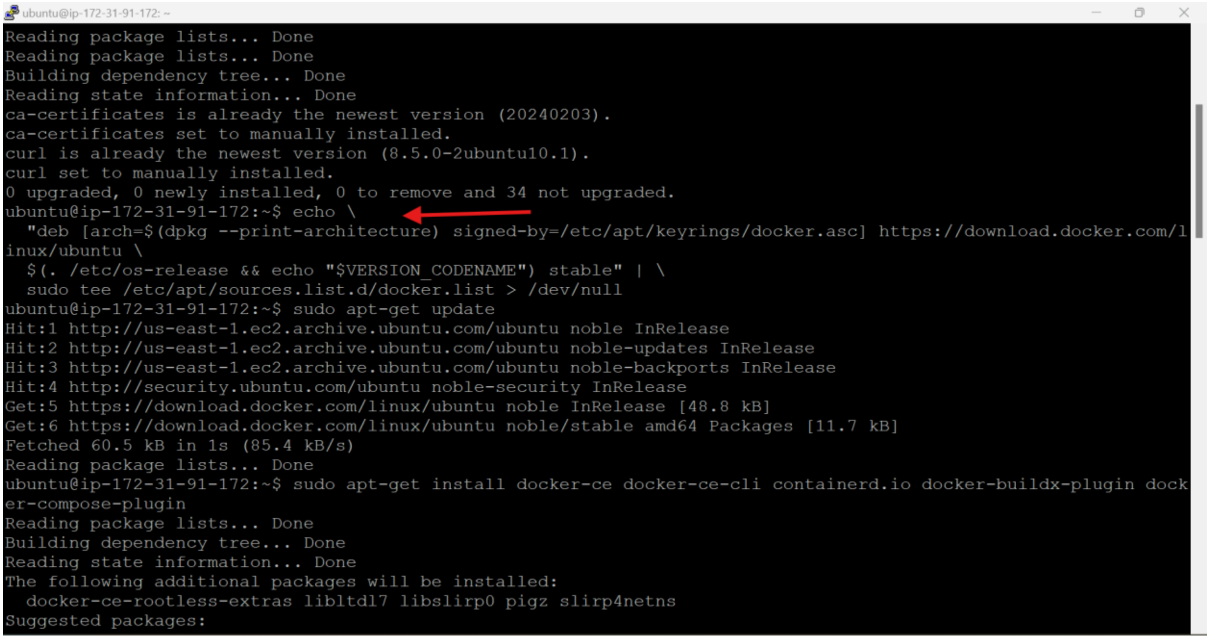
$ sudo chmod a+r /etc/apt/keyrings/docker.asc



```
ubuntu@ip-172-31-91-172:~$ docker --version
Command 'docker' not found, but can be installed with:
sudo snap install docker         # version 24.0.5, or
sudo apt  install podman-docker  # version 4.9.3+ds1-1ubuntu0.1
sudo apt  install docker.io      # version 24.0.5-0ubuntu1
See 'snap info docker' for additional versions.
ubuntu@ip-172-31-91-172:~$ sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20240203).
ca-certificates set to manually installed.
curl is already the newest version (8.5.0-2ubuntu10.1).
curl set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 34 not upgraded.
ubuntu@ip-172-31-91-172:~$ echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/l
inux/ubuntu \
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
ubuntu@ip-172-31-91-172:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
```

- Add Docker's official repository to your system's package sources list and then update the package list to include packages from this new repository.

```
 echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc]
https://download.docker.com/linux/ubuntu \
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
```

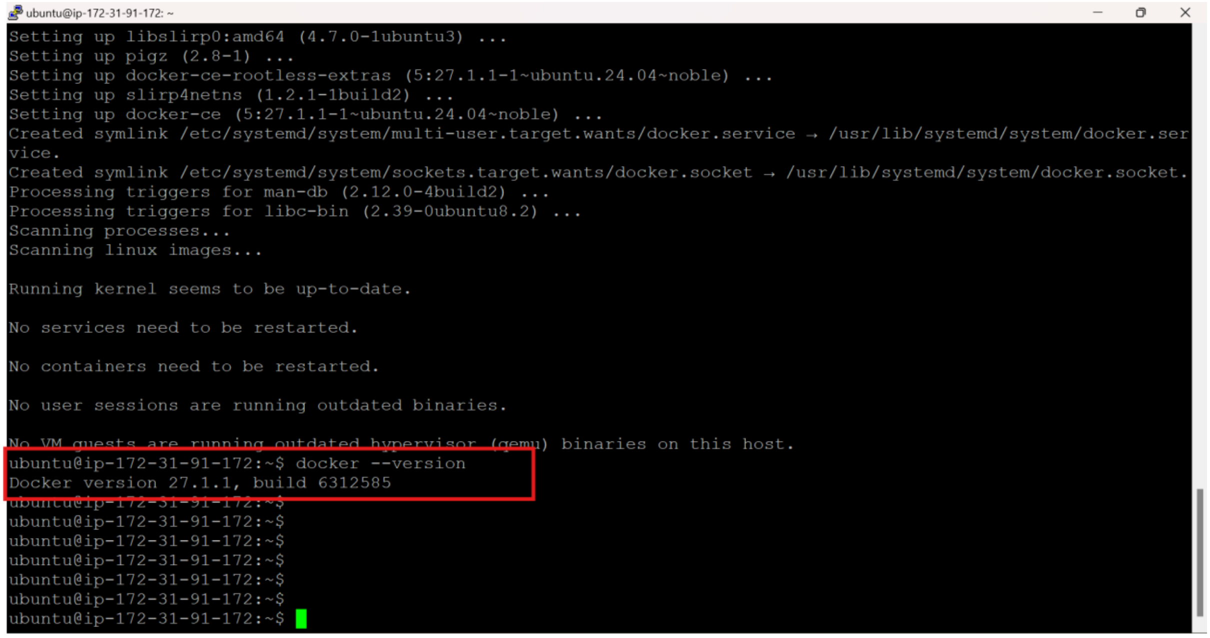


```
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20240203).
ca-certificates set to manually installed.
curl is already the newest version (8.5.0-2ubuntu10.1).
curl set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 34 not upgraded.
ubuntu@ip-172-31-91-172:~$ echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/l
inux/ubuntu \
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
ubuntu@ip-172-31-91-172:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:5 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Get:6 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [11.7 kB]
Fetched 60.5 kB in 1s (85.4 kB/s)
Reading package lists... Done
ubuntu@ip-172-31-91-172:~$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin dock
er-compose-plugin
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  docker-ce-rootless-extras libltdl7 libslirp0 pigz slirp4netns
Suggested packages:
```

- Installs various Docker-related packages on an Ubuntu system like docker engine, docker cli, container runtime, docker compose.

$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

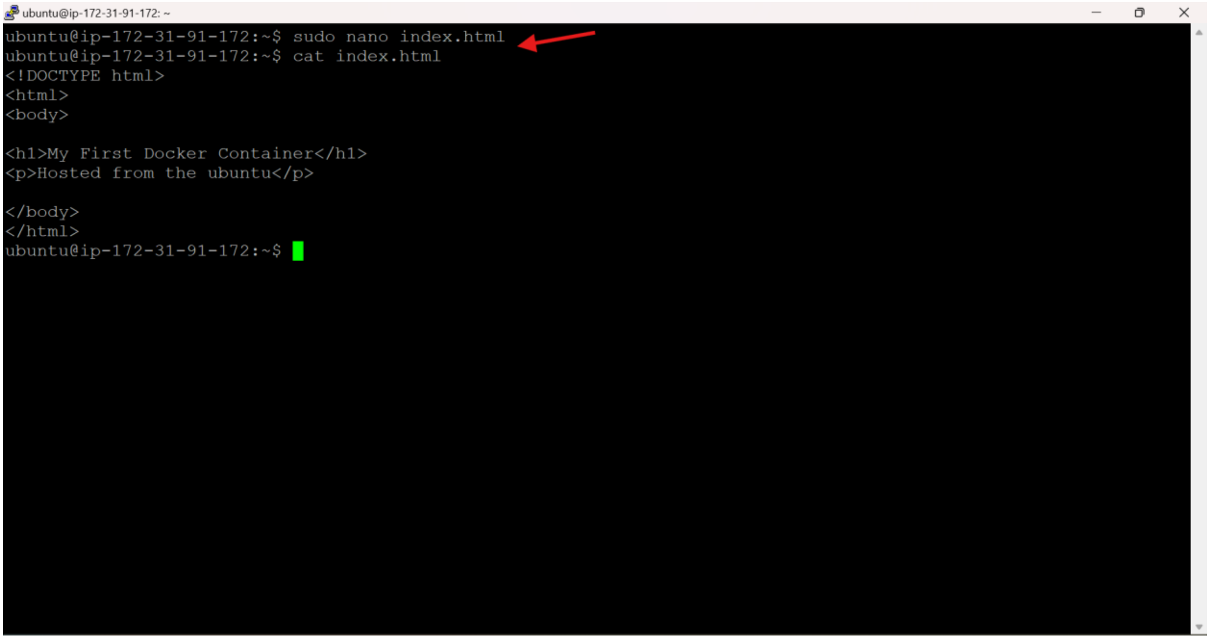
$ docker –version



```
Setting up libslirp0:amd64 (4.7.0-1ubuntu3) ...
Setting up pigz (2.8-1) ...
Setting up docker-ce-rootless-extras (5:27.1.1-1~ubuntu.24.04~noble) ...
Setting up slirp4netns (1.2.1-1build2) ...
Setting up docker-ce (5:27.1.1-1~ubuntu.24.04~noble) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.ser
vice.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-91-172:~$ docker --version
Docker version 27.1.1, build 6312585
ubuntu@ip-172-31-91-172:~$
ubuntu@ip-172-31-91-172:~$
ubuntu@ip-172-31-91-172:~$
ubuntu@ip-172-31-91-172:~$
ubuntu@ip-172-31-91-172:~$
ubuntu@ip-172-31-91-172:~$
ubuntu@ip-172-31-91-172:~$
```

Create a html sample website:

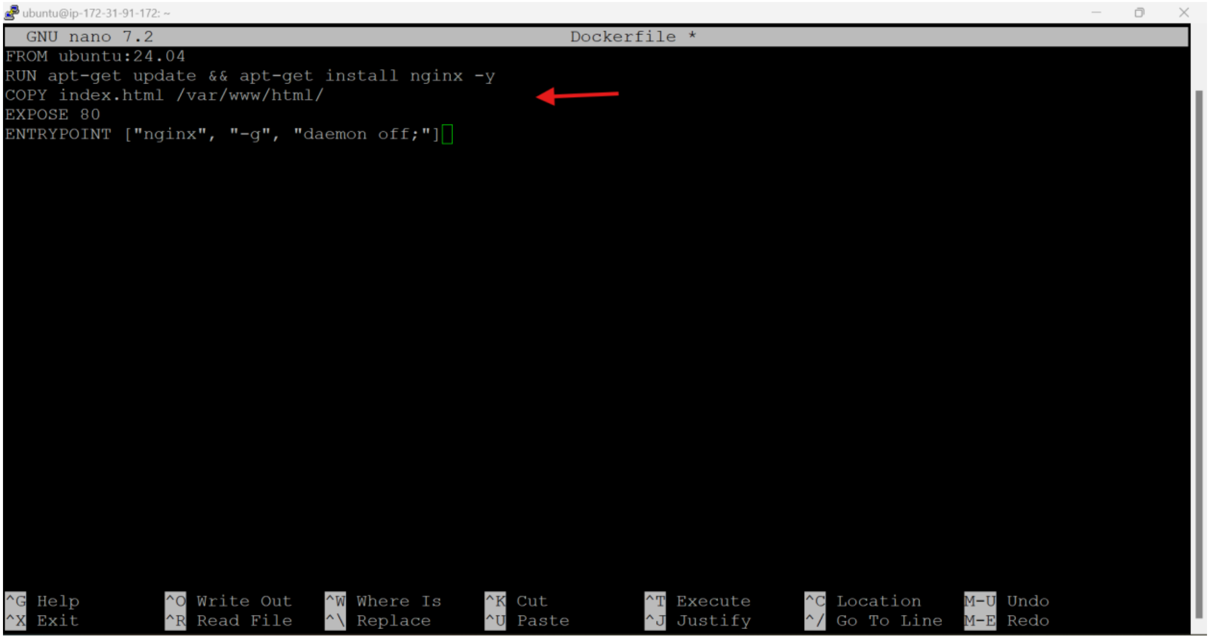
```
<!DOCTYPE html>
<html>
<body>

<h1>My First Docker container</h1>
<p>Hosted from ubuntu</p>

</body>
</html>
```



```
ubuntu@ip-172-31-91-172:~$ sudo nano index.html
ubuntu@ip-172-31-91-172:~$ cat index.html
<!DOCTYPE html>
<html>
<body>

<h1>My First Docker Container</h1>
<p>Hosted from the ubuntu</p>

</body>
</html>
ubuntu@ip-172-31-91-172:~$
```

Create an Dockerfile:

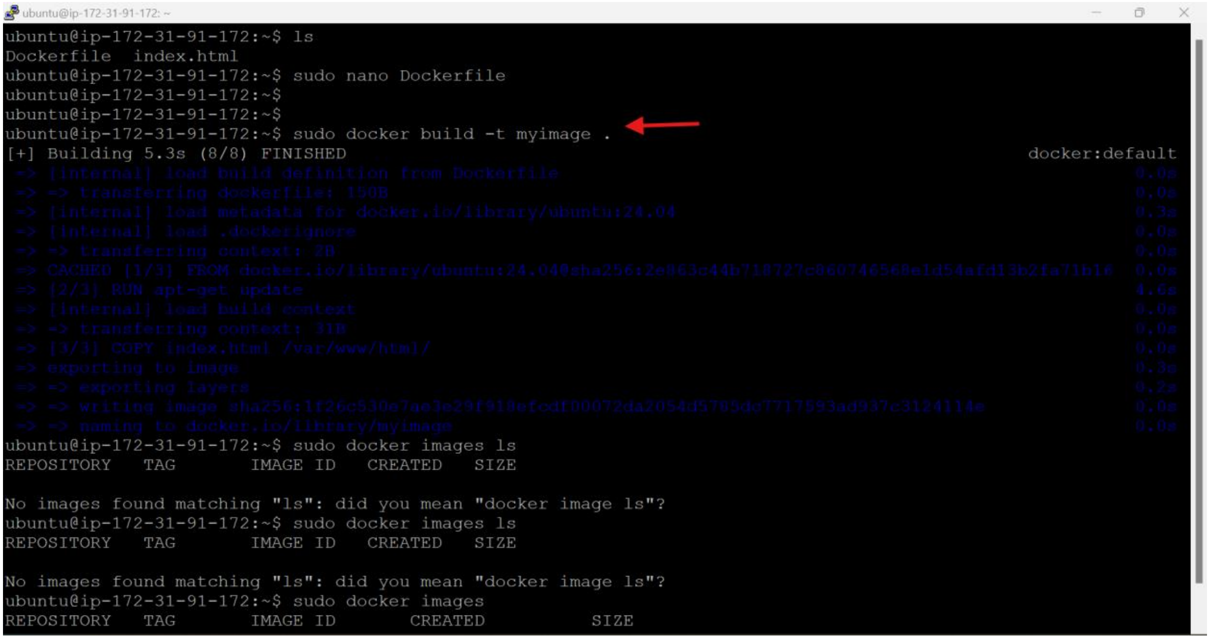
```
FROM ubuntu:24.04

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

COPY index.html /var/www/html/

EXPOSE 80

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



```
FROM ubuntu:24.04
RUN apt-get update && apt-get install nginx -y
COPY index.html /var/www/html/
EXPOSE 80
ENTRYPOINT ["nginx", "-g", "daemon off;"]
```

Build an image from the Dockerfile

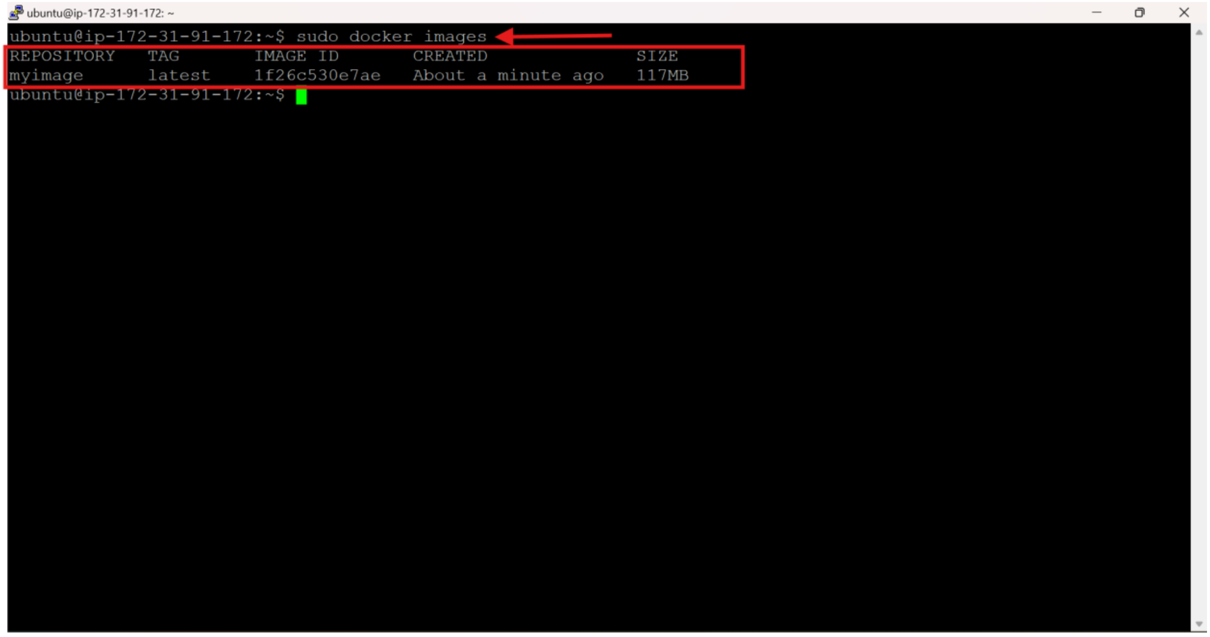
$ sudo docker build -t myimage .



```
ubuntu@ip-172-31-91-172:~$ ls
Dockerfile  index.html
ubuntu@ip-172-31-91-172:~$ sudo nano Dockerfile
ubuntu@ip-172-31-91-172:~$
ubuntu@ip-172-31-91-172:~$
ubuntu@ip-172-31-91-172:~$ sudo docker build -t myimage .
[+] Building 5.3s (8/8) FINISHED                                                  docker:default
 => [internal] load build definition from Dockerfile                                         0.0s
 => => transferring dockerfile: 150B                                                         0.0s
 => [internal] load metadata for docker.io/library/ubuntu:24.04                              0.3s
 => [internal] load .dockerignore                                                            0.0s
 => => transferring context: 2B                                                              0.0s
 => CACHED [1/3] FROM docker.io/library/ubuntu:24.04@sha256:2e863c44b718727c860746568e1d54afd13b2fa71b16  0.0s
 => [2/3] RUN apt-get update                                                                 4.6s
 => [internal] load build context                                                            0.0s
 => => transferring context: 31B                                                             0.0s
 => [3/3] COPY index.html /var/www/html/                                                     0.0s
 => exporting to image                                                                       0.3s
 => => exporting layers                                                                      0.2s
 => => writing image sha256:1f26c530e7ae3e29f918efcdf00072da2054d5785dc7717593ad937c3124114e  0.0s
 => => naming to docker.io/library/myimage                                                   0.0s
ubuntu@ip-172-31-91-172:~$ sudo docker images ls
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE

No images found matching "ls": did you mean "docker image ls"?
ubuntu@ip-172-31-91-172:~$ sudo docker images ls
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE

No images found matching "ls": did you mean "docker image ls"?
ubuntu@ip-172-31-91-172:~$ sudo docker images
REPOSITORY   TAG       IMAGE ID       CREATED          SIZE
```

List the images created:

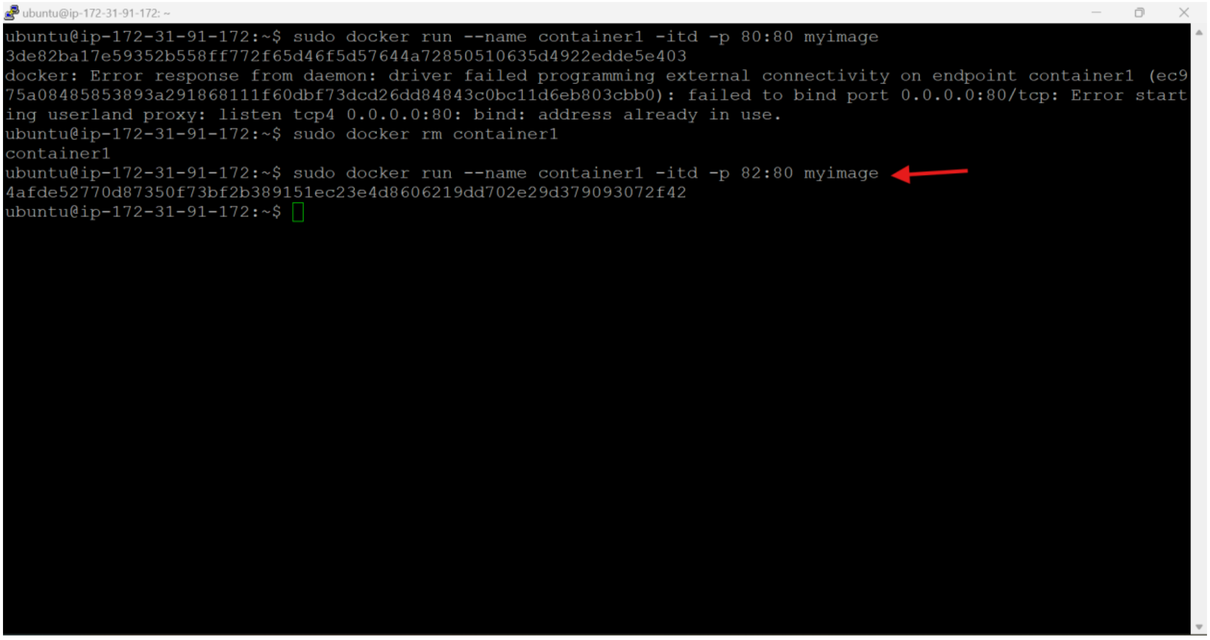
$ sudo docker images



```
ubuntu@ip-172-31-91-172:~$ sudo docker images
REPOSITORY    TAG       IMAGE ID       CREATED              SIZE
myimage       latest    1f26c530e7ae   About a minute ago   117MB
ubuntu@ip-172-31-91-172:~$
```

Create a container from the image created

$ sudo docker run -itd -p 82:80 myimage

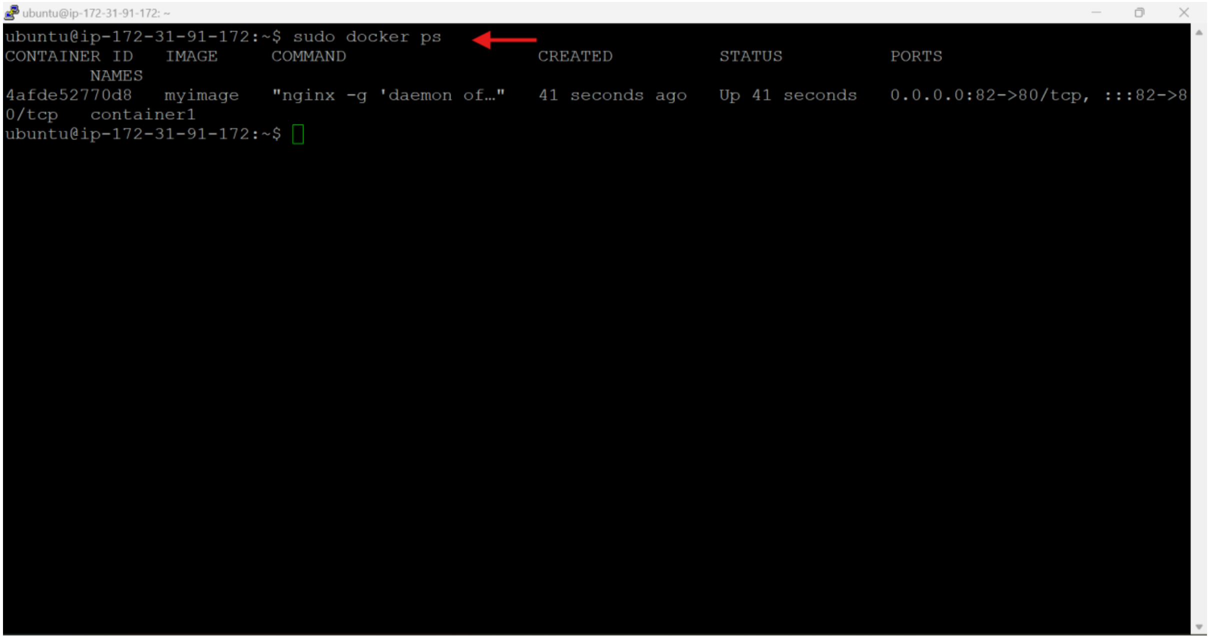


```
ubuntu@ip-172-31-91-172:~$ sudo docker run --name container1 -itd -p 80:80 myimage
3de82ba17e59352b558ff772f65d46f5d57644a72850510635d4922edde5e403
docker: Error response from daemon: driver failed programming external connectivity on endpoint container1 (ec9
75a08485853893a291868111f60dbf73dcd26dd84843c0bc11d6eb803cbb0): failed to bind port 0.0.0.0:80/tcp: Error start
ing userland proxy: listen tcp4 0.0.0.0:80: bind: address already in use.
ubuntu@ip-172-31-91-172:~$ sudo docker rm container1
container1
ubuntu@ip-172-31-91-172:~$ sudo docker run --name container1 -itd -p 82:80 myimage
4afde52770d87350f73bf2b389151ec23e4d8606219dd702e29d379093072f42
ubuntu@ip-172-31-91-172:~$
```

Container has been created!

List the container running

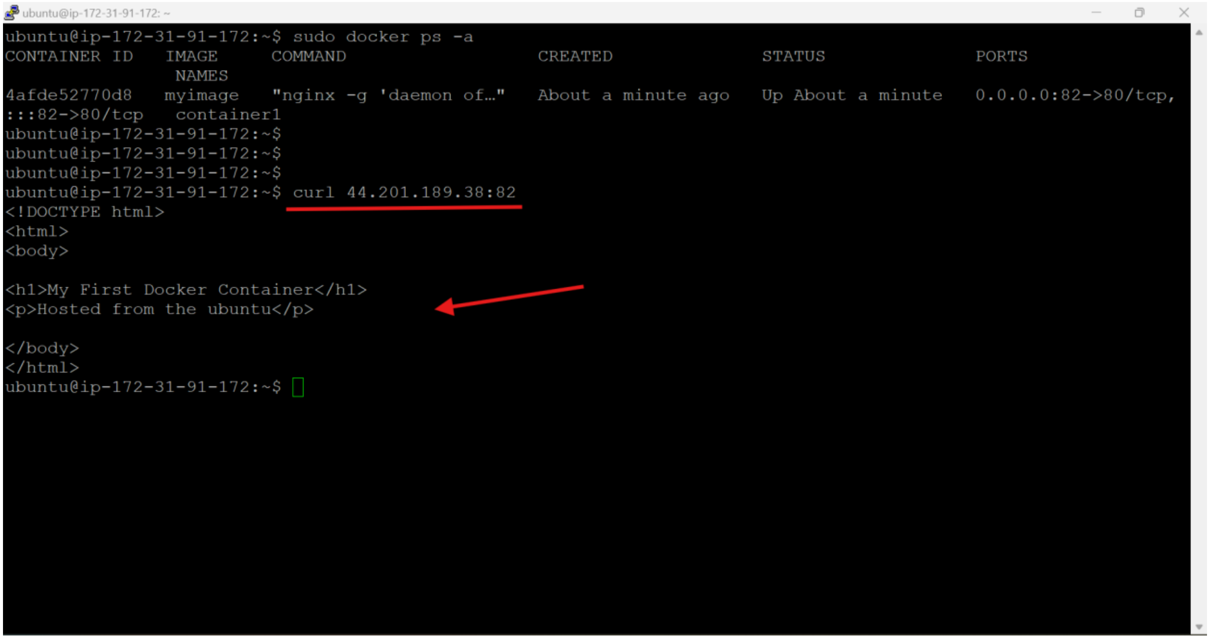
$ sudo docker ps



```
ubuntu@ip-172-31-91-172:~$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS
        NAMES
4afde52770d8   myimage   "nginx -g 'daemon of…"   41 seconds ago   Up 41 seconds   0.0.0.0:82->80/tcp, :::82->8
0/tcp   container1
ubuntu@ip-172-31-91-172:~$
```

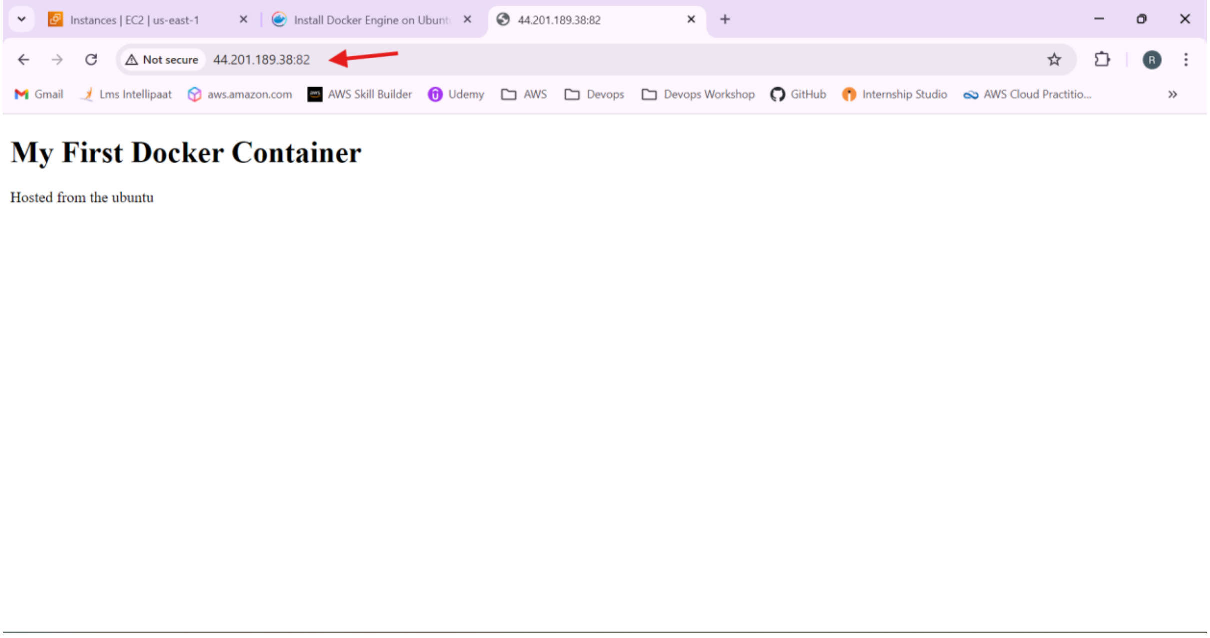
Test the port with curl

$ curl localhost:82



```
ubuntu@ip-172-31-91-172:~$ sudo docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED              STATUS              PORTS
                NAMES
4afde52770d8   myimage   "nginx -g 'daemon of…"   About a minute ago   Up About a minute   0.0.0.0:82->80/tcp,
:::82->80/tcp   container1
ubuntu@ip-172-31-91-172:~$
ubuntu@ip-172-31-91-172:~$
ubuntu@ip-172-31-91-172:~$
ubuntu@ip-172-31-91-172:~$ curl 44.201.189.38:82
<!DOCTYPE html>
<html>
<body>

<h1>My First Docker Container</h1>
<p>Hosted from the ubuntu</p>

</body>
</html>
ubuntu@ip-172-31-91-172:~$
```

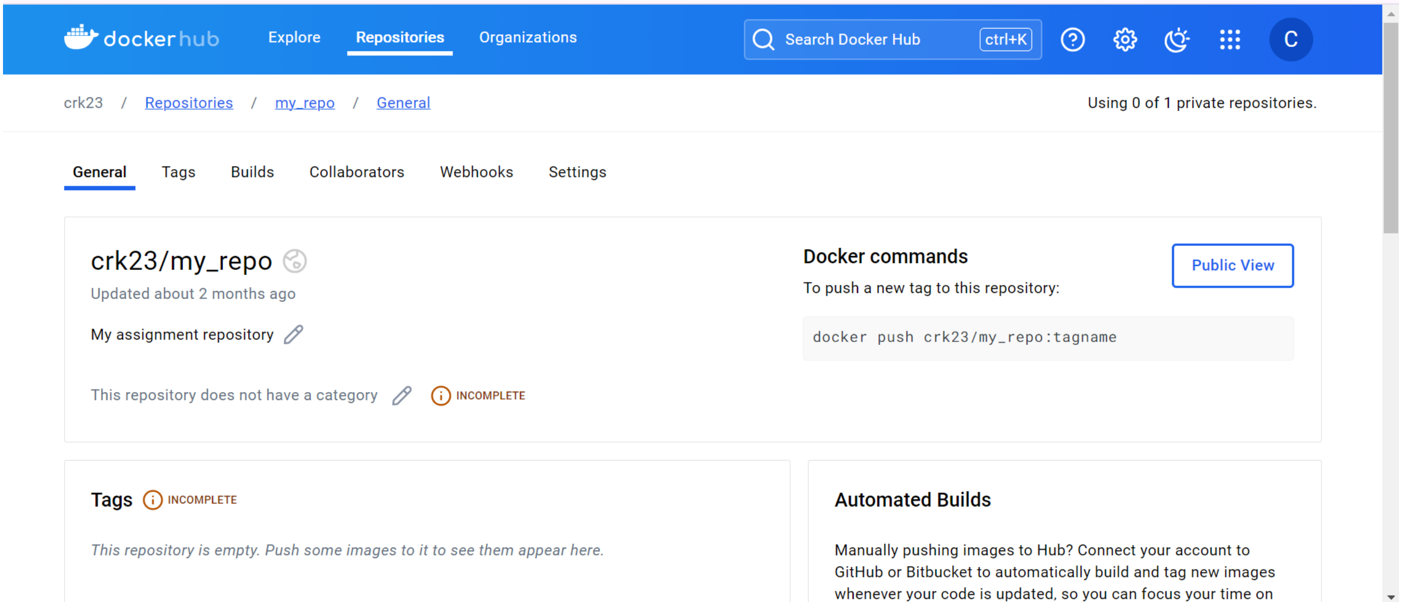
Go to the browser and search localhost:82 an we can see the html contents as shown below:



My First Docker Container

Hosted from the ubuntu

- Login to the Dockerhub



Using 0 of 1 private repositories.

crk23/my_repo

Updated about 2 months ago

My assignment repository

This repository does not have a category INCOMPLETE

Docker commands

To push a new tag to this repository:

```
docker push crk23/my_repo:tagname
```

Tags INCOMPLETE

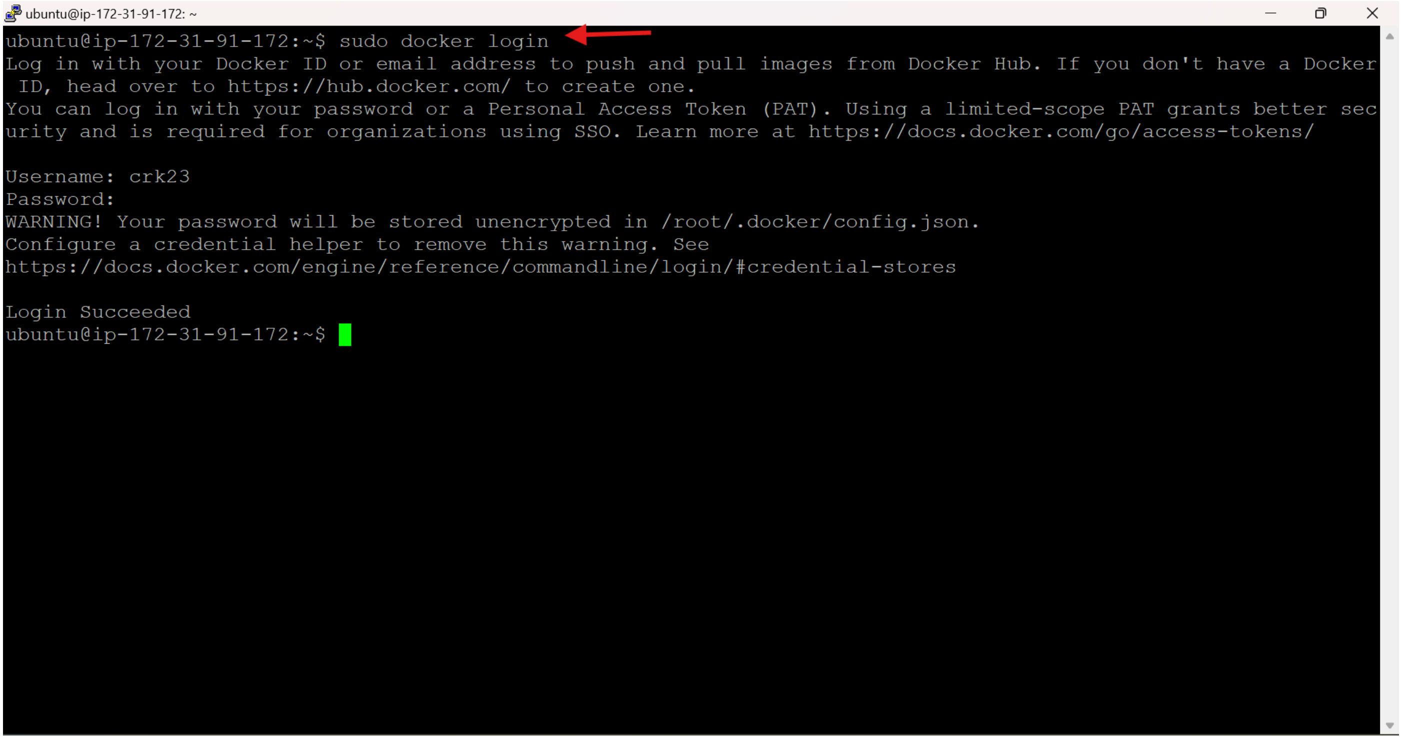
This repository is empty. Push some images to it to see them appear here.

Automated Builds

Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on

Login to Docker hub from the terminal to authenticate a user with Docker registries like dockerhub to manage images

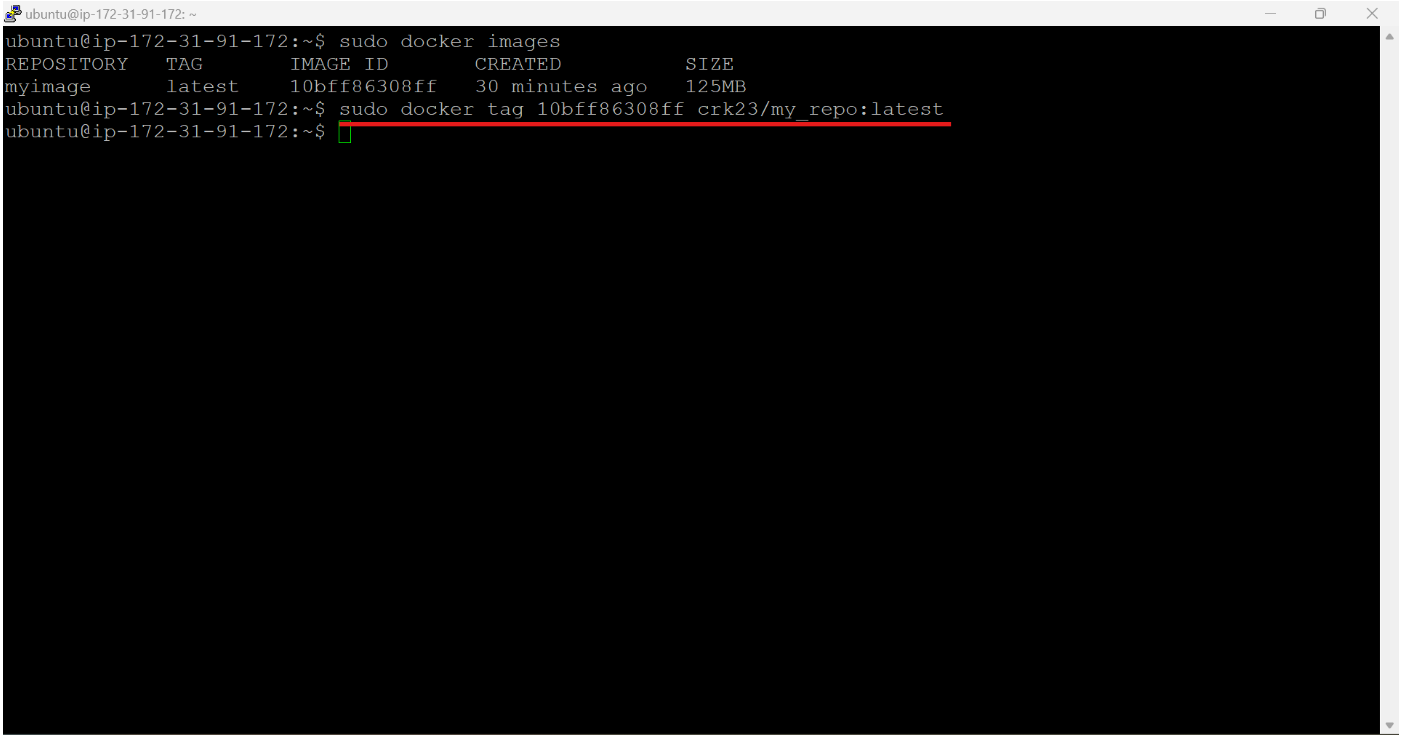
$ sudo docker login



```
ubuntu@ip-172-31-91-172:~$ 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 create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants better sec
urity and is required for organizations using SSO. Learn more at https://docs.docker.com/go/access-tokens/

Username: crk23
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/#credential-stores

Login Succeeded
ubuntu@ip-172-31-91-172:~$
```

Tag the created image (Tag images before pushing them to Docker Hub for Identifying Versions, Naming Convention, Pushing to Registry, Pulling Specific Versions)

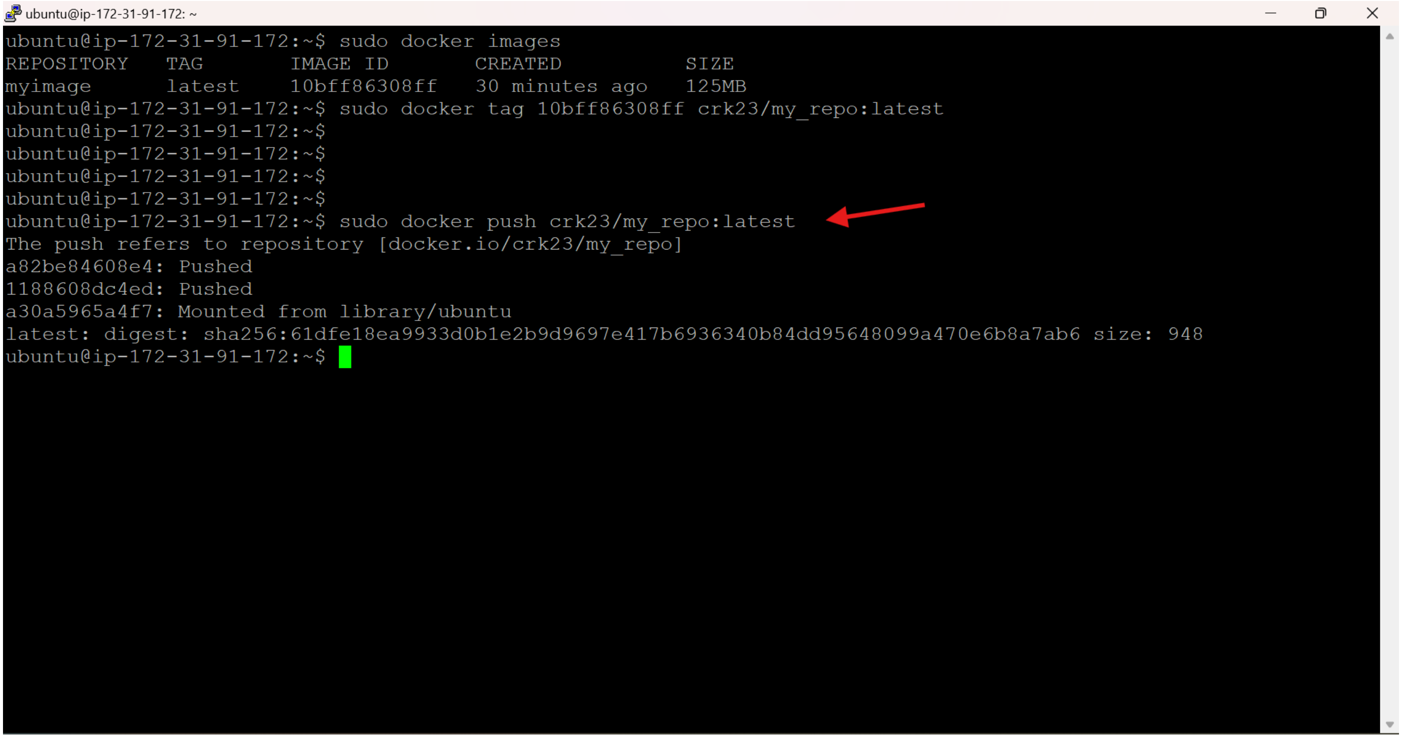
$ sudo docker tag <image id> <dockerhub username>/<Repository name>:<Name to be given for tag>



```
ubuntu@ip-172-31-91-172:~$ sudo docker images
REPOSITORY   TAG       IMAGE ID       CREATED          SIZE
myimage      latest    10bff86308ff   30 minutes ago   125MB
ubuntu@ip-172-31-91-172:~$ sudo docker tag 10bff86308ff crk23/my_repo:latest
ubuntu@ip-172-31-91-172:~$
```

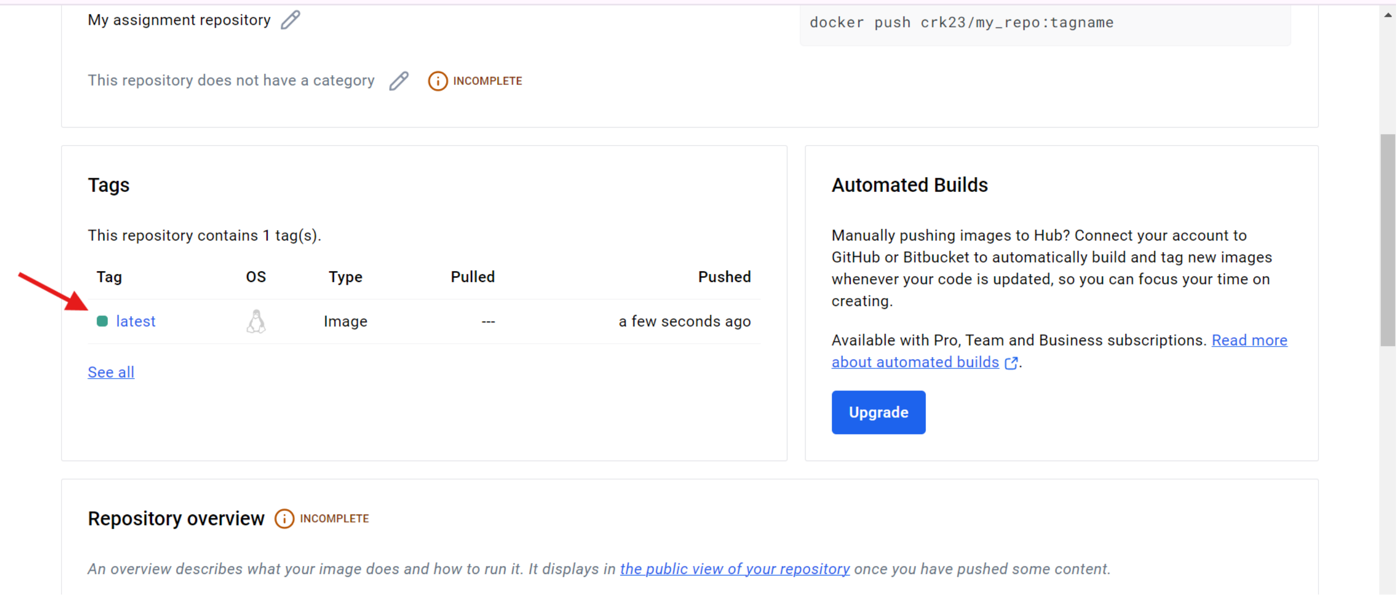
Push to the Dockerhub

$ sudo docker push <dockerhub username>/<Repository name>:<tag name>



```
ubuntu@ip-172-31-91-172:~$ sudo docker images
REPOSITORY   TAG       IMAGE ID       CREATED          SIZE
myimage      latest    10bff86308ff   30 minutes ago   125MB
ubuntu@ip-172-31-91-172:~$ sudo docker tag 10bff86308ff crk23/my_repo:latest
ubuntu@ip-172-31-91-172:~$
ubuntu@ip-172-31-91-172:~$
ubuntu@ip-172-31-91-172:~$
ubuntu@ip-172-31-91-172:~$
ubuntu@ip-172-31-91-172:~$ sudo docker push crk23/my_repo:latest
The push refers to repository [docker.io/crk23/my_repo]
a82be84608e4: Pushed
1188608dc4ed: Pushed
a30a5965a4f7: Mounted from library/ubuntu
latest: digest: sha256:61dfe18ea9933d0b1e2b9d9697e417b6936340b84dd95648099a470e6b8a7ab6 size: 948
ubuntu@ip-172-31-91-172:~$
```

Image added to the Dockerhub registry successfully!



My assignment repository

```
docker push crk23/my_repo:tagname
```

This repository does not have a category

Tags

This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
latest		Image	---	a few seconds ago

See all

Automated Builds

Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available with Pro, Team and Business subscriptions. Read more about automated builds.

Repository overview

An overview describes what your image does and how to run it. It displays in the public view of your repository once you have pushed some content.