

DevOps – Final Hands-On Project

Capstone Project: Full GitOps Pipeline on AWS with Terraform, and Secrets Management

Objective:

Provision and deploy a secure and production-ready infrastructure and CI/CD pipeline using **Terraform**, **Amazon EKS**, and modern DevOps tooling. Students will integrate **Jenkins**, **ArgoCD**, **Argo Image Updater**, and **External Secrets Operator**, and deploy a NodeJs web app with MySQL and Redis.

Project Scope

1. Infrastructure Provisioning – With Terraform (Mandatory)

Provision the following using **Terraform** only:

- **VPC** with 3 public and 3 private subnets across 3 AZs
 - **NAT Gateway, Internet Gateway, Route Tables**
 - **Amazon EKS Cluster**
 - Control plane and node groups in **private subnets**
-

2. CI Tool – Jenkins

- Install Jenkins via Helm into EKS
 - Use a Jenkins pipelines to:
 - Run your terraform code
 - Clone NodeJs app repo
 - Build and push Docker images to **Amazon ECR**
-

3. CD Tool – ArgoCD + Argo Image Updater

- Install ArgoCD via Helm in a separate namespace
 - Set up ArgoCD to:
 - Sync k8s manifests from Git repository
 - Auto-deploy via GitOps
 - Use **Argo Image Updater** to:
 - Monitor image tags in ECR
 - Auto-update image tags in Git
 - Trigger GitOps flow
-

4. Secrets Management – External Secrets Operator [Bonus]

- Install **external secrets operator** via Helm
 - Connect to **AWS Secrets Manager**
 - Automatically sync secrets into Kubernetes Secrets:
 - Database and Redis credentials
-

5. Application: NodeJS App with MySQL and Redis

- Deploy a NodeJS web application (https://github.com/mahmoud254/jenkins_nodejs_example.git)
 - Set up **MySQL and Redis as pods within the EKS cluster**
 - Connect to:
 - The **MySQL pod** using environment variables for configuration
 - The **Redis pod** for caching purposes
 - Use Helm or Kustomize for Kubernetes manifests
-

6. Ingress and HTTPS [Bonus]

- Deploy **NGINX Ingress Controller** or AWS Load Balancer Controller
 - Use **Ingress resources** to expose the app securely
 - Use **cert-manager** and **Let's Encrypt** for HTTPS
-

Deliverables

1. GitHub Repository

Must include:

- **Terraform code** (modular structure)
 - Jenkins pipeline (`Jenkinsfile`)
 - Helm values or Kustomize manifests
 - ArgoCD + Image Updater configs
 - External secrets operator manifests
-

2. Documentation (`README.md`)

Should include:

- Project overview and architecture diagram
 - Setup instructions
 - CI/CD flow explanation
-

3. Demo Presentation (5–10 mins)

Cover:

- Infrastructure deployment with Terraform
 - Pipeline flow (code → build → deploy)
 - App running in EKS
-

III Evaluation Criteria

Area	Points
Terraform infrastructure (EKS, RDS, Redis, VPC)	25
Jenkins CI pipeline	15
ArgoCD GitOps + Image Updater	15
External Secrets with AWS Secrets Manager [Bonus]	5
NodeJs app integration with RDS & Redis	15
Ingress & HTTPS [Bonus]	5
Code quality, structure, Helm/Kustomize usage	10
Documentation & Presentation	20
Total	100

🛠 Tool Stack Summary

Area	Tool
IaC	Terraform
Cloud	AWS (VPC, EKS, Secrets Manager)
CI	Jenkins
CD	ArgoCD + Argo Image Updater
Secrets	External Secrets Operator
App	NodeJs, MySQL, Redis
Other	Helm, Kustomize, cert-manager, NGINX Ingress