

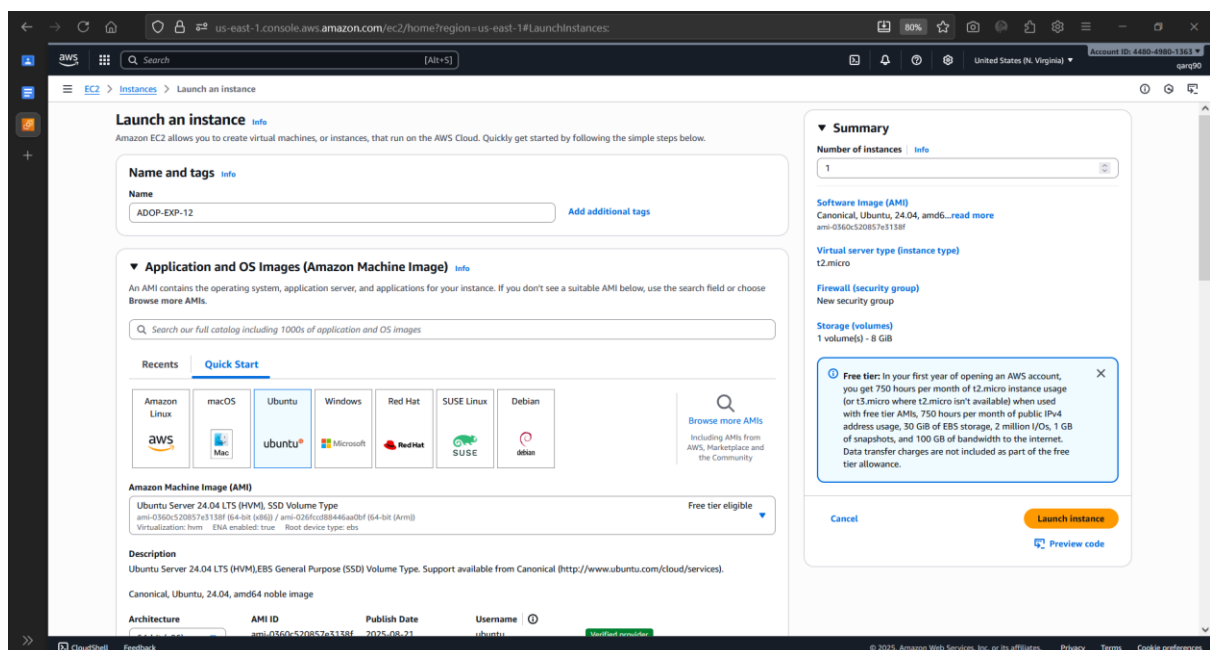
Name: Abdurrahman Qureshi

Roll No: 242466

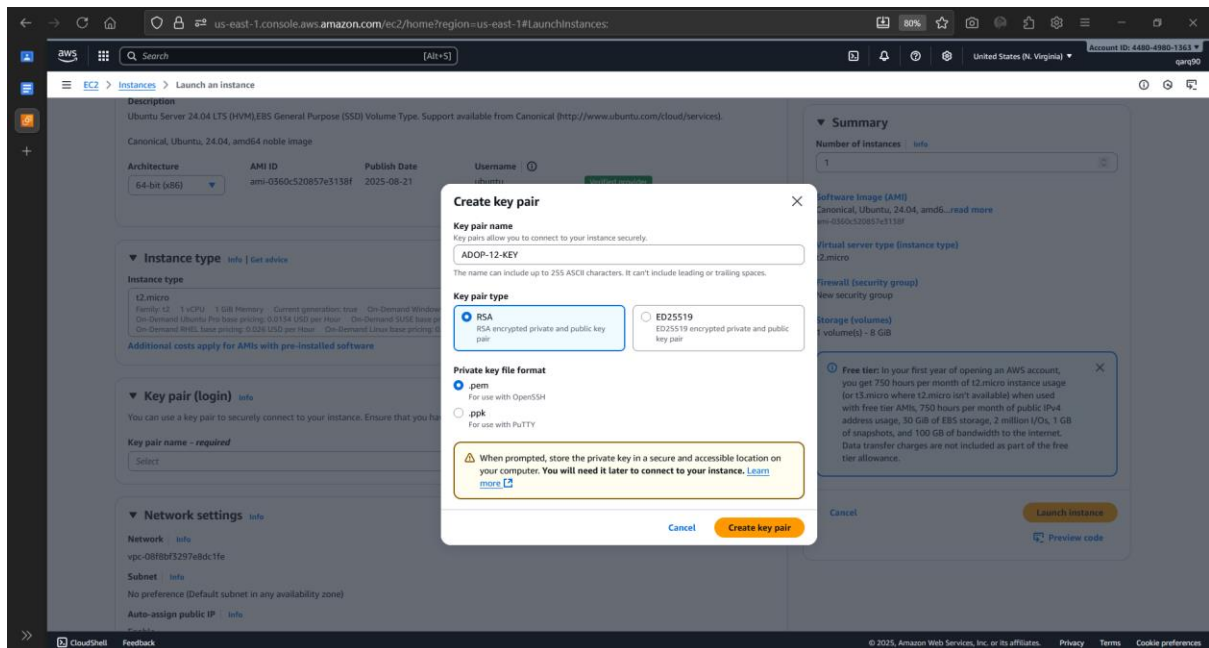
Practical No: 12

Date Of Performance: 23/09/2025

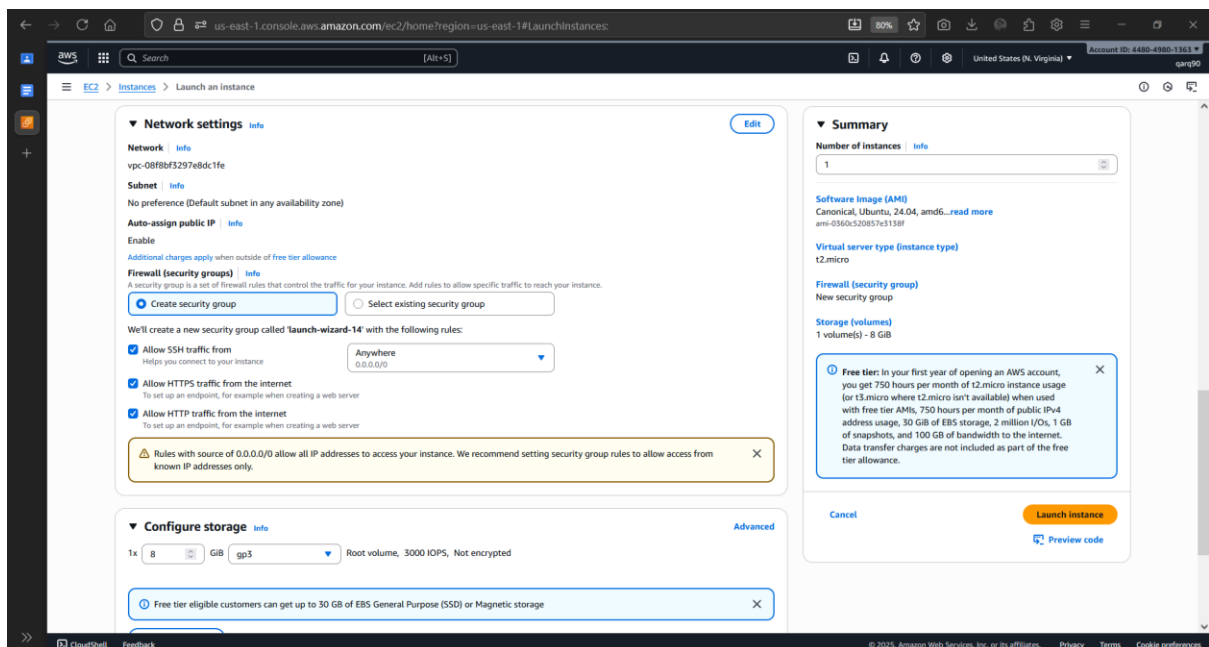
Aim: To demonstrate the complete workflow of installing Docker on an AWS EC2 Ubuntu instance, containerizing a Flask application, and properly managing cloud resources.



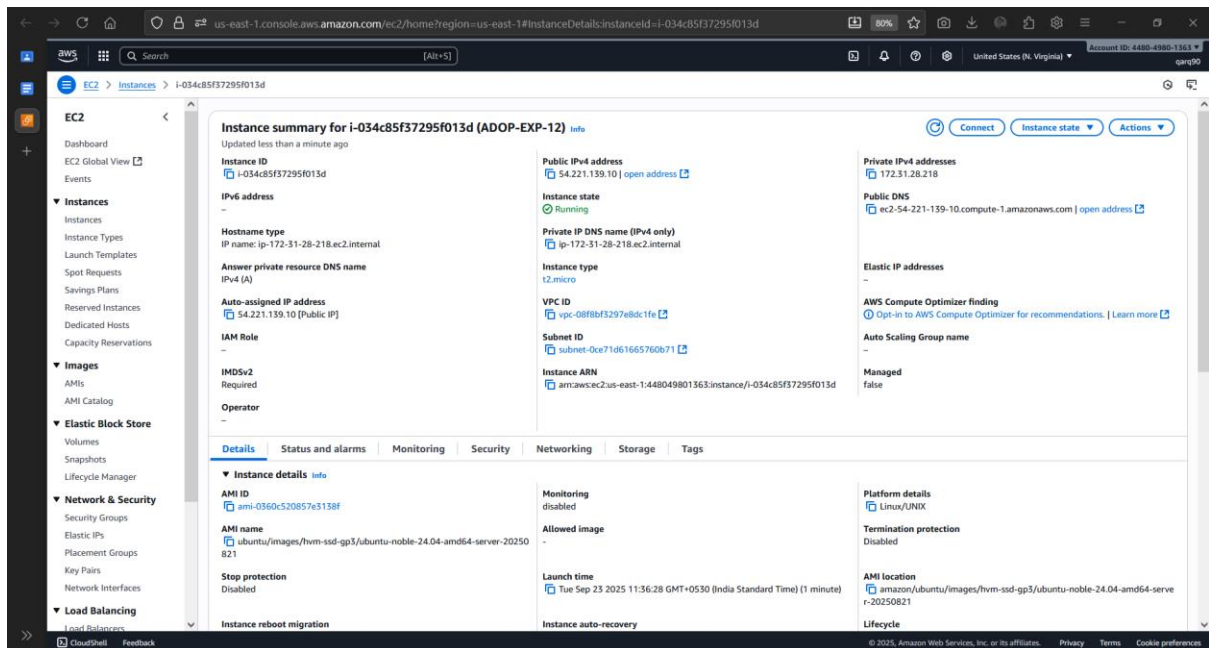
Creating a new EC2 Instance



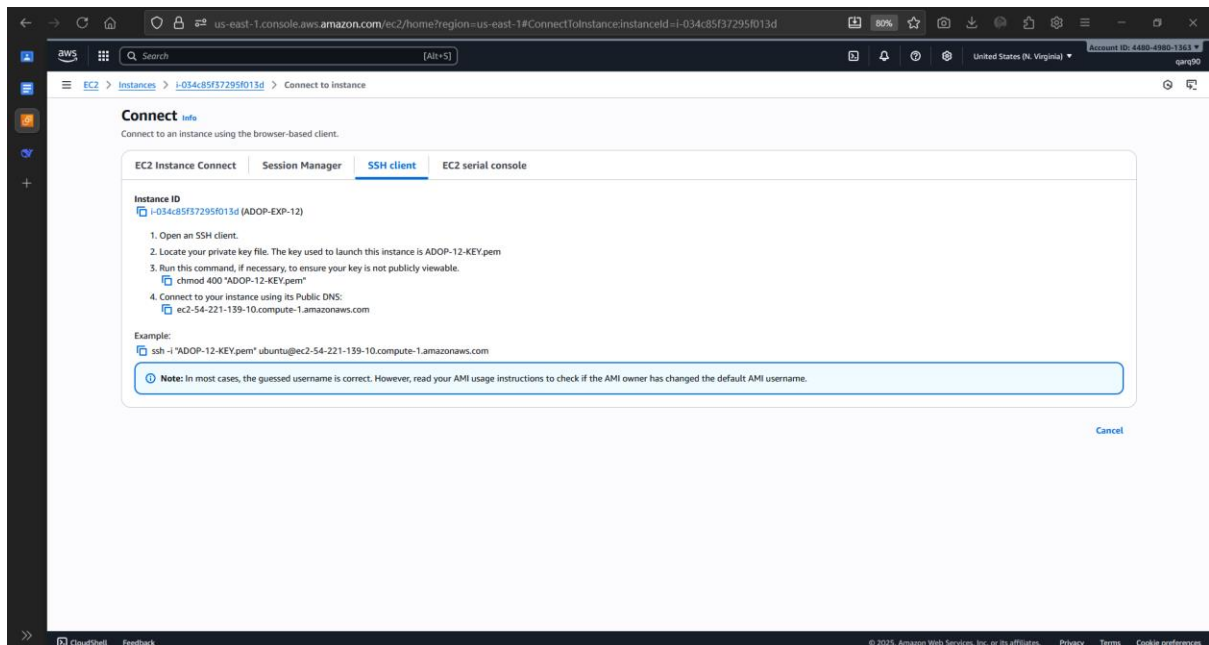
## Creating new key



## Configuring network settings for EC2



## Instance Details



## Commands to Connect to Instance

```
Abdurrahman.Qureshi@DESKTOP-H2RVSMQ MINGW64 ~/Downloads
$ ssh -i "ADOP-12-KEY.pem" ubuntu@ec2-54-221-139-18.compute-1.amazonaws.com
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1011-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Tue Sep 23 06:10:43 UTC 2025

System load:  0.19           Processes:           114
Usage of /:   37.0% of 6.7168   Users logged in:    0
Memory usage: 27%           IPv4 address for enx0: 172.31.28.218
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

43 updates can be applied immediately.
31 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Last login: Tue Sep 23 06:07:55 2025 from 1.38.148.24
ubuntu@ip-172-31-28-218:~$
```

Connection Successful

```
Server: Docker Engine - Community
Engine:
  Version:           28.4.0
  API version:       1.51 (minimum version 1.24)
  Go version:        go1.24.7
  Git commit:        249d679
  Built:             Wed Sep 3 20:57:32 2025
  OS/Arch:           linux/amd64
  Experimental:      false
containerd:
  Version:           1.7.27
  GitCommit:        05044ec0a9a75232cad458027ca83437aee3f4da
runc:
  Version:           1.2.5
  GitCommit:        v1.2.5-0-g59923ef
docker-init:
  Version:           0.19.0
  GitCommit:        de40ad0

=====

To run Docker as a non-privileged user, consider setting up the
Docker daemon in rootless mode for your user:

  dockerd-rootless-setuptool.sh install

Visit https://docs.docker.com/go/rootless/ to learn about rootless mode.

To run the Docker daemon as a fully privileged service, but granting non-root
users access, refer to https://docs.docker.com/go/daemon-access/

WARNING: Access to the remote API on a privileged Docker daemon is equivalent
to root access on the host. Refer to the 'Docker daemon attack surface'
documentation for details: https://docs.docker.com/go/attack-surface/

=====

ubuntu@ip-172-31-28-218:~$
```

Installed Docker

```
ubuntu@ip-172-31-28-218: ~/flask-docker-app
ubuntu@ip-172-31-28-218:~/flask-docker-app$ # Create Dockerfile in the SAME directory
ubuntu@ip-172-31-28-218:~/flask-docker-app$ cat > Dockerfile << 'EOF'
> FROM python:3.9-slim
> WORKDIR /app
> COPY requirements.txt .
> RUN pip install --no-cache-dir -r requirements.txt
> COPY . .
> EXPOSE 5000
> ENV FLASK_APP=app.py
> CMD ["python", "app.py"]
> EOF
ubuntu@ip-172-31-28-218:~/flask-docker-app$ # Create Dockerfile in the SAME directory
ubuntu@ip-172-31-28-218:~/flask-docker-app$ cat > Dockerfile << 'EOF'
> FROM python:3.9-slim
> WORKDIR /app
> COPY requirements.txt .
> RUN pip install --no-cache-dir -r requirements.txt
> COPY . .
> EXPOSE 5000
> ENV FLASK_APP=app.py
> CMD ["python", "app.py"]
> EOF
ubuntu@ip-172-31-28-218:~/flask-docker-app$ ls -la
total 20
drwxrwxr-x 2 ubuntu ubuntu 4096 Sep 23 06:13 .
drwxr-xr-x 5 ubuntu ubuntu 4096 Sep 23 06:11 ..
-rw-rw-r-- 1 ubuntu ubuntu 176 Sep 23 06:13 Dockerfile
-rw-rw-r-- 1 ubuntu ubuntu 362 Sep 23 06:13 app.py
-rw-rw-r-- 1 ubuntu ubuntu 15 Sep 23 06:13 requirements.txt
ubuntu@ip-172-31-28-218:~/flask-docker-app$
```

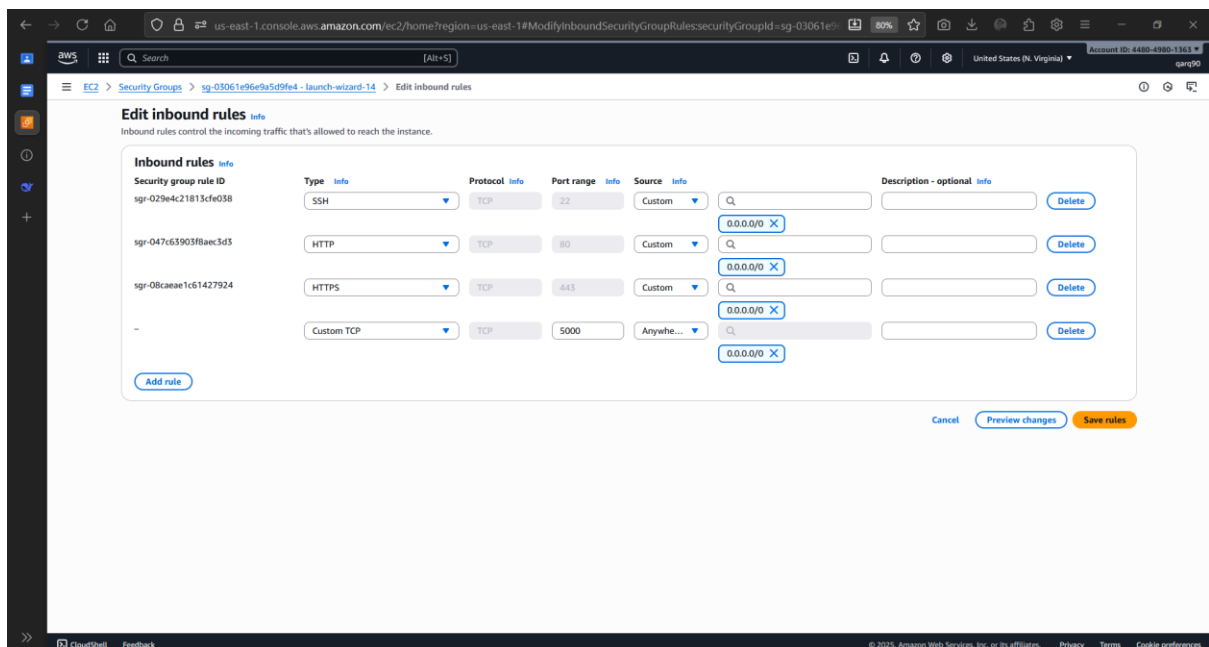
Setup Complete for flask application

```
ubuntu@ip-172-31-28-218: ~/flask-docker-app
> CMD ["python", "app.py"]
> EOF
ubuntu@ip-172-31-28-218:~/flask-docker-app$ ls -la
total 20
drwxrwxr-x 2 ubuntu ubuntu 4096 Sep 23 06:13 .
drwxr-xr-x 5 ubuntu ubuntu 4096 Sep 23 06:11 ..
-rw-rw-r-- 1 ubuntu ubuntu 176 Sep 23 06:13 Dockerfile
-rw-rw-r-- 1 ubuntu ubuntu 362 Sep 23 06:13 app.py
-rw-rw-r-- 1 ubuntu ubuntu 15 Sep 23 06:13 requirements.txt
ubuntu@ip-172-31-28-218:~/flask-docker-app$ docker build -t flask-app .
[+] Building 8.4s (10/10) FINISHED
=> [internal] load build definition from Dockerfile                                docker:default
=> => transferring dockerfile: 215B                                              0.1s
=> [internal] load metadata for docker.io/library/python:3.9-slim                0.0s
=> [internal] load .dockerignore                                                 0.5s
=> [internal] load context: 28                                                  0.0s
=> => transferring context: 28                                                    0.0s
=> [1/5] FROM docker.io/library/python:3.9-slim@sha256:cf0704507972b63c9b20382dd6f05248572d6b25961410305f96479bf2e8a23c 3.3s
=> => resolve docker.io/library/python:3.9-slim@sha256:cf0704507972b63c9b20382dd6f05248572d6b25961410305f96479bf2e8a23c 0.0s
=> => sha256:ce1261c6d567efa8e3b457673eeeb474a0a8066df6bb95ca9a6a94a31e219dd3 29.77MB / 29.77MB 0.6s
=> => sha256:1d454ace0e384876850a0aa5ef6b8c45705445114ab233959bdab71a577b9200 1.29MB / 1.29MB 0.2s
=> => sha256:41dc2499d8fe1ea2351cc01f3716ce6a95ad0e9bf90c0819fd0c4a93cf4e9b24 13.37MB / 13.37MB 0.5s
=> => sha256:cf0704507972b63c9b20382dd6f05248572d6b25961410305f96479bf2e8a23c 10.36kB / 10.36kB 0.0s
=> => sha256:161727d2d61fdfe4836d11f82fb437a3fcb2f4ce5b85951805f0717687ce110f 1.74kB / 1.74kB 0.0s
=> => sha256:56ceae0119ab69043114ce215d355f9f343a55b74b58001450df2e00478fb3529 5.39kB / 5.39kB 0.0s
=> => extracting sha256:ce1261c6d567efa8e3b457673eeeb474a0a8066df6bb95ca9a6a94a31e219dd3 1.4s
=> => extracting sha256:1d454ace0e384876850a0aa5ef6b8c45705445114ab233959bdab71a577b9200 0.1s
=> => extracting sha256:41dc2499d8fe1ea2351cc01f3716ce6a95ad0e9bf90c0819fd0c4a93cf4e9b24 0.0s
=> => extracting sha256:7fcd9369fa90e0413fe19da3d316fb6c3bfbd7371fa4ce617617cac3e8de12 0.0s
=> [internal] load build context                                                0.0s
=> => transferring context: 674B                                                  0.0s
=> [2/5] WORKDIR /app                                                         0.1s
=> [3/5] COPY requirements.txt .                                              0.1s
=> [4/5] RUN pip install --no-cache-dir -r requirements.txt                  3.9s
=> [5/5] COPY . .                                                            0.1s
=> => exporting image                                                            0.1s
=> => exporting layers                                                            0.3s
=> => writing image sha256:635141dc59c01685dd8c66b693e260ab368339cec5d018680a7589b9ccb4c0e5 0.0s
=> => naming to docker.io/library/flask-app                                    0.0s
ubuntu@ip-172-31-28-218:~/flask-docker-app$
```

Building the flask application

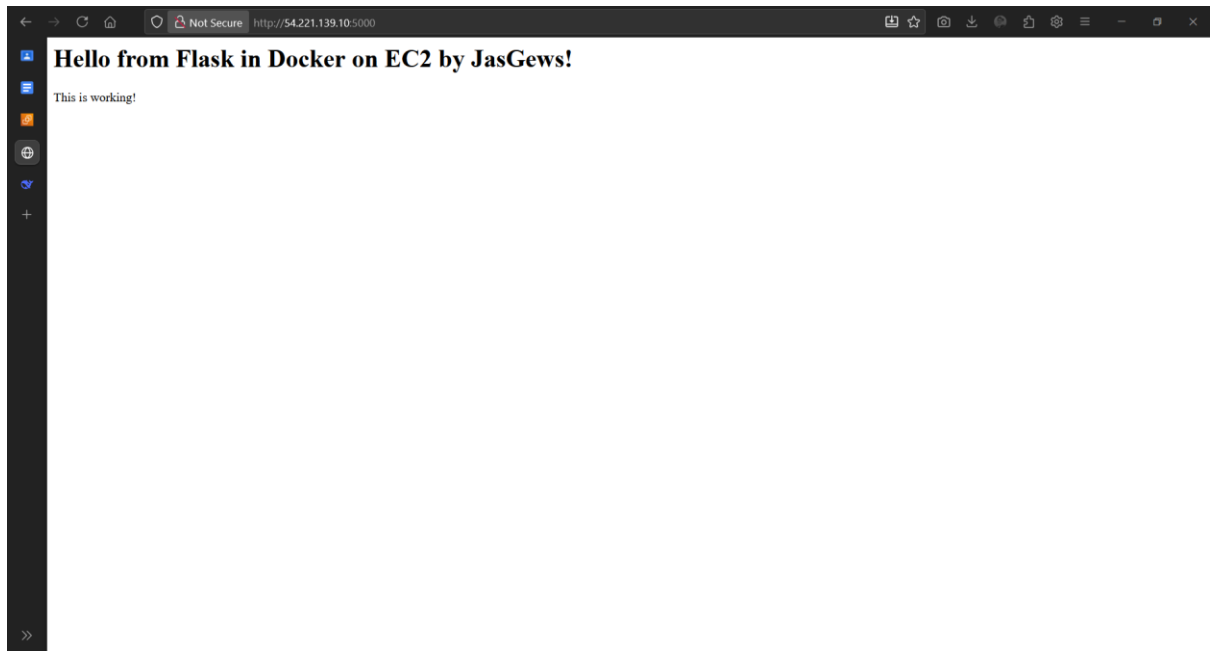
```
ubuntu@ip-172-31-28-218: ~/flask-docker-app
ubuntu@ip-172-31-28-218:~/flask-docker-app$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
flask-app latest 635141dc59c0 22 seconds ago 131MB
ubuntu@ip-172-31-28-218:~/flask-docker-app$ docker run -d -p 5000:5000 --name my-flask-app flask-app
c0e4fcd78b4668971977144ca8f135fa85f933be91d873acb27f9bb5d6b19e0c
ubuntu@ip-172-31-28-218:~/flask-docker-app$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
c0e4fcd78b46 flask-app "python app.py" 10 seconds ago Up 10 seconds 0.0.0.0:5000->5000/tcp, [::]:5000->5000/tcp my-fla
sk-app
ubuntu@ip-172-31-28-218:~/flask-docker-app$ curl http://localhost:5000
<h1>Hello from Flask in Docker on EC2 by JasGews!</h1><p>This is working!</p>ubuntu@ip-172-31-28-218:~/flask-docker-app$ ^C
ubuntu@ip-172-31-28-218:~/flask-docker-app$ docker logs my-flask-app
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://172.17.0.2:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 602-577-905
172.17.0.1 - - [23/Sep/2025 06:15:35] "GET / HTTP/1.1" 200 -
ubuntu@ip-172-31-28-218:~/flask-docker-app$
```

Build complete



Editing Inbound rules of the instance

## OUTPUT:



Output of the flask application

Q. What is Dockerfile?

- A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble a Docker image.

DockerFile:

1: FROM python:3.9-slim

- This specifies the base image to build upon, using the official Python 3.9 runtime on a slim Linux distribution

2: WORKDIR /app

- Sets the working directory inside the container to /app, where all subsequent commands will be executed

3: COPY requirements.txt .

- Copies only the requirements.txt file from your host machine to the current working directory (/app) in the container

4: RUN pip install --no-cache-dir -r requirements.txt

- Installs the Python dependencies listed in requirements.txt using pip, without saving cache to reduce image size

5: COPY . .

- Copies all remaining application files from the host directory to the container's working directory

6: EXPOSE 5000

- Documents that the container will listen on port 5000 (the default Flask port)

7: ENV FLASK\_APP=app.py

- Sets an environment variable that tells Flask which application to run (app.py)

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

- Specifies the default command to run when the container starts, which will execute the Flask application using Python



## App.py:

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello():
    return '<h1>Hello from Flask in Docker on EC2 by JasGews!</h1><p>This is
working!</p>'

@app.route('/health')
def health():
    return {'status': 'healthy', 'message': 'Flask app running'}

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000, debug=True)
```

## requirements.txt

Flask==2.3.3