TP

Dans ce TP, nous verrons comment faciliter le cycle de vie d'une application grâce à la méthodologie DevOps.
N'hésitez pas à poser des questions et à échanger entre-vous.
Un compte rendu par personne est demandé contenant les réponses aux questions et les commandes exécutées. Vous les enverrez à l'adresse pro.guillaume.leroy@gmail.com et indiquerez comme objet [TP DevOps] TP2 Nom Prénom au plus tard le 9 octobre 2020.

Installation

En suivant la documentation officielle de Docker, installez-le sur votre machine.

Premiers pas

Toujours en vous appuyant sur la documentation officielle, lancez un conteneur postgres. Par défaut, une base PostgreSQL écoute sur le port 5432.

1. Connectez vous à la base et créez une table avec l'instruction SQL suivante :

```
CREATE EXTENSION "uuid-ossp";
CREATE TABLE techo (id UUID NOT NULL PRIMARY KEY, name varchar(50) NOT NULL UNIQUE);
```

2. Redémarrez votre conteneur. La table existe t-elle toujours ? D'après vous, où est localisé le stockage de votre conteneur ?

3. Supprimez votre conteneur et re-créez le. Quelle différence observez vous ?

4. Recommencez la même opération avec les contraintes suivantes :

- la base de données doit être accessible sur l'hôte sur le port 2345 ;
- la base de données doit se nommer tp_devops ;

- les identifiants de connexion doivent être admin / foo123 ;
- le conteneur a une mémoire limitée à 50 MB ;
- le conteneur ne peut consommer plus d'un CPU ;
- le stockage doit être persistant et les données de la base sauvegardée dans un dossier data à la racine du projet.

5. Observez les ressources utilisées par votre conteneur et expliquez ce que représente chaque métrique.

Docker Compose - partie 1

6. Forkez ce dépôt. Vous y trouverez une application web écrite en Kotlin qui met à disposition un endpoint permettant de récupérer le statut de sa connection à sa base de données.

7. Lancez l'application avec Gradle et vérifiez son bon fonctionnement en réalisant une requête GET sur http://localhost:8080. D'après les logs de l'application, pourquoi le statut de l'application est "down" ?

8. En suivant la documentation officielle, installez Docker Compose.

9. Créez un fichier docker-compose.yml qui contient un service PostgreSQL configuré de telle sorte à ce que notre application puisse s'y connecter. Vérifiez le bon fonctionnement en réalisant une nouvelle fois une requête GET (le statut affiché devrait être "up").

Dockerfile

10. Construisez le projet avec Gradle et exécutez maintenant l'application sans Gradle. Le jar construit se trouve dans le dossier build/libs. Faîtes en sorte que l'application se connecte à la base de données.

11. Créez un dossier docker à la racine du projet et écrivez votre premier Dockerfile décrivant une image qui exécute la commande précédente. Faîtes en sorte que les variables pour se connecter à la base de données soient paramétrables.

12. Construisez cette image et exécutez là.

13. Modifiez le script de build de telle sorte que ce soit Gradle qui construise notre image (appuyez-vous sur ce plugin).

Docker Compose - partie 2

14. Modifiez votre docker-compose.yml de telle sorte à ce qu'un conteneur de notre application soit également lancé, en plus de la base de données. Faîtes en sorte que les deux conteneurs soient présents dans le même réseau.

Docker Hub

15. Créez un compte sur Docker Hub.

16. Taggez votre image de telle sorte à ce qu'elle respecte les conventions de Docker Hub (<nom du compte>/tp-devops).

17. Après avoir fait un docker login, poussez votre image.

GitHub Actions (facultatif)

18. En suivant la documentation officielle, créez un fichier dans votre dépôt pour automatiser la création de votre image et l'upload de cette-ci sur le Docker Hub.

Un peu d’OPS...

19. Créez 4 VMs avec la distribution Linux de votre choix avec les noms d’hôte suivant : lb-01, app-01, app-02 et db-01. Faîtes en sorte que celles-ci puissent communiquer entre elles.

20. Sur db-01, installez un serveur PostgreSQL (>= 9.6).

21. Sur app-01 et app-02, installez l’application comme service systemd. Vérifiez que tout se lance automatiquement lorsque vous redémarrez la machine et que le statut affiché est bien “up” pour les deux instances.

22. Sur lb-01, installez HAProxy et configurez le de telle sorte à ce qu’il redirige les requêtes vers app-01 et app-02.

Pour les plus courageux… (facultatif)

23. Automatisez toute la partie “Un peu d’OPS” avec Ansible ou Puppet.