Lab Task: Collaborative Flask Application with CI/CD & Docker

Objective: To build, containerize, and continuously deploy a Flask web application collaboratively using GitHub, Docker, and a CI/CD pipeline (GitHub Actions).

Team Setup

- **Total Members:** 3
- **Roles:**
    - **Member 1 (Backend Lead):** Implements core Flask routes and logic.
    - **Member 2 (Frontend/API Integration):** Works on templates, static files, and API endpoints.
    - **Member 3 (DevOps Engineer):** Handles Docker configuration, testing, and CI/CD pipeline setup.

Project Structure

```
flask_lab_project/
│
├── main/                    # Main directory
│   ├── app.py               # Main Flask app
│   ├── requirements.txt
│   ├── Dockerfile
│   ├── tests/               # Unit tests
│   │   └── test_app.py
│   ├── templates/
│   ├── static/
│   └── .github/
│       └── workflows/
│           └── ci-cd.yml    # GitHub Actions pipeline
│
├── member1_backend/         # Subdirectory for backend work
├── member2_frontend/        # Subdirectory for frontend/API work
└── member3_devops/          # Subdirectory for CI/CD & Docker work
```

Tasks Breakdown

Step 1 – Repository Setup

1. Create a new **GitHub repository** named flask-lab-project.
2. Add all 3 members as **collaborators**.
3. Each member clones the repo and creates their **own branch** (backend, frontend, devops).

Step 2 – Flask Application

- Create a simple Flask app in main/app.py with at least:
    - A homepage route (/)
    - A health check route (/health)
    - A simple POST endpoint (/data)

Step 3 – Docker Setup

1. Create a Dockerfile in the main/ directory
2. Test locally

Step 4 – Unit Tests

In main/tests/test_app.py: filewrite some unit tests for example:

```
from app import app

def test_home():
    response = app.test_client().get('/')
    assert response.status_code == 200

def test_health():
    response = app.test_client().get('/health')
    assert b"OK" in response.data
```

Step 5 – CI/CD Pipeline (GitHub Actions)

Create a .github/workflows/ci-cd.yml file:

Specifications:

- Triggers when **any member pushes to the main branch**.
- Runs automated tests.
- Builds Docker image.
- Optionally pushes image to Docker Hub.

Step 6 – Collaboration Workflow

- Each member works in their own directory & branch.
- Merge via **Pull Requests (PRs)**.
- The CI/CD pipeline runs automatically on merges or commits to main.

Step 7 – Submission

Each group must submit:

- GitHub repo link
- Screenshot of a successful CI/CD run
- Screenshot of the app running via Docker
- Brief documentation (README.md) describing:
    - Each member's role
    - How to build, test, and run the project