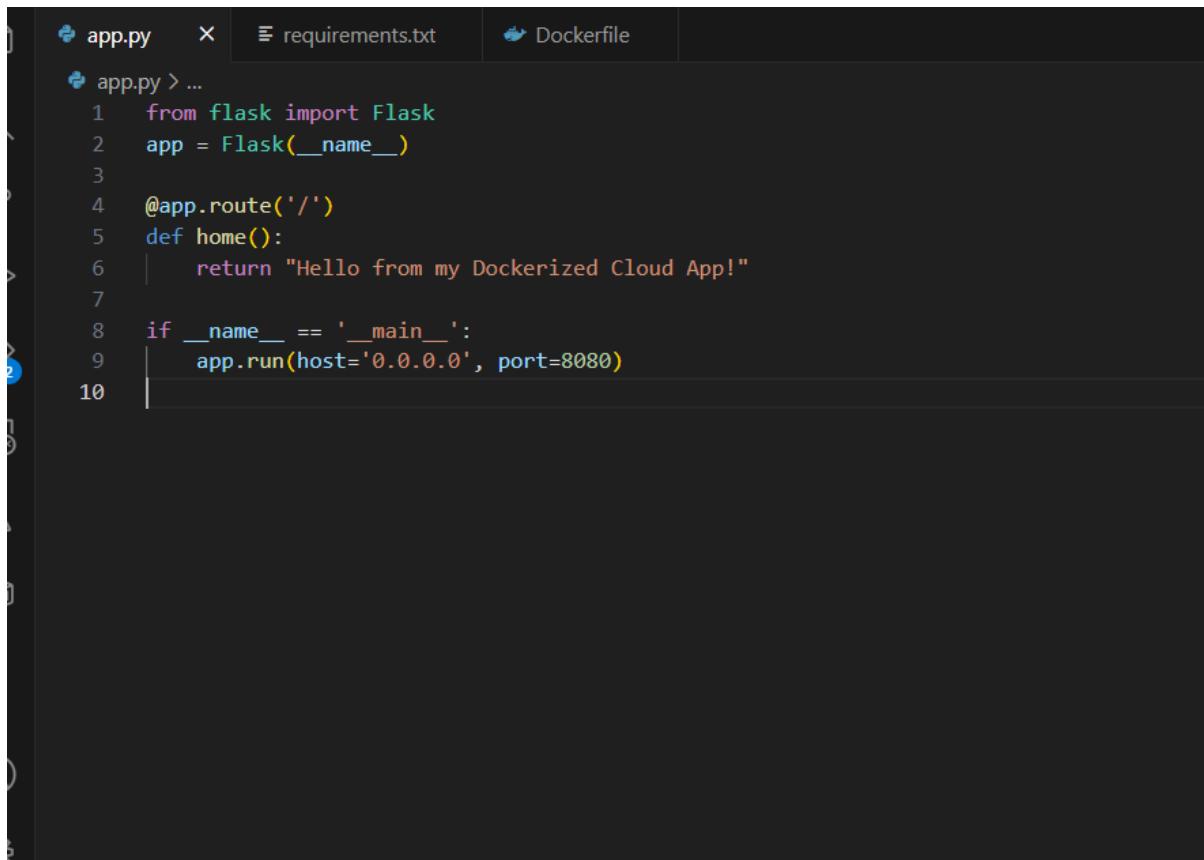


Task 8: Deploy a Dockerized Web Application on the Cloud

Screenshot of running container locally



The screenshot shows a code editor interface with three tabs: `app.py`, `requirements.txt`, and `Dockerfile`. The `app.py` tab is active and displays the following Python code:

```
 1  from flask import Flask
 2  app = Flask(__name__)
 3
 4  @app.route('/')
 5  def home():
 6      return "Hello from my Dockerized Cloud App!"
 7
 8  if __name__ == '__main__':
 9      app.run(host='0.0.0.0', port=8080)
```

Screenshot of deployed service on the cloud

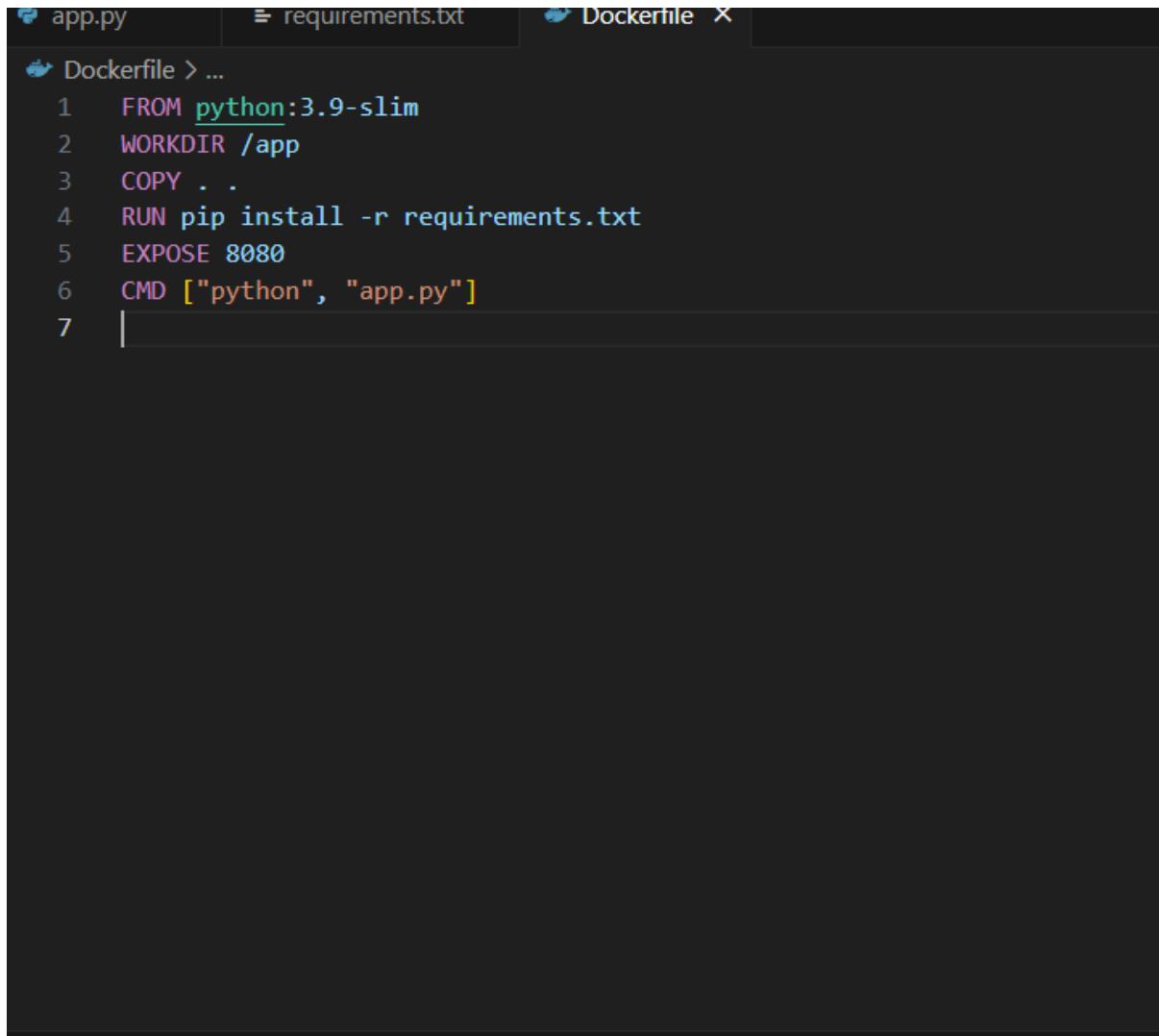
The screenshot shows a code editor interface with three tabs:

- `app.py`: Contains the Python code for the application.
- `requirements.txt`: Contains the dependency list, showing `flask` as the only requirement.
- `Dockerfile`: Contains the Docker configuration.

```
requirements.txt
1 flask
2
```

Hello from my Dockerized Cloud App!

Dockerfile code



The screenshot shows a code editor with three tabs open:

- Dockerfile**: The active tab, containing the following Dockerfile code:

```
1 FROM python:3.9-slim
2 WORKDIR /app
3 COPY . .
4 RUN pip install -r requirements.txt
5 EXPOSE 8080
6 CMD ["python", "app.py"]
7 |
```

Short note (4–6 lines) explaining what the container does

I created a simple Flask web app, containerized it using Docker, and deployed it to Google Cloud Run. The container runs a lightweight Python app and can scale automatically based on incoming requests. The public Cloud Run URL shows my deployed app running live