DEVOPS INTERMEDIATE 5 DAYS

Docker and Kubernetes Bootcamp

Build a strong foundation in Docker and Kubernetes, from container fundamentals to production-ready deployment and operations. Learn how to troubleshoot real-world failures, apply security best practices, and build reliable workloads with scalable, enterprise-grade Kubernetes patterns.

Why Docker and Kubernetes?

Faster delivery

Docker standardizes runtime environments and Kubernetes makes deployments repeatable and automated.

High availability and reliability

Self-healing workloads, health-based routing, and safe rolling updates reduce outages.

Cost efficiency at scale

Right-sizing resources and scaling only when needed improves infrastructure governance.

Security readiness

Safer defaults like runAsNonRoot reduce risk and improve compliance posture.

Standardization across teams

Common deployment patterns and troubleshooting workflows simplify onboarding.

Scalable platform foundation

Enables microservices, APIs, jobs, and event-driven architectures without changing deployment strategy.

What you'll be able to do

- Containerize applications using Docker with production-ready practices.
- Create optimized Docker images using multi-stage builds and tagging strategies.
- Deploy multi-service applications using Docker Compose.
- Understand Kubernetes architecture and core objects.
- Deploy applications using Deployments, Services, ConfigMaps, and Secrets.
- Configure storage and persistence using PersistentVolumeClaims (PVC).
- Manage safe deployments using probes, rolling updates, and rollback strategies.
- Implement Kubernetes security basics using securityContext and runAsNonRoot.
- Configure resource requests and limits for controlled CPU and memory usage.
- Diagnose and fix common Kubernetes failures including CrashLoopBackOff and OOMKilled.
- Deploy applications using Helm charts with values overrides.
- Troubleshoot broken workloads and fix misconfigurations confidently.
- Deliver a production-style Kubernetes capstone aligned with real enterprise practices.

Who this is for

- Software Engineers containerizing and deploying applications
- DevOps and SRE Engineers managing Kubernetes operations
- Cloud and Platform Engineers building scalable delivery platforms
- QA and Automation Engineers creating consistent test environments
- Technical Leads driving cloud-native adoption and reliability practices

Curriculum

01 Containerization Fundamentals with Docker

- What containers are and why Docker is used in modern delivery
- Docker architecture (images, containers, registry basics)
- Writing a production-ready Dockerfile (layers, caching, best practices)
- Running and managing containers (logs, exec, environment variables)
- **Hands-on:** Containerize a simple application and run it locally using Docker

02 Production-Ready Docker Image Optimization

- Multi-stage builds for smaller, faster, safer images
- Choosing minimal base images and reducing attack surface
- Tagging strategies (semantic versioning, git-sha tags, avoiding latest)
- Image hygiene (pin versions, remove secrets, least privilege runtime)
- **Hands-on:** Build an optimized multi-stage Docker image and compare size + performance

03 Deploying Multi-Service Apps with Docker Compose

- Why Compose is used for local multi-service development environments
- Compose fundamentals (services, networks, volumes)
- Managing app dependencies (database + backend + frontend)
- Environment handling using .env and configuration patterns
- **Hands-on:** Deploy a multi-service application using Docker Compose

04 Kubernetes Architecture and Core Concepts

- Why Kubernetes exists (orchestration, scaling, reliability)
- Kubernetes architecture (control plane, nodes, kubelet, scheduler)
- Core objects overview (Pods, Deployments, Services, Namespaces)
- kubectl workflow (apply, get, describe, logs, exec)
- **Hands-on:** Setup a Kubernetes cluster environment and deploy your first app

05 Deploying Applications with Deployments and Services

- Deployments explained (replicas, rolling updates, self-healing)
- Service types (ClusterIP, NodePort, LoadBalancer)
- Labels and selectors (how Kubernetes connects resources)
- Zero-downtime rollout concepts
- **Hands-on:** Deploy an app using Deployment + Service and access it from outside the cluster

06 Configuration Management with ConfigMaps and Secrets

- Why configuration must be externalized from images
- ConfigMaps for app configuration (env vars and mounted files)
- Secrets for sensitive values (base64 storage concept)
- Safe practices for managing secrets (no hardcoding in YAML)
- **Hands-on:** Deploy an app using ConfigMaps and Secrets and validate configuration injection

07 Storage and Persistence with PersistentVolumeClaims (PVC)

- Why stateful workloads need persistent storage
- Volumes vs PersistentVolumes vs PersistentVolumeClaims
- StorageClass concepts and dynamic provisioning basics
- Common storage patterns for databases and shared storage
- **Hands-on:** Deploy a database workload using PVC and validate persistence after restart

08 Safe Deployments with Probes, Rolling Updates, and Rollbacks

- Readiness vs liveness probes (why both matter)
- Startup probe concepts (optional intro)
- Rolling update strategy (maxSurge, maxUnavailable)
- Rollback strategies and deployment history inspection
- **Hands-on:** Add probes to an app and simulate a failed rollout followed by rollback

09 Kubernetes Security Basics (securityContext and runAsNonRoot)

- Why Kubernetes hardening matters (runtime risks and privilege escalation)
- securityContext essentials (runAsNonRoot, readOnlyRootFilesystem)
- Dropping Linux capabilities and disabling privilege escalation
- Safe runtime defaults checklist for workloads
- **Hands-on:** Harden a Deployment YAML using securityContext best practices

10 Resource Requests and Limits for Stability

- Requests vs limits explained
- CPU and memory management fundamentals
- Preventing noisy neighbor issues and cluster instability
- Right-sizing strategy for dev vs production environments
- **Hands-on:** Configure requests and limits and observe scheduling + performance behavior

11 Diagnosing Kubernetes Failures (CrashLoopBackOff and OOMKilled)

- Common failure states (CrashLoopBackOff, ImagePullBackOff, Pending)
- Debug workflow using kubectl describe, logs, events
- Root cause analysis patterns (bad config, missing env vars, wrong ports)
- OOMKilled debugging and remediation (increase memory or optimize app)
- **Hands-on:** Fix a broken workload scenario with CrashLoopBackOff and OOMKilled issues

12 Helm Charts Fundamentals + Values Overrides

- Why Helm is used (packaging, reuse, environment consistency)
- Helm chart structure (Chart.yaml, templates, values.yaml)
- Install, upgrade, rollback workflows
- Values override strategies per environment (dev, stage, prod)
- **Hands-on:** Deploy an application using Helm and override values for different environments

13 Troubleshooting Broken Workloads and Misconfigurations

- Debugging misconfigured Deployments and Services
- Common YAML mistakes (labels mismatch, wrong selectors, wrong ports)
- Debugging failed readiness and liveness probes
- Step-by-step troubleshooting checklist for production incidents
- **Hands-on:** Troubleshoot a broken Kubernetes app and fix end-to-end connectivity issues

14 Capstone Project (Production-Style Kubernetes Deployment)

- Capstone goal: Deliver a production-style microservice deployment on Kubernetes
- Containerize application using Docker multi-stage build and safe tagging
- Deploy multi-service stack (API + DB + frontend) on Kubernetes
- Use Deployments, Services, ConfigMaps, Secrets, PVC
- Apply probes, rolling updates, rollback, requests/limits, and securityContext hardening
- **Hands-on:** Build and present a complete end-to-end Kubernetes deployment with troubleshooting demonstration