(Nipur Jain)

GitOps Workflow Implementation using ArgoCD on Kubernetes

Project Repository: https://github.com/NipurJain4/nginx.git

Introduction

This project demonstrates the implementation of a complete GitOps workflow using ArgoCD on Kubernetes. GitOps is a modern approach to continuous deployment that uses Git repositories as the single source of truth for declarative infrastructure and applications. The project showcases automated deployment synchronization, version control integration, and infrastructure as code principles.

Abstract

Successfully implemented a GitOps pipeline that automatically synchronizes Kubernetes deployments from a Git repository using ArgoCD. The solution includes a containerized nginx application with 2 replicas, automated deployment updates through Git commits, and real-time monitoring of application health. The implementation demonstrates modern DevOps practices including infrastructure as code, automated deployments, and declarative configuration management.

Tools Used

■ Docker Desktop - Kubernetes cluster platform
■■ Kubernetes - Container orchestration
■ ArgoCD - GitOps continuous deployment
■ GitHub - Git repository hosting
■ Nginx - Web server application
■■ kubectl CLI - Kubernetes management

Steps Involved in Building the Project

1. Environment Setup

• Configured Docker Desktop with Kubernetes enabled
• Verified cluster connectivity and kubectl access

2. ArgoCD Installation

• Deployed ArgoCD on Kubernetes cluster in dedicated namespace
• Installed all required components (server, controller, repo-server, redis)
• Verified ArgoCD services are running and healthy

3. Repository Preparation

• Created Git repository with Kubernetes manifests
• Added deployment.yaml for nginx application (2 replicas)
• Added service.yaml for ClusterIP service
• Committed and pushed to GitHub repository

4. ArgoCD Application Configuration

• Configured ArgoCD application to monitor Git repository
• Set up automatic synchronization policies
• Established connection between ArgoCD and GitHub repo

5. GitOps Workflow Testing

• Made changes to deployment manifests
• Committed changes to Git repository
• Observed automatic synchronization by ArgoCD
• Verified deployment updates in Kubernetes cluster

6. Monitoring & Verification

• Confirmed application health status: Healthy
• Verified sync status: Synced
• Validated proper functioning of GitOps pipeline

Key Achievements

■ Automated Deployment: Zero-downtime deployments triggered by Git commits
■ Infrastructure as Code: All configurations stored in version control
■ Real-time Sync: ArgoCD continuously monitors and syncs repository changes
■ Application Health: Automated health checks and status monitoring

Project Status

```
ArgoCD Application: nginx-app
Sync Status: Synced
Health Status: Healthy
Deployment: nginx-deployment (2/2 replicas ready)
Service: nginx-service (ClusterIP)
```

Conclusion

The GitOps workflow implementation successfully demonstrates modern DevOps practices with automated, reliable, and traceable deployments. The solution provides a robust foundation for continuous deployment with Git as the single source of truth. ArgoCD's declarative approach ensures consistency between desired and actual cluster state, while the automated synchronization reduces manual intervention and potential errors. This implementation serves as a scalable template for enterprise-grade GitOps workflows and establishes best practices for cloud-native application deployment.

Project Status: ■ Completed Successfully