

# DevOps

# Core Topics

The **DevOps Core Course** is designed to give beginners technical skills which are necessary for **Real-Time** .

## Who It's For

- Beginners starting their **DevOps** journey from scratch .
- Developers who want **faster, automated** deployments
- System admins **moving** into Cloud & DevOps roles
- IT professionals aiming for **high-growth, high-paying** careers
- Students who want practical, **job-ready** skills

## What You'll Learn

- Master **Git & GitHub** for version control and collaboration
- Build powerful **CI/CD pipelines using Jenkins**
- Work with **Docker** for containerization and **Kubernetes** for orchestration
- Automate infrastructure using **Ansible & Terraform**
- Monitor systems and applications using **Zabbix**
- Implement complete **real-world DevOps workflows**

## Benefits You'll Gain

- Build real-world projects to showcase in your portfolio
- Automate repetitive tasks and save hours of manual work
- Gain in-demand skills companies are actively hiring for
- Boost your chances of landing high-paying DevOps roles
- Stand out from the crowd with practical, hands-on experience

## Course Details

- **Duration:** 50 days (1 hour per day)
- **Mode:** Fully online – attend live
- **Start Date:** Flexible enrollment – begin anytime!

## Topics that will be covered

### DevOps Foundation

- ◆ What is DevOps?
- ◆ Why DevOps?
- ◆ Evolution of Software Methodologies
- ◆ Dev Challenges v/s DevOps Solution
- ◆ Ops Challenges v/s DevOps Solution
- ◆ Stages Of DevOps Lifecycle
- ◆ Continuous Development
- ◆ Continuous Testing
- ◆ Continuous Integration

- ◆ Continuous Deployment
- ◆ Continuous Monitoring
- ◆ The Various DevOps Tools

## **VCS with Git and GitHub Repositories**

- ◆ Why VCS?
- ◆ VCS tools
- ◆ Distributed VCS
- ◆ What is Git & Why Git?
- ◆ Features Of Git
- ◆ Git Workflow
- ◆ Git Configurations
- ◆ Creating Git Repository
- ◆ Syncing Repositories
- ◆ Adding Origin
- ◆ Pushing changes
- ◆ Pulling changes
- ◆ Clone operation
- ◆ Perform, Review & Commit Changes
- ◆ Stacking Unfinished Changes
- ◆ Move, Rename & Delete Operations
- ◆ Tagging Versions In Repository

## **Jenkins (Continuous Integration)**

- ◆ Challenges before Continuous Integration

- ◆ What is Continuous Integration?
- ◆ Benefits of Continuous Integration
- ◆ Tools of Continuous Integration
- ◆ Introduction to Jenkins
- ◆ Configuring Jenkins
- ◆ Build Setup in Jenkins
- ◆ Jenkins Dashboard
- ◆ Creating jobs in Jenkins
- ◆ Configuring Security in Jenkins
- ◆ Plugin Management in Jenkins
- ◆ Notification System
- ◆ Jenkins Best Practices

## **Docker Containers (Containerization)**

- ◆ What is Docker?
- ◆ Challenges before Containerization
- ◆ Understanding microservices
- ◆ VMs for microservices
- ◆ What is a Container?
- ◆ VM vs Containers
- ◆ Benefits of Containerization
- ◆ Introduction to Docker
- ◆ Docker Fundamentals
- ◆ Architecture of Docker

- ◆ Creating & Executing Docker Images
- ◆ Image Distribution
- ◆ Docker Registry
- ◆ What is Docker Hub?
- ◆ Creating Docker Files
- ◆ Using Docker Compose to compose scripts
- ◆ Using Docker Volumes
- ◆ Create 'Dockerized' Application
- ◆ Docker Networking
- ◆ Docker Swarm

## **Kubernetes (Container Orchestration)**

- ◆ Understand container orchestration concepts and architecture
- ◆ Learn master and worker node components (API server, scheduler, kubelet)
- ◆ Create, manage, and scale applications using pods and deployments
- ◆ Expose applications using ClusterIP, NodePort, and LoadBalancer
- ◆ Manage configuration and sensitive data securely
- ◆ Work with persistent storage and volume management
- ◆ Implement scaling and self-healing features in applications

## **Ansible (Configuration Management)**

- ◆ **Introduction to Ansible:** Understand agentless automation and how Ansible works
- ◆ **Installation & Setup:** Configure Ansible environment and inventory files
- ◆ **Ad-hoc Commands:** Execute quick automation tasks without scripts
- ◆ **Playbooks:** Write YAML-based playbooks for automation workflows
- ◆ **Modules:** Use built-in modules for tasks like package install, file management, and services
- ◆ **Inventory Management:** Manage hosts and groups for scalable automation
- ◆ **Variables & Facts:** Work with dynamic data and system information
- ◆ **Conditionals & Loops:** Control execution flow in playbooks
- ◆ **Roles:** Organize and reuse automation code efficiently
- ◆ **Ansible Galaxy:** Use and manage community roles
- ◆ **Handlers:** Trigger actions based on changes
- ◆ **Error Handling:** Manage failures and ensure smooth execution
- ◆ **Real-world Use Cases:** Automate server setup, deployments, and configuration management

## **Terraform (Infrastructure as Code)**

- ◆ Understand Infrastructure as Code (IaC) concepts using Terraform

- ◆ Install and set up Terraform for real-world usage
- ◆ Write and manage Terraform configuration files (HCL)
- ◆ Work with providers like AWS, Azure, and GCP
- ◆ Create, update, and destroy infrastructure using Terraform commands
- ◆ Use variables and outputs for flexible configurations
- ◆ Manage state files and understand remote state concepts
- ◆ Organize code using modules for reusability
- ◆ Handle dependencies and resource lifecycle
- ◆ Implement best practices for scalable and secure infrastructure
- ◆ Perform real-world deployments using Terraform projects

### **Zabbix (Continuous Monitoring)**

- ◆ Introduction to Zabbix and monitoring fundamentals
- ◆ Installing and configuring Zabbix server and agents
- ◆ Monitoring servers, applications, and network devices
- ◆ Creating hosts, items, triggers, and graphs
- ◆ Setting up alerts, notifications, and escalation rules
- ◆ Building dashboards for real-time monitoring
- ◆ Integrating Zabbix with other DevOps tools
- ◆ Basic troubleshooting and performance tuning of Zabbix setup

**To Join this Course Contact or Ping on Whatsapp : [93427 37660](https://wa.me/9342737660)**