

# Program 1:

## Build a Docker container from a custom Dockerfile

### 31/10/2025

#### Project Directory:

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

#### Program code and steps for execution:

Step 1: Create a directory with the name mylab for all the lab programs

Then again create another directory with the name prog1 move to the prog1 directory

```
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~$ mkdir mylab
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~$ cd mylab/
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab$ mkdir prog1
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ █
```

Step 2: Inside the prog1 now create the Dockerfile

```
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ sudo nano Dockerfile
[sudo] password for 1RV24MC030_chinmayi_m_h:
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ cat Dockerfile
###Dockerfile

#This is the Dockerfile which contains the information for Docker to build the container images

#use the slim python as base image
FROM python:3.9-slim

#set the working directory for the container
WORKDIR /app

#copy the text/content from the requirements.txt and install the dependencies
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt

#copy the application code
COPY . .

#set the flask environment variables
ENV FLASK_APP=app.py
ENV FLASK_RUN_HOST=0.0.0.0
ENV FLASK_RUN_PORT=5000

#expose the port that flask will run an app on
EXPOSE 5000

#define the command to run the flask application
CMD ["flask", "run"]
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ █
```

### Step 3: Inside the prog1 now create the app.py

```
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ sudo nano app.py
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ cat app.py
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)
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ █
```

### Step 4: Inside the prog1 now create the requirements.txt

```
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ sudo nano requirements.txt
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ cat requirements.txt
flask
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$
```

### Step 5: Now the project directory of prog1 looks like the below

```
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ ls
app.py  Dockerfile  requirements.txt
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$
```

### Step 6: Now build the docker from the Dockerfile prog1

```
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ docker build -t prog1 .
[+] Building 2.8s (10/10) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 698B
=> [internal] load metadata for docker.io/library/python:3.9-slim
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/5] FROM docker.io/library/python:3.9-slim@sha256:545badebace9a958b98d3e272f0f0d46c0a1a389ac77e24c33f2e7b548ce1b6b
=> [internal] load build context
=> => transferring context: 950B
=> 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 sha256:65b11833f00ef71494be7858be453011fafcdd818da101495267b52aaee32d2e
=> => naming to docker.io/library/prog1
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$
```

Step 7: Now run the docker of prog1 with the port of 5000

Step 8: Now check for the running containers

```
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ docker run -d -p 5000:5000 --name flask-container prog1
b77d0371ed7463b3f84d51b550e82c732118089a7ae6a8092892568b3e2140ad
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS                               NAMES
b77d0371ed74   prog1     "flask run"   6 seconds ago   Up 5 seconds   0.0.0.0:5000->5000/tcp, [::]:5000->5000/tcp   flask-container
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$
```

Step 8: In the browser search for <http://localhost:5000/>



Step 9: Stop the created container, as we named it as flask-container use that name itself,  
Also remove the container by using the same name

```
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ docker stop flask-container
flask-container
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ docker rm flask-container
flask-container
```

Step 10: Now delete the image which you have created for prog1

```
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
prog2         latest   fe9b99b6dc6a   15 minutes ago  137MB
prog1         latest   65b11833f00e   25 minutes ago  132MB
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ docker rmi prog1
Untagged: prog1:latest
(base) 1RV24MC030_chinmayi_m_h@chinmayi-m-h-HP-Pavilion-Laptop-15-eg3xxx:~/mylab/prog1$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
prog2         latest   fe9b99b6dc6a   15 minutes ago  137MB
```