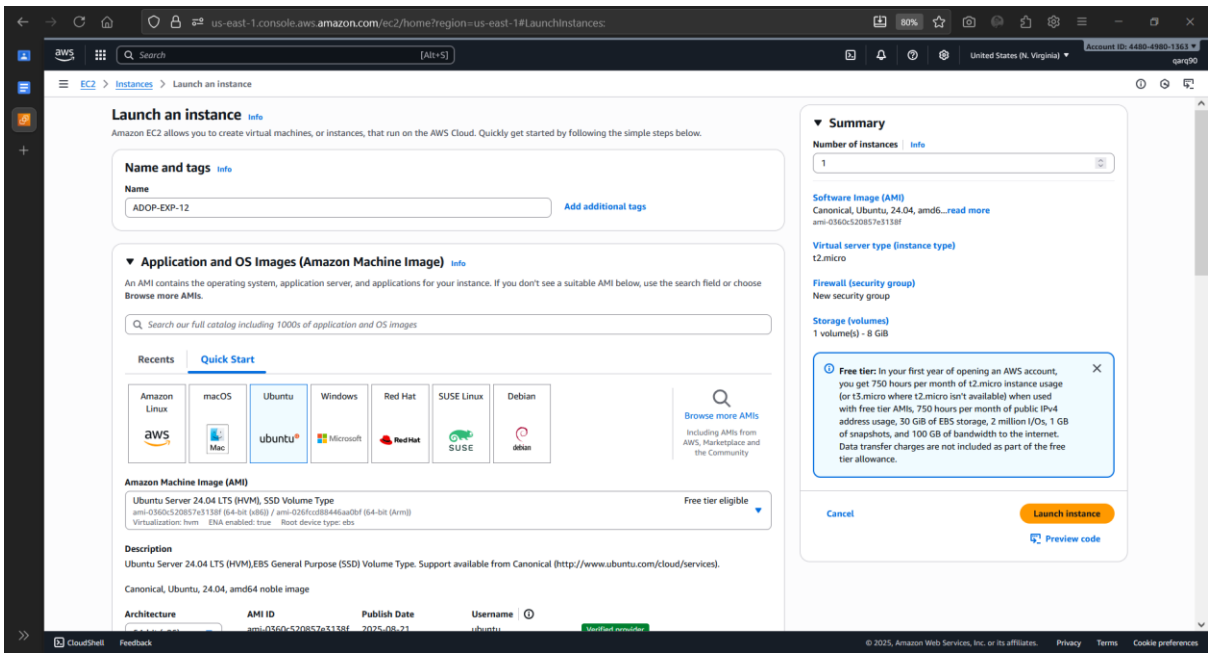
Name: Abdurrahman Qureshi

Roll No: 242466

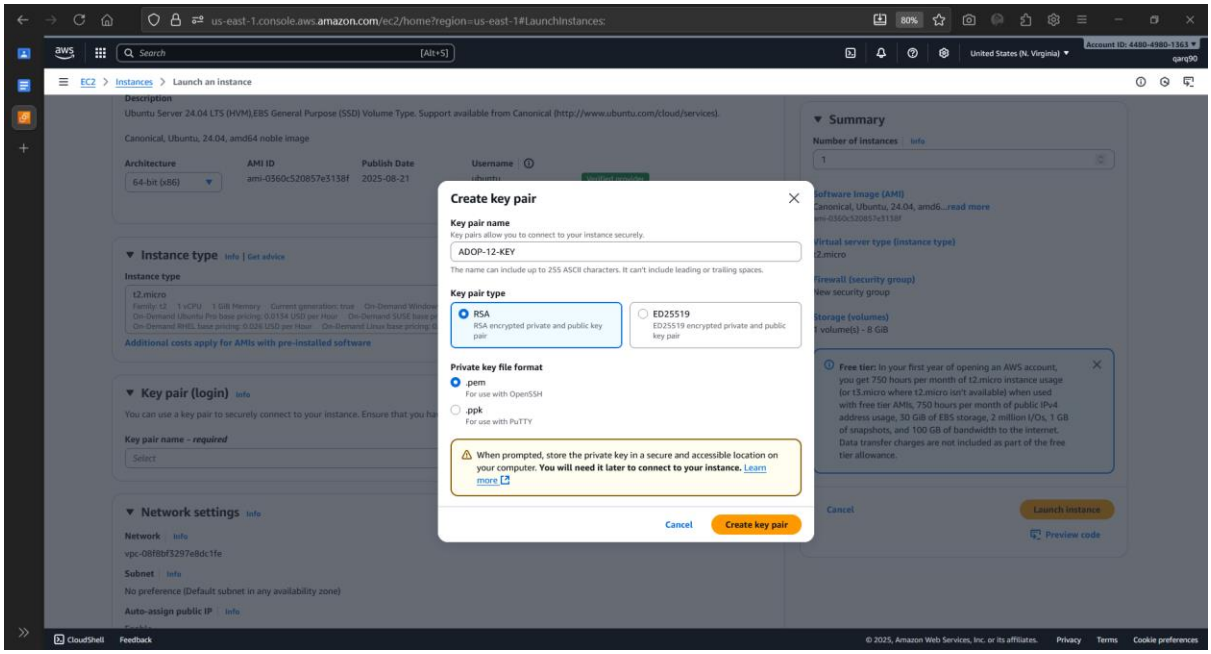
Practical No: 12

Date Of Performance: 23/09/2025

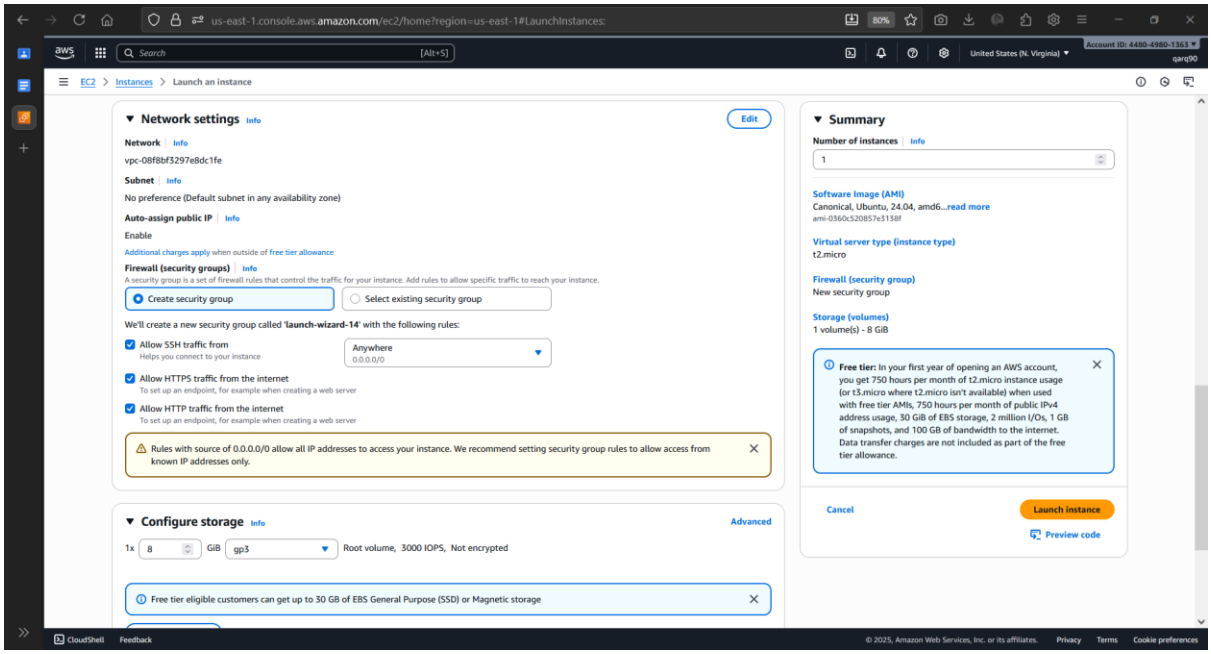
Aim: To demonstrate the complete workflow of installing Docker on an AWS EC2 Ubuntu instance, containerizing a Flask application, and properly managing cloud resources.



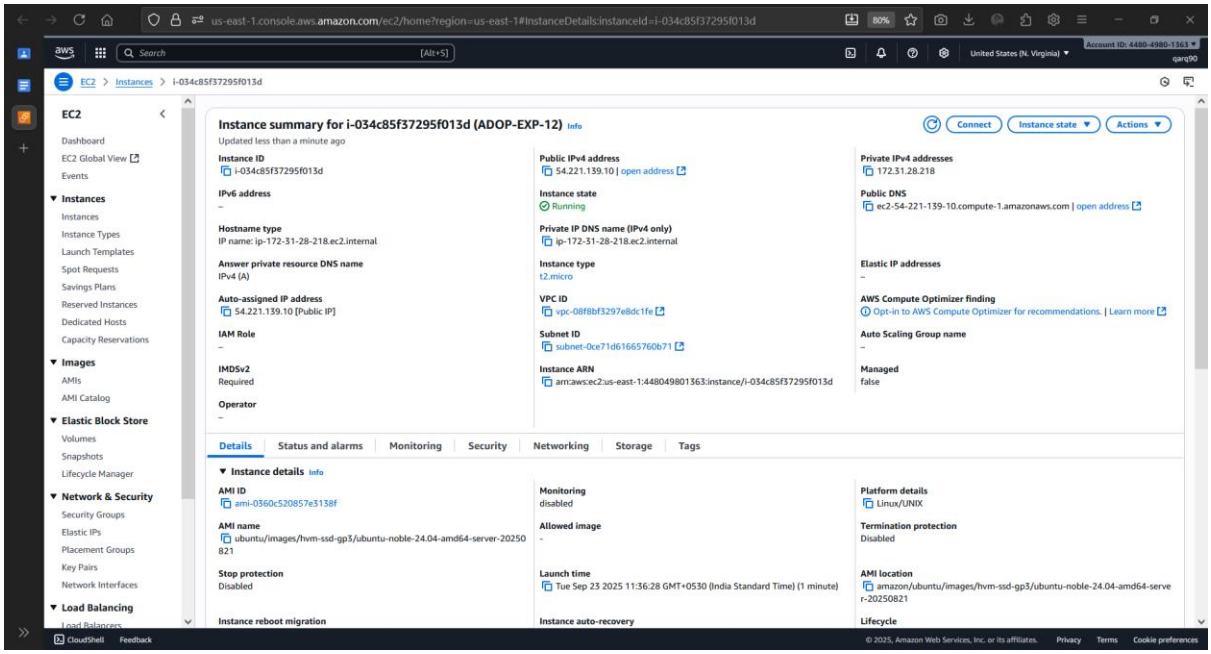
Creating a new EC2 Instance



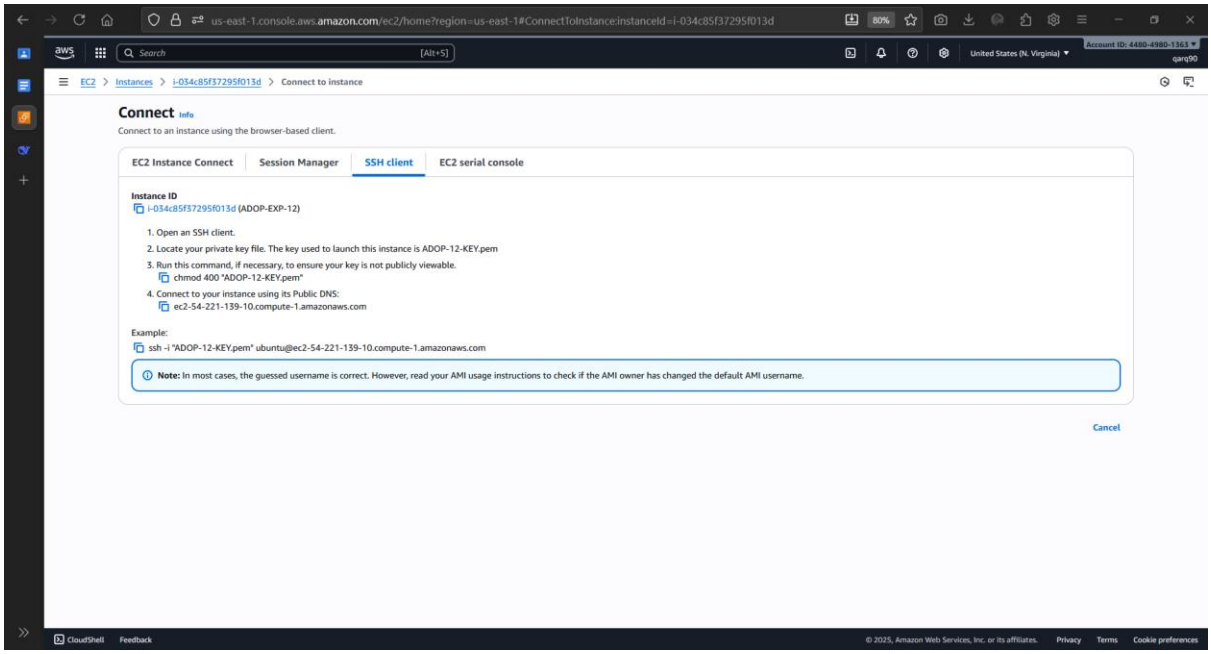
Creating new key



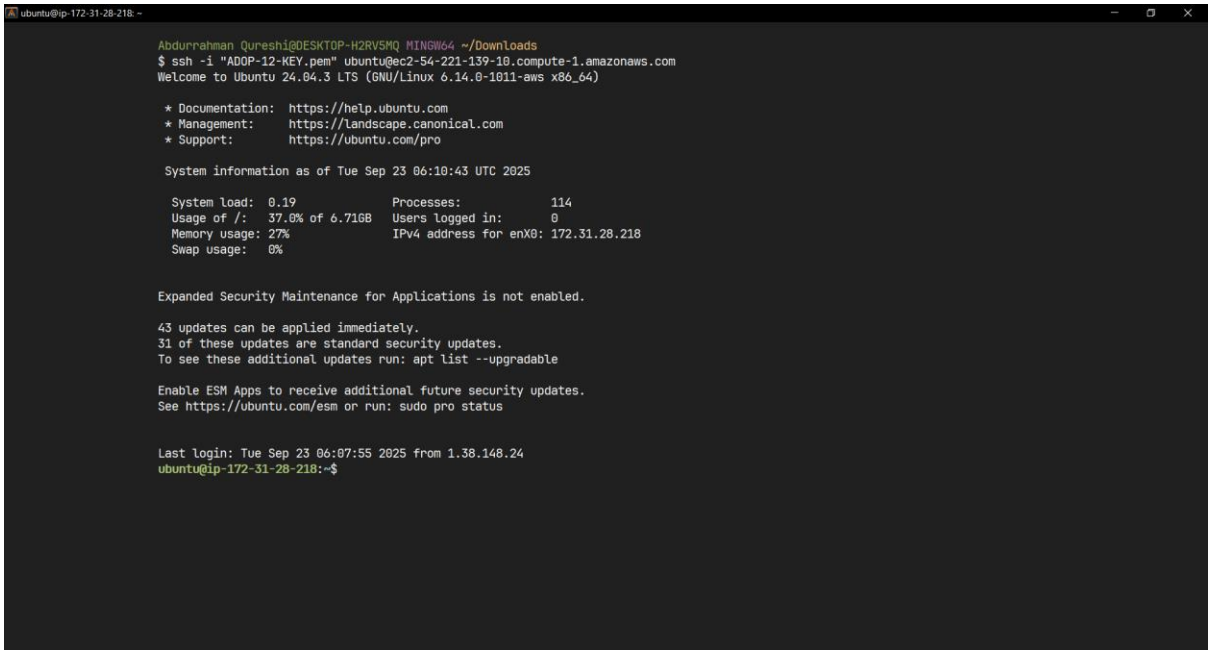
Configuring network settings for EC2



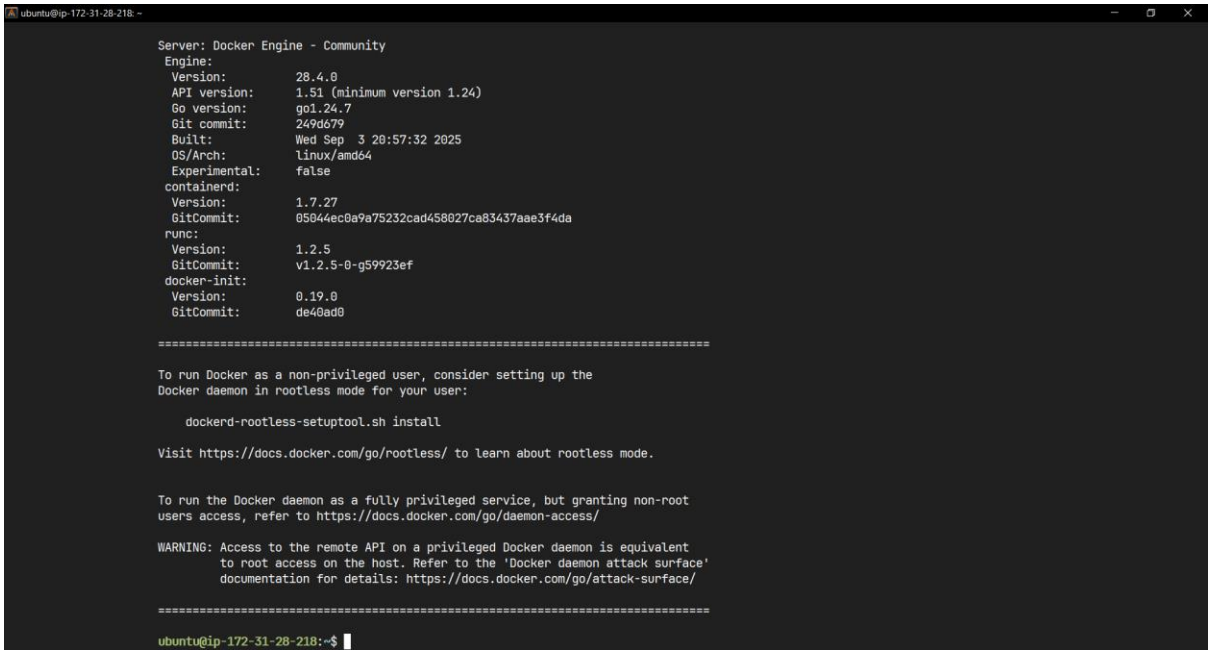
Instance Details



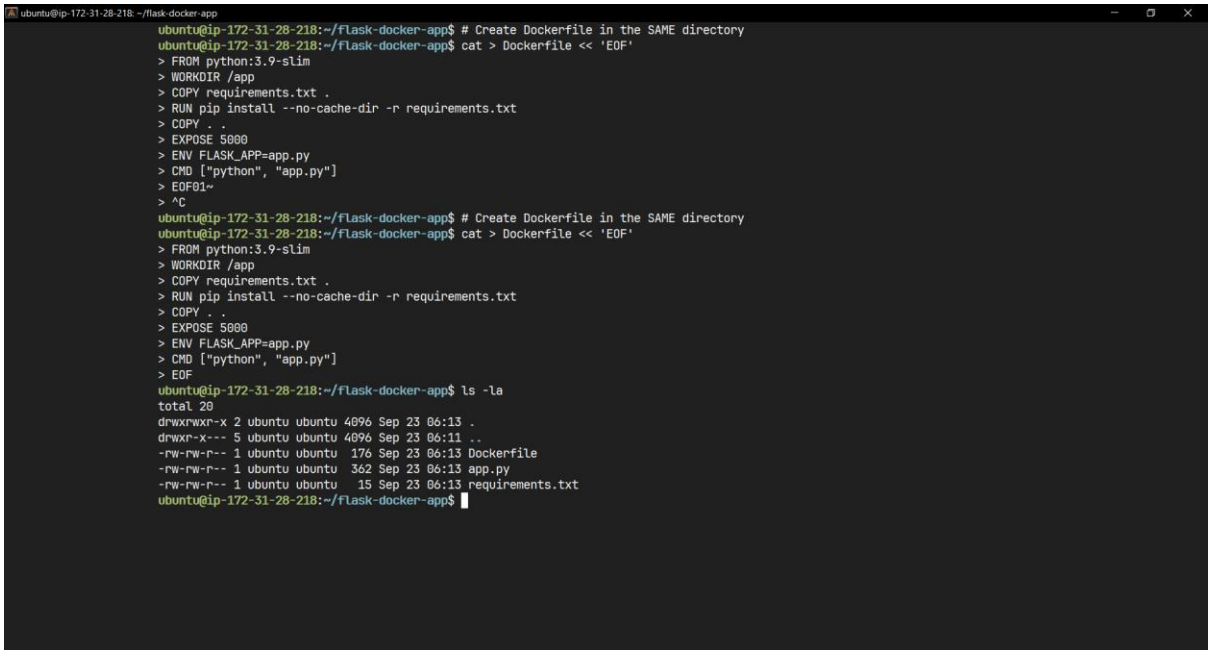
Commands to Connect to Instance



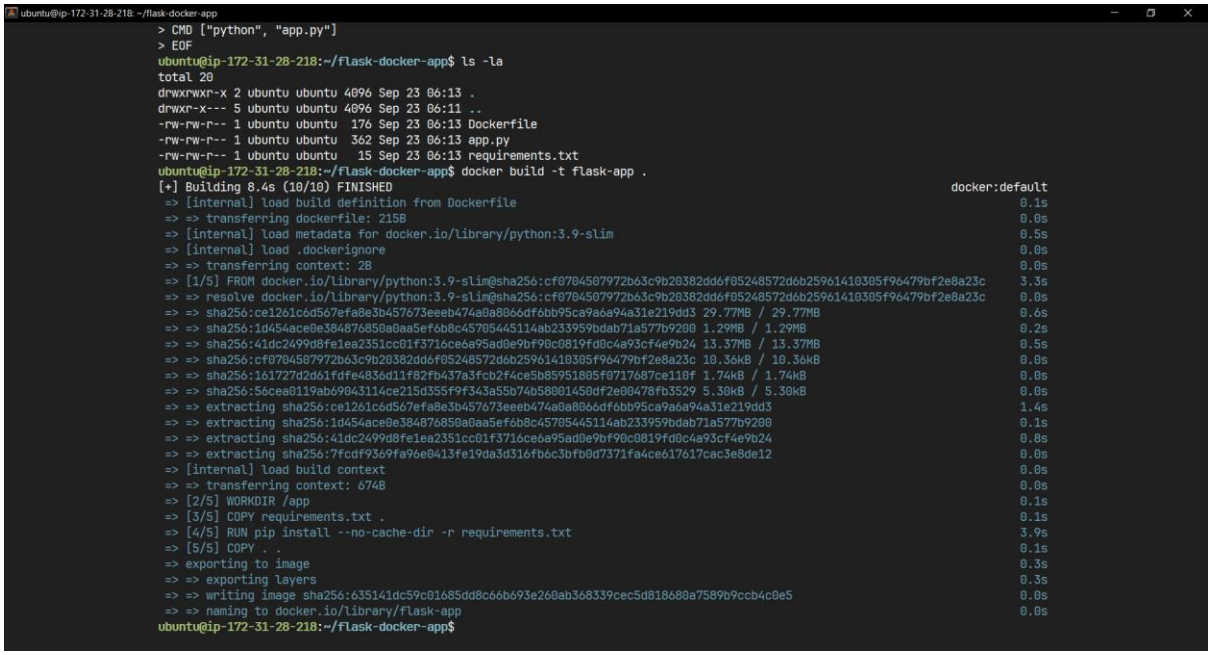
Connection Successful



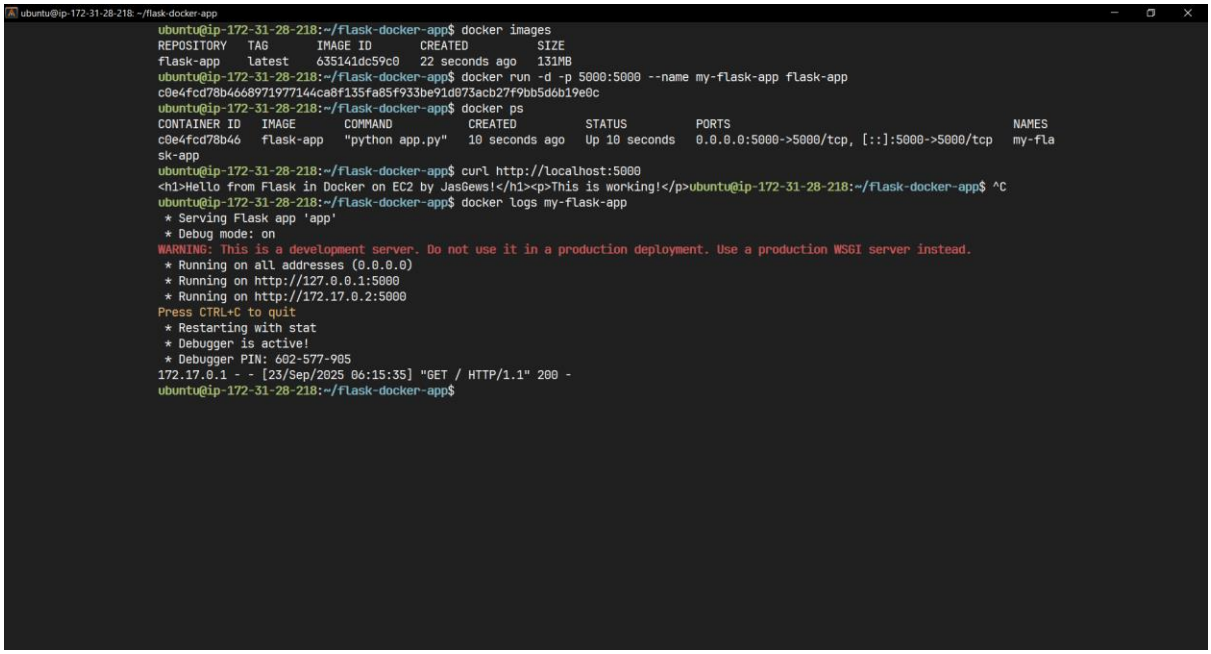
Installed Docker



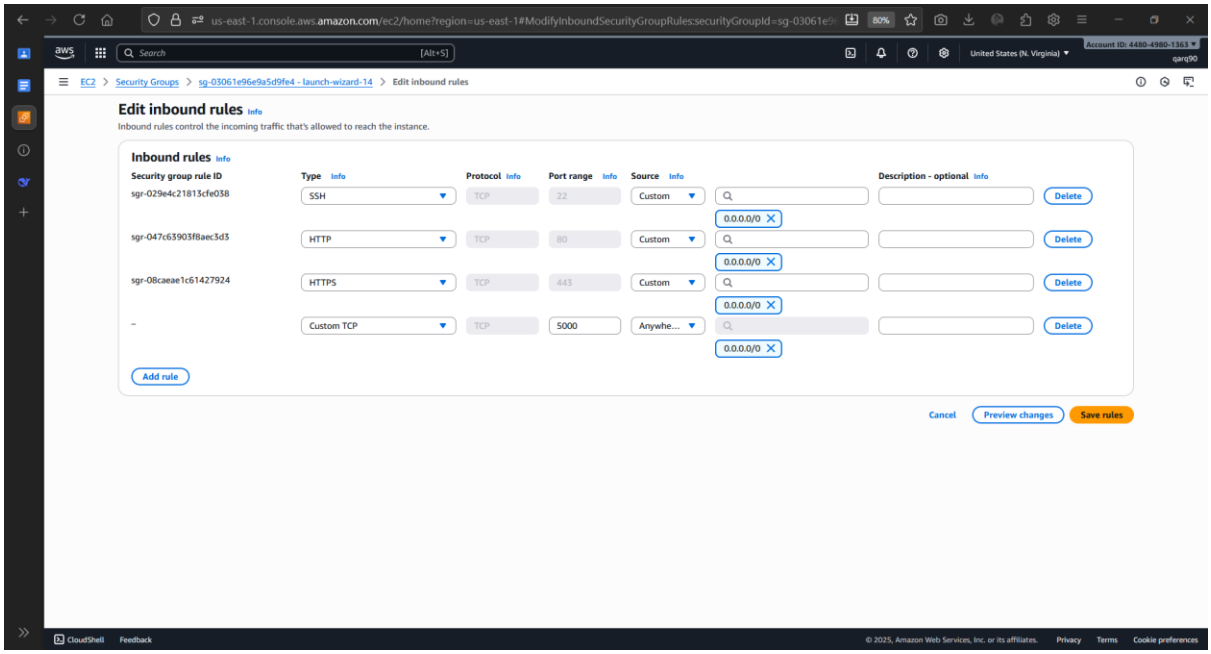
Setup Complete for flask application



Building the flask application

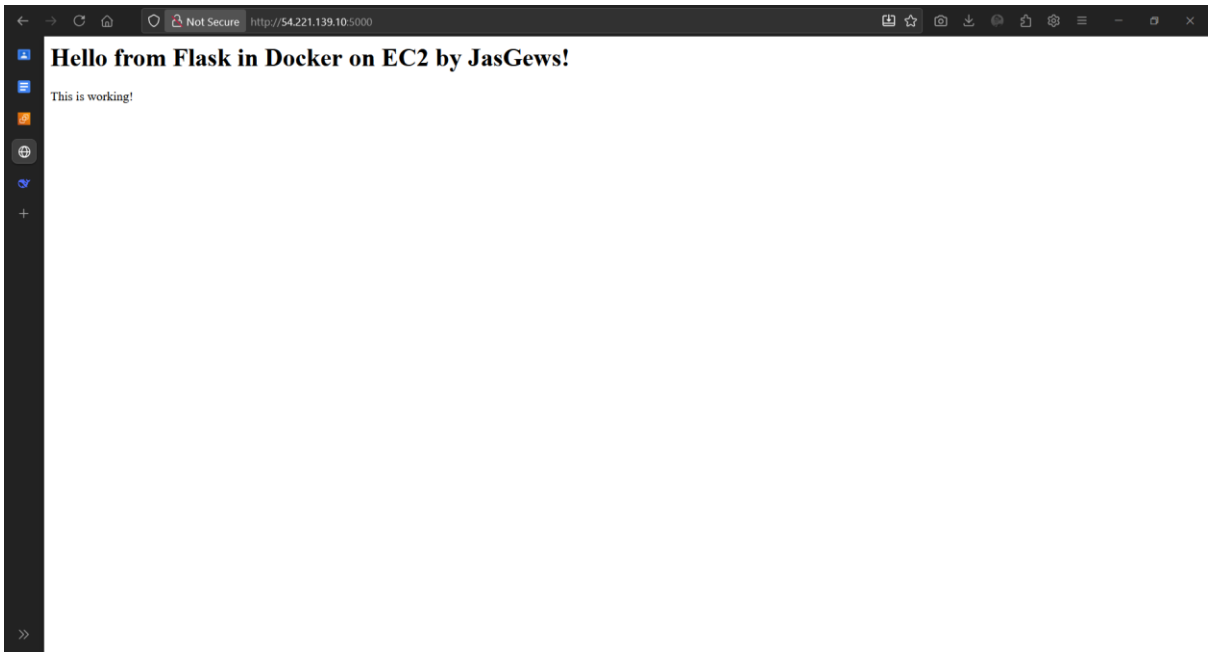


Build complete



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
```