Projet de Fin de Module : Déploiement et Gestion de Microservices avec Docker et Kubernetes

Durée : 4 heures

Objectif : Évaluer la maîtrise des fondamentaux des microservices, la conception et modélisation avec Docker, l'orchestration avec Kubernetes, la scalabilité, la haute disponibilité et la migration des applications vers des microservices.

Contexte

Vous êtes embauché comme ingénieur DevOps pour une entreprise fictive, **MicroShop**, qui souhaite migrer son application monolithique vers une architecture de microservices pour améliorer la scalabilité et la résilience. Votre mission est de déployer plusieurs microservices, gérer les configurations et secrets, surveiller les applications, et assurer la scalabilité et la haute disponibilité.

Étapes du Projet

Partie 1: Déploiement de Microservices avec Docker et Kubernetes

1. **Décomposition en Microservices**
    - Décomposez l'application **MicroShop** en trois microservices : `Product-Service`, `Order-Service`, et `User-Service`.
    - Chaque microservice doit avoir une API REST basique qui renvoie un message simple (e.g., "Hello from Product-Service"). L'API des Products est en Node.js, celle des Orders est en Python et celle des Users est en Golang.
2. **Création des Images Docker**
    - Écrivez les Dockerfiles pour chaque microservice.
    - Construisez les images Docker localement.
3. **Déploiement sur Kubernetes**
    - Créez des fichiers YAML pour déployer chaque microservice sur un cluster Kubernetes.
    - Définissez les déploiements et les services pour chaque microservice.

Partie 2: Configuration et Secrets

1. **Gestion des Secrets**
    - Créez un fichier YAML pour définir un secret Kubernetes contenant une clé API fictive.
2. **(bonus) Utilisation des Secrets dans les Microservices**

- Modifiez le code des microservices pour récupérer la clé API depuis le secret.
- Montez le secret en tant que variable d'environnement dans le fichier de déploiement Kubernetes.

Partie 3: Surveillance et Logging

1. **Configuration du Logging**
   - Configurez les microservices pour envoyer les logs dans un fichier (par exemple, /var/log/user.log).
   - Créez un sidecar container par service qui permet de lire les logs de ce service

Partie 4: Scalabilité et Haute Disponibilité

1. **Stratégies de Scalabilité**

   - Configurez les déploiements Kubernetes pour qu'ils puissent s'adapter automatiquement à la charge en utilisant les Horizontal Pod Autoscalers (HPA). Il doit y avoir au minimum 2 réplicas du service et au maximum 10. La mémoire CPU utilisée doit être de maximum 50%.

2. **(bonus)Test de Résilience**

   - Simulez des pannes pour vérifier la résilience et la récupération automatique des services.

Critères d'évaluation:

- Qualité et fonctionnalité des Dockerfiles.
- Configuration correcte des fichiers YAML pour les déploiements et services Kubernetes.
- Gestion et Utilisation des secrets Kubernetes.
- Mise en place d’outils de surveillance et de logging.
- Configuration de la scalabilité automatique et les tests de résilience.

Ressources

Les secrets

- https://kubernetes.io/docs/tasks/configmap-secret/ (https://kubernetes.io/docs/tasks/configmap-secret/)
- https://kubernetes.io/docs/concepts/configuration/secret/#secret-types (https://kubernetes.io/docs/concepts/configuration/secret/#secret-types)
- https://kubernetes.io/docs/tasks/configmap-secret/ (https://kubernetes.io/docs/tasks/configmap-secret/)

Surveillances et logging

- https://kubernetes.io/docs/concepts/cluster-administration/logging/ (https://kubernetes.io/docs/concepts/cluster-administration/logging/)

Scalabilité et Haute Disponibilité

- https://creativetech-fr.devoteam.com/2023/07/13/kubernetes-techniques-et-astuces-pour-optimiser-votre-deploiement/ (https://creativetech-fr.devoteam.com/2023/07/13/kubernetes-techniques-et-astuces-pour-optimiser-

votre-deploiement/)