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 (par exemple *Minikube* ou *Kind*). Les services managés comme AKS (Azure Kubernetes Service) ou EKS (Amazon Kubernetes Service) restent optionnels mais ne sont pas requis.

Le projet inclut également la mise en place d'un système de monitoring après le déploiement, intégrant Prometheus et Grafana pour l'observabilité de l'application.

2. Objectifs du projet

- Partir d'une application web déjà développée par l'étudiant (architecture MVC ou microservices).
- Conteneuriser l'application à l'aide de Docker (Dockerfile et Docker Compose).
- Déployer l'application sur un cluster Kubernetes local.
- Mettre en place un monitoring post-déploiement avec Prometheus et Grafana.

3. Contenu du projet

3.1. Application

L'application utilisée pour ce projet doit être déjà fonctionnelle et développée par l'étudiant. Une architecture MVC ou microservices est recommandée, et l'application doit être compatible avec la conteneurisation (pas de dépendances incompatibles).

3.2. Conteneurisation

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

- Un ou plusieurs **Dockerfile(s)** adaptés à chaque service ou composant.
- Un fichier **docker-compose.yml** pour lancer et tester l'application en local.

3.3. Intégration continue avec Jenkins

Un pipeline Jenkins sera utilisé pour automatiser les étapes de construction et distribution. Le **Jenkinsfile** contiendra :

- **Build** : Construction de l'image Docker à partir du *Dockerfile*.
- **Scan des vulnérabilités** : Utilisation de **Trivy** pour détecter les vulnérabilités de sécurité.
- **Push sur Docker Hub** : Si la construction et le scan sont concluants, l'image sera poussée dans un registre Docker.

3.4. Cluster Kubernetes local

Le déploiement se fera sur un cluster Kubernetes local (par exemple *Minikube*, *Kind* ou *k3s*). Les étapes incluront :

- Création et configuration du cluster local.
- Application des manifestes Kubernetes (Deployments, Services, ConfigMap, etc.).
- Utilisation de **Helm Charts** pour gérer le déploiement.
- Intégration de **ArgoCD** pour la mise en place d'une stratégie GitOps.

3.5. Monitoring et Observabilité

L'intégration d'un système de monitoring est obligatoire :

- Déploiement de **Prometheus** pour collecter les métriques de l'application et du cluster.
- Intégration de **Grafana** pour visualiser ces métriques à travers des tableaux de bord personnalisés.

4. Bonus (Optionnel)

- Déploiement sur un cluster managé AKS ou 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 le déploiement local.
- Helm Charts, ArgoCD.
- Fichiers de configuration Prometheus/Grafana.
- Documentation détaillée : instructions de déploiement, configuration, et utilisation.
- (optionnel) Captures sur le déploiement AKS/EKS.