Tests
Docker - Kubernetes – Terraform

Objectif du test : déployer un projet WordPress sur un cluster Kubernetes à l'aide de Terraform

L'utilisation d'un outil tel que Lens pour visualiser ce qu'il se passe dans votre cluster est recommandé.

Informations utiles :
- Le projet est un WordPress 6.3
- Il nécessite un serveur Apache - PHP 8.2
vous pourrez donc utiliser l'image du Docker hub: **php:8.2-apache**
- Le projet nécessite une base de données MySQL 5.7
vous pourrez donc utiliser l'image du Docker hub: **mysql:5.7**

1. **Docker - création de l'image du projet (15 min)**

Générer une image Docker incluant le code du projet, situé dans le dossier "project" du repo, et envoyer là dans le registry privé fourni.

Information utile : le code doit être déposé dans le dossier /var/www/html de l'image.

Vous pouvez vous connecter au registry privé en utilisant la commande :

docker login rg.fr-par.scw.cloud/smartfire-devops-tests -u nologin
[Entrez le password fourni lors de la préparation du test]

Ne pas passer plus de 15 min sur cette question en cas de problème.

2. **Kubernetes / Terraform (40 min)**

À défaut de succès de l'étape 1, utiliser une image officielle du docker hub, par exemple **php:8.2-apache**

Déployer sur le cluster Kubernetes fourni à l'aide de Terraform, en première étape, le projet en version minimale :

- HTTP only (pas de génération de certificat SSL) (voir questions bonus)
- Pas de persistance de données (voir questions bonus)
- Pas de phpmyadmin (voir questions bonus)

Pour la gestion du réseau, vous pourrez utiliser le LoadBalancer fourni, ainsi que l'ingress controler traefik pré-installé.

Concernant l'accès au registry privé, vous pourrez rajouter la clé dans la spec de votre template de deployment :

```
image_pull_secrets {

      name = var.registry_secret_name

}
```

3. Questions bonus

a. Persister le dossier wp-content/uploads du projet, permettant de conserver les images entre plusieurs redémarrages du pod.
b. Persister le fichier wp-config.php, permettant le stockage des informations de connexion à la base de données.
c. Persister les données de la base de données. Informations utiles : il faut persister le dossier "/var/lib/mysql" du container.
d. Installer un phpmyadmin pour pouvoir visualiser les données de la base de données.
e. Prévoir l'utilisation d'un certificat SSL pour l'accès en https (utiliser le domaine tests-devops.smartfire.dev). Lors du débrief, nous pourrons tester votre solution.
f. Ajouter une protection sur l'usage de ressources de l'application.
g. Changer le port d'écoute du container applicatif, par exemple sur 8080.
h. Exposer l'application à l'aide d'un NodePort plutôt qu'un LoadBalancer.