D Y PATIL INTERNATIONAL UNIVERSITY
AKURDI PUNE

School of Computer Science Engineering and Application

BCA TY SEM VI

Subject Name: Container and Orchestration Practical

Assignment No 8

Aim: Build Image with two dependencies (Flask, Reddis) and create container with 5 replicas with docker stack

Submitted By

Name: Muskan Jakir Shaikh

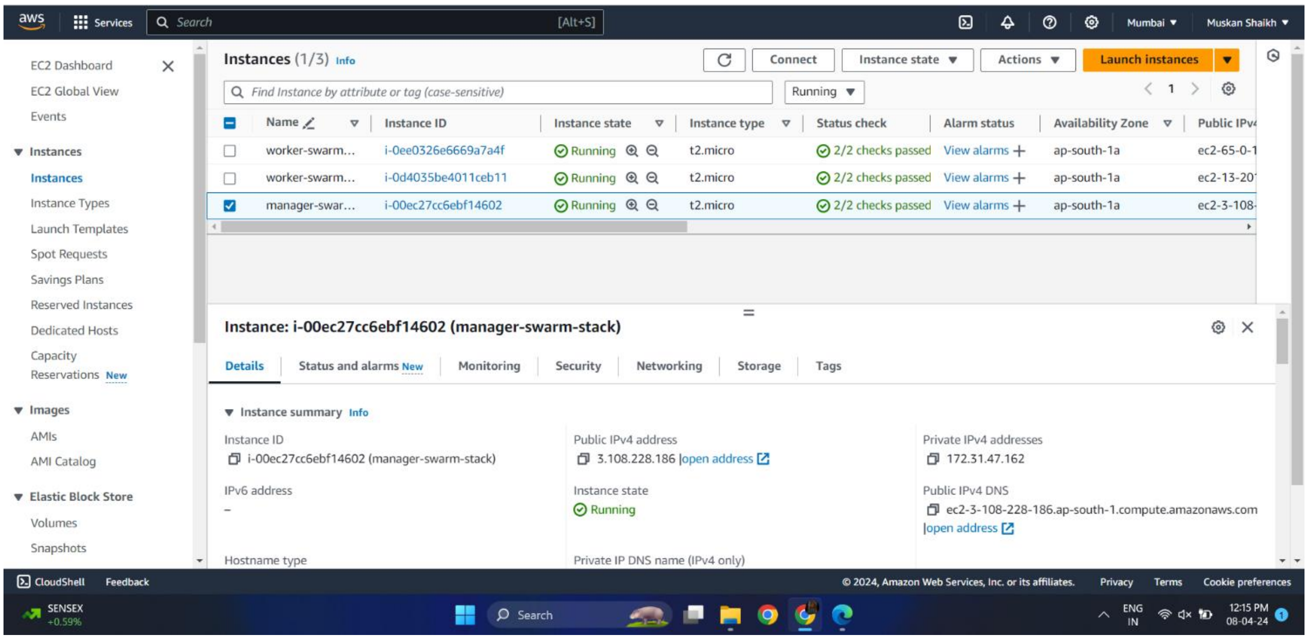
PRN: 20210801020

Date: (15-04-2024)

Aim: Build Image with two dependencies (Flask, Reddis) and create container with 5 replicas with docker stack

Technology Used: Docker, Container, AWS

Step1: Create three instances one for manager node and another two for worker node.



Instances (1/3)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
worker-swarm...	i-0ee0326e6669a7a4f	Running	t2.micro	2/2 checks passed	View alarms	ap-south-1a	ec2-65-0-1
worker-swarm...	i-0d4035be4011ceb11	Running	t2.micro	2/2 checks passed	View alarms	ap-south-1a	ec2-13-20
manager-swar...	i-00ec27cc6ebf14602	Running	t2.micro	2/2 checks passed	View alarms	ap-south-1a	ec2-3-108-

Instance: i-00ec27cc6ebf14602 (manager-swarm-stack)

Instance summary

Instance ID: i-00ec27cc6ebf14602 (manager-swarm-stack)

Public IPv4 address: 3.108.228.186 | open address

Private IPv4 addresses: 172.31.47.162

IPv6 address: –

Instance state: Running

Public IPv4 DNS: ec2-3-108-228-186.ap-south-1.compute.amazonaws.com | open address

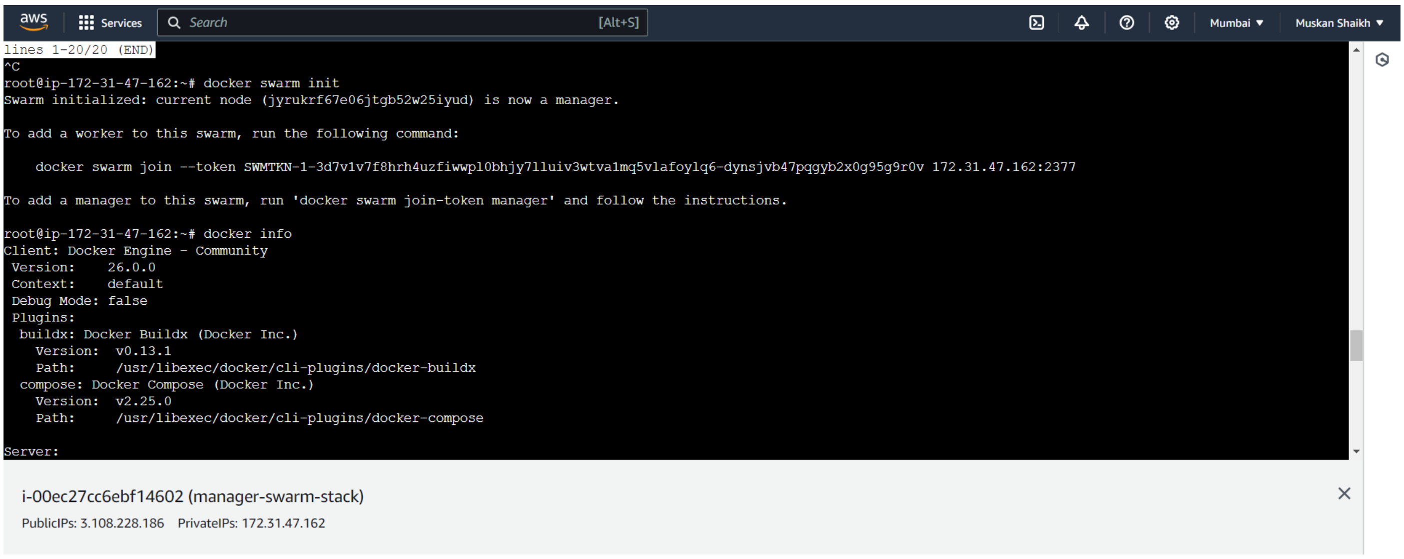
Hostname type

Private IP DNS name (IPv4 only)

Step 2: Launch the manager instance and install docker and run it in swarm mode.

#docker swarm init

#docker info



```
lines 1-20/20 (END)
^C
root@ip-172-31-47-162:~# docker swarm init
Swarm initialized: current node (jyrukrf67e06jtgb52w25iyud) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-3d7v1v7f8hrh4uzfiwwpl0bhjy7lluiv3wtva1mq5vlafoylq6-dynsjvb47pqgyb2x0g95g9r0v 172.31.47.162:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.

root@ip-172-31-47-162:~# docker info
Client: Docker Engine - Community
 Version:    26.0.0
 Context:    default
 Debug Mode: false
 Plugins:
  buildx: Docker Buildx (Docker Inc.)
    Version:  v0.13.1
    Path:     /usr/libexec/docker/cli-plugins/docker-buildx
  compose: Docker Compose (Docker Inc.)
    Version:  v2.25.0
    Path:     /usr/libexec/docker/cli-plugins/docker-compose

Server:
```

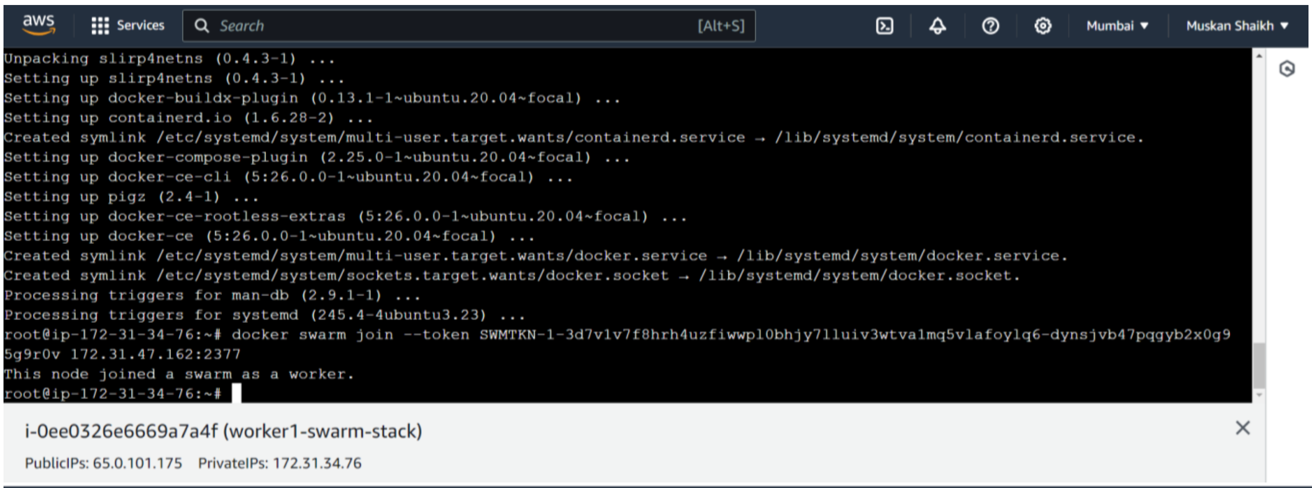
i-00ec27cc6ebf14602 (manager-swarm-stack)

PublicIPs: 3.108.228.186 PrivateIPs: 172.31.47.162

Step 3: Install docker and join the 2 worker nodes to the swarm of the manager node

Run the command produced by the docker swarm init output in the worker node

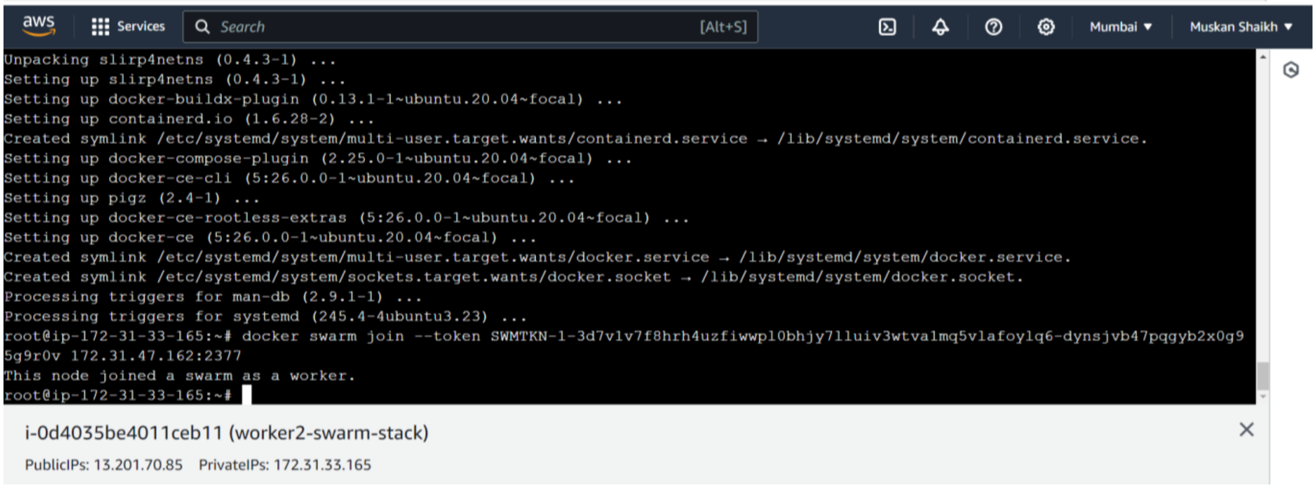
#docker swarm join \ --token SWMTKN-1- 49nj1cmql0jkz5s954yi3oex3nedyz0fb0xx14ie39trti4wxv-8vxv8rssmk743ojnwacrr2e7c \ 192.168.99.100:2377



```
Unpacking slirp4netns (0.4.3-1) ...
Setting up slirp4netns (0.4.3-1) ...
Setting up docker-buildx-plugin (0.13.1-1~ubuntu.20.04~focal) ...
Setting up containerd.io (1.6.28-2) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /lib/systemd/system/containerd.service.
Setting up docker-compose-plugin (2.25.0-1~ubuntu.20.04~focal) ...
Setting up docker-ce-cli (5:26.0.0-1~ubuntu.20.04~focal) ...
Setting up pigz (2.4-1) ...
Setting up docker-ce-rootless-extras (5:26.0.0-1~ubuntu.20.04~focal) ...
Setting up docker-ce (5:26.0.0-1~ubuntu.20.04~focal) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for systemd (245.4-4ubuntu3.23) ...
root@ip-172-31-34-76:~# docker swarm join --token SWMTKN-1-3d7v1v7f8hrh4uzfiwwpl0bhjy7lluiv3wtva1mq5vlafoylq6-dynsjvb47pqgyb2x0g9
5g9r0v 172.31.47.162:2377
This node joined a swarm as a worker.
root@ip-172-31-34-76:~#
```

i-0ee0326e6669a7a4f (worker1-swarm-stack)

PublicIPs: 65.0.101.175 PrivateIPs: 172.31.34.76



```
Unpacking slirp4netns (0.4.3-1) ...
Setting up slirp4netns (0.4.3-1) ...
Setting up docker-buildx-plugin (0.13.1-1~ubuntu.20.04~focal) ...
Setting up containerd.io (1.6.28-2) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /lib/systemd/system/containerd.service.
Setting up docker-compose-plugin (2.25.0-1~ubuntu.20.04~focal) ...
Setting up docker-ce-cli (5:26.0.0-1~ubuntu.20.04~focal) ...
Setting up pigz (2.4-1) ...
Setting up docker-ce-rootless-extras (5:26.0.0-1~ubuntu.20.04~focal) ...
Setting up docker-ce (5:26.0.0-1~ubuntu.20.04~focal) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for systemd (245.4-4ubuntu3.23) ...
root@ip-172-31-33-165:~# docker swarm join --token SWMTKN-1-3d7v1v7f8hrh4uzfiwwpl0bhjy7lluiv3wtva1mq5vlafoylq6-dynsjvb47pqgyb2x0g9
5g9r0v 172.31.47.162:2377
This node joined a swarm as a worker.
root@ip-172-31-33-165:~#
```

i-0d4035be4011ceb11 (worker2-swarm-stack)

PublicIPs: 13.201.70.85 PrivateIPs: 172.31.33.165

Step 4: Create a directory for the project in the manager node:

mkdir demoproj

cd demoproj

Step 5: Create a file called app.py in your project directory and paste the following code:

from flask import Flask

from redis import Redis, RedisError

import os

```
import socket

#Connect to Redis

redis = Redis(host=”redis” , db=0, socket_connect_timeout=2, socket_timeout=2)

app = Flask(__name__)

@app.route('/')

def hello():

    try:

        visits = redis.incr(“counter”)

    except RedisError:

        visits = “<i>cannot connect to Redis, counter disabled</i>”

    html = “<h3>Hello {name} ! </h3>” \

             “<b>Hostname:</b> {hostname}<br/>” \

             “<b>Visits:</b> {visits}”

   return html.format (name=os.getenv(“NAME”, “world”), hostname=socket.gethostname(),
visits=visits)

if __name__ == ”__main__”:

          app.run(host=”0.0.0.0”, port=80)
```

Step 6: Create a file called requirements.txt and paste the following code:

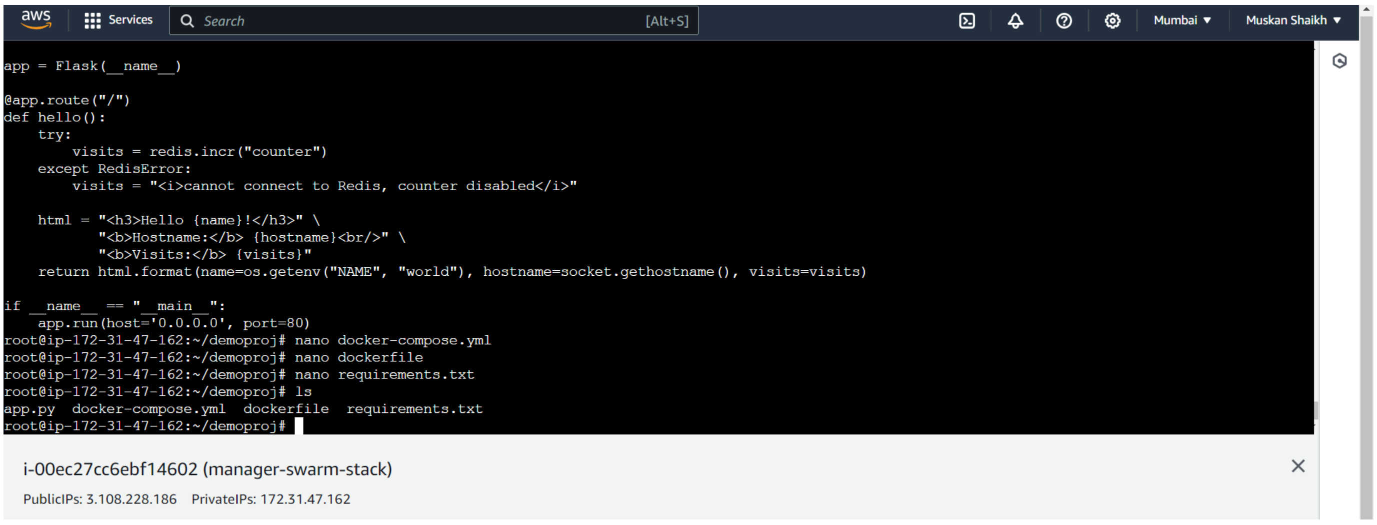
```
Flask

Redis
```

Step 7: Create Dockerfile and paste the following code:

```
FROM python:3.12-slim

WORKDIR / app

COPY ./app

RUN pip install –trusted-host pypi.python.org -r requirements.txt

EXPOSE 80

ENV NAME World

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

Step 8: Create a file called docker-compose.yml and paste the following:

```
version: “3”
services:
 web:
       #image: dockerhubusername/repo:tag
       image: muskanshaikh10/web_app:1.0
 deploy:
       replicas: 5
       resources:
              limits:
                     cpus: “0.1”
                     memory: 50M
       restart_policy:
              condition: on_failure
  ports:
   - "4000:80 "
  networks:
       -webnet
networks:
       webnet:
```



```
app = Flask(__name__)

@app.route("/")
def hello():
    try:
        visits = redis.incr("counter")
    except RedisError:
        visits = "<i>cannot connect to Redis, counter disabled</i>"

    html = "<h3>Hello {name}!</h3>" \
           "<b>Hostname:</b> {hostname}<br/>" \
           "<b>Visits:</b> {visits}"
    return html.format(name=os.getenv("NAME", "world"), hostname=socket.gethostname(), visits=visits)

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=80)
root@ip-172-31-47-162:~/demoproj# nano docker-compose.yml
root@ip-172-31-47-162:~/demoproj# nano dockerfile
root@ip-172-31-47-162:~/demoproj# nano requirements.txt
root@ip-172-31-47-162:~/demoproj# ls
app.py  docker-compose.yml  dockerfile  requirements.txt
root@ip-172-31-47-162:~/demoproj#
```

i-00ec27cc6ebf14602 (manager-swarm-stack)

PublicIPs: 3.108.228.186   PrivateIPs: 172.31.47.162

Step 9: Build the image and push it to Docker Hub

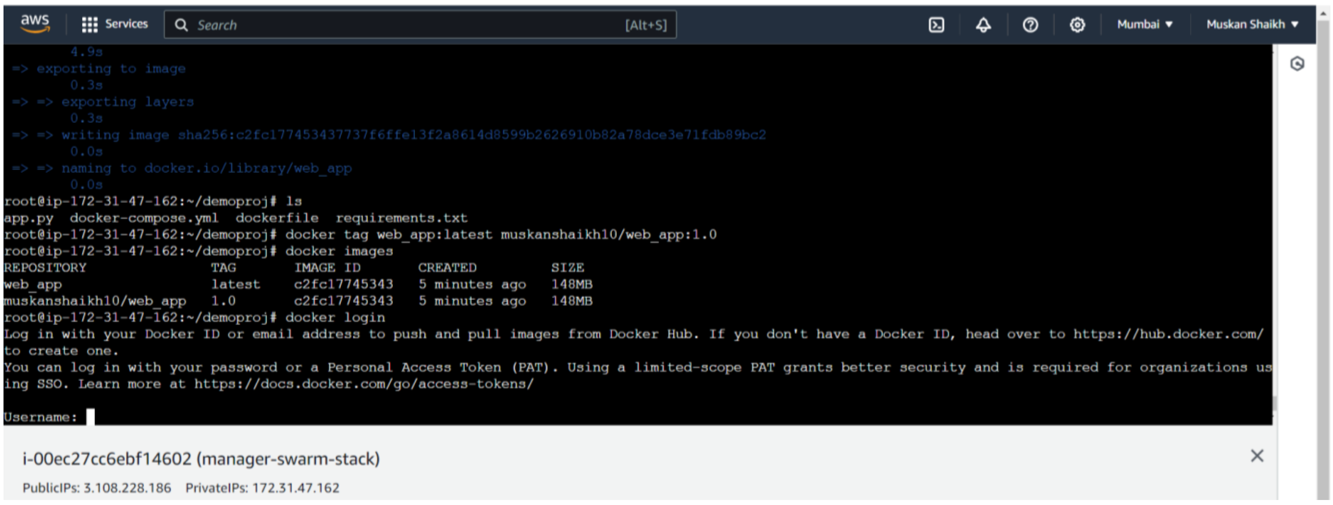
#docker build -t web_app

#docker tag web_app:latest muskanshaikh/web_app:1.0

#docker images

#docker login

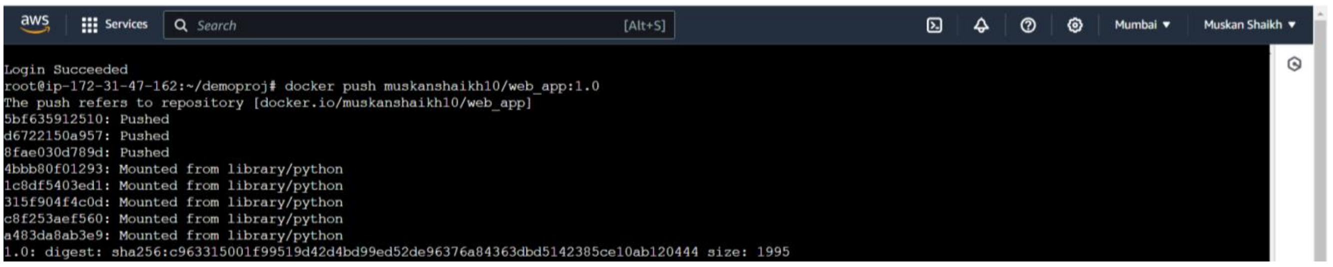
#docker push muskanshaikh10/web_app:1.0



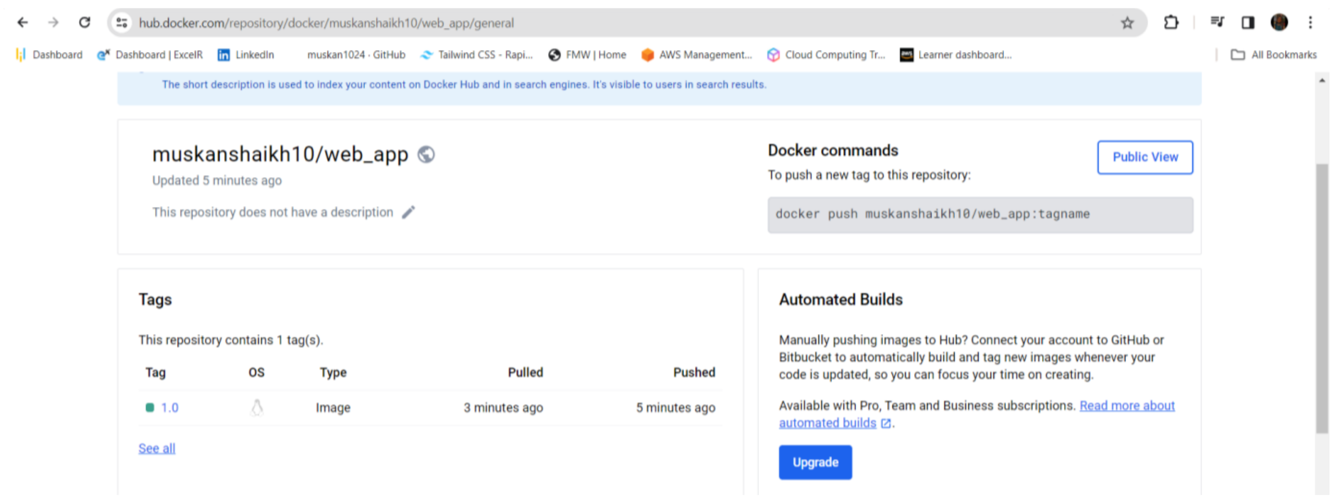
```
         4.9s
 => exporting to image
         0.3s
 => => exporting layers
         0.3s
 => => writing image sha256:c2fc177453437737f6ffe13f2a8614d8599b2626910b82a78dce3e71fdb89bc2
         0.0s
 => => naming to docker.io/library/web_app
         0.0s
root@ip-172-31-47-162:~/demoproj# ls
app.py  docker-compose.yml  dockerfile  requirements.txt
root@ip-172-31-47-162:~/demoproj# docker tag web_app:latest muskanshaikh10/web_app:1.0
root@ip-172-31-47-162:~/demoproj# docker images
REPOSITORY               TAG       IMAGE ID       CREATED         SIZE
web_app                  latest    c2fc17745343   5 minutes ago   148MB
muskanshaikh10/web_app   1.0       c2fc17745343   5 minutes ago   148MB
root@ip-172-31-47-162:~/demoproj# 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 security and is required for organizations us
ing SSO. Learn more at https://docs.docker.com/go/access-tokens/

Username: 
```

i-00ec27cc6ebf14602 (manager-swarm-stack)

PublicIPs: 3.108.228.186 PrivateIPs: 172.31.47.162



```
Login Succeeded
root@ip-172-31-47-162:~/demoproj# docker push muskanshaikh10/web_app:1.0
The push refers to repository [docker.io/muskanshaikh10/web_app]
5bf635912510: Pushed
d6722150a957: Pushed
8fae030d789d: Pushed
4bbb80f01293: Mounted from library/python
1c8df5403ed1: Mounted from library/python
315f904f4c0d: Mounted from library/python
c8f253aef560: Mounted from library/python
a483da8ab3e9: Mounted from library/python
1.0: digest: sha256:c963315001f99519d42d4bd99ed52de96376a84363dbd5142385ce10ab120444 size: 1995
```



hub.docker.com/repository/docker/muskanshaikh10/web_app/general

The short description is used to index your content on Docker Hub and in search engines. It's visible to users in search results.

muskanshaikh10/web_app

Updated 5 minutes ago

This repository does not have a description

Docker commands

To push a new tag to this repository:

`docker push muskanshaikh10/web_app:tagname`

Tags

This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
1.0		Image	3 minutes ago	5 minutes 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.

Step 10: Deploy the stack to the swarm and check it:

#docker stack deploy –help

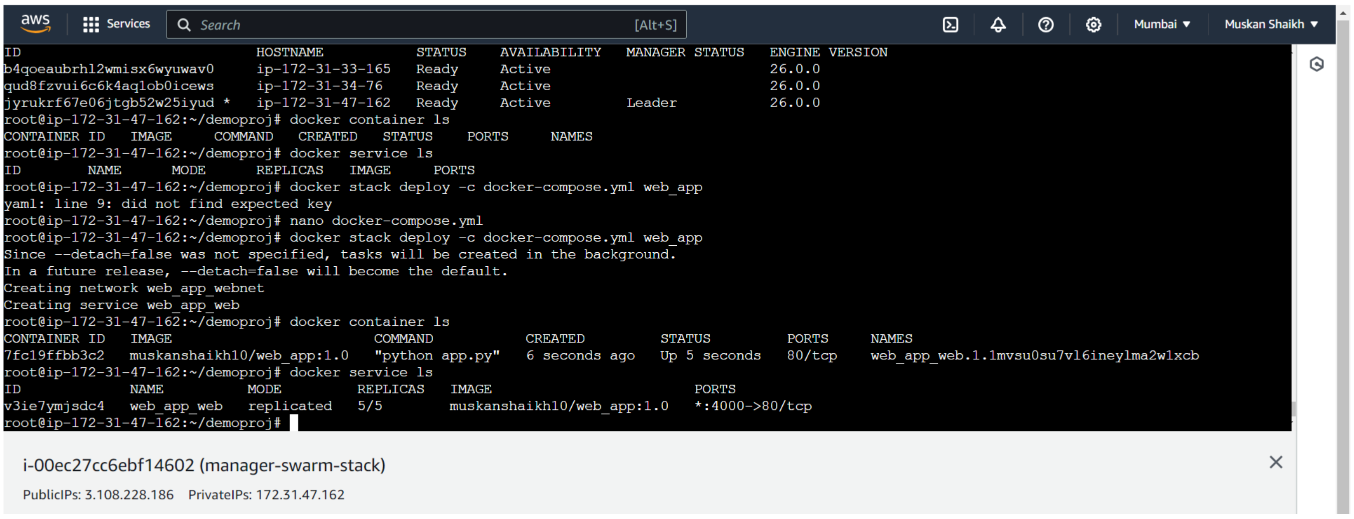
Create the stack with docker stack deploy:

#docker stack deploy -c dokcer-compose.yml web_app

#docker node ls

#docker container ls

#docker service ls



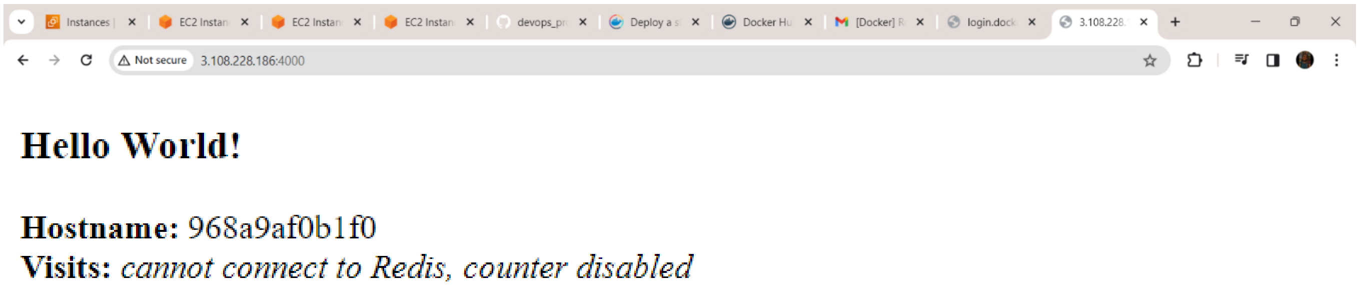
```
ID                            HOSTNAME           STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
b4qoeaubrhl2wmisx6wyuwav0     ip-172-31-33-165   Ready     Active                          26.0.0
qud8fzvui6c6k4aqlob0icews     ip-172-31-34-76    Ready     Active                          26.0.0
jyrukrf67e06jtgb52w25iyud *   ip-172-31-47-162   Ready     Active         Leader           26.0.0
root@ip-172-31-47-162:~/demoproj# docker container ls
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
root@ip-172-31-47-162:~/demoproj# docker service ls
ID        NAME      MODE      REPLICAS   IMAGE     PORTS
root@ip-172-31-47-162:~/demoproj# docker stack deploy -c docker-compose.yml web_app
yaml: line 9: did not find expected key
root@ip-172-31-47-162:~/demoproj# nano docker-compose.yml
root@ip-172-31-47-162:~/demoproj# docker stack deploy -c docker-compose.yml web_app
Since --detach=false was not specified, tasks will be created in the background.
In a future release, --detach=false will become the default.
Creating network web_app_webnet
Creating service web_app_web
root@ip-172-31-47-162:~/demoproj# docker container ls
CONTAINER ID   IMAGE                     COMMAND           CREATED         STATUS         PORTS     NAMES
7fc19ffbb3c2   muskanshaikh10/web_app:1.0   "python app.py"   6 seconds ago   Up 5 seconds   80/tcp    web_app_web.1.1mvsu0su7vl6ineylma2w1xcb
root@ip-172-31-47-162:~/demoproj# docker service ls
ID             NAME          MODE         REPLICAS   IMAGE                        PORTS
v3ie7ymjsdc4   web_app_web   replicated   5/5        muskanshaikh10/web_app:1.0   *:4000->80/tcp
root@ip-172-31-47-162:~/demoproj#
```

i-00ec27cc6ebf14602 (manager-swarm-stack)

PublicIPs: 3.108.228.186 PrivateIPs: 172.31.47.162

Step 11: Check if the application is running

publicIP:4000



Hello World!

Hostname: 968a9af0b1f0
Visits: *cannot connect to Redis, counter disabled*

End of the practical

Sign

Dr. Swapnil Waghmare