

Program 1: Build a Docker Container from a Custom Dockerfile

Step 1 : Create a Dockerfile

```
GNU nano 7.2                                            Dockerfile
# Use a slim Python base image
FROM python:3.9-slim
# Set the working directory inside the container
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 the Flask app will run on
EXPOSE 5000
# Define the command to run the Flask application
CMD ["flask", "run"]
```

Step 2 : Create python file [app.py](#)

```
GNU nano 7.2                                         app.py
from flask import Flask

app = Flask(__name__)

@app.route('/')
def home():
    return "Hello, Docker!"

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

Step 3 : Create a simple text file - requirements.txt

Step 4 : Execute the Docker build command on the terminal

```
> docker build -t program1 .
```

```
--> naming to Dockerfile /app/Dockerfile
1rv24mc101_shreya@shreya-Lenovo-IdeaPad-S145-15IWL:~/program1$ docker build -t program1 .
[+] Building 4.4s (10/10) FINISHED v.4s
      docker:default
--> [internal] load build definition from Dockerfile          0.1s
--> => transferring dockerfile: 552B                          0.0s
--> [internal] load metadata for docker.io/library/python:3.9-slim 3.4s
--> [internal] load .dockerignore                            0.1s
--> => transferring context: 2B                            0.0s
--> [1/5] FROM docker.io/library/python:3.9-slim@sha256:2d97f6910b16bd338 0.0s
--> [internal] load build context                          0.1s
--> => transferring context: 93B                          0.0s
--> CACHED [2/5] WORKDIR /app                           0.0s
--> CACHED [3/5] COPY requirements.txt .                  0.0s
--> CACHED [4/5] RUN pip install --no-cache-dir -r requirements.txt 0.0s
--> CACHED [5/5] COPY . .                                0.0s
--> exporting to image                                    0.1s
--> => exporting layers                                 0.0s
--> => writing image sha256:08a5c2d7ce2f1025fad0228cb314e3f6315a97aa7fb4c 0.0s
--> => naming to docker.io/library/program1             0.0s
1rv24mc101_shreya@shreya-Lenovo-IdeaPad-S145-15IWL:~/program1$ docker run -d -p 5000:5000 --name flask-container p
^[[M1000 070100 H00 00070 77 001 00 10000000 00 00000000 7
```

Step 5 : Run the Docker run command

```
> docker run -d -p 5000:5000 -program1
```

```
--> exporting layers
--> => writing image sha256:e7f426a24079f07745c078baa19ea20e12f0c50fdfaf6 0.0s
--> => naming to docker.io/library/program1                0.0s
2528239db53a9b261b7ccc3c13d2d971a597798a5a7475a5f00892dabde2b839
1rv24mc101_shreya@shreya-Lenovo-IdeaPad-S145-15IWL:~/program1$ docker ps
CONTAINER ID   IMAGE       COMMAND     CREATED      STATUS      PORTS          NAMES
2528239db53a   program1   "flask run"  9 seconds ago  Up 8 seconds  0.0.0.0:5000->5000/tcp, [::]:5000->5000/tcp   flask-conta
iner
```

Step 6 : Test and verify the localhost text -> “Hello Docker!”

