Évaluation Écrite – Module DevOps (CI/CD/CM)

Classe : 4DS11 — Durée : 1h30

1. Compléter le code Jenkinsfile (5 pts)

Complétezlespartiesmanquantespour obtenir une pipeline Jenkins qui : - clone le projet depuis GitHub,-compileavecMaven (sans exécuter les tests), - analyse la qualité avec SonarQube. groovy pipeline

```
{
    agent any stages
        {
            stage('Checkout')
                {
                steps
                    {
                        gitbranch:'main', url: ' https://github.com/example/repo.git '
                    }
                }
            stage('Build')
                {
                 steps {
                        sh'mvn compile -DskipTests '
                    }
                }
             stage('SonarQubeAnalysis')
                {
                 steps {
                        withSonarQubeEnv('sonarqube-server') {
                            sh'mvn sonar:sonar '
                        }
                    }
                 }
        }
}
```

- Expliquezbrièvementlerôledubloc `withSonarQubeEnv`.

Ce bloc Jenkins permet d'exécuter une analyse statique du code via SonarQube pour vérifier la qualité du code, détecter les défauts potentiels et s'assurer du respect des standards avant le déploiement

2. Explication du workflow CI/CD (3 pts)

Expliquez le cheminement complet (workflow) après un **push sur le dépôt Git** :

-Que fait Jenkins automatiquement ?

-Comment Maven et SonarQube interviennent dans la qualité du code ?

-Quelle est la sortie attendue à la fin de cette étape ?

Réponse :

- Jenkins fais l'automatisation de l'orchestration de la partie CI (intégration continue)
- Maven est un outil de build et de gestion de projet par construction automatisé et sonarqube analyse la qualité du code
- la sortie attendue à la fin de cette étape : un rapport contenant les bugs ,erreurs , vulnérabilités permettant de rejeter ou de valider le build

3. Étapes de déploiement Docker (3 pts)

Détaillez les étapes nécessaires pour : 1. Construire une image Docker à partir d'un projet Spring Boot. 2. L'exécuter localement. 3. La pousser sur Docker Hub. (Utilisez les commandes Linux nécessaires.)

Réponse :

1. docker build -t utilisateur/mon-app:latest .
2. docker run -d -p 8080:8080 --name mon-app utilisateur/mon-app-springboot:lates
3. docker login
4. docker push mon-registry/mon-app:latest

4. Corriger le Dockerfile suivant (2 pts)

Corrigez le fichier pour qu'il fonctionne avec une application Spring Boot (JDK17, JAR nommé `app.jar`). Dockerfile

FROM ubuntu	FROM openjdk:17-jdk-slim
COPY app.jar /tmp	COPY target/app.jar app.jar
CMD java -jar /tmp/app.jar	CMD ["java", "-jar", "app.jar"]

- Pourquoi vaut-il mieux utiliser une image JDK officielle plutôt que Ubuntu brut ?

Réponse :

- plus légère et optimisée pour java
- sécurisée et maintenue régulièrement (car elle est officielle)
- évite l'installation manuelle du jdk
- meilleures pratiques Docker

5. Compléter le fichier YAML (3 pts)

Complétez ce fichier `mysql-deployment.yaml` : yaml

apiVersion: apps/v1

kind: Deployment

metadata:

name: mysql

spec: replicas: 1

selector: matchLabels:

app: mysql

template:

metadata:

labels:

app: mysql

spec:

containers:

-name:mysql

image:mysql:8.0

env:

-name:MYSQL_ROOT_PASSWORD

value: mot-de-passe

ports:

-containerPort: 3306

■ Quellecommandepermetd'appliquer ce fichier dans un cluster Kubernetes ?

Réponse : kubectl apply -f mysql-deployment.yaml

6. Expliquer le workflow Kubernetes (3 pts)

Expliquez ce qui se passe après la création du fichier `Deployment` dans Kubernetes :

-Quels composants du cluster interviennent ?

-Comment Kubernetes maintient-il le **desired state** ?

-Que se passe-t-il si un pod tombe en panne ou est supprimé manuellement ?

Réponse :

- les composants intervenants : api-server , etcd , shceduler , controller manager, kubelet
- grace au controller manager
- il sera recréé pour maintenir l'état désiré (maintenir le nombre de réplicas souhaités)

7. Rôle et intégration de Prometheus & Grafana (2 pts)

Expliquez comment Prometheus et Grafana peuvent être utilisés pour:

-surveiller les performances d'une application sur Kubernetes,

-détecter automatiquement les problèmes dans une pipeline CI/CD.

Réponse :

- prometheus collecte les métriques sur les performances de l'application sur kubernetes dans une base de données temporelle , ces métriques peuvent etre utilisées pour créer un dashboard personnalisée , et des alertes seront envoyées sur la dégradation de la performance , détection d'anomalies (prometheus

8. Workflow global CI/CD/CM (2 pts)

Expliquez le rôle de chaque outil dans le pipeline :

>Git → Jenkins → Maven → SonarQube → Docker → Kubernetes → Prometheus/Grafana

Etprécisez comment ils interagissent pour automatiser la livraison et le maintien du service.

Réponse :

- git : un logiciel libre responsable de la gestion des versions du code assurant la collaboration
- jenkins : automatisation de l'orchestration de la CI (intégration continue)
- Maven : build, gestion des dépendances et packaging
- sonarqube : analyse de la qualité du code et sécurité
- Docker : conteneurisation de l'application assurant intégrité et portabilité et optimisation des ressources facilitant la livraison continue
- Kubernetes : orchestration des conteneurs en production
- prometheus/grafana: monitoring et obsevabilité

interraction : toute modification du code (un simple push (git)) elle déclanche jenkins qui orchestrera tous les prochaines étapes en utilisant le jenkinsfile , il fera des tests commençant par le build , les tests packaging (maven) , donner un raport sur la qualité du code avec sonarqube , si le build est validé , création de la nouvelle version de l'image contenant les nouvelles modifications qui sera pushed dans la registry , et utilisées pour faire le déploiement de l'application , ces conteneurs seront orchestrés avec kubernetes , et tous les metriques seront supervisés par prometheus qui collecte les métriques (cpu , mémoire , réseau ..) dans des bases de données temporaires ... ces métriques peuvent etre utilisées pour créer des dashboards àvec grafana afin de visualiser / observer ces métriques