CI/CD Pipeline with GitHub Actions & Docker

Abstract

This project demonstrates the implementation of a Continuous Integration and Continuous Deployment (CI/CD) pipeline using GitHub Actions and Docker. The pipeline builds a Node.js application, runs tests, creates a Docker image, pushes it to Docker Hub, and deploys it locally using Docker or Minikube. The project aims to showcase automation in software delivery without relying on external cloud providers.

Introduction

CI/CD pipelines are essential in modern software development, ensuring faster delivery, consistency, and reliability. This project focuses on designing a complete CI/CD pipeline that integrates GitHub Actions for automation and Docker for containerization. By deploying locally, the setup provides a hands-on demonstration without the need for cloud infrastructure.

Tools Used

- **GitHub Actions**: Automates build, test, and deployment processes. - **Docker & Docker Compose**: Used to containerize and run the application. - **Docker Hub**: Serves as the container registry. - **Node.js & Express**: Provides the sample web application. - **Minikube (Optional)**: Local Kubernetes cluster for deployment.

Steps Involved in Building the Project

1. Created a simple Node.js application with Express. 2. Wrote a Dockerfile to containerize the application. 3. Configured docker-compose.yml for local deployment. 4. Created a GitHub Actions workflow (.yml file) to build, test, and push the Docker image. 5. Linked Docker Hub with GitHub Secrets for authentication. 6. Verified pipeline execution in GitHub Actions. 7. Pulled and deployed the image locally using Docker and optionally Minikube.

Conclusion

This project successfully demonstrates a fully automated CI/CD pipeline using GitHub Actions and Docker. It highlights the power of DevOps practices in achieving consistent, repeatable deployments. The approach is scalable and can be extended to real-world projects with cloud-based or on-premise environments.