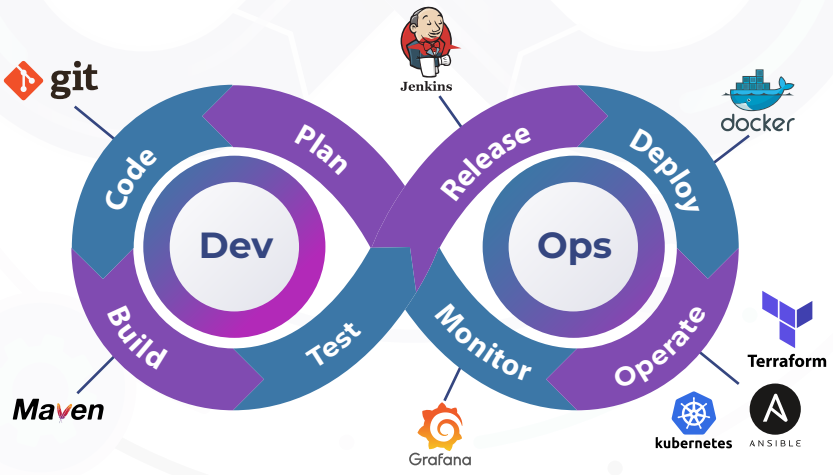
SR Technologies_DevOps with AWS
Curriculum

Module-1 Introduction to DevOps

➢ What is Software Application Development?

➢ SDLC (Software Development Life Cycle)

1. What is DevOps?
2. Who can learn DevOps?
3. What are the prerequisites to learn DevOps?
4. DevOps Market Trends
5. Why are DevOps skills in high demand across industries?
6. History and Evolution of DevOps
7. Traditional IT vs Agile vs DevOps
8. DevOps Core Principles: CALMS (Culture, Automation, Lean, Measurement, Sharing)
9. DevOps Lifecycle Overview
   (Plan → Develop → Build → Test → Release → Deploy → Operate → Monitor)



➢ Benefits and Business Value of DevOps

1. Why DevOps? (Faster Delivery, Reduced Failures)
2. DevOps Impact on Software Delivery

➢ Linux Basics:

- What is an operating System? What is the difference between Windows and Linux?
- Introduction to Linux and Its Distributions (Ubuntu, CentOS, RHEL)
- Basic Linux Commands (ls, cd, pwd, mkdir, rm, etc.)

- File Permissions, Ownership, and chmod, chown, umask
- User and Group Management (useradd, groupadd, passwd)
- File and Directory Management (cp, mv, find, locate, du, df)
- Process Management (ps, top, kill, nice, jobs, bg, fg)
- Package Management (APT/YUM/DNF – install, remove, update packages)
- Networking Commands (ip, ping, netstat, ss, curl, wget)
- Shell Scripting Basics (variables, conditions, loops, functions)
- System Services and Logs (systemctl, journalctl, log files in /var/log)

Module-2 Introduction to Git & Version Control

1. What is Version Control?
2. Centralized vs. Distributed Version Control
3. What is Git? Why use Git?
4. Installing Git (Windows, macOS, Linux)
5. Git vs GitHub vs GitLab vs Bitbucket
6. Initializing a Git Repository
7. Checking the Status
8. Git Add, Commit, and Log
9. Understanding the Working Directory, Staging Area, and Repository
10. Viewing History with git log

➤ Branching and Merging

1. What is a Branch?
2. Creating, Switching, and Deleting Branches
3. Merging Branches
4. Fast-forward vs Three-way Merge
5. Merge Conflicts: How to resolve them
6. Visualizing Branches with git log --graph

➤ Remote Repositories

1. Setting up GitHub or GitLab
2. Cloning Repositories
3. Adding Remote Repositories
4. Pushing and Pulling Changes
5. Fetch vs Pull
6. Tracking and Untracking Files

➤ Collaboration Workflow

1. Forking and Pull Requests
2. Git Workflow Models (Feature Branch, GitFlow, Trunk-Based)
3. Code Review with Pull Requests
4. Best Practices for Team Collaboration

➤ Undoing Changes

1. git checkout vs git restore
2. git reset (soft, mixed, hard)
3. git revert
4. Recovering Lost Commits
5. Stash: Saving Temporary Changes

➤ Maven: Build Tool

➤ What is the Build Process?

➤ Introduction to Maven

1. What is Apache Maven?
2. History & Evolution of Build Tools
3. Maven vs Ant vs Gradle
4. Installing Maven (Windows, macOS, Linux)
5. Verifying Maven Installation

➤ Maven Project Structure

1. Understanding the Maven Standard Directory Layout
2. What is pom.xml?
3. Basic Elements of a POM (Project Object Model)
4. GroupId, ArtifactId, Version (GAV)
5. Project Lifecycle Overview

➤ Maven Build Lifecycle & Phases

1. Build Lifecycles: Default, Clean, Site
2. Core Phases: validate, compile, test, package, install, deploy
3. Maven Goals vs Phases
4. Running Maven Goals from CLI

➤ Maven Dependencies

1. What is a Dependency?
2. Adding Dependencies to pom.xml
3. Transitive Dependencies
4. Managing Conflicts (Dependency Mediation)
5. Dependency Scopes (compile, test, provided, runtime, system)

➤ Plugins and Goals

1. What are Maven Plugins?
2. Build Plugins vs Reporting Plugins
3. Commonly Used Plugins:
   - o maven-compiler-plugin
   - o maven-surefire-plugin
   - o maven-jar-plugin
   - o maven-clean-plugin
4. Executing Plugin Goals

Module-3 Jenkins (CICD Tool)

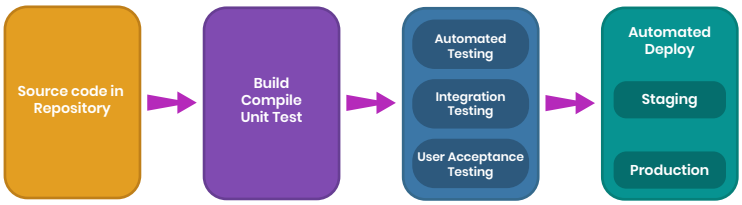
➤ What is CICD (Continuous Integration and Continuous Delivery/ Deployment)?

➤ What is the Need for Jenkins?

➤ Introduction to Jenkins

1. What is Jenkins?
2. Jenkins History and Evolution
3. Features of Jenkins
4. Jenkins vs Other CI/CD Tools (GitLab CI, GitHub Actions, CircleCI)
5. Jenkins Use Cases in DevOps

CI/CD PIPELINE



➤ Installing Jenkins

1. System Requirements
2. Installing Jenkins on:
    - o Linux (Ubuntu/CentOS)
    - o Windows
    - o Docker
3. Running Jenkins as a Service
4. Initial Setup and Admin Configuration
5. Installing Plugins During Setup

➤ Getting Started with Jenkins UI

1. Exploring the Jenkins Dashboard
2. Creating Your First Job (Freestyle Project)
3. Understanding Build Triggers
4. Build Steps and Post-build Actions
5. Build History and Console Output

➤ Jenkins Plugins

1. What are Plugins?
2. Must-Have Plugins for DevOps
    - o Git
    - o Maven
    - o Pipeline
    - o Email Extension
    - o Docker
3. Installing and Managing Plugins
4. Plugin Compatibility and Updates

➤ Jenkins Pipeline Basics

1. What is a Jenkins Pipeline?
2. Declarative vs Scripted Pipelines
3. Creating a Simple Pipeline (UI and Jenkinsfile)
4. Pipeline Stages, Steps, and Agents
5. Running and Debugging Pipelines

➤ Jenkinsfile Deep Dive

1. Writing Jenkinsfiles from Scratch
2. Parameters and Environment Variables
3. Using tools, when, input, and post blocks
4. Parallel Stages and Matrix Builds
5. Shared Libraries and Reusability

➤ Jenkins Architecture

1. Jenkins Master and Agent (Node) Architecture
2. Distributed Builds
3. How Jenkins Executes Jobs
4. Agent Configuration (SSH, JNLP, Docker-based)

➤ Integrating with Git & GitHub

1. Connecting Jenkins to Git/GitHub/GitLab
2. Webhooks for Triggering Builds on Push
3. Building on Pull Request Events
4. Git Credentials and SSH Key Setup
5. GitHub Integration Plugin

➤ Jenkins with Build Tools

1. Jenkins with Maven
2. Jenkins with Gradle
3. Using Ant (if needed)
4. Managing Dependencies and Artifacts

➤ Jenkins with Docker and Containers

1. Running Jenkins in Docker
2. Using Docker in Pipelines (Docker Plugin)
3. Building and Pushing Docker Images with Jenkins
4. Docker Inside Jenkins vs Jenkins Inside Docker

➤ Jenkins with CI/CD

1. CI/CD Concepts Refresher
2. Automating Build-Test-Deploy Pipelines
3. Rolling Deployments with Jenkins
4. Deployment to Tomcat, EC2, or Kubernetes
5. Blue/Green and Canary Deployments (basic intro)

➤ Notifications and Reporting

1. Email Notifications Setup
2. Slack/MS Teams Integration
3. Test Reports with JUnit
4. Code Coverage Reports
5. Publishing Artifacts

➤ Security and User Management

1. Managing Users and Roles
2. Matrix-Based Authorization
3. Setting up Credentials Securely
4. Secrets Management (Vault, Jenkins Credentials Plugin)

➤ Jenkins Backup, Restore & Maintenance

1. Jenkins Backup Strategy
2. Restoring Jenkins
3. Log Rotation and Cleanup
4. Upgrading Jenkins and Plugins

➤ Jenkins in the Cloud

1. Jenkins on AWS EC2
2. Jenkins with EFS/S3 for storage
3. Jenkins on Kubernetes (Jenkins X, Helm)
4. Jenkins with Terraform or Ansible Integration

Module-4 Docker – Containerization Tool

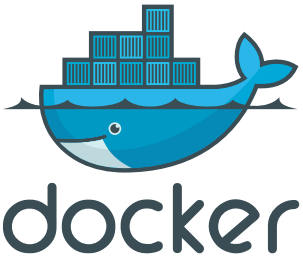
➤ What is a Virtual server?

➤ What is the difference between a physical server and a virtual server?

➤ What is the difference between a virtual server and a container?

➤ Introduction to Docker

1. What is Docker?
2. Why Docker? Benefits Over VMs
3. Docker Architecture (Client, Daemon, Images, Containers)
4. Understanding Containers vs Images
5. Installing Docker (Linux, macOS, Windows)



➤ Getting Started with Docker

1. Running Your First Container
2. docker run Command Breakdown
3. Listing and Stopping Containers
4. Docker Help & Command Line Basics
5. Removing Containers and Images

➤ Working with Docker Images

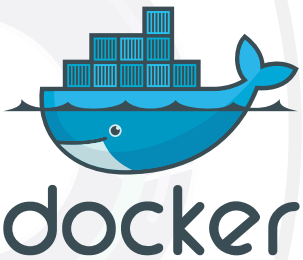
1. What is a Docker Image?
2. Pulling Images from Docker Hub
3. Inspecting and Tagging Images
4. Removing, Saving, and Loading Images
5. Difference Between Official and Custom Images

➤ Building Custom Images with Dockerfile

1. What is a Dockerfile?
2. Dockerfile Instructions: FROM, RUN, COPY, CMD, ENTRYPOINT, etc.
3. Building a Custom Image (docker build)
4. Best Practices for Writing Dockerfiles
5. Multi-Stage Builds

➤ Managing Containers

1. Container Lifecycle
2. Detached Mode vs Interactive Mode
3. Accessing Containers (docker exec, docker attach)
4. Environment Variables in Containers
5. Volumes and Persistent Storage



➤ Docker Networking

1. Docker Default Networks (bridge, host, none)
2. Creating Custom Networks
3. Linking Containers
4. Network Inspection and Troubleshooting
5. Exposing and Publishing Ports

➤ Docker Volumes and Storage

1. Volumes vs Bind Mounts
2. Creating and Using Docker Volumes
3. Sharing Data Between Containers
4. Backing Up and Restoring Volumes
5. Managing Volume Lifecycle

➤ Docker Compose

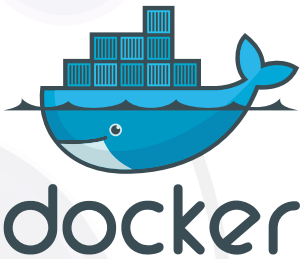
1. What is Docker Compose?
2. docker-compose.yml File Structure
3. Defining Multi-Container Apps
4. Networking in Compose
5. Running, Scaling, and Tearing Down Compose Projects

➤ Docker Registry & Image Management

1. Docker Hub and Docker Registry
2. Pushing and Pulling from Docker Hub
3. Setting Up a Private Docker Registry
4. Image Tagging Strategies (latest, semantic versioning)
5. Automating Builds with Webhooks

➤ Docker in CI/CD

1. Docker with Jenkins/GitHub Actions/GitLab CI
2. Building Images in CI Pipelines
3. Scanning Images for Vulnerabilities (Trivy, Snyk)
4. Caching and Optimizing Build Times
5. Deploying Containers in CI/CD Pipelines



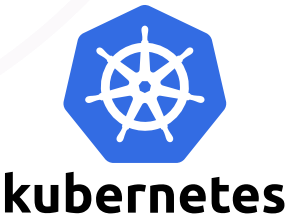
➤ Docker and Kubernetes (Intro)

1. Why Orchestrate Containers?
2. Docker vs Kubernetes
3. Docker Swarm Overview (optional)
4. Docker in Kubernetes Clusters
5. Creating Pods with Docker Images

➤ Kubernetes:

➤ Introduction to Kubernetes

1. What is Kubernetes?
2. Why Kubernetes? Evolution from VMs to Containers to K8s
3. Key Features and Use Cases
4. Kubernetes vs Docker Swarm vs ECS vs Nomad
5. CNCF and the Cloud-Native Ecosystem



➤ Kubernetes Architecture

1. Kubernetes Cluster Components
    - o Control Plane: API Server, Scheduler, Controller Manager, etcd
    - o Worker Nodes: Kubelet, Kube-proxy, Container Runtime
2. Pods, Nodes, and Containers
3. Communication Between Components
4. High Availability Architecture

➤ Setting Up Kubernetes (On-premises K8S, GKE, EKS)

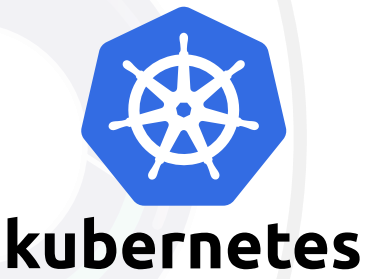
1. Kubernetes Setup Options:
    - o Minikube (local)
    - o kubeadm (manual cluster)
    - o Managed (EKS, GKE, AKS)
2. Installing kubectl and CLI Tools
3. Connecting to a Cluster
4. Exploring the Kubernetes Dashboard

➤ Working with Pods

1. What is a Pod?
2. Creating Pods with YAML
3. Pod Lifecycle & Status
4. Viewing Logs and Exec into Containers
5. Init Containers and Multi-Container Pods

➤ ReplicaSets and Deployments

1. ReplicaSet Basics
2. Deployments and Rolling Updates
3. Rollbacks and Revisions
4. Scaling Deployments Manually and Automatically
5. Self-healing and High Availability



➤ Services and Networking

1. Cluster Networking Overview
2. Service Types:
    - o ClusterIP
    - o NodePort
    - o LoadBalancer
    - o ExternalName
3. DNS in Kubernetes
4. Headless Services and Stateful Networking
5. Ingress and Ingress Controllers

➤ Volumes and Persistent Storage

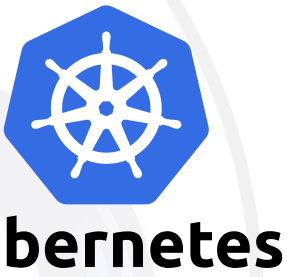
1. Kubernetes Volume Types (emptyDir, hostPath, NFS, etc.)
2. Persistent Volumes (PV) and Claims (PVC)
3. Storage Classes and Dynamic Provisioning
4. Volume Mounts and Access Modes
5. StatefulSets and Persistent Storage

➤ Helm – Kubernetes Package Manager

1. What is Helm?
2. Installing Helm
3. Charts and Chart Repositories
4. Deploying Applications using Helm Charts
5. Creating and Managing Custom Charts

➤ Autoscaling and Resource Management

1. Resource Requests and Limits
2. Horizontal Pod Autoscaler (HPA)
3. Vertical Pod Autoscaler (VPA)
4. Cluster Autoscaler
5. Best Practices for Efficient Resource Use



➤ Monitoring and Logging

1. Metrics Server and Resource Metrics
2. Prometheus + Grafana Setup for Monitoring
3. Logging with EFK Stack (Elasticsearch, Fluentd, Kibana)
4. Using kubectl top and Logs
5. Alerting Basics with Prometheus

➤ CI/CD Integration

1. CI/CD Concepts Recap
2. Jenkins with Kubernetes (Jenkins-X or Kubernetes Plugin)
3. GitOps with ArgoCD or Flux
4. Building and Deploying Containers Automatically
5. Canary Deployments and Blue-Green Deployments

➤ Security in Kubernetes

1. Role-Based Access Control (RBAC)
2. Service Accounts and Tokens
3. PodSecurityPolicies (Deprecated), Pod Security Standards
4. Network Policies
5. Securing Secrets and API Access

Module-5 Terraform (Infrastructure Management Tool)

➤ What is IT infrastructure?

➤ Introduction to Infrastructure as Code (IaC) and Terraform

1. What is Infrastructure as Code?
2. Why Terraform? Features and Use Cases
3. Terraform vs Other IaC Tools (CloudFormation, Pulumi, Ansible)
4. Terraform Architecture and Workflow
5. Supported Providers Overview (AWS, Azure, GCP, Kubernetes)

➤ Getting Started with Terraform

1. Installing Terraform (Windows/Linux/macOS)
2. Terraform CLI Basics
3. First Terraform Configuration – Provisioning EC2 on AWS
4. Terraform Workflow: Init → Plan → Apply → Destroy
5. Hands-on: Your First Terraform Project

➤ Terraform Configuration Language (HCL)

1. Overview of HCL Syntax
2. Resources, Providers, and Variables
3. Data Sources and Outputs
4. Local Values and Expressions
5. Input Validation and Descriptions

➤ Variables and Outputs

1. Defining Input Variables
2. Variable Types and Validation Rules
3. Using Environment Variables and terraform.tfvars
4. Output Values and Dependencies
5. Sensitive Data Handling

➤ Terraform State Management

1. What is Terraform State?
2. Local vs Remote State Files
3. State Locking and State File Structure
4. Terraform Workspaces (Managing Environments)

➤ Providers and Resources

1. What is a Provider?
2. Provider Block Configuration
3. Using Multiple Providers (e.g., AWS + Kubernetes)
4. Resources – Basics and Arguments
5. Data Sources – Referencing Existing Infrastructure

➤ Modules in Terraform

1. What are Modules?
2. Creating and Reusing Modules
3. Organizing Code with Root and Child Modules
4. Calling Modules with Parameters
5. Using Public Modules from the Terraform Registry

➤ Terraform with AWS (or Azure/GCP)

1. Setting up Terraform with AWS Credentials
2. Creating VPC, Subnets, Security Groups, and EC2 Instances
3. Using IAM Roles and Policies
4. Deploying S3 Buckets, RDS, and Load Balancers
5. Hands-on: Full Infrastructure Deployment on AWS

➤ Remote State and Backend Configuration

1. What is a Backend?
2. Configuring Remote Backends (S3)
3. State Locking with DynamoDB (AWS)
4. Secure State Storage Best Practices
5. Hands-on: Set up S3 Backend with Locking

➤ Provisioners and Templating

1. Using local-exec and remote-exec Provisioners
2. Connection Block for Remote Execution
3. File Provisioner to Upload Files
4. Template Files and Interpolation
5. When (and When NOT) to Use Provisioners

➤ Terraform Functions and Expressions

1. Built-in Functions (lookup, join, length, etc.)
2. Conditional Expressions and Loops (for, count, for_each)
3. Dynamic Blocks and Complex Structures
4. Hands-on: Create a Scalable, Parameterized Deployment

➤ Terraform Cloud and Enterprise

1. Introduction to Terraform Cloud
2. Remote Execution and Workspaces in Terraform Cloud
3. Variable and Secret Management
4. Terraform Enterprise Features (RBAC, Policy as Code)
5. Hands-on: Deploy via Terraform Cloud

➤ Ansible:

➤ Introduction to Configuration Management

1. What is Configuration Management?
2. Introduction to Ansible
3. Ansible vs Other Tools (Puppet, Chef, Salt)
4. Use Cases and Benefits of Ansible
5. Architecture Overview – Control Node and Managed Nodes

➤ Setting Up Ansible

1. Installing Ansible (Linux, Mac, Windows via WSL)
2. Understanding Ansible Inventory (Static and Dynamic)
3. Setting up SSH Key Authentication
4. Running Your First Ad-Hoc Command
5. Basic Troubleshooting and Common Errors

➤ Ansible Core Concepts

1. Modules and Ad-Hoc Commands
2. Playbooks – Structure and YAML Syntax
3. Tasks, Handlers, and Tags
4. Variables – Types and Precedence
5. Facts and setup Module

➤ Writing and Running Playbooks

1. Anatomy of a Playbook
2. Executing Playbooks with ansible-playbook
3. Conditional Execution (when)
4. Looping in Playbooks (with_items, loop)
5. Using Tags for Task Selection

➤ Variables and Templates

1. Defining Variables (Inventory, Playbook, Host Vars, Group Vars)
2. Registering Variables from Commands
3. Using Jinja2 Templating
4. Template Module and .j2 Files
5. Variable Precedence and Best Practices

➤ Roles and Reusability

1. Introduction to Roles in Ansible
2. Creating and Using Roles
3. Role Directory Structure
4. Importing Roles from Ansible Galaxy
5. Best Practices for Role-Based Projects

➤ Ansible Inventory Management

1. Static Inventory – INI and YAML
2. Dynamic Inventory (AWS, Azure, GCP, etc.)
3. Inventory Grouping and Variables
4. Host and Group Variable Files
5. Inventory Plugins and Scripts

➤ Error Handling and Debugging

1. Using ignore_errors, failed_when, and block
2. Error Messaging with debug and msg
3. Rescue Blocks and Error Recovery
4. Validating Playbooks (ansible-lint, --syntax-check)
5. Debugging Playbooks and Verbose Modes

➤ Ansible Vault – Managing Secrets

1. What is Ansible Vault?
2. Encrypting and Decrypting Files
3. Using Vault in Playbooks
4. Editing Encrypted Files Securely
5. Vault IDs and Best Practices

➤ Advanced Playbook Techniques

1. Import vs Include
2. Handlers and Notifications
3. Delegation and Local Actions
4. Asynchronous Tasks and Polling
5. Working with Collections

➤ Ansible Tower (AWX)

1. What is Ansible Tower/AWX?
2. Installing and Setting Up AWX
3. Managing Projects, Inventories, and Job Templates
4. Role-Based Access Control (RBAC)
5. Workflow Automation and Notifications
    - o App deployment from Git
    - o Secure access with Vault and RBAC
    - o CI/CD pipeline integration

➤ Grafana + Prometheus

➤ Introduction to Observability

1. What is Observability? (Metrics, Logs, Traces)
2. Monitoring vs Observability
3. Why Grafana and Prometheus?
4. Overview of the Prometheus + Grafana Ecosystem
5. Where They Fit in the DevOps/Cloud Stack

➤ Getting Started with Prometheus

1. What is Prometheus?
2. Prometheus Architecture (TSDB, Exporters, Pull-based model)
3. Installing Prometheus
4. Prometheus Configuration File (prometheus.yml)
5. Running Prometheus and Exploring the UI

AWS (Amazon Web Services)

Module-1 Introduction to AWS and Cloud Computing

1. What is a Data Centre? What is Cloud?
2. What are cloud providers(AWS, Azure, GCP, Oracle cloud, IBM cloud……etc)
3. Benefits of cloud?
4. What is Cloud Computing?
5. AWS Global Infrastructure (Regions, AZs, Edge Locations)
6. AWS Free Tier and Account Setup
7. AWS Management Console and CLI

Module-2 Core AWS Services – Compute

1. Amazon EC2 – Instance Types, Launch, and SSH Access
2. AMI, EBS Volumes, Snapshots
3. EC2 Pricing Models (On-Demand, Reserved, Spot)
4. Auto Scaling and Elastic Load Balancing (ELB)

Module-3 Core AWS Services – Storage

1. Amazon S3 – Buckets, Objects, Storage Classes.
2. Lifecycle Policies and Versioning
3. S3 Permissions and Bucket Policies
4. Amazon EFS and Amazon FSx
5. S3 Transfer Acceleration and Static Website Hosting

Module-4 Core AWS Services – Networking

1. Amazon VPC – Subnets, Route Tables, NACLs, Security Groups
2. Internet Gateway, NAT Gateway, VPC Peering
3. PrivateLink and Transit Gateway
4. Elastic IPs and DNS with Route 53
5. VPC Endpoints

Module-5 AWS IAM–Identity and Access Management

1. IAM Users, Groups, Roles, and Policies
2. Policy JSON – Structure and Examples
3. MFA and IAM Best Practices
4. Resource-Based vs Identity-Based Policies
5. AWS Organizations and Service Control Policies (SCPs)

Module-6

1. Amazon RDS (MySQL, PostgreSQL, etc.)
2. AWS CLI

➤ Monitoring, Logging, Auto-scaling, and Automation

1. Amazon CloudWatch – Logs, Metrics, Alarms
2. AWS CloudTrail – Audit and Governance

➤ Serverless

1. AWS Lambda – Functions and Triggers
2. API Gateway – REST and HTTP APIs

Module-7 Final Capstone Project– Real-World DevOps Pipeline

1. Project Overview
   - o End-to-End DevOps Pipeline for a Sample Web App (Java, Node.js, Python, etc.)
2. Version Control
   - o Store Code in GitHub/GitLab
   - o Branching Strategy (main/dev)
3. Build & Test
   - o Build using Maven/Gradle
   - o Code Quality with SonarQube
   - o Unit Testing
4. CI/CD Integration
   - o Configure Jenkins
   - o Automated Build and Deployment
5. Containerization
   - o Create Dockerfile and Build Docker Image
   - o Push Image to Docker Hub
6. Orchestration & Deployment
   - o Deploy App on Kubernetes Cluster (GKE/EKS/Minikube)