CI/CD Pipeline with GitHub Actions & Docker

Introduction

In modern software development, Continuous Integration and Continuous Deployment (CI/CD) pipelines are essential for automating builds, testing, and deployments. This project demonstrates a CI/CD pipeline using GitHub Actions, Docker, and Docker Hub, with deployment on a local Minikube or virtual machine setup.

Abstract

The project implements a streamlined CI/CD pipeline to ensure efficient and reliable application delivery. The pipeline automates source code integration, Docker image creation, testing, and deployment. GitHub Actions is used for automation, Docker Hub for container registry, and Minikube for local deployment. This eliminates the dependency on cloud services while maintaining professional DevOps practices.

Tools Used

- GitHub Actions – Automating CI/CD workflows.
- Docker – Containerization of the application.
- Docker Hub – Hosting and sharing container images.
- Minikube – Local Kubernetes cluster for deployment.
- GitHub Repository – Source code and workflow management.

Steps Involved in Building the Project

1. Developed a Dockerfile and docker-compose.yml to containerize the application.
2. Configured GitHub Actions workflow (.github/workflows/ci-cd.yml) for automated builds and tests.
3. Built the Docker image and pushed it to Docker Hub.
4. Pulled the Docker image from Docker Hub into Minikube/local VM.
5. Deployed and verified the application successfully.
6. Captured workflow results and deployment screenshots as deliverables.

Conclusion

This project demonstrates how to set up a robust CI/CD pipeline using GitHub Actions, Docker, and Docker Hub, with deployment on a local environment. The solution shows that automation, testing, and deployment can be achieved without relying on paid cloud services. This project provides a solid foundation for students and professionals to understand DevOps practices and apply them in real-world projects.