**Projet de mise en œuvre d'une infrastructure Docker pour POZOS**

**(Student-List)**

**Objectifs :**

Les objectifs de cet examen pratique sont de vous assurer que vous êtes capable de gérer une infrastructure Docker, en vous évaluant sur les points suivants :

**Thèmes :**

- Améliorer le processus de déploiement d'une application existante.

- Versionner votre infrastructure.

- Appliquer les meilleures pratiques lors de la mise en œuvre d'une infrastructure Docker.

- Infrastructure en tant que code.

**Contexte** :

POZOS est une société informatique basée en France qui développe des logiciels pour les lycées. Le département d'innovation souhaite perturber l'infrastructure existante pour garantir qu'elle soit évolutive, facilement déployable et automatisée au maximum.

POZOS souhaite que vous construisiez une preuve de concept (POC) pour montrer comment Docker peut vous aider et à quel point cette technologie est efficace.

Pour ce POC, POZOS vous fournira une application et souhaite que vous construisez une infrastructure "découplée" basée sur Docker.

Actuellement, l'application tourne sur un seul serveur sans scalabilité ni haute disponibilité. Lorsque POZOS doit déployer une nouvelle version, des problèmes surviennent à chaque fois.

En conclusion, POZOS a besoin d'agilité dans son système logiciel.

**Infrastructure :**

Pour ce POC, vous utiliserez une seule machine avec Docker installé. La construction et le déploiement se feront sur cette machine.

POZOS recommande l'utilisation de CentOS 7.6, car c'est le système d'exploitation le plus utilisé dans l'entreprise.

Veuillez noter que vous êtes autorisé à utiliser une machine virtuelle basée sur CentOS 7.6 et non votre machine physique.

La sécurité est un aspect très critique pour le DSI de POZOS, veuillez donc ne pas désactiver le pare-feu ou d'autres mécanismes de sécurité, sauf si vous expliquez vos raisons dans votre livraison.

**Application** :

L'application sur laquelle vous travaillerez s'appelle "student\_list". Cette application est très basique et permet à POZOS d'afficher la liste des élèves avec leur âge.

"student\_list" comporte deux modules :

1. Le premier module est une API REST (avec une authentification de base requise) qui renvoie la liste désirée des élèves au format JSON.

2. Le deuxième module est une application Web écrite en HTML + PHP qui permet à l'utilisateur final d'obtenir une liste d'élèves.

Votre travail consiste à construire un conteneur pour chaque module et à les faire interagir entre eux.

Le code source de l'application est disponible [ici](Lien Drive).

Les fichiers que vous devez fournir (dans votre livraison) sont `Dockerfile` et `docker-compose.yml` (actuellement tous deux vides).

Maintenant, expliquons le rôle de chaque fichier :

- `docker-compose.yml` : pour lancer l'application (API et application Web).

- `Dockerfile` : le fichier qui sera utilisé pour construire l'image de l'API (les détails seront donnés).

- `requirements.txt` : contient tous les packages à installer pour exécuter l'application.

- `student\_age.json` : contient le nom des élèves avec leur âge au format JSON.

- `student\_age.py` : contient le code source de l'API en Python.

- `index.php` : page PHP où l'utilisateur final se connectera pour interagir avec le service pour obtenir une liste d'élèves avec leur âge. Vous devez mettre à jour la ligne suivante avant de lancer le conteneur du site Web pour que `api\_ip\_or\_name` et `port` correspondent à votre déploiement :

$url = 'http://<api\_ip\_or\_name:port>/pozos/api/v1.0/get\_student\_ages';

**Builder et tester** (7 points) :

POZOS vous donnera les informations pour construire le conteneur API.

- Image de base : Pour construire l'image API, vous devez utiliser "python:3.8-buster".

- Ajouter le code source : Vous devez copier le code source de l'API dans le conteneur à la racine "/".

- Prérequis : L'API utilise le moteur FLASK, vous devez donc installer certains packages. Copiez le fichier `requirements.txt` dans le conteneur dans le répertoire racine "/" pour installer les packages nécessaires. Voilà les comandes à executer.

apt update -y && apt install python-dev python3-dev libsasl2-dev python-dev libldap2-dev libssl-dev -y

pip install flask==1.1.2 flask\_httpauth==4.1.0 flask\_simpleldap python-dotenv==0.14.0

- Données persistantes (volume) : Créez un dossier `data` à la racine "/" où les données seront stockées et déclarez-le comme volume.

- Port de l'API : Pour interagir avec cette API, exposez le port 5000.

- CMD : Lorsque le conteneur démarre, il doit exécuter `student\_age.py` (copié dans l’application), donc cela devrait ressembler à quelque chose comme :

CMD [ "python", "./student\_age.py" ]

Construisez votre image et essayez de la lancer (n'oubliez pas de monter le fichier `student\_age.json` dans `/data/student\_age.json` dans le conteneur), vérifiez les journaux et assurez-vous que le conteneur écoute et est prêt à répondre.

Exécutez cette commande pour vous assurer que l'API répond correctement (prenez une capture d'écran à des fins de livraison) :

***curl -u toto:python -X GET http://<host IP>:<API exposed port>/pozos/api/v1.0/get\_student\_ages***

Félicitations ! Vous êtes maintenant prêt pour l'étape suivante (docker-compose.yml).

**Infrastructure As Code** (5 points) :

Après avoir testé votre image API, vous devez tout mettre en place et le déployer en utilisant docker-compose.

Le fichier `docker-compose.yml` déploiera deux services :

- **website** : l'interface utilisateur final avec les caractéristiques suivantes :

- Image : php:apache

- Environnement : vous fournirez `USERNAME` et `PASSWORD` pour permettre à l'application Web d'accéder à l'API via l'authentification.

- Volumes : pour éviter que l'image `php:apache` ne fonctionne avec le site Web par défaut, nous lierons le site Web fourni par POZOS pour l'utiliser. Vous devez avoir quelque chose comme `./website:/var/www/html`.

- Dépendance : vous devez vous assurer que l'API démarrera avant le site Web.

- Port : n'oubliez pas d'exposer le port.

- **API** : l'image construite précédemment doit être utilisée avec les spécifications suivantes :

- Image : le nom de l'image construite précédemment

-Volumes : vous monterez le fichier `student\_age.json` dans `/data/student\_age.json`

- Port : n'oubliez pas d'exposer le port

- Réseaux : n'oubliez pas d'ajouter un réseau spécifique pour votre projet.

Supprimez votre conteneur précédemment créé.

Exécutez votre `docker-compose.yml`.

Enfin, accédez à votre site Web et cliquez sur le bouton "Lister les élèves".

Si la liste des élèves apparaît, vous avez réussi à dockeriser l'application POZOS ! Félicitations (faites une capture d'écran).

**Registre Docker** (4 points) :

POZOS a besoin que vous déployiez un registre privé Docker et stockiez les images construites.

Vous devez donc déployer :

- Un registre Docker (https://docs.docker.com/registry/).

- Une interface Web pour voir les images poussées comme un conteneur (https://hub.docker.com/r/joxit/docker-registry-ui/).

Ou vous pouvez utiliser Portus (http://port.us.org/)pour exécuter les deux.

N'oubliez pas de pousser votre image sur votre registre privé et de les montrer dans votre livraison.

**Livraison** (4 points)

Votre livraison doit être compressée dans un fichier ZIP nommé "prenom.zip" (remplacez "prenom" par votre prénom) et doit contenir :

Un fichier doc ou PDF avec vos captures d'écran et explications.

Les fichiers de configuration utilisés pour réaliser l'exercice noté (docker-compose.yml et Dockerfile).

Votre livraison sera évaluée sur :

La qualité des explications.

La qualité des captures d'écran (pertinence, visibilité).

La qualité de la présentation.

Envoyez votre livraison à benissasonia@gmail.com et nous vous fournirons le lien vers la solution.