

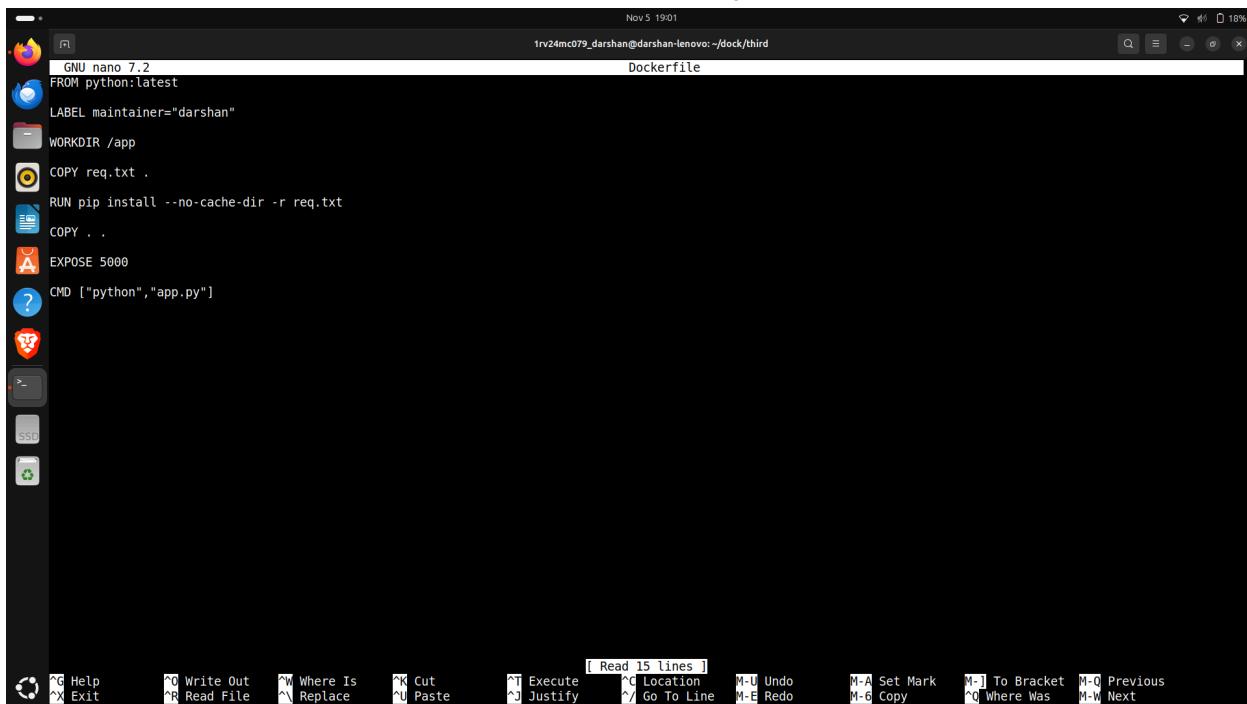
Program3

Code a Dockerized python Flask or Node.js application

Step 1 - Create the following file structure

```
prgm3
|---- Dockerfile
|---- app.py
|---- requirements.txt
```

Step 4 - Write the Dockerfile, requirements.txt and [app.py](#)



The screenshot shows a terminal window titled "Dockerfile" with the following content:

```
GNU nano 7.2
FROM python:latest
LABEL maintainer="darshan"
WORKDIR /app
COPY req.txt .
RUN pip install --no-cache-dir -r req.txt
COPY . .
EXPOSE 5000
CMD ["python", "app.py"]
```

The terminal window has a dark theme and includes standard nano editor navigation keys at the bottom.

Nov 5 19:01
1rv24mc079_darshan@darshan-lenovo: ~/dock/third

```
GNU nano 7.2
from flask import Flask
app = Flask(__name__)
@app.route("/")
def home():
    return("Hello from python ")
app.run(host="0.0.0.0",port=5000)
```

The terminal window shows a Python Flask application code in `app.py`. The code defines a single endpoint '/' that returns the string "Hello from python ". It then runs the application with host set to "0.0.0.0" and port set to 5000.

Step 5 - Build the Docker image

```
docker build -t prgm3 .
```

Nov 5 19:01
1rv24mc079_darshan@darshan-lenovo: ~/dock/third\$ docker build -t prgm3 .

```
[+] Building 2.6s (10/10) FINISHED
  => [internal] load build definition from Dockerfile
  => => transferring dockerfile: 2078
  => [internal] load metadata for docker.io/library/python:latest
  => [internal] load dockerignore
  => => transferring context: 2B
  => [1/5] FROM docker.io/library/python:latest@sha256:1ad1a43b5e2478e62056bbc28028af858185d73bf4d6a439ccb858b6800a96d
  => => resolve docker.io/library/python:latest@sha256:1ad1a43b5e2478e62056bbc28028af858185d73bf4d6a439ccb858b6800a96d
  => [internal] load build context
  => => transferring context: 848
  => CACHED [2/5] WORKDIR /app
  => CACHED [3/5] COPY req.txt .
  => CACHED [4/5] RUN pip install --no-cache-dir -r req.txt
  => CACHED [5/5] COPY . .
  => => exporting to image
  => => exporting layers
  => => writing image sha256:41aca1ceea58167b18fbebeba3528dd62c4f50ec9fc8dd8b3ab139f42fbfa
  => => naming to docker.io/library/prgm3
1rv24mc079_darshan@darshan-lenovo:~/dock/third$
```

The terminal window shows the output of the `docker build -t prgm3 .` command. The build process completed successfully in 2.6 seconds, creating a Docker image named `prgm3`.

Step 6 - Build and Run the container

```
docker run -p 5000:5000 prgm3
```

Step 7 - Verify the output

<https://localhost:5000>

