DevOps Final Project Requirements

Project Overview

Enhance the mid-term project by integrating advanced DevOps technologies and practices learned during the course.

Duration: 2 weeks

Team Size: Individual or pairs

Complexity: Intermediate

Technical Requirements

1. Application Enhancement

- • Upgrade the Python application based on mid-term feedback
- • Implement file-based data persistence (instead of RAM only)
- • Containerize the application and push to Docker Hub
- • **Application must launch with initial dummy data**

2. Infrastructure as Code

Use **Terraform** to provision AWS infrastructure:

- • 3 EC2 instances
- • Load Balancer
- • Required networking components

3. Configuration Management

Use **Ansible** to configure:

- • Kubernetes cluster on the EC2 instances
- • NFS server for persistent storage
- • **Any additional components required for proper system operation**

Note: This list covers the main requirements. Students are expected to identify and implement any additional components needed for a fully functional system.

4. Kubernetes Deployment

- • Deploy application using **Helm** charts
- • Configure port forwarding and NFS access
- • Ensure high availability across nodes

5. CI/CD Pipeline

- • Implement **GitHub Actions** workflow that starts with running tests, then includes:
  - ○ Building and pushing the container to Docker Hub
  - ○ Running Terraform to provision infrastructure
  - ○ Running Ansible to configure the systems
  - ○ Running Helm to deploy the application
- • **Entire deployment must be automated** through a single CI/CD process
- • Include basic testing in the pipeline
- • If using GitHub Secrets for sensitive data (AWS credentials, passwords, etc.), this must be clearly documented in the README

6. Networking Configuration

- • Configure the infrastructure so that the Load Balancer forwards public port 80 to the appropriate Kubernetes cluster port
- • Ensure end-to-end connectivity from internet to application

Documentation Requirements

Create a comprehensive **README.md** including:

Essential Information

- • Personal details (name, ID)
- • Project title and description
- • GitHub repository URL
- • **Complete step-by-step instructions for evaluators:**
  - ○ How to clone the latest version
  - ○ All tools that need to be installed (with versions)
  - ○ All configurations required for successful deployment
  - ○ Clear instructions for triggering the CI/CD pipeline
- • **Instructions for where and how to add AWS credentials**
- • **How to find the Load Balancer's public IP address**
- • **Link to separate application user guide (separate .md file)**

Technical Documentation

- • Prerequisites and required tools
- • Pipeline stages description

Operations Guide

Testing:

- • Health check procedures
- • Test URLs and endpoints
- • Expected outputs

Troubleshooting:

- • Known issues and solutions
- • Log file locations
- • System cleanup instructions

Additional Documentation

Application User Guide (separate .md file) with:

- • How to use the application
- • Available features
- • API endpoints (if applicable)

Testing & Validation

- • **Primary validation:** Access the application via Load Balancer's public IP on port 80
- • The web application should be fully functional upon first access
- • Dummy data should be visible immediately after deployment

Strong Recommendation

Test your project thoroughly before submission:

1. Create a new GitHub account
2. Clone your project on a clean machine with no pre-installed tools
3. Follow your own README instructions exactly
4. This will help you:
    - • Verify your documentation is complete and clear
    - • Ensure the project will pass evaluation successfully
    - • Identify any missing dependencies or configurations

Submission Guidelines

- • All code must be in a GitHub repository
- • All project files must be zipped and submitted through the Moodle system
- • Except for AWS authentication, the entire process must run automatically

Evaluation Criteria

- • Automation completeness
- • Code organization and documentation
- • Successful integration of all components
- • README clarity and completeness
- • Application accessibility via Load Balancer (port 80)

Good luck!

Focus on creating a working solution rather than perfection.