Projet Kubernetes - Déploiement d'une application web

Architecture du système

L'application est structurée en trois services:

1. **Frontend** : Une application web React servie via un déploiement Kubernetes.
2. **Backend** : Un serveur API Node.js responsable de la communication avec la base de données MySQL.
3. **Base de Données MySQL** : Un serveur MySQL déployé sur Kubernetes avec une persistance des données via un PVC.

Guide de déploiement

Déploiement du frontend

Commandes à exécuter

Déployer le frontend :
`kubectl apply -f frontend-deployment.yaml`

Créer le service LoadBalancer pour exposer le frontend :
`kubectl apply -f frontend-service.yaml`

Variables d'environnement à personnaliser

- **API_URL** :
  - Définit l'URL du backend. Par défaut, `http://localhost:3000`. Modifiez cette valeur si le backend est exposé sur une autre adresse.

frontend-deployment.yaml : Définit le déploiement de l'application frontend avec l'image Docker spécifiée. Configurer la variable d'environnement `API_URL` pour connecter le frontend au backend.

frontend-service.yaml : Expose le déploiement frontend avec LoadBalance

Déploiement du backend

Commandes à exécuter

Déployez le backend en appliquant le fichier *backend-deployment.yaml* :
`kubectl apply -f backend-deployment.yaml`

Créez le service NodePort pour exposer le backend en local ou en externe :
`kubectl apply -f backend-service.yaml`

backend-deployment.yaml : Définit le déploiement du backend avec l'image Docker spécifiée. Configure les variables d'environnement nécessaires pour se connecter à la base de données MySQL.

backend-service.yaml : Expose le backend via un service de type NodePort pour un accès local ou externe.

Variables d'environnement à personnaliser

- **DB_HOST** : Adresse de la base de données MySQL (par défaut *mysql*).
- **DB_PORT** : Port MySQL (par défaut *3306*).
- **DB_USER** et **DB_PASSWORD** : Identifiants de la base de données.
- **DB_NAME** : Nom de la base de données (par défaut *projetkub*).

Déploiement de MySQL

Commandes à exécuter

Créer la ConfigMap :
`kubectl apply -f mysql-configmap.yaml`

Créer le Secret pour sécuriser les identifiants MySQL :
`kubectl apply -f mysql-secret.yaml`

Déployer MySQL :
`kubectl apply -f mysql-deployment.yaml`

Déployer le service pour exposer MySQL en interne au cluster :
`kubectl apply -f mysql-service.yaml`

Se connecter au pod pour créer les tables :
`kubectl exec -it <nom du pod> -- bash`
`mysql -u <user> -p<password>`

Variables d'environnement à modifier

- **MYSQL_DATABASE** : Nom de la base de données.
- **MYSQL_ROOT_PASSWORD** : Mot de passe administrateur (sécurisé dans le Secret).

Pour tester localement :

- **Frontend**: `minikube service frontend-service`
- **Backend**: `kubectl port-forward pod/<nom du pod> 3000:3000`

Guide de Maintenance

Surveillance

Vérifier les pods en cours d'exécution :

```
kubectl get pods -o wide
```

Vérifier si les services exposent correctement les applications :

```
kubectl get services
```

Inspecter les logs d’un pod spécifique :

```
kubectl get logs
kubectl logs <nom du pod>
```

Surveiller l’état des Déploiements :

```
kubectl get deployments
kubectl describe deployment <nom du déploiement>
```

Mise à l’Échelle

Pour ajuster le nombre de répliques d’un déploiement.

Mise à l’Échelle Manuelle

```
kubectl get deployments
kubectl scale deployment <nom du déploiement> --replicas=<nombre de
réplicas>
```

Mise à l’Échelle Automatique

```
kubectl autoscale deployment <nom du déploiement> --cpu-percent=80 --min=1
--max=5
```

Résolution de Problèmes

Inspecter les événements du Pod :

```
kubectl get pods
kubectl describe pod <nom du pod>
```

Consulter les Logs :

```
kubectl logs <nom du pod>
```

Redémarrer un Pod (un nouveau sera recréé automatiquement) :

```
kubectl delete pod <nom du pod>
```

Vérifier les Services :
```
kubectl get services
```

Tester la Connectivité Entre Pods : créer un conteneur temporaire pour tester la connectivité :
```
kubectl run curl-test --image=appropriate/curl --restart=Never -- curl <nom du service>:<port>
```

Gestion des Volumes et Données Persistantes

Vérifier les PVC :
```
kubectl get pvc
```

Surveiller l'Utilisation de l'Espace de stockage :
```
kubectl exec -it <nom du pod mysql> -- df -h /var/lib/mysql
```

Augmenter l'espace de stockage :
```
spec:
  resources:
    requests:
      storage: 10Gi
```

Appliquer les changements :
```
kubectl apply -f mysql-pvc.yaml
```

Sauvegarde de la Base de Données

Se connecter au pod MySQL de l'intérieur :
```
kubectl exec -it <nom du pod> -- bash
```

Sauvegarder la base de données:
```
mysqldump -u root -p<mot de passe> <nom de la bdd> > /backup.sql
```

Récupérer la sauvegarde :
```
kubectl cp <nom du pod>:/backup.sql ./backup.sql
```