Syllabus for DevOps

MiND Scripts®
Success Simplified

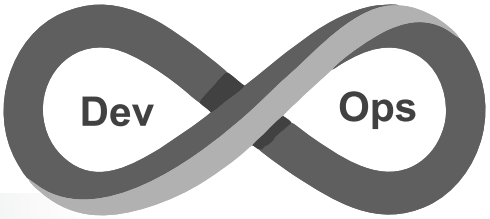
Course Duration For DevOps Training Course :

- 5 Weekends (Weekend Batches)

DevOps

Objective For DevOps Training Course :

- Become a DevOps practitioner and apply the latest in DevOps methodology and its best practices to multiple SDLC steps.
- Get hands-on knowledge on various DevOps tools like Git, Docker, Jenkins, Puppet, Chef and Nagios.



Eligibility For DevOps Training Course :

- IT industry experience

Syllabus

DevOps

DevOps Essentials

- ✓ 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 Deveopment
  - Continuous Testing
  - Continuous Integration
  - Continuous Deployment
  - Continuous Monitoring
- ✓ Dark Launching Technique
- ✓ The DevOps Ecosystem
- ✓ The Various DevOps Tools

Git & GitHub (Managing Source Code and Automating Builds)

- ✓ What is Version Control System(VCS)?
- ✓ 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

Syllabus for DevOps

MiND Scripts®
Success Simplified

Automated Testing and Test Driven Development

- ✓ Automated Testing and Test Driven Development
- ✓ Integrated Development Environments
- ✓ Test Driven Development Approach
- ✓ Behavior Driven Development
- ✓ Integration Testing and Mocking
- ✓ Software Integration Tools
- ✓ Code Quality Principles
- ✓ Code Quality Tools
- ✓ Continuous Code Quality

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 Maven Integration
- ✓ Jenkins Best Practices

Docker (Containerization)

- ✓ Challenges before Containerization
  - Understanding microservices
  - VMs for microservices
- ✓ What is a Container?
- ✓ VM v/s Containers
- ✓ Benefits of Containerization
- ✓ Introduction to Docker
- ✓ Docker Fundamentals
- ✓ Architecture of Docker
- ✓ Creating & Executing Docker Images
- ✓ Image Distribution
- ✓ Docker Registry
- ✓ What is Docker Hub?
- ✓ Docker Containers
- ✓ Creating Docker Files
- ✓ Using Docker Compose to compose scripts
- ✓ Using Docker Volumes
- ✓ Create 'Dockerized' Application
- ✓ Docker Networking
- ✓ Docker Swarm

Puppet (Configuration Management & Deployment)

- ✓ Challenges before Configuration Management
- ✓ Advantages of Inheritance
- ✓ What is Configuration Management?
- ✓ Configuration management Components
- ✓ Configuration management Tools
- ✓ What is Puppet?
- ✓ Puppet Environment
- ✓ Master-slave Architecture of Puppet
- ✓ How Puppet Works?
- ✓ Components of Puppet
- ✓ Generating Master & Agent Certificates
- ✓ Basic Puppet Terminologies
- ✓ Puppet Language Constructs
- ✓ Node Classification
- ✓ Puppet Resources & Classes
- ✓ Puppet Manifests
- ✓ Puppet Modules
- ✓ Puppet Template

Chef (Configuration Management & Deployment -2)

- ✓ Chef Fundamentals
- ✓ Chef environment
- ✓ Chef Cookbooks
- ✓ Knife Commands
- ✓ Node Object & Search
- ✓ Chef Data-bags
- ✓ Roles in Chef
- ✓ Deploying Nodes in Production
- ✓ Vagrant file

Nagios (Continuous Monitoring)

- ✓ What is Continuous Monitoring?
- ✓ Introduction to Nagios
- ✓ Nagios Setup
- ✓ Nagios Plugins
- ✓ Introduction to Events
- ✓ Objects in Nagios
- ✓ Nagios Commands
- ✓ Nagios Notification