FULL CURRICULUM

DevOps & Cloud Engineering Masters Bootcamp

90-day weekend program · Saturday & Sunday sessions · 12 weeks · 3 phases · 4 Masters Capstone projects

Program at a Glance

Phase	Weeks	Focus
Phase 1	Weeks 1–4	Foundations — Linux, Git & Docker
Phase 2	Weeks 5–8	AWS Cloud Engineering & Infrastructure as Code
Phase 3	Weeks 9–12	Container Orchestration & Observability
Capstones	Weeks 7–12	Four Masters Capstone projects, built progressively across the program

Each phase builds directly on the last: the projects from Phase 1 are containerized, the infrastructure from Phase 2 is automated, and everything comes together in Phase 3 under orchestration, GitOps and full observability.

PHASE 1 — Foundations: Linux, Git & Docker

Weeks 1–4 · Systems, operating systems, scripting, version control and containers

Week 1 · Saturday — Introduction to DevOps & Program Overview

DevOps Fundamentals

- What DevOps is and why it exists
- SDLC — the Software Development Life Cycle
- The DevOps lifecycle
- DevOps engineer: roles & responsibilities

Program Orientation

- Program structure: how the 90 days are organized
- How the projects build on each other

Introduction to Cloud Computing

- Public cloud vs private cloud vs hybrid cloud
- Cloud service models: IaaS, PaaS, SaaS
- Major cloud providers overview: AWS, Azure, GCP

Computing & Hardware Fundamentals

- What a computer system is vs a server
- CPU: cores, threads, clock speed
- RAM: volatile memory & performance
- Hard disk vs SSD: storage fundamentals
- GPU: role in computing and AI workloads
- Networking hardware basics: NIC, switch, router

- How these components work together
- Client–server architecture

Week 1 · Sunday — Operating Systems Fundamentals

Operating System Basics

- What an operating system is
- OS architecture & kernel basics
- Windows OS overview
- Linux OS overview
- Windows vs Linux: use cases in DevOps
- Why Linux dominates servers & the cloud

Getting Started with Linux

- Introduction to Linux & Linux distributions
- Multiuser & multitasking operating systems
- Linux philosophy & the File System Hierarchy Standard

Virtualization & Lab Setup

- Introduction to virtualization
- Hypervisors: Type 1 vs Type 2
- Installing & configuring VMware Workstation
- Installing & configuring VirtualBox
- Creating your first Linux virtual machine
- Connecting to a Linux system (console, SSH)

Week 2 · Saturday — Linux Fundamentals — File System, Users & Access

File System & Permissions

- Linux directory structure
- Navigating the file system
- File & directory operations
- Users, groups & ownership
- File permissions: rwx, chmod, chown
- sudo & root access

Remote Access & Sessions

- SSH & key-based authentication
- Terminal multiplexers: screen & tmux basics

System Administration

- Package management: APT & YUM/DNF
- Process management: ps, top, htop, kill
- Service management with systemd
- Job scheduling with cron
- Environment variables
- Log files & journalctl basics
- System monitoring basics
- User & group administration (advanced)

Week 2 · Sunday — Linux Disk Management & Networking

Disk & Storage Management

- Understanding partitions
- File systems: ext4 & XFS
- Mounting & unmounting disks
- Logical Volume Manager (LVM) basics
- Disk usage monitoring: df & du
- Swap space configuration

Linux Networking

- Networking fundamentals recap: IP, subnet, gateway
- Network interfaces in Linux
- Checking connectivity: ping, traceroute, curl, wget
- SSH protocol deep dive: port 22 & key exchange
- Ports & protocols: TCP/UDP
- DNS basics from a Linux perspective
- Firewall basics: UFW & iptables introduction
- Network troubleshooting commands

Week 3 · Saturday — Web Servers & Application Delivery Basics

- What a web server is
- How browsers & servers communicate: the HTTP request/response cycle
- Apache vs Nginx: overview
- Installing & running Nginx on Linux
- Serving a static website
- Reading web server logs
- Starting & stopping web services with systemd

Week 3 · Sunday — Shell Scripting for Automation

- Introduction to Bash scripting
- Variables & data types
- Conditional statements
- Loops: for & while
- Functions in Bash
- Text processing: grep, awk, sed, cut
- Automating real admin tasks with scripts
- Error handling & exit codes

> **Mini Project:** Server Health Check Script

Week 4 · Saturday — Git & GitHub — Version Control

- Introduction to version control systems
- Git installation & configuration
- Git workflow: init, add, commit, push
- Branching & merging
- Resolving merge conflicts

- SSH vs HTTPS authentication with GitHub
- Working with remote repositories
- Pull requests & code review basics
- .gitignore & best practices
- Building a basic HTML project

> **Mini Project:** Static Website Hosted with GitHub Pages

Week 4 · Sunday — Virtualization vs Containerization — Docker

From Virtual Machines to Containers

- Recap: virtual machines
- What containerization is
- Virtualization vs containerization: key differences
- Why the industry moved to containers

Docker Fundamentals

- Introduction to Docker
- Docker architecture: client, daemon, registry
- Installing Docker
- Docker images vs Docker containers
- Writing your first Dockerfile
- Building & running Docker images
- Docker volumes: types & use cases
- Docker Hub: pulling & pushing images
- Docker networking: bridge, host, none
- Containerizing the HTML/Nginx project from Week 3

> **Mini Project:** Dockerized Static Website

PHASE 2 — AWS Cloud Engineering & Infrastructure as Code

Weeks 5–8 · From AWS fundamentals to a fully automated 3-tier architecture with CI/CD

Week 5 · Saturday — AWS Fundamentals — Accounts, IAM & Global Infrastructure

The Cloud Provider Landscape

- Cloud provider landscape recap
- Market share & industry adoption
- AWS vs Azure vs GCP: service comparison
- Pricing philosophy differences
- Why this course focuses on AWS

Getting Started with AWS

- AWS global infrastructure overview
- Introduction to AWS
- Creating an AWS account (Free Tier)

- AWS Management Console overview

Identity, Access & Billing

- IAM: users, groups, roles & policies
- IAM best practices: MFA & least privilege
- AWS billing & cost management basics
- Setting billing alerts

Regions & AWS Networking

- Regions & Availability Zones
- Edge locations overview
- How AWS networking is organized
- Choosing a region: latency, cost & compliance

Amazon CloudWatch is woven in where it fits best: alarms with Auto Scaling (Week 6), monitoring in the capstone build (Week 7), and compared against Prometheus/Grafana (Week 12).

Week 5 · Sunday — Amazon VPC — Virtual Private Cloud Deep Dive

- VPC concepts & components
- CIDR notation & IP addressing
- Public vs private subnets
- Internet Gateway
- Route tables
- Security Groups vs NACLs
- Creating a custom VPC
- VPC peering

Week 6 · Saturday — AWS Compute & Storage — EC2, EBS, S3 & RDS

EC2 Compute

- EC2 introduction
- Instance families & use cases
- Pricing models: On-Demand, Reserved, Spot & Dedicated
- Placement groups & placement strategies
- Amazon Machine Images (AMI)
- Launching & connecting to an EC2 instance

EBS Block Storage

- EBS volumes & volume types
- EBS encryption
- Snapshots & backups

S3 Object Storage

- Introduction to S3
- S3 buckets, objects & storage classes
- S3 access control & bucket policies

RDS Managed Databases

- Introduction to RDS & supported database engines
- Creating an RDS instance
- Backups, Multi-AZ & read replicas (overview)

Week 6 · Sunday — DNS, Load Balancing & 3-Tier System Design

The complete 3-tier architecture is designed here first, taught through the concepts of high availability and scalability — then built with Terraform in Week 7.

DNS & Load Balancing

- Route 53: DNS management
- Routing policies overview
- Introduction to Elastic Load Balancing
- Application Load Balancer vs Network Load Balancer
- Health checks & target groups

System Design — High Availability & Scalability

- What system design means for DevOps engineers
- High availability: concepts & patterns
- Scalability: vertical vs horizontal
- Redundancy across Availability Zones
- Auto Scaling Groups
- Amazon CloudWatch: the metrics & alarms that drive scaling
- Caching layer concepts
- Designing a fault-tolerant 3-tier system
- Trade-offs: cost vs availability vs complexity

Week 7 · Saturday — Infrastructure as Code — Terraform & CloudFormation

- Why Infrastructure as Code
- Terraform introduction & installation
- Terraform core concepts: providers, resources & state
- Writing Terraform configurations for VPC, EC2 & S3
- Terraform variables & outputs
- The plan → apply → destroy workflow
- Remote state basics
- CloudFormation overview & comparison with Terraform
- When to choose Terraform vs CloudFormation

Week 7 · Sunday — Capstone Build — 3-Tier Architecture on AWS via Terraform

- Designing the 3-tier architecture: web / app / database
- Translating the design into Terraform modules
- Provisioning the VPC & subnets with Terraform
- Provisioning the EC2 web/app tier with Terraform
- Provisioning the RDS database tier with Terraform
- Attaching a load balancer via Terraform
- Configuring Route 53 for the project
- Adding CloudWatch monitoring
- End-to-end testing & teardown

> **Capstone 1 · Infrastructure:** the stack built here is deployed through full CI/CD pipelines in Week 8.

Week 8 · Saturday & Sunday — CI/CD Fundamentals — GitHub Actions & Jenkins

The Terraform-provisioned EC2 3-tier project from Week 7 is deployed end-to-end here, using both pipelines.

- What CI/CD is
- Introduction to GitHub Actions
- Building a GitHub Actions pipeline
- Introduction to Jenkins
- Building a Jenkins pipeline
- GitHub Actions vs Jenkins: when to use which

> **Masters Capstone 1 completed:** Complete 3-Tier Architecture with CI/CD.

PHASE 3 — Container Orchestration & Observability

Weeks 9–12 · Kubernetes, EKS, ECS, GitOps with ArgoCD, and production-grade monitoring

Weeks 9 & 10 · Sat & Sun — Kubernetes — Complete Track

Kubernetes Foundations

- Why Kubernetes: the problem it solves
- Kubernetes architecture overview (high level)
- Pods, nodes & clusters (conceptual)
- Where Kubernetes fits after Docker

Amazon EKS

- EKS architecture overview
- Setting up an EKS cluster
- kubectl & EKS configuration
- Deploying Pods, Deployments & Services on EKS
- EKS networking: VPC CNI overview
- EKS vs ECS: decision framework

Amazon ECS

- ECS concepts: clusters, tasks & services
- Task definitions
- Fargate vs EC2 launch type
- Deploying a container to ECS
- ECS service auto scaling basics

GitOps with ArgoCD

- What GitOps is
- ArgoCD architecture & installation
- Connecting ArgoCD to a Git repository
- Application sync & auto-sync
- Rollbacks with ArgoCD
- ArgoCD vs traditional CI/CD-driven deployment

Week 11 · Saturday & Sunday — Deployment Projects — EKS & ECS with CI/CD + ArgoCD

Two full deployment projects, run with CI/CD and GitOps only — monitoring & observability are layered on in Week 12.

Capstone Preparation — Multi-Language Containerization & Amazon ECR

These polyglot microservices (Python, PHP, Go, Rust) become the applications deployed in the capstone projects, each bringing a different containerization perspective.

- Do you always need Apache/Nginx? Web servers vs app runtimes
- Pattern 1: Reverse proxy — Nginx in front of an app server
- Pattern 2: Self-hosted app servers — Gunicorn, Actix, net/http
- Containerizing a Python app: Gunicorn / Uvicorn
- Containerizing a PHP/Laravel app: Nginx + PHP-FPM
- Containerizing a Go app: multi-stage builds
- Containerizing a Rust app: multi-stage builds
- Introduction to Amazon ECR
- Pushing & pulling images from ECR
- ECR vs Docker Hub: when to use which

Project 1 → Masters Capstone 2: Kubernetes microservices on EKS — CI with GitHub Actions & Jenkins, CD via ArgoCD.

Project 2 → Masters Capstone 3: ECS deployment — images pushed to ECR, automated pipeline with GitHub Actions.

Week 12 · Saturday & Sunday — Monitoring & Observability — Prometheus, Grafana & OpenTelemetry

- Observability vs monitoring: key differences
- Introduction to Prometheus
- Prometheus architecture & metrics scraping
- Introduction to Grafana
- Building dashboards in Grafana
- Introduction to OpenTelemetry
- Traces, metrics & logs: the three pillars of observability
- CloudWatch vs Prometheus/Grafana: when to use which

Masters Capstone 4: the culminating project that brings everything together — microservices on EKS with ArgoCD, CI/CD, and the full Prometheus + Grafana observability stack.

MASTERS CAPSTONE PROJECTS

Four end-to-end builds, completed progressively through the program

Capstone 1 — Complete 3-Tier Architecture with CI/CD · *Built in Weeks 7–8*

- Project architecture overview (Terraform-provisioned)
- Jenkins pipeline: build & test
- GitHub Actions: deployment trigger
- Full deployment to the EC2 / RDS / ALB stack
- Monitoring & validation

Capstone 2 — Kubernetes Microservices with ArgoCD · *Built in Week 11*

- Microservices architecture overview
- Containerizing multiple services
- CI with GitHub Actions & Jenkins
- CD via ArgoCD to EKS

Capstone 3 — ECS Deployment with GitHub Actions CI/CD · *Built in Week 11*

- Project architecture overview
- Building the application image
- Pushing to ECR via GitHub Actions
- Automated ECS deployment pipeline
- End-to-end testing

Capstone 4 — Kubernetes Microservices with ArgoCD + Full Observability · *Built in Week 12*

Extends Capstone 2 with the complete monitoring stack — the final, production-grade platform.

- Microservices architecture overview
- Containerizing multiple services
- CI with GitHub Actions & Jenkins
- CD via ArgoCD to EKS
- Monitoring & observability with Prometheus and Grafana