Lab 1: Building and Running a Python Flask App with Docker

This document outlines the folder structure and step-by-step procedure to complete the lab program.

1. Folder Structure

Your project should have a single folder (e.g., Program 1) containing the following three files. This flat structure is all that is needed for this project.

- **Program 1/** (Your main project folder)
  - **Dockerfile**: The text file containing instructions for Docker to build the image.
  - **app.py**: The Python web application code using the Flask framework.
  - **requirements.txt**: A list of Python dependencies (libraries) that app.py needs to run.

2. Source Code

Make sure the contents of your three files are correct.

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)
```

requirements.txt

```
Flask
```

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"]
```

3. Step-by-Step Procedure

Follow these commands in your terminal to build the Docker image and run the container.

Prerequisite: You must have Docker installed and running on your computer.

Step 1: Open Your Terminal

Open your terminal or command prompt (e.g., Terminal, PowerShell, cmd).

Step 2: Navigate to Your Project Folder

Use the cd (change directory) command to go into the folder where your three files are located.

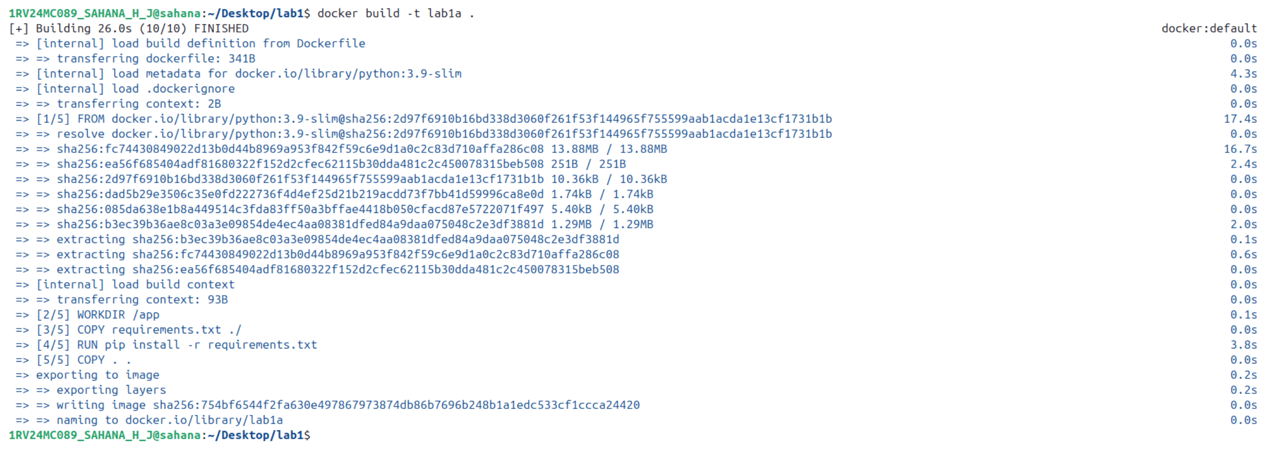
```
# Example path based on your screenshots:
cd /home/Sahana/DevOps/Program 1

# Note: If your folder path has spaces, you might need to use quotes:
# cd "/home/Sahana/DevOps/Program 1"
```

Step 3: Build the Docker Image

This command tells Docker to read your Dockerfile and build an image.

docker build -t my-flask-app .

- docker build: The command to start the build process.
- -t my-flask-app: The -t flag "tags" your image with a human-readable name (e.g., my-flask-app).
- .: This dot is important. It tells Docker to find the Dockerfile in the current directory.



```
1RV24MC089_SAHANA_H_J@sahana:~/Desktop/lab1$ docker build -t lab1a .
[+] Building 26.0s (10/10) FINISHED                                                                               docker:default
 => [internal] load build definition from Dockerfile                                                                        0.0s
 => => transferring dockerfile: 341B                                                                                        0.0s
 => [internal] load metadata for docker.io/library/python:3.9-slim                                                          4.3s
 => [internal] load .dockerignore                                                                                           0.0s
 => => transferring context: 2B                                                                                             0.0s
 => [1/5] FROM docker.io/library/python:3.9-slim@sha256:2d97f6910b16bd338d3060f261f53f144965f755599aab1acda1e13cf1731b1b   17.4s
 => => resolve docker.io/library/python:3.9-slim@sha256:2d97f6910b16bd338d3060f261f53f144965f755599aab1acda1e13cf1731b1b    0.0s
 => => sha256:fc74430849022d13b0d44b8969a953f842f59c6e9d1a0c2c83d710affa286c08 13.88MB / 13.88MB                           16.7s
 => => sha256:ea56f685404adf81680322f152d2cfec62115b30dda481c2c450078315beb508 251B / 251B                                  2.4s
 => => sha256:2d97f6910b16bd338d3060f261f53f144965f755599aab1acda1e13cf1731b1b 10.36kB / 10.36kB                            0.0s
 => => sha256:dad5b29e3506c35e0fd222736f4d4ef25d21b219acdd73f7bb41d59996ca8e0d 1.74kB / 1.74kB                              0.0s
 => => sha256:085da638e1b8a449514c3fda83ff50a3bffae4418b050cfacd87e5722071f497 5.40kB / 5.40kB                              0.0s
 => => sha256:b3ec39b36ae8c03a3e09854de4ec4aa08381dfed84a9daa075048c2e3df3881d 1.29MB / 1.29MB                              2.0s
 => => extracting sha256:b3ec39b36ae8c03a3e09854de4ec4aa08381dfed84a9daa075048c2e3df3881d                                   0.1s
 => => extracting sha256:fc74430849022d13b0d44b8969a953f842f59c6e9d1a0c2c83d710affa286c08                                   0.6s
 => => extracting sha256:ea56f685404adf81680322f152d2cfec62115b30dda481c2c450078315beb508                                   0.0s
 => [internal] load build context                                                                                           0.0s
 => => transferring context: 93B                                                                                            0.0s
 => [2/5] WORKDIR /app                                                                                                      0.1s
 => [3/5] COPY requirements.txt ./                                                                                          0.0s
 => [4/5] RUN pip install -r requirements.txt                                                                               3.8s
 => [5/5] COPY . .                                                                                                          0.0s
 => exporting to image                                                                                                      0.2s
 => => exporting layers                                                                                                     0.2s
 => => writing image sha256:754bf6544f2fa630e497867973874db86b7696b248b1a1edc533cf1ccca24420                                0.0s
 => => naming to docker.io/library/lab1a                                                                                    0.0s
1RV24MC089_SAHANA_H_J@sahana:~/Desktop/lab1$
```

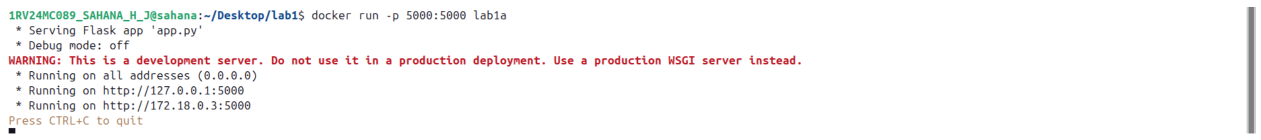
Step 4: Run the Docker Container

Once the image is built, use this command to run it as a live container.

docker run -p 5000:5000 my-flask-app

- docker run: The command to create and start a container from an image.
- -p 5000:5000: This "publishes" or "maps" the port. It connects **port 5000 on your computer** (the first 5000) to **port 5000 inside the container** (the second 5000), which is where the Flask app is running.
- my-flask-app: The name of the image you want to run.

Your terminal will now show output from the Flask server, indicating it is running.



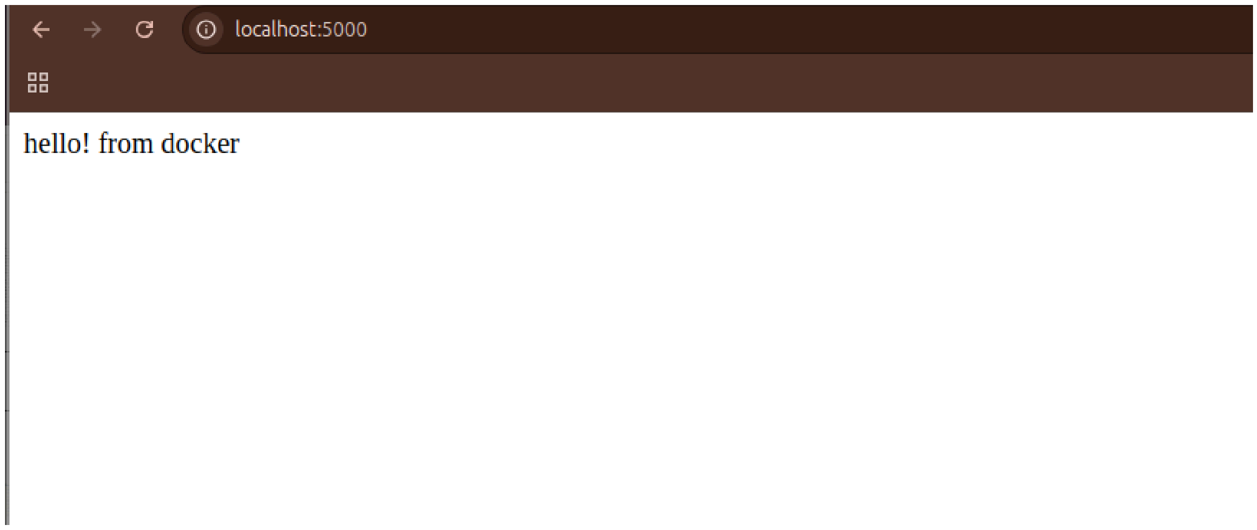
```
1RV24MC089_SAHANA_H_J@sahana:~/Desktop/lab1$ docker run -p 5000:5000 lab1a
 * Serving Flask app 'app.py'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.18.0.3:5000
Press CTRL+C to quit
```

Step 5: Test Your Application

Open your web browser and navigate to the following address:

http://localhost:5000

You should see a webpage that displays the text: **Hello, Docker!**



```
hello! from docker
```

Step 6: Stop the Container

When you are finished, you can stop the running container by going back to your terminal and pressing **Ctrl + C**.