Institut Supérieur des Études Technologiques de Mahdia

Licence Nationale en Technologies de l'Informatique
Parcours : Réseaux et Services Informatiques

End-to-End DevSecOps Platform

for a Cloud-Native Web Application

Project Documentation

Student : Omar Trabelsi
Academic Year : 2025 – 2026

Contents

Chapter 1 Project Charter

1.1 Project Title

End-to-End DevSecOps Platform for a Cloud-Native Web Application.

1.2 Project Context

Modern software systems require automation, security, scalability, and reliability. Traditional deployment methods are manual, error-prone, and insecure. This project is carried out as part of the final-year internship (PFE) to address these challenges using DevSecOps practices.

1.3 Problem Statement

Organizations often face:

- Manual and slow deployments
- Lack of integrated security
- Poor monitoring and observability
- Limited scalability and availability

1.4 Project Objectives

The main objectives of this project are:

- Automate the CI/CD pipeline
- Integrate security at every stage (DevSecOps)
- Deploy applications using containers and Kubernetes

- Monitor and log system performance
- Provide full technical documentation

1.5 Project Scope

This project covers:

- Application development
- CI/CD automation
- Containerization and orchestration
- Infrastructure as Code
- Security, monitoring, and logging

Chapter 2 Tool Selection Justification

2.1 Version Control

Git and GitHub are used for source code management due to their popularity, collaboration features, and integration with CI/CD tools.

2.2 CI/CD

GitHub Actions is selected for its seamless integration with GitHub repositories and ease of pipeline automation.

2.3 Containerization

Docker enables application portability and consistency across environments.

2.4 Orchestration

Kubernetes is chosen for container orchestration due to its scalability, self-healing, and industry adoption.

2.5 Infrastructure as Code

- **Terraform** for infrastructure provisioning
- **Ansible** for configuration management

2.6 Security Tools

- SonarQube – static code analysis

- Trivy – container vulnerability scanning
- OWASP Dependency-Check – dependency security

2.7 Monitoring and Logging

- Prometheus – metrics collection
- Grafana – visualization and dashboards
- ELK Stack – centralized logging (optional)

Chapter 3 Repository Structure

3.1 Global Repository Organization

```
devsecops-project/
|
+-- app/
|   +-- frontend/
|   +-- backend/
|   `-- database/
|
+-- docker/
|   +-- Dockerfile
|   `-- docker-compose.yml
|
+-- k8s/
|   +-- deployments/
|   +-- services/
|   +-- ingress/
|   `-- secrets/
|
+-- cicd/
|   `-- github-actions.yml
|
+-- terraform/
|   +-- main.tf
|   +-- variables.tf
|   `-- outputs.tf
|
+-- ansible/
|   +-- playbooks/
|   `-- roles/
|
+-- monitoring/
```

```
|   +-- prometheus/
|   `-- grafana/
|
+-- docs/
|   `-- documentation.tex
|
`-- README.md
```

3.2 Structure Justification

This structure ensures:

- Separation of concerns
- Scalability
- Ease of maintenance
- Clear documentation and reproducibility