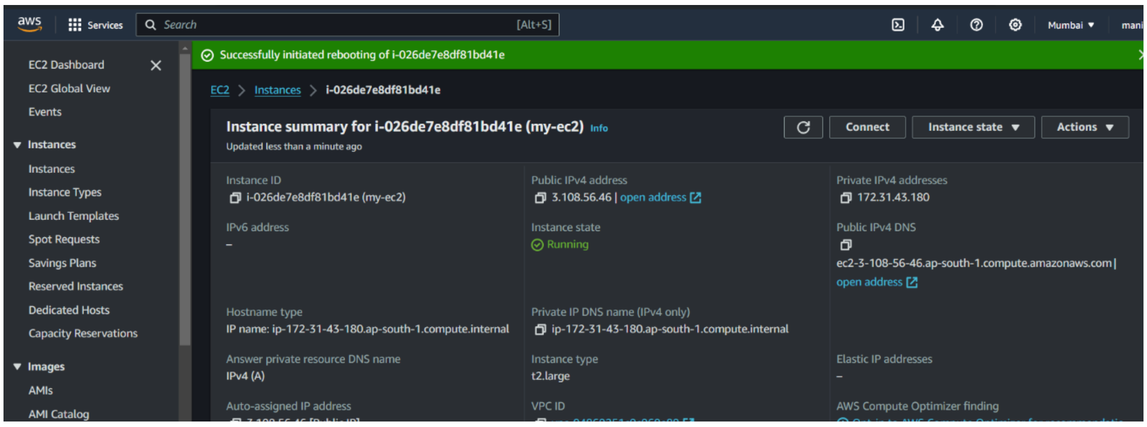
EXPOSING A SIMPLE PYTHON SCRIPT IN DOCKER CONTAINERS THROUGH PORTS

- First launch an EC2 instance of instance-type “t2.large”



Successfully initiated rebooting of i-026de7e8df81bd41e

EC2 > Instances > i-026de7e8df81bd41e

Instance summary for i-026de7e8df81bd41e (my-ec2)

Updated less than a minute ago

Instance ID: i-026de7e8df81bd41e (my-ec2)

Public IPv4 address: 3.108.56.46 | open address

Private IPv4 addresses: 172.31.43.180

IPv6 address: –

Instance state: Running

Public IPv4 DNS: ec2-3-108-56-46.ap-south-1.compute.amazonaws.com | open address

Hostname type: IP name: ip-172-31-43-180.ap-south-1.compute.internal

Private IP DNS name (IPv4 only): ip-172-31-43-180.ap-south-1.compute.internal

Answer private resource DNS name: IPv4 (A)

Instance type: t2.large

Elastic IP addresses: –

Auto-assigned IP address

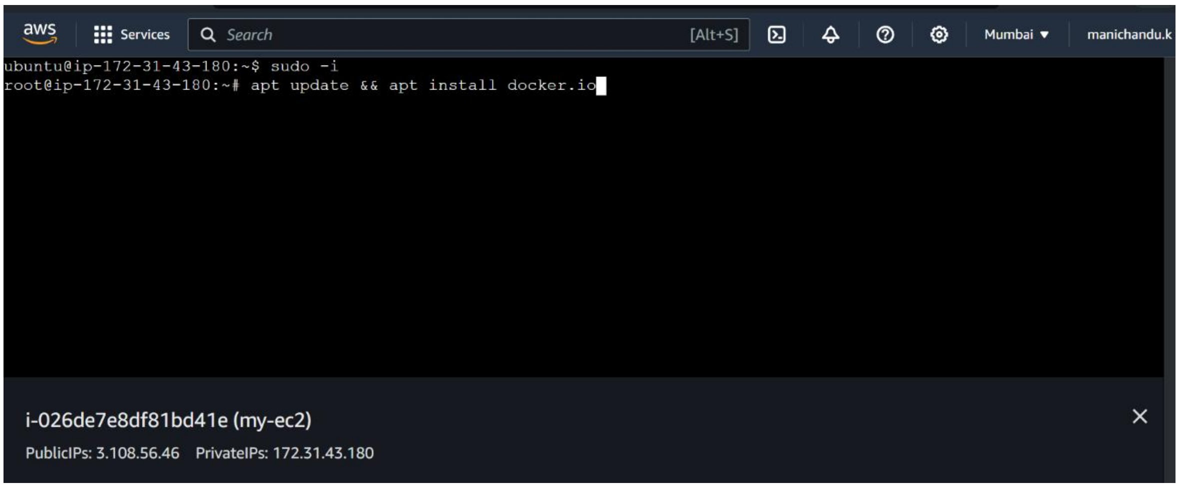
VPC ID

AWS Compute Optimizer finding

- Next connect to the instance and run these commands

Sudo -i

apt update && apt install docker.io

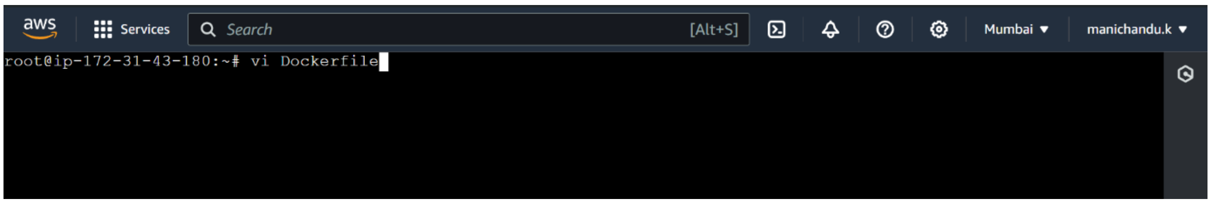


```
ubuntu@ip-172-31-43-180:~$ sudo -i
root@ip-172-31-43-180:~# apt update && apt install docker.io
```

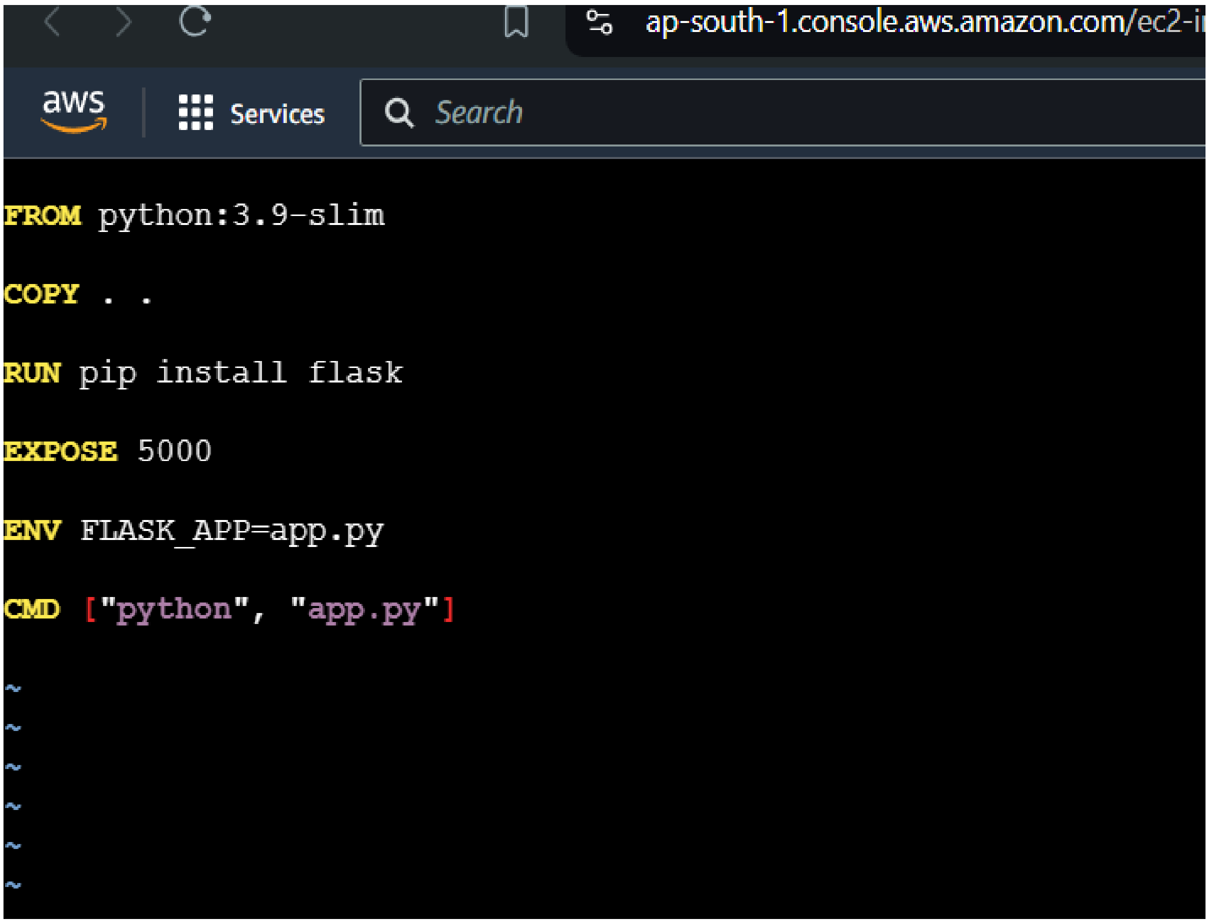
i-026de7e8df81bd41e (my-ec2)

PublicIPs: 3.108.56.46 PrivateIPs: 172.31.43.180

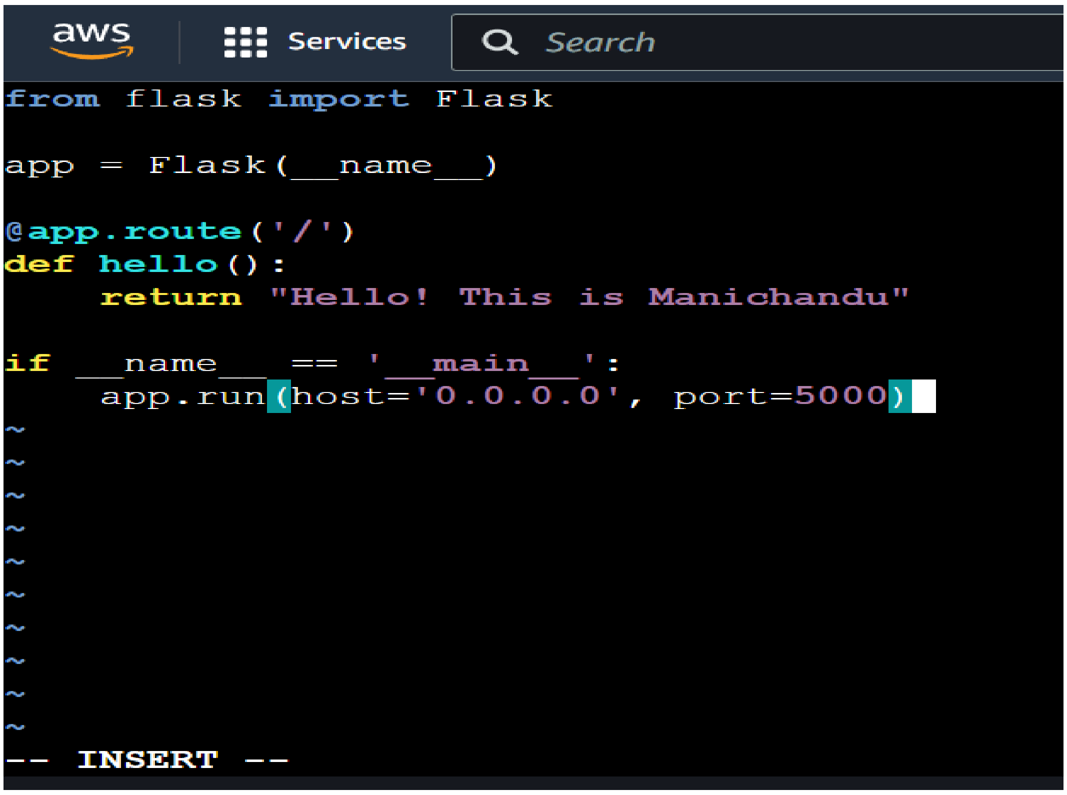
- After installing docker write “Dockerfile” and python application script using “.py” extension.



```
root@ip-172-31-43-180:~# vi Dockerfile
```

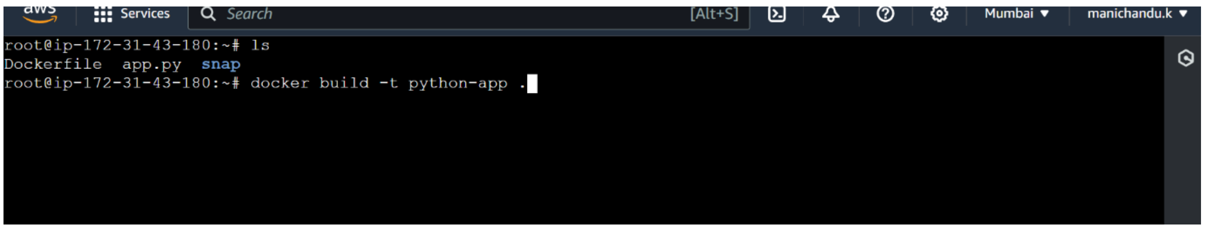


```
FROM python:3.9-slim

COPY . .

RUN pip install flask

EXPOSE 5000

ENV FLASK_APP=app.py

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

~
~
~
~
~
~
```



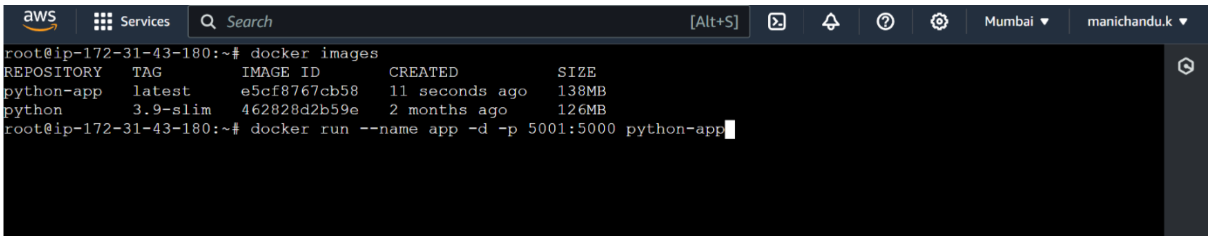
```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def hello():
    return "Hello! This is Manichandu"

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000)
~
~
~
~
~
~
~
~
~
~
-- INSERT --
```

- Now it is time to build the image from our Dockerfile.



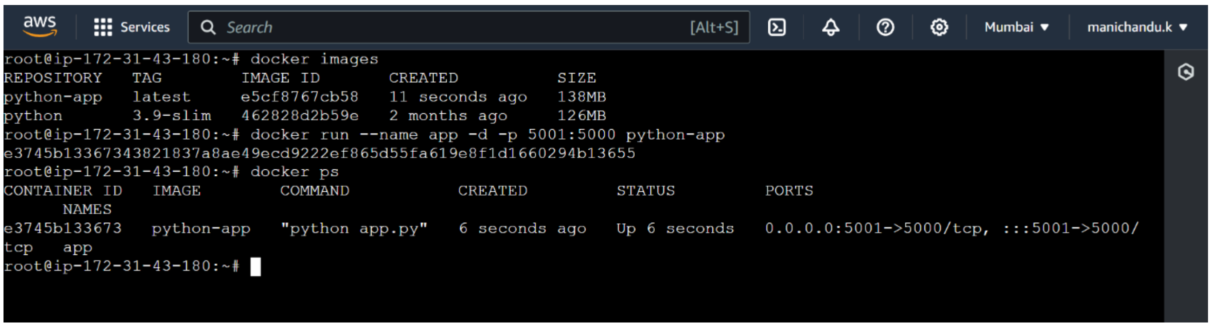
```
root@ip-172-31-43-180:~# ls
Dockerfile  app.py  snap
root@ip-172-31-43-180:~# docker build -t python-app .
```

- Now create a container using this image.



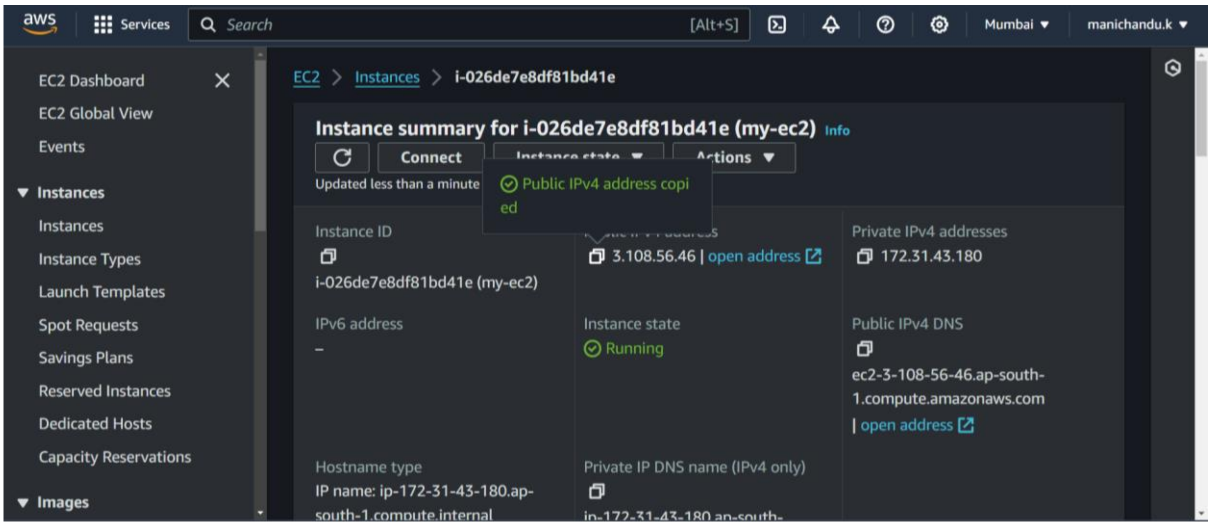
```
root@ip-172-31-43-180:~# docker images
REPOSITORY   TAG        IMAGE ID       CREATED          SIZE
python-app   latest     e5cf8767cb58   11 seconds ago   138MB
python       3.9-slim   462828d2b59e   2 months ago     126MB
root@ip-172-31-43-180:~# docker run --name app -d -p 5001:5000 python-app
```

- Check whether the container is running.



```
root@ip-172-31-43-180:~# docker images
REPOSITORY   TAG        IMAGE ID       CREATED          SIZE
python-app   latest     e5cf8767cb58   11 seconds ago   138MB
python       3.9-slim   462828d2b59e   2 months ago     126MB
root@ip-172-31-43-180:~# docker run --name app -d -p 5001:5000 python-app
e3745b13367343821837a8ae49ecd9222ef865d55fa619e8f1d1660294b13655
root@ip-172-31-43-180:~# docker ps
CONTAINER ID   IMAGE        COMMAND           CREATED         STATUS         PORTS
      NAMES
e3745b133673   python-app   "python app.py"   6 seconds ago   Up 6 seconds   0.0.0.0:5001->5000/tcp, :::5001->5000/
tcp    app
root@ip-172-31-43-180:~#
```

- Now access the application using the public-ip address of the instance and the port number.



Instance summary for i-026de7e8df81bd41e (my-ec2)

Updated less than a minute

Public IPv4 address copied

Instance ID
i-026de7e8df81bd41e (my-ec2)

3.108.56.46 | open address

Private IPv4 addresses
172.31.43.180

IPv6 address
–

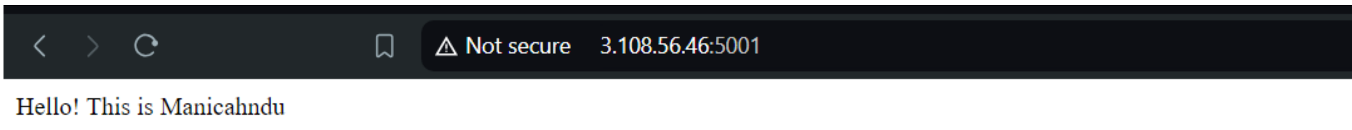
Instance state
Running

Public IPv4 DNS
ec2-3-108-56-46.ap-south-1.compute.amazonaws.com
| open address

Hostname type
IP name: ip-172-31-43-180.ap-south-1.compute.internal

Private IP DNS name (IPv4 only)
ip-172-31-43-180.ap-south-

- We successfully run our python script in docker container.



```
Hello! This is Manicahndu
```