



# Development & Operations

## DevOps Introduction

### Learning Objective

- ❖ **Understanding DevOps**

### Topics Covered

- ❑ What is DevOps?
- ❑ Evolution of Software Methodologies
  - ❑ Waterfall
  - ❑ Agile
  - ❑ Disadvantages of traditional SDLC
- ❑ Why DevOps?
- ❑ 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 Introduction
- ❑ Roles and Responsibilities of a DevOps Engineer
- ❑ How DevOps fits in the whole Software Development Lifecycle

# Physical and Virtual Computing Environment

## Learning Objective

- ❖ **Computing Environments and Getting Started with Virtualization**

## Topics Covered

- ❑ Computing Environments and Operating Systems
  - ❑ Personal Computing, Server- Client
- ❑ Understanding Virtualization
- ❑ Understanding Hypervisor
  - ❑ Hosted
  - ❑ Baremetal

## Hands-On

- ❑ Setting up a Virtual Machine with Linux OS

# Linux for Devops

## Learning Objective

- ❖ **Understanding Linux OS and Basic Shell Commands**

## Topics Covered

- ❑ What is Linux and Open Source
- ❑ Linux Distributions
- ❑ Understanding Local User in Linux
- ❑ Understanding Linux Hierarchy
- ❑ Understanding Shell
- ❑ Understanding Absolute and Relative Path
- ❑ Working with Text Editors
- ❑ Creating Files and Directory Using CLI
- ❑ Understanding Linux Command and Using Options for Linux Commands

- ❑ Understanding Globbing, Pipe, Tee
- ❑ Understanding Variables, Operators

## Hands-On

- ❑ Executing Basic Shell Commands - cd, ls, vim, nano, touch, mkdir
- ❑ File and Directory Management - cp, mv, rm
- ❑ Commands with Pattern Matching
- ❑ Writing Simple Shell Scripts

## Programming with Python

### Learning Objective

- ❖ **Getting Started with Python and Writing Python Programs**

### Topics Covered

- ❑ Introduction to Python: What is Python and why to learn Python as a DevOps engineer?
- ❑ Installation and Setup Local Development Environment
- ❑ Write Simple Python program
- ❑ Python IDE vs simple File Editor
- ❑ Strings and Number Data Types
- ❑ Variables
- ❑ Encapsulate Logic with Functions
- ❑ Accepting User Input
- ❑ Conditionals (if / else) and Boolean Data Type
- ❑ Error Handling with Try / Except
- ❑ While Loops
- ❑ Lists and For Loops
- ❑ Comments in Python
- ❑ Sets
- ❑ Built-In Functions
- ❑ Dictionary Data Type
- ❑ Modularize your project with Modules
- ❑ Packages, PyPI and Pip
- ❑ Object Oriented Programming: Classes and Objects

## Hands-On

- Installation and Setup Local Development Environment
- Writing Python Scripts
- Project: Countdown App
- Project: Automation with Python (Working with Spreadsheets)
- Project: API Request to GitLab

## Cloud Computing - AWS

### Learning Objective

- ❖ **Cloud Computing and Introduction to AWS.**

### Topics Covered

- What is Cloud Computing
- History & Comparison with Client Server computing
- Advantages of Cloud Computing
- Why AWS is different from other Vendors.
- Future of Cloud
- Service Model of Cloud
  - IAAS
  - PAAS
  - SAAS
- Deployment Model of Cloud
  - Private
  - Public
  - Hybrid
- AWS Infrastructure (Regions and Availability Zone)
- Design Diagram Tools for AWS
- Parts of Cloud
  - Frontend
  - Backend
- Accessing AWS
  - Management Console
  - AWS CLI
  - AWS SDK
- AWS Account Plans and Free Tier

- ❑ AWS Domain and Services
- ❑ AWS Certifications

## **AWS Elastic Compute Cloud in Compute Domain**

### **Learning Objective**

- ❖ **Understanding EC2, Instance, AMI, Security Group, KeyPair**
- ❖ **Launch Instance and types with Volumes, and Images.**
- ❖ **Mastering Elastic Compute Cloud**
- ❖ **Mastering Amazon Machine Image**
- ❖ **Mastering EC2 Pricing**
- ❖ **Mastering Instance types**

### **Topics Covered:**

- ❑ Launching EC2 Instance and Connect
  - ❑ Windows Instance
  - ❑ Linux Instance
- ❑ Instance Userdata and Metadata
- ❑ EC2 Instance Types and Family
- ❑ Types of AMIs to Launch EC2 Instance
  - ❑ AWS Published
  - ❑ AWS Marketplace
  - ❑ Creating from existing Instance
  - ❑ Upload Virtual Services
- ❑ AWS Service Limits and Support Plans

### **Hand-on Lab:**

- ❑ Launch EC2 Instance (Windows) with Standard SSD Storage, Connect to Windows Instance Using Remote Desktop Protocol.
- ❑ Launch EC2 Instance (Linux) with Standard SSD Storage, Connect to Linux instance Using Secure Shell

## **VCS with Git**

## Learning Objective

- ❖ **Understanding Git & GitHub (Managing Source Code and What is Version Control System(VCS)?**
- ❖ **Understanding AWS Code Commit**

## Topics Covered

- 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
- Concepts of Branches
- Merge Requests
- Deleting Branches
- Resolving Merge Conflicts
- Git Ignore
- Git Stash
- Merging Branches

## Hand-on Lab:

- Launch EC2 Instance (Windows) and Configure Git
- Launch EC2 Instance (Linux) and Configure Git
- Configuring all Git Operations (Creating Local and Setting up Remote Repository in GitHub and Code Commit), push, pull, clone, creating branches, merge.

# Build & Package Manager Tools

## Learning Objective

- ❖ **What are Build Tools and Package Managers?**

## Topics Covered

- How to build an artifact?
- How to run the application artifact?
- How to publish the application artifact to the artifact repository?
- Build Tools for Java (gradle and maven examples)
- Dependency Management in Software Development
- Package Manager in JavaScript applications - Build and run applications in JS
- Build Tools
- Why Build Tools are relevant for DevOps Engineers?

# Artifact Repository Manager with Nexus

## Learning Objective

- ❖ **Understanding Artifact Repository Manager, Types**

## Topics Covered

- What is an Artifact Repository Manager?
- Install and run Nexus on Cloud Server
- Different Repository Types (proxy, hosted, etc.) explained
- Different Repository Formats (maven, docker, npm, etc.) explained
- Upload Jar File to Nexus (maven and gradle projects)
- Nexus API and Repository URLs
- Blob stores
- Browsing Components - Components vs Assets
- Cleanup Policies
- Scheduled Tasks

# Continuous Integration with Jenkins

## Learning Objective

- ❖ Understanding Integration with Jenkins

## Topics Covered

- Challenges before Continuous Integration
- What is Continuous Integration?
- Benefits of Continuous Integration
- Tools of Continuous Integration
- Introduction to Jenkins
- Jenkins Plugins
- Build Setup in Jenkins
- Jenkins Pipeline (Use Cases)
- Create a simple Pipeline Job
- Full Jenkinsfile Syntax Demo
- Create a full Pipeline Job
- Build Java App
- Build Docker Image
- Push to Private DockerHub
- Create a Multi-Branch Pipeline Job
- Credentials in Jenkins
- Jenkins Shared Library
- WebHooks - Trigger Jenkins Jobs automatically
- Versioning Application in Continuous Deployment
- Concepts of Versioning in Software Development
- Increment Application version from Jenkins Pipeline
- Set new Docker Image version from Jenkins Pipeline
- Commit Version Bump from Jenkins Pipeline

## Hand-on Lab:

- Launch EC2 Instance (Linux) and Install Jenkins



- ❑ Creating a simple freestyle job
- ❑ Configure Git Repository and Build a java application
- ❑ Build docker images and push it to docker hub
- ❑ Create a simple and multi pipeline

## **AWS Command Line Interface**

### **Learning Objective**

- ❖ **Understanding AWS Command Line Tool**
- ❖ **Install AWS Command Line Tool**
- ❖ **AWS CLI Configuration**
- ❖ **Launch AWS Resources using CLI Tool**

### **Topics Covered**

- ❑ Install and Configure AWS CLI
- ❑ AWS CLI Reference
- ❑ Build AWS Resources using AWS CLI

### **Hands-On**

- ❑ Getting started with AWS CLI Commands

## **Identity and Access Management**

### **Learning Objective**

- ❖ **Understanding Fundamentals of AWS IAM**
- ❖ **Understanding IAM Principles**
- ❖ **Build Secure Administration using IAM Components**
  - **Users**
  - **Groups**
  - **Policies**
  - **Roles**

## Topics Covered

- IAM Principles
- Creating Users
- Creating Groups
- Understanding Policies
- Understanding Console and Programmatic Access
- Access Keys and Secret Key
- IAM Roles
- Security and Policies

## Hands-On

- Create user access AWS Resources CLI
- Create and assign Roles to Resource

## Serverless and PAAS

### Learning Objective

- ❖ Understanding What is Serverless
- ❖ Understanding Lambda
- ❖ Manual invoke and Cloud Watch(Event bridge trigger)
- ❖ Understanding PAAS and Elastic Beanstalk

## Topics Covered

- Lambda functions
- Configuration limitations and pricing
- Configuring Elastic Beanstalk
- Understand the deployment types

## Hands-On

- Create a lambda function for stopping and starting ec2 instance
- Integrate with cloudwatch event and trigger lambda
- Create a simple sample application and deploy using elastic beanstalk

# Containerization

## Learning Objective

- ❖ Understanding Traditional(Physical and Virtual) Application Deployment Methods) and Containerization
- ❖ Advantages
- ❖ Understanding Docker and Components
- ❖ Managing Docker in a Standalone Instance

## Topics Covered

- Virtualization vs Containerization
- What are Containers and Advantages of Containers
- Architecture of Docker Container
- Components of Docker
  - Images
    - Registries (Docker Hub, Elastic Container Registry)
- Managing Docker Service
- Running a Container(Attached/Detached), Logging in to Container
- Starting / Stopping / Restarting Containers
- Container Networking
  - Bridge
  - Host
  - Overlay
- Managing Storage for Containers
- Understanding Docker File
- Docker Hub - Pushing Images to Repository

## Hand-on Lab:

- Launch EC2 Instance (Windows and Linux) and Configure Docker Engine
- Run a Simple Standalone Webapp Container
- Run an Ubuntu Container to Check Connectivity between Containers
- Creating Volume for Containers and Mounting it Persistently
- Creating a Custom Container Image from Another Container / Docker File

# Containerization Orchestration

## Learning Objective

- ❖ Understanding What is Container Orchestration
- ❖ Kubernetes
- ❖ ECS/EKS

## Topics Covered

- Introduction to Kubernetes
- Understand the Main Kubernetes Components
- Node, Pod, Service, Ingress, ConfigMap, Secret, Volume, Deployment, StatefulSet
- Kubernetes Architecture
- Minikube and Kubectl - Local Setup
- Main Kubectl Commands - K8s CLI
- Create and Debug Pod in a Miniclustert
- Kubernetes YAML Configuration File
- Create and Configure Deployment and Service Components
- Organizing your components with K8s Namespaces
- Kubernetes Service Types
- Making your App accessible from outside with Kubernetes Ingress
- Persisting Data in Kubernetes with Volumes
- Persistent Volume
- Persistent Volume Claim
- Storage Class
- ConfigMap and Secret Kubernetes Volume Types
- Deploying Stateful Apps with StatefulSet
- Deploying Kubernetes cluster on a Managed Kubernetes Service (K8s on Cloud)
- Helm - Package Manager of Kubernetes
- Creating a ECS Cluster
- Creating a EKS Cluster

## Infrastructure as Code with Terraform & Ansible

## Learning Objective

- ❖ Terraform and it's components

## Topics Covered

- What is Terraform? How it works
- Architecture
- Providers
- Resources & Data Sources
- Variables & Output Values
- Environment variables in Terraform
- Terraform commands
- Terraform State
- Provisioners
- Modules
- Remote State
- Terraform & AWS

## Hand-on Lab:

- Create Security Group and Provision EC2 windows or Linux Instance using Terraform
- Configure Terraform in Jenkins
- Automate provisioning EC2 instance from Jenkins pipeline and deploy the application with docker-compose

## Learning Objective

- ❖ Automation with Ansible

## Topics Covered

- What is Ansible, Uses and How Ansible Works.
- Architecture

- Control Node
- Managed Node
- Inventory
- Module
- Play and Playbook
- Managing Static Inventory
- Creating Ansible Project Directory and Configurations
- Understanding Ansible Ad-hoc Commands
- Privilege Escalation Configuration
- Understanding YAML and Writing Simple Ansible Playbook
- Using Variables in Ansible Playbook
- Using Loop
- Using Conditions
- Ansible Roles from Ansible Galaxy

## **Hand-on Lab:**

- Launch an EC2 Linux Instance and Configure Ansible Engine
- Setting Up Linux Managed Nodes
- Writing a Playbook for Deploying Web Application
- Deploying Applications from Ansible Galaxy
- Project: Ansible & Terraform
- Project: Run Docker applications
- Project: Run Ansible from Jenkins Pipeline

## **Automation with Python**

### **Learning Objective**

- ❖ **Automation with Python**

### **Topics Covered**

- Cloud Automation - AWS & Python
- Introduction to Boto (AWS Library for Python)

- ❑ Install Boto3 and connect to AWS
- ❑ Getting familiar with Boto Library
- ❑ Automate creating VPC and Subnets
- ❑ Terraform vs Python - understand the differences and when to use which tool
- ❑ Automation Tasks around EC2 Instance:
  - ❑ Health Check: Automatically check the status of EC2 Instances
  - ❑ Scheduler: Write a scheduled task that executes the status check in a specified interval automatically
  - ❑ Configure Server: Automate adding tags to EC2 Instances with the environment label