**Project 6: Flask Web Application with CI/CD**

**✅ Project Objective:**

To **develop a Flask-based web application** and set up a **CI/CD pipeline** that automates the following:

* Code integration
* Docker image building
* Deployment to a container
* Optional push to cloud

**🧰 Tools & Technologies Used:**

* **Flask (Python Web Framework)**
* **GitHub** (Code repository)
* **Jenkins** (CI/CD Server)
* **Docker** (Containerization)
* **GitHub Webhooks**
* **Linux (Ubuntu/CentOS environment)**

**⚙️ Use Case:**

A web app that displays basic content like:

python

Copy code

@app.route("/")

def home():

return "Welcome to My Flask App with CI/CD!"

Once code is pushed to GitHub:

* Jenkins detects the push
* Jenkins builds the Docker image
* The image is deployed automatically
* You can access the running app from the browser

**🔄 CI/CD Pipeline Steps:**

1. **GitHub → Jenkins Webhook:**
   * Connect GitHub repo to Jenkins using webhook
2. **Jenkins Job:**
   * Pulls code from GitHub
   * Builds a Docker image
   * Stops old container
   * Runs new container with latest app
3. **Dockerfile Sample:**

Dockerfile

Copy code

FROM python:3.10

WORKDIR /app

COPY . .

RUN pip install -r requirements.txt

CMD ["python", "app.py"]

1. **Jenkins Pipeline Script:**

groovy

Copy code

pipeline {

agent any

stages {

stage('Clone') {

steps {

git 'https://github.com/username/repo'

}

}

stage('Build Docker') {

steps {

sh 'docker build -t flask-ci-app .'

}

}

stage('Run Docker') {

steps {

sh 'docker rm -f flask-container || true'

sh 'docker run -d --name flask-container -p 5000:5000 flask-ci-app'

}

}

}

}

**🖥️ Expected Output:**

* Jenkins shows all build stages passing.
* A Docker container runs your Flask app.
* App accessible on http://<your-ip>:5000
* Web browser showing the Flask app output

**💡 Benefits:**

* Automates testing and deployment
* Ensures every change is immediately reflected in production
* Beginner-friendly introduction to DevOps tools