DEVOPS INTERMEDIATE 3 DAYS

Kubernetes and GitOps with FluxCD

Build a strong foundation in pull-based GitOps delivery using FluxCD, from cluster bootstrapping to multi-environment deployments. Learn how to deploy with Helm and Kustomize, enable drift detection and self-healing, implement notifications, and deliver production-ready workflows with rollback and troubleshooting practices.

Why FluxCD GitOps?

Faster and safer deployments

Git becomes the source of truth for controlled, auditable releases.

Better security and compliance

Changes are tracked, reviewed, and rolled back through Git history.

Improved reliability

FluxCD continuously reconciles cluster state and auto-corrects drift.

Operational efficiency

Reduce manual kubectl work with automated and predictable delivery workflows.

Standardized environments

Consistent deployment patterns across dev, staging, and production clusters.

What you'll be able to do

- Understand GitOps principles and why pull-based deployment is the modern standard.
- Understand FluxCD architecture and core concepts including sources, reconciliation, and controllers.
- Install and configure FluxCD in a Kubernetes cluster.
- Design GitOps repository structures for single and multi-environment delivery (dev/stage/prod).
- Deploy and manage applications using FluxCD with raw YAML, Kustomize, and HelmReleases.
- Implement production-ready Kubernetes workloads using probes, resource requests/limits, securityContext, and config patterns.
- Apply GitOps lifecycle workflows including promotion via pull requests, rollback via Git revert, and drift self-healing.
- Troubleshoot Kubernetes and FluxCD failures including sync issues, Kustomize build errors, Helm reconciliation problems, and workload failures.
- Deliver a complete multi-service GitOps deployment as a capstone.

Who this is for

- DevOps Engineers
- SRE Engineers
- Platform Engineers
- Kubernetes Administrators
- Cloud Engineers deploying workloads on Kubernetes
- Developers working with cloud-native applications

Curriculum

01 GitOps Foundations (Why Pull-Based Deployment Wins)

- What GitOps is and why pull-based CD is the modern standard
- Git as the single source of truth (declarative desired state)
- Reconciliation loop concept (continuous convergence)
- GitOps vs push-based deployments (security, auditability, rollbacks)
- **Hands-on:** Activity: Convert a traditional deployment workflow into a GitOps pull-based flow

02 FluxCD Architecture and Core Concepts

- FluxCD components overview (sources, controllers, reconciliation)
- Key objects (GitRepository, Kustomization, HelmRepository, HelmRelease)
- Reconciliation intervals and drift detection behavior
- Multi-tenancy and namespace-scoped GitOps concepts
- **Hands-on:** Lab: Explore Flux controllers and inspect reconciliation status for a sample source

03 Installing and Configuring FluxCD on Kubernetes

- Installation workflow using flux CLI (bootstrap concepts)
- Connecting Flux to Git repository (auth concepts)
- Flux system components and secure setup basics
- Validating installation (health checks, logs, reconcile status)
- **Hands-on:** Lab: Install FluxCD and bootstrap a GitOps repo for a Kubernetes cluster

04 GitOps Repository Design (Single Env and Multi-Env Delivery)

- Repo structure patterns (apps vs infra, mono-repo vs multi-repo)
- Single environment structure (simple production-ready layout)
- Multi-environment structure (dev/stage/prod) and promotion workflow
- Ownership boundaries and scalable structure for multiple teams
- **Hands-on:** Workshop: Design a multi-environment GitOps repo layout with promotion rules

05 Deploying Apps with Raw YAML and Kustomize (Flux Kustomization)

- Deploying raw YAML using Flux Kustomization
- Kustomize fundamentals (bases, overlays, patches)
- Config patterns using ConfigMaps and Secrets (safe injection concepts)
- Validation and build failures (common Kustomize error patterns)
- **Hands-on:** Lab: Deploy an application using Kustomize overlays for dev and prod environments

06 Helm GitOps with Flux (HelmReleases and Reconciliation)

- Helm in GitOps workflows (values as code)
- HelmRepository and HelmRelease objects
- Version pinning, upgrades, and rollback patterns
- Helm reconciliation troubleshooting (timeouts, chart errors, values issues)
- **Hands-on:** Lab: Deploy an application with HelmRelease and override values per environment

07 Production-Ready Kubernetes Workloads for GitOps

- Probes (liveness/readiness) and safe rollout behavior
- Resource requests/limits and stability patterns (OOMKilled prevention)
- securityContext hardening (runAsNonRoot, drop capabilities concepts)
- Configuration best practices (ConfigMaps, Secrets, env patterns)
- **Hands-on:** Lab: Harden a workload with probes, resources, and securityContext and deploy via Flux

08 GitOps Lifecycle (Promotion, Rollback, Drift Self-Heal)

- Promotion via pull requests (environment overlays and approvals)
- Rollback via Git revert and last-known-good commit pinning
- Drift detection and self-healing behavior in Flux
- Release safety practices (progressive changes and verification)
- **Hands-on:** Lab: Promote a version dev → prod, simulate drift, and rollback using Git revert

09 Troubleshooting FluxCD and Kubernetes Failures

- Sync issues (auth problems, repo fetch failures, source not ready)
- Kustomize build failures (invalid patches, missing resources, wrong paths)
- Helm reconciliation problems (chart fetch failures, values mismatch, timeouts)
- Workload failures (CrashLoopBackOff, ImagePullBackOff, OOMKilled, Service/Ingress issues)
- **Hands-on:** Lab: Fix broken GitOps deployments by reading Flux status, events, and logs

10 Capstone Project (Multi-Service GitOps Deployment with FluxCD)

- Capstone goal: Deliver a complete multi-service deployment using FluxCD
- Deploy services using Kustomize overlays and/or HelmReleases
- Implement production-ready Kubernetes settings (probes, resources, securityContext, config)
- Apply GitOps lifecycle workflows (PR promotion, self-heal, rollback)
- Demonstrate troubleshooting capability with controlled failure scenarios
- **Hands-on:** Capstone Lab: Build and demo a working multi-service GitOps repo managed by FluxCD