DevOps Java Application Deployment Project Documentation

Project Overview

This project demonstrates the complete lifecycle of building, containerizing, and deploying a Java application using modern DevOps methodologies. The workflow integrates Java build automation with Maven, Docker containerization, continuous integration via Jenkins, Docker Hub image hosting, and Kubernetes deployment using Minikube. It is designed to equip fresh DevOps engineers with hands-on skills essential for industry roles and to build a robust professional portfolio.

Technologies Used

- Java & Maven
- Docker
- Jenkins
- Docker Hub
- Kubernetes (Minikube)
- kubectl CLI

Project Timeline and Tasks

Day 1: Java Build and GitHub Integration

- Developed a Java application using Maven.
- Committed and pushed source code to a GitHub repository.

Day 2: Dockerizing the Application

- Created a Dockerfile for the Java app.
- Built and tested the Docker image locally.

Day 3: Jenkins Continuous Integration

- Installed and configured Jenkins server.
- Created Jenkins pipeline to automate Java app build and Docker image creation.
- Integrated Jenkins with GitHub to trigger builds on code commit.

Day 4: Docker Hub Integration

- Configured Jenkins to push the Docker image to Docker Hub.
- Verified the image is publicly accessible on Docker Hub.

Day 5: Kubernetes Deployment (Minikube)

- Installed and launched Minikube cluster.
- Created Kubernetes Deployment and Service manifests in YAML.
- Deployed the Docker image to Kubernetes using `kubectl apply`.
- Verified pod status and exposed service to access the application.

Day 6: Jenkins Deployment to Kubernetes (Optional Advanced)

- Planned Jenkins pipeline step for automated Kubernetes deployment post image push.
- Requires Jenkins-Kubernetes integration and cluster access setup.

Challenges Faced

- Managing Docker storage space on local environment.

- Troubleshooting delays during Kubernetes pod startup.
- Configuring Jenkins credentials securely for Docker Hub and Kubernetes access.

Skills Gained

- Proficiency in Java build automation and Docker containerization.
- Ability to create and manage Jenkins CI/CD pipelines.
- Experience handling Docker image repositories on Docker Hub.
- Practical knowledge of Kubernetes cluster management and YAML manifest configuration.
- Troubleshooting skills in container orchestration environments.

Conclusion

This project provided valuable practical experience with key DevOps tools and workflows, emphasizing automation of software delivery from code to production. Mastery of these processes substantially improves readiness for roles that involve continuous integration, container orchestration, and cloud-native deployments, making it a strong foundation for a career in DevOps.