Cahier des Charges - Mini-projet DevOps

1. Introduction

Ce mini-projet DevOps a pour but de conteneuriser et déployer une application web déjà développée par l'étudiant. L'orchestration se fera sur un cluster Kubernetes local via Docker Desktop. Les services managés (AKS/EKS) restent optionnels mais ne sont pas requis.

Le projet couvre l'automatisation CI, le déploiement Kubernetes, et l'observabilité (monitoring). Des extensions optionnelles (service mesh et IaC) peuvent être ajoutées.

2. Objectifs du projet

- Partir d'une application web déjà développée par l'étudiant (MVC ou microservices).
- Conteneuriser l'application avec Docker (Dockerfile et Docker Compose).
- Déployer l'application sur Kubernetes local (Docker Desktop).
- Mettre en place l'observabilité post-déploiement (Prometheus + Grafana).

3. Contenu du projet

3.1. Application

L'application doit être déjà fonctionnelle et compatible avec la conteneurisation (services, dépendances, variables d'environnement, etc.). Une architecture MVC ou microservices est recommandée.

3.2. Conteneurisation

Le projet exige d'empaqueter l'application dans des conteneurs Docker. Les livrables incluront :

- Un ou plusieurs **Dockerfile(s)** (selon l'architecture).
- Un fichier **docker-compose.yml** pour exécuter et tester l'application en local.

3.3. Intégration continue avec Jenkins

Un pipeline Jenkins automatisera la construction et la distribution. Le **Jenkinsfile** contiendra :

- **Build** : construction de l'image Docker.
- **Scan des vulnérabilités** : utilisation de **Trivy**.
- **Push sur Docker Hub** : publication de l'image si les étapes précédentes sont validées.

3.4. Kubernetes local (Docker Desktop)

Le déploiement se fera sur **Kubernetes (Docker Desktop)**. Les étapes incluront :

- Activation/configuration du cluster Kubernetes local.
- Application des manifestes Kubernetes (Deployments, Services, ConfigMap, etc.).
- Utilisation de **Helm Charts** pour gérer le déploiement.
- (recommandé) Intégration de **ArgoCD** pour une approche GitOps.

3.5. Monitoring et Observabilité

L'intégration d'un système d'observabilité est obligatoire :

- Déploiement de **Prometheus** (métriques application + cluster).
- Intégration de **Grafana** (dashboards, visualisation, alertes simples si souhaité).

4. Bonus (Optionnel)

- **Service Mesh Observability** : ajout de **Istio + Kiali** (visualisation du trafic, topologie, métriques mesh).
- **IaC avec Terraform** : provisionnement reproductible (ex: registre, VM, ou ressources cloud si utilisées) et versionné dans Git.
- Déploiement sur EKS en plus de l'environnement local.

5. Livrables

- Code source de l'application (développée par l'étudiant).
- Dockerfiles et fichier docker-compose.
- Jenkinsfile pour l'intégration continue.
- Manifestes Kubernetes pour Docker Desktop.
- Helm Charts, et ArgoCD.
- Fichiers de configuration Prometheus/Grafana.
- (optionnel) Config Istio/Kiali.
- (optionnel) Code Terraform.
- Documentation : prérequis, installation, déploiement, tests, et captures.