Program 1

Project Structure:

```
lab1-devops/
│
├── app.py
├── requirements.txt
├── Dockerfile
├── flenv/            # Python virtual environment (local)
└── README.md
```

Dockerfile:

```
# lab 1 file
FROM python:3.9-slim

WORKDIR /app

COPY requirements.txt .
RUN pip install -r requirements.txt

LABEL maintainer="Kunal"

LABEL version="1.0"

COPY . .

ENV FLASK_APP=app.py
ENV FLASK_RUN_HOST=0.0.0.0
ENV FLASK_RUN_PORT=5000

EXPOSE 5000

CMD ["flask", "run"]
```

app.py

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def hello():
   return "hello world from container"

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

Requirements.txt
Flask

Setup and Installation:

Make sure Python 3 is installed:

```
python3 --version
```

If not installed, you can install it (on Ubuntu/Debian):

```
sudo apt update
sudo apt install python3 python3-venv python3-pip -y
```

In your project folder (e.g., `~/lab1-devops`):

```
python3 -m venv flenv
```

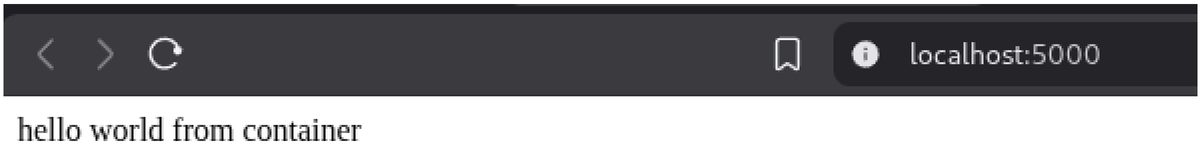
Activate the environment:

```
source flenv/bin/activate
```

```
(flenv)─(singh㉿singh)-[~/lab1-devops]
 └─$ sudo docker build -t pythonapp:latest .

┌──(flenv)─(singh㉿singh)-[~/lab1-devops]
└─$ sudo docker run -d -p 5000:5000 --name contain-myapplication pythonapp
```

Open:
http://localhost:5000



hello world from container

Program 2

Project Structure:

```
lab2-devops/
│
├── Dockerfile
├── package.json
├── package-lock.json
├── server.js
├── node_modules/
└── README.md
```

Dockerfile:

```
FROM node:18-alpine AS builder

ENV APP_PORT=3000

WORKDIR /app

COPY package*.json ./

RUN npm install

COPY . .

RUN npm run build

EXPOSE ${APP_PORT}

ENV PORT=${APP_PORT}

CMD ["node", "server.js"]


## developer stage
FROM node:18-alpine AS tester

ENV APP_PORT=3000

WORKDIR /app

COPY --from=builder /app /app
```

```
RUN adduser -D app

USER app

EXPOSE ${APP_PORT}

ENV PORT=${APP_PORT} NODE_ENV=development

CMD ["node", "server.js"]
```

server.js:

```
const express = require('express');
const app = express();
const port = 3000;

app.get('/', (req, res) => {
 res.send('Hello World from docker container!');
});

app.listen(port, () => {
 console.log(`Server is running at http://localhost:${port}`);
});
```

Package.json:

```
{
 "dependencies": {
   "express": "^5.1.0"
 },
 "scripts": {
   "start": "node server.js",
   "build": "echo \"running build\" "
 },
 "name": "lab2-devops",
 "version": "1.0.0",
 "main": "server.js",
 "devDependencies": {},
 "repository": {
   "type": "git",
   "url": "git+https://github.com/vighneshhm02/lab2-devops.git"
 },
 "keywords": [],
 "author": "",
 "license": "ISC",
 "bugs": {
   "url": "https://github.com/vighneshhm02/lab2-devops/issues"
 },
 "homepage": "https://github.com/vighneshhm02/lab2-devops#readme",
 "description": ""
}
```

Setup and Installation:

Verify Node.js and npm installation:
Node –version
Npm –version

Initialize npm:
Npm init -y
(This creates a package.json file with default settings)

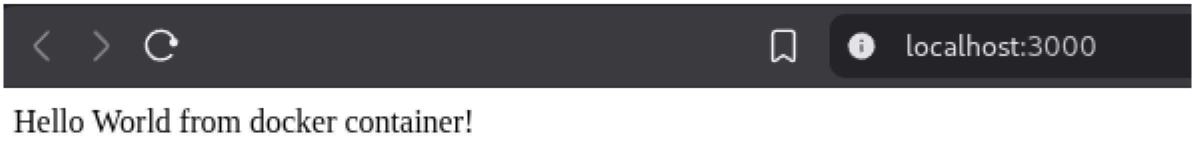
Install express:
Npm install express
(Express will be added to dependencies in package.json)

Build Builder Image
sudo docker build –target builder -t jsapp-bui .

Build Tester Image
sudo docker build –target tester -t jsapp-tes .
(This creates the final image using the built files from the first stage.)

```
(singh㉿singh)-[~/lab2-devops]
└─$ sudo docker run -d -p 3000:3000 --name jsbui jsapp-bui
```

Open:
http://localhost:3000



Hello World from docker container!