PROJECT REPORT

CI/CD Pipeline with GitHub Actions & Docker 🚀

Introduction

Continuous Integration and Continuous Deployment (CI/CD) streamline software development by automating testing, building, and deployment ⚙️. This project demonstrates a complete CI/CD pipeline for a Flask web application using GitHub Actions, Docker, and Minikube for local Kubernetes deployment, eliminating the need for cloud services ☁️❌.

Abstract

The project involves a Flask application that displays a message on a background 🌟. The CI/CD pipeline automates testing ✅, Docker image building 🐳, and pushes the image to Docker Hub 📦. Using Minikube, the Docker image is pulled and deployed locally as a Kubernetes pod 🧩, providing a realistic simulation of cloud deployments while remaining entirely on local infrastructure.

Tools Used 🛠️

★ Flask – Python framework for building the web application 🐍.

★ Docker – Containerization platform for packaging the app 🐳.

★ GitHub Actions – Automates the CI/CD workflow ⚙️.

★ Docker Hub – Repository for storing Docker images 📦.

★ Minikube – Local Kubernetes environment for deployment and testing 🖥️.

★ kubectl – Command-line tool for interacting with Kubernetes clusters 💻.

★ Steps Involved in Building the Project 📝

★ Created app.py page directly from Flask 🐍.

★ Wrote a Dockerfile to containerize the Flask app 🐳.

★ Verified image runs correctly locally via docker build and docker run ✅.

PROJECT REPORT

★ CI/CD Workflow with GitHub Actions

★ Configured .github/workflows/ci-cd.yml to:

★ Checkout the repository code 📂.

★ Run automated tests using pytest ✅.

★ Build Docker image 🐳.

★ Push the image to Docker Hub 📦.

★ Local Deployment with Minikube

★ Started Minikube cluster locally 🖥️.

★ Loaded Docker image into Minikube 🐳➡️🖥️.

★ Deployed the app as a Kubernetes pod 🧩.

★ Accessed the deployed app using: minikube service my-local-app-service 🌟.

★ Verified all tests passed in GitHub Actions ✅.

★ Confirmed the Flask app displayed correctly in the browser via Minikube 🌐.

Conclusion

This project demonstrates a complete local CI/CD pipeline using GitHub Actions, Docker, and Minikube 🖥️. Automating testing, image building, and deployment provides faster ⚡, consistent 🔄, and reliable ✅ delivery. Using Minikube for local Kubernetes deployment simulates cloud environments ☁️, giving hands-on experience with real-world CI/CD practices while remaining fully local.