CI/CD Pipeline Using GitHub Actions, Docker, and Kubernetes (Minikube)

Introduction

In today's software development process, Continuous Integration and Continuous Deployment (CI/CD) play a vital role in automating code build, testing, and deployment. This project demonstrates how to create a fully automated CI/CD pipeline using GitHub Actions, Docker, Docker Hub, and Kubernetes (Minikube) — all executed locally within an Amazon Linux EC2 instance.

Abstract

The project showcases a complete DevOps workflow starting from source code management to deployment automation. A simple Node.js Express application is containerized with Docker, integrated with GitHub Actions for CI/CD automation, and deployed on Minikube (a local Kubernetes environment). The pipeline automatically triggers on every code commit, builds a Docker image, runs tests, and pushes the updated image to Docker Hub.

Tools Used

• Amazon Linux EC2 Instance – Execution environment
• Git & GitHub – Source code management
• GitHub Actions – Automated build workflow
• Docker & Docker Hub – Containerization and image repository
• Minikube & kubectl – Local Kubernetes cluster setup
• Node.js & Express.js – Application framework

Steps Involved in Building the Project

Step 1: Launch EC2 Instance & Install Dependencies

```
sudo yum update -y
sudo yum install -y git docker
sudo systemctl enable docker --now
sudo usermod -aG docker $USER && newgrp docker
```

Step 2: Install Minikube and kubectl

```
curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
sudo install minikube-linux-amd64 /usr/local/bin/minikube
curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
```

Step 3: Initialize Node.js Application

```
mkdir ci-cd-node && cd ci-cd-node
npm init -y
npm install express
```

Step 4: Build and Test Docker Image

```
docker build -t ci-cd-node:local .
docker run -d -p 3000:3000 ci-cd-node:local
```

Step 5: Push Image to Docker Hub

```
docker tag ci-cd-node:local <your_dockerhub_username>/ci-cd-node:latest
docker login
docker push <your_dockerhub_username>/ci-cd-node:latest
```

Step 6: Configure GitHub Actions

Create .github/workflows/ci.yml to automate build and push steps.

Step 7: Deploy on Kubernetes (Minikube)

```
minikube start
kubectl apply -f k8s/deployment.yml
kubectl apply -f k8s/service.yml
kubectl get pods
kubectl get svc
```

Step 8: Access Application

```
minikube service ci-cd-service
```

Conclusion

This project successfully demonstrates an end-to-end CI/CD pipeline setup using open-source tools. It automates the process of building, testing, and deploying applications locally using GitHub Actions and Docker. The deployment on Minikube simulates real-world Kubernetes environments, providing a cost-effective and hands-on DevOps learning experience.