

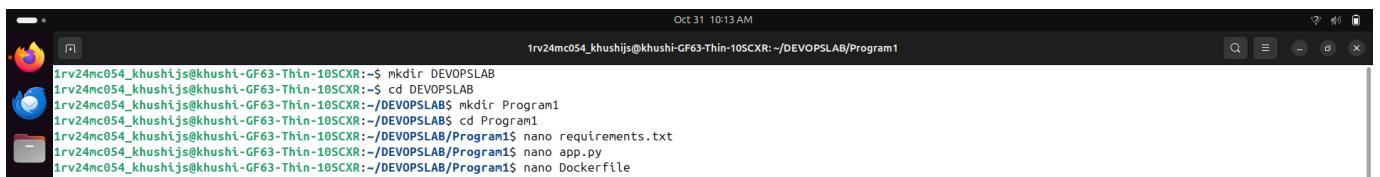
# Program 1 : Build a Docker Container from a Custom Dockerfile

## Folder Structure

### DEVOPSLAB

```
|--Program1  
|  |--app.py  
|  |--requirements.txt  
|  |--Dockerfile
```

**Step 1:** I've created a directory called DEVOPSLAB navigate to DEVOPSLAB directory and within that create a sub-directory with name Program1, navigate to Program1 directory and create the files such as [app.py](#), requirements.txt and Dockerfile.



```
Oct 31 10:13 AM  
1rv24mc054_khushij@khushi-GF63-Thin-105CXR:~/DEVOPSLAB$ mkdir DEVOPSLAB  
1rv24mc054_khushij@khushi-GF63-Thin-105CXR:~/DEVOPSLAB$ cd DEVOPSLAB  
1rv24mc054_khushij@khushi-GF63-Thin-105CXR:~/DEVOPSLABS$ mkdir Program1  
1rv24mc054_khushij@khushi-GF63-Thin-105CXR:~/DEVOPSLABS$ cd Program1  
1rv24mc054_khushij@khushi-GF63-Thin-105CXR:~/DEVOPSLAB/Program1$ nano requirements.txt  
1rv24mc054_khushij@khushi-GF63-Thin-105CXR:~/DEVOPSLAB/Program1$ nano app.py  
1rv24mc054_khushij@khushi-GF63-Thin-105CXR:~/DEVOPSLAB/Program1$ nano Dockerfile
```

**Step 2:** Add the following code in requirements.txt file, which is used to install the Dependencies.



```
Oct 31 10:45 AM  
1rv24mc054_khushij@khushi-GF63-Thin-105CXR:~/DEVOPSLAB/Program1$ nano requirements.txt  
GNU nano 7.2  
flask
```

The screenshot shows a terminal window with the command "nano requirements.txt" running. The nano editor interface is visible, showing the word "flask" as the initial content of the file. The terminal window has a dark theme with white text. The title bar shows the path "1rv24mc054\_khushij@khushi-GF63-Thin-105CXR:~/DEVOPSLAB/Program1". The bottom of the screen displays the nano editor's keyboard shortcuts.

**Step 3:** In the app.py file, implement the following code to run a basic web server. When accessed, this server will display "Hello, Docker!".

```
GNU nano 7.2
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello Docker!"

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

The screenshot shows a terminal window titled "Oct 31 10:45 AM" with the command "1rv24mc054\_khushij@khushi-GF63-Thin-10SCXR: ~/DEVOPSLAB/Program1". The file "app.py" is open, displaying the provided Python code. The terminal interface includes a toolbar with icons for file operations like Open, Save, and Print, and a menu bar with options like Help, Exit, Write Out, Read File, Where Is, Replace, Cut, Paste, Execute, Justify, Location, Go To Line, Set Mark, To Bracket, Copy, Undo, Redo, Previous, Next, Back, and Forward. The bottom of the window shows a series of keyboard shortcuts for these functions.

**Step 4:** The following Dockerfile creates a lightweight container. It installs Python and Flask dependencies, copies the code to /app, and then runs the Flask server on port 5000.

```
GNU nano 7.2
#Use a slim Python base image
FROM python:3.9-slim

#Set working directory
WORKDIR /app

#Copy the requirements file and install dependencies
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt

#Copy the application code
COPY .

#Set Flask environment variables
ENV FLASK_APP=app.py
ENV FLASK_RUN_HOST=0.0.0.0
ENV FLASK_RUN_PORT=5000

#Expose the port
EXPOSE 5000

#Define the commands to run the flask application
CMD ["flask", "run"]
```

The screenshot shows a terminal window titled "Oct 31 10:45 AM" with the command "1rv24mc054\_khushij@khushi-GF63-Thin-10SCXR: ~/DEVOPSLAB/Program1". The file "Dockerfile" is open, displaying the provided Dockerfile code. The terminal interface includes a toolbar with icons for file operations like Open, Save, and Print, and a menu bar with options like Help, Exit, Write Out, Read File, Where Is, Replace, Cut, Paste, Execute, Justify, Location, Go To Line, Set Mark, To Bracket, Copy, Undo, Redo, Previous, Next, Back, and Forward. The bottom of the window shows a series of keyboard shortcuts for these functions.

**Step 5:** Next, use the docker build command to create a new image from the Dockerfile.

```
1rv24mc054_khushij@khushi-GF63-Thin-10SCXR:~/DEVOPSLAB/Program1$ sudo docker build -t flask_app .
[sudo] password for 1rv24mc054_khushij:
[+] Building 7.6s (10/10) FINISHED
--> [internal] load build definition from Dockerfile
--> == transferring dockerfile: 501B
--> [internal] load metadata for docker.io/library/python:3.9-slim
--> [internal] load .dockerrcignore
--> == transferring context: 2B
--> [internal] load build context
--> == transferring context: 757B
--> [1/5] FROM docker.io/library/python:3.9-slim@sha256:545badebace9a958b98d3e272f0f0d46c0a1a389ac77e24c33f2e7b548ce1b6b
--> CACHED [2/5] WORKDIR /app
--> CACHED [3/5] COPY requirements.txt .
--> CACHED [4/5] RUN pip install --no-cache-dir -r requirements.txt
--> [5/5] COPY .
--> exporting to image
--> exporting layers
--> == writing image sha256:be7143bfa32c010d9241a915f704578d13d465359b37bb8b3ac623e956dc88ce
--> == naming to docker.io/library/flask_app
```

**Step 6:** Execute the docker run command to create a container from the previously built image.

```
1rv24nc054_khushij@khushi-GF63-ThIn-105CXR:~/DEVOPSLAB/Program1$ sudo docker run -p 5000:5000 flask_app
 * Serving Flask app 'app.py'
 * Debug mode: off
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://172.18.0.2:5000
Press CTRL+C to quit
172.18.0.1 - - [31/Oct/2025 10:12:47] "GET / HTTP/1.1" 200 -
172.18.0.1 - - [31/Oct/2025 10:12:54] "GET / HTTP/1.1" 200 -
172.18.0.1 - - [31/Oct/2025 10:12:54] "GET /favicon.ico HTTP/1.1" 404 -
```

