Day 17: Docker Project for DevOps Engineers.

Dockerfile

Docker is a tool that makes it easy to run applications in containers. Containers are like small packages that hold everything an application needs to run. To create these containers, developers use something called a Dockerfile.

A Dockerfile is like a set of instructions for making a container. It tells Docker what base image to use, what commands to run, and what files to include. For example, if you were making a container for a website, the Dockerfile might tell Docker to use an official web server image, copy the files for your website into the container, and start the web server when the container starts.

For more about Dockerfile visit here

Task:

- Create a Dockerfile for a simple web application (e.g. a Node.js or Python app)

Dockerfile is nothig but a set of commands to execute or automate the application deployments

Follow below steps to deploy flask app to DockerHub

Launch AWS ubuntu ec2 instance

Ref:- https://medium.com/@misalPav103/deploying-nodejs-app-on-aws-ec2-instance-step-by-step-1b00f807cdce

After That Update the system and install docker, using below command

sudo apt-get update && sudo apt-get install docker.io

After successfully installed docker, clone the git repository

```
ubuntu@ip-172-31-90-57:~$ git clone https://github.com/mmumshad/simple-webapp-color.git
Cloning into 'simple-webapp-color'...
remote: Enumerating objects: 15, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 15 (delta 4), reused 4 (delta 4), pack-reused 9
Receiving objects: 100% (15/15), done.
Resolving deltas: 100% (5/5), done.
ubuntu@ip-172-31-90-57:~$ sudo apt-get install docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-doc rinse zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io pigz runc ubuntu-fan
0 upgraded, 8 newly installed, 0 to remove and 6 not upgraded.
Need to get 66.8 MB of archives.
After this operation, 287 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 pigz amd64 2.6-1 [63.6 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 bridge-utils amd64 1.7-1ubuntu3 [34.4 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 runc amd64 1.1.0-0ubuntu1.1 [4242 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 containerd amd64 1.5.9-0ubuntu3.1 [28.1 MB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 dns-root-data all 2021011101 [5256 B]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 dnsmasq-base amd64 2.86-1.1ubuntu0.1 [354 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 docker.io amd64 20.10.12-0ubuntu4 [34.0 MB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 ubuntu-fan all 0.12.16 [35.2 kB]
Fetched 66.8 MB in 1s (44.9 MB/s)
Preconfiguring packages ...
Selecting previously unselected package pigz.
(Reading database ... 63567 files and directories currently installed.)
Preparing to unpack .../0-pigz_2.6-1_amd64.deb ...
Unpacking pigz (2.6-1) ...
Selecting previously unselected package bridge-utils.
Preparing to unpack .../1-bridge-utils_1.7-1ubuntu3_amd64.deb ...
Unpacking bridge-utils (1.7-1ubuntu3) ...
```

After that you create a Dockerfile

```
ubuntu@ip-172-31-90-57:~/simple-webapp-color$ cat Dockerfile
FROM python:latest
WORKDIR /app/
COPY . /app/
RUN pip install -r requirements.txt
ENTRYPOINT FLASK_APP=/app/app.py flask run --host=0.0.0.0 --port=9001
```

FROM python: latest

fetching the latest python image from docker hub

WORKDIR /app/

now we create a directory to hold application code inside the image. This is current working directory of image

COPY . /app/

Copy all local application data to image, inside /app/ folder

RUN pip install -r requirements.txt

Using Run Command, we have to install some dependencies. All dependencies are stored inside requirements.txt.

Using Run Command, we install all the dependencies recursively.

ENTRYPOINT FLASK_APP=/app/app.py flask run --host=0.0.0.0 --port=9001

ENTRYPOINT is a command is responsible for execute a command, allocate port number to application, allocate host ip, etc...

- Build the image using the Dockerfile and run the container

Before building dockerfile,Run the below command to set current user All permissions of executing the Dockerfile.

sudo –a –G docker $USER && sudo reboot

After rebooting, again create a connection

To Build a image using Dockerfile.

docker build . -t new-flask-image

```
ubuntu@ip-172-31-90-57:~/simple-webapp-color$ docker build . -t new-flask-image
Sending build context to Docker daemon  65.02kB
Step 1/5 : FROM python:latest
latest: Pulling from library/python
bbeef03cda1f: Pull complete
f049f75f014e: Pull complete
56261d0e6b05: Pull complete
9bd150679dbd: Pull complete
5b282ee9da04: Pull complete
03f027d5e312: Pull complete
db6ee1ace097: Pull complete
0a86d528f1ea: Pull complete
4cfb032ae58b: Pull complete
Digest: sha256:a3c0c6766535f85f18e7304d3a0111de5208d73935bcf1b024217005ad5ce195
Status: Downloaded newer image for python:latest
 ---> b44268c8cbc0
Step 2/5 : WORKDIR /app/
 ---> Running in 0cc6394ad70a
Removing intermediate container 0cc6394ad70a
 ---> a5a5a5001c0f
Step 3/5 : COPY . /app/
 ---> 8cf37cc52939
Step 4/5 : RUN pip install -r requirements.txt
 ---> Running in 767db248f7a5
Collecting Flask
  Downloading Flask-2.2.2-py3-none-any.whl (101 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 101.5/101.5 kB 10.7 MB/s eta 0:00:00
Collecting Werkzeug>=2.2.2
  Downloading Werkzeug-2.2.2-py3-none-any.whl (232 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 232.7/232.7 kB 4.4 MB/s eta 0:00:00
Collecting Jinja2>=3.0
  Downloading Jinja2-3.1.2-py3-none-any.whl (133 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 133.1/133.1 kB 1.7 MB/s eta 0:00:00
Collecting itsdangerous>=2.0
  Downloading itsdangerous-2.1.2-py3-none-any.whl (15 kB)
Collecting click>=8.0
  Downloading click-8.1.3-py3-none-any.whl (96 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 96.6/96.6 kB 2.0 MB/s eta 0:00:00
Collecting MarkupSafe>=2.0
  Downloading MarkupSafe-2.1.2-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (27 kB)
Installing collected packages: MarkupSafe, itsdangerous, click, Werkzeug, Jinja2, Flask
```

After Building a image Run this image to create a container.

docker run -d -p 9001:9001 --name new-flask-ctr new-flask-image: latest

```
ubuntu@ip-172-31-90-57:~/simple-webapp-color$ docker images
REPOSITORY        TAG       IMAGE ID       CREATED          SIZE
new-flask-image   latest    8b061de05342   9 seconds ago    948MB
python            latest    b44268c8cbc0   27 hours ago     932MB
ubuntu@ip-172-31-90-57:~/simple-webapp-color$ docker run -d -p 9001:9001 --name new-flask-ctr new-flask-image:latest
467beab6aeddbca7a8a4474bf59886d929c677a1ddbecc0a5612407ca77970d1
ubuntu@ip-172-31-90-57:~/simple-webapp-color$ docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED         STATUS         PORTS                                                NAMES
467beab6aedd   new-flask-image:latest   "/bin/sh -c 'FLASK_A…"   4 seconds ago   Up 3 seconds   0.0.0.0:9001->9001/tcp, :::9001->9001/tcp   new-flask-ctr
```

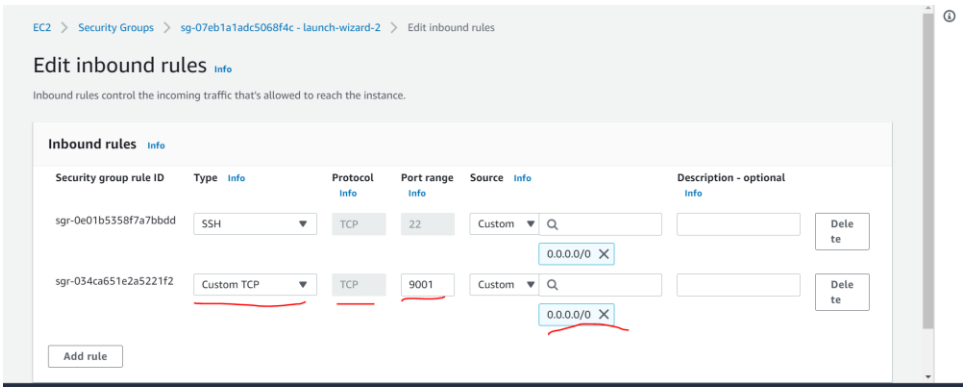
- Verify that the application is working as expected by accessing it in a web browser

Now, container is running, now you go to ec2 instance page, and add a 9001 port in Security Group.

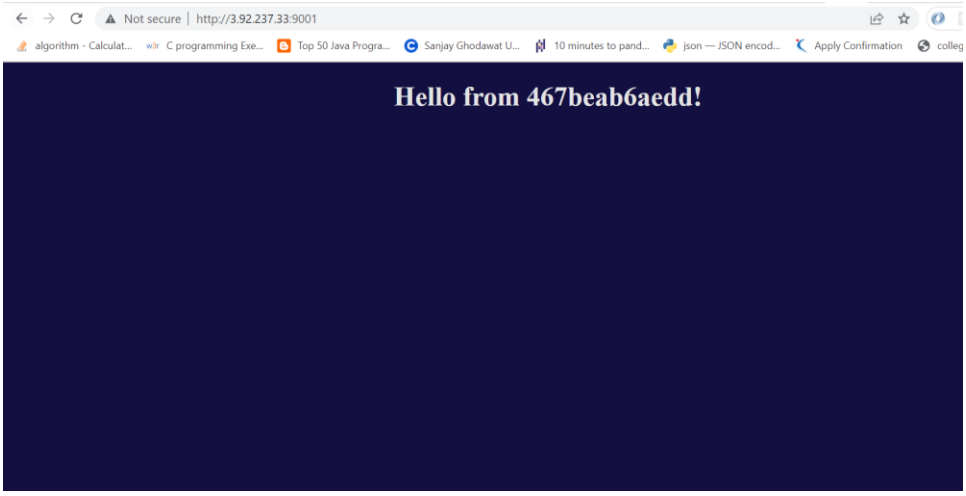
1] select ec2 instance checkbox and click security tab

2] click on security group

3]inside inbound rules add your mapped port number and select custom ip.



Open the public IPv4 address of your EC2 instance with port.no. 9001



- Push the image to a public or private repository (e.g., Docker Hub)

Before directly Pushing the image into DockerHub,first login using DockerHub account credentials.

docker login

-add your username and password

After that rename or tag your image

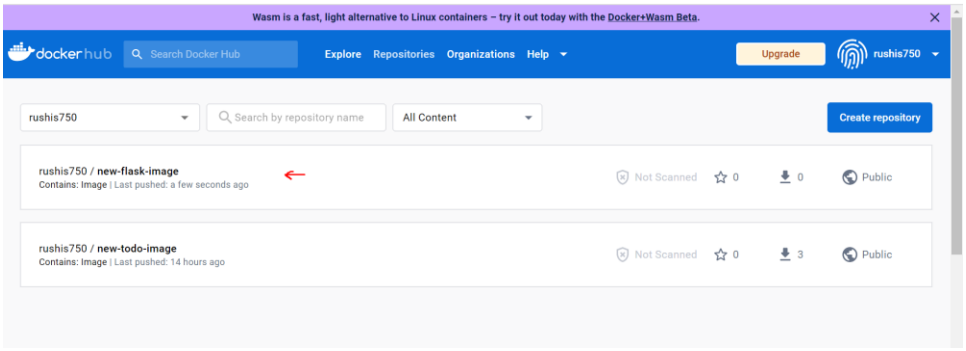
docker image tag <your image name> <dockerhub username>/<image Name>

Push this latest image to DockerHub

docker push <latest image name>

```
ubuntu@ip-172-31-90-57:~/simple-webapp-color$ docker login
Login with your Docker ID 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.
Username: rushis750
Password:
WARNING! Your password will be stored unencrypted in /home/ubuntu/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
ubuntu@ip-172-31-90-57:~/simple-webapp-color$ docker image tag new-flask-image:latest rushis750/new-flask-image:latest
ubuntu@ip-172-31-90-57:~/simple-webapp-color$ docker images
REPOSITORY                  TAG       IMAGE ID       CREATED          SIZE
new-flask-image             latest    8b061de05342   9 minutes ago    948MB
rushis750/new-flask-image   latest    8b061de05342   9 minutes ago    948MB
python                      latest    b44268c8cbc0   27 hours ago     932MB
ubuntu@ip-172-31-90-57:~/simple-webapp-color$ docker push rushis750/new-flask-image:latest
The push refers to repository [docker.io/rushis750/new-flask-image]
2dd60eb01da3: Pushed
c997b7c1a92a: Pushed
e79a422a1acf: Pushed
c110d2a211c2: Mounted from library/python
f0516dcd8c5f: Mounted from library/python
31c905cb9fd7: Mounted from library/python
dc6462f7bb8b: Mounted from library/python
a4db1a405763: Mounted from library/python
9f4f964da727: Mounted from library/python
49b333f7bad4: Mounted from library/python
a463dbda4664: Mounted from library/python
a9099c3159f5: Mounted from library/python
latest: digest: sha256:72593f8d61703bb0d793b5bdaacecfb979224b398ed5c2fa9175da75d2abff79 size: 2844
ubuntu@ip-172-31-90-57:~/simple-webapp-color$
```



- ----------Happy Learning😊----------