DevSecOps Project Details

Our course includes **10 Major Real-Time Projects**, designed to provide hands-on experience in DevSecOps. These projects focus on CI/CD security, infrastructure as code, container security, and cloud security. Students will work with tools like AWS, Git, Jenkins, Docker, Kubernetes, Terraform, and Ansible. By implementing secure DevOps pipelines, they will learn to build and protect modern applications. This course enhances expertise in integrating security into development workflows.

10 Major Real-Time Projects

1. Automation Deployment Using Jenkins Free Style and Pipeline Jobs:

Tools: Git, GitHub, Jenkins, Maven, Nexus, Sonar & Tomcat.

Description: Write a Jenkins pipeline to get the source code from GitHub to the CI server and build the code using Maven, storing the artifacts on Nexus with the ability to roll back if it fails. Scan the source code using SonarQube to check for bugs and code smells. Deploy the web application on an application server like Tomcat.

2. Microservices Project:

Tools: Git, GitHub, Jenkins, Sonar, Trivy & Docker.

Description: Get the source code from GitHub and scan it using SonarQube. Write a Dockerfile to deploy a static website in Apache, push it to GitHub, and build the Dockerfile in Jenkins by integrating Git with Jenkins. By writing Jenkins declarative pipelines, after building the Docker image, we need to scan it using Trivy and check the dependency installations using OWASP. Then, share the image to Docker Hub and containerize it using Docker.

3. Monolithic Project:

Tools: Git, GitHub, Jenkins, Terraform & Ansible.

Description: Write code for infrastructure using Terraform and automate it with Jenkins. Once the infrastructure is created, write a playbook to install all dependencies and deploy the application by writing the Jenkins declarative pipeline.

4. CI/CD Using Ansible:

Tools: Git, GitHub, Ansible, Jenkins & Tomcat.

Description: Write a playbook to set up an application server (Tomcat) using Ansible and deploy the application with Ansible by integrating it with Jenkins.

5. CI/CD Using AWS CodePipeline:

Services: Git, S3, IAM, GitHub, CodeBuild, CodeDeploy, CodePipeline.

Description: Create an AWS CodePipeline to implement continuous integration and continuous deployment. Whenever a developer pushes code into the central repository, the pipeline is automatically triggered and deployed.

6. Deploy 3-Tier Application Using Docker Compose.

Tools Used: Git, Maven, Jenkins, Docker, Docker-Compose, Tomcat, Datadog.

Description: Deploy a Java-based 3-tier web application using Docker Compose, automated with Jenkins. Monitor the application using Datadog.

7. Deploy 3-Tier Application Using AWS Cloud with High Availability.

Services Used: EC2, VPC, Database, Load Balancers, Auto Scaling Groups.

Description: Deploy a Node.js application on AWS Cloud by creating a high-availability cluster with 1 VPC, 3 private subnets, 2 public subnets, 1 load balancer, and 1 auto scaling group. It's a 3-tier application where users can store and retrieve data at any time.

8. Deploy Blood Bank Application Using Kubernetes Kops Cluster

Services Used: EC2, IAM, S3, AWS CLI, Kops, Kubectl, Prometheus & Grafana.

Description: Deploy a 3-tier application on a Kubernetes Operations Cluster by writing deployment and stateful files to create the pods with high availability. Monitor cluster resources and infrastructure using Prometheus and Grafana.

9. Deploy an E-Commerce Application on EKS Cluster

Services Used: EC2, IAM, S3, AWS CLI, EKS, Kubectl, Prometheus & Grafana.

Description: Create an EKS cluster and deploy an e-commerce application on the EKS cluster by writing deployment and stateful files to create the pods with high availability. To automate the process, we will write multi-branch pipelines in Jenkins.

10. Deploy a Python-Based Application Using Argo CD

Services Used: EC2, IAM, AWS CLI, Kops, Kubectl, Helm & Argo CD.

Description: Create a Kops cluster and deploy a Python application by writing deployment and stateful files to create the pods with high availability. To automate the process, we are implementing GitOps with Argo CD.