MODULE DOCKER : TP Noté

- Groupes de 3 à 6
- 3 h maximum
- Vous gérez votre temps comme vous voulez (Pauses etc…)
- Aide sur Internet autorisé
- 3 questions maximum par groupes (pénalités pour les autres questions)
- C'est un TP de découverte, la note ne sera pas binaire, mais en fonction de vos efforts de recherches et de l'avancement de celui-ci

Informations sur les axes du TP

- Architecture de base : Comprendre le problème et proposer une architecture appropriée en microservice
- Création d'une image de base avec une notion de taille
- Intégration continue : Afin d'améliorer le temps de création de l'image et d'éviter les erreurs humaines
- Docker compose : Création d'un docker compose afin de faciliter le développement de l'application
- Orchestration (Bonus) : Notions de base de Kubernetes et le rolling update

Technologies :

- NodeJS : applicatif
- MongoDB : Data

Bon courage !

https://github.com/Pierre-Malherbe/IsenTPnote

Introduction :

Votre équipe découvre un nouveau projet, une application de gestion de produits pour une grande entreprise de Retail dans le nord de la France. L'équipe de dév a fait une ébauche dans la techno **NodeJS**. Il y a en tout 5 endpoints sur cette application :

- GET : {BaseURL}/products/{id}
- CREATE : {BaseURL}/products/create
    - x-www-form-urlencoded :
        - Name : String
        - Price : Int
- UPDATE : {BaseURL}/products/{id}/update
    - x-www-form-urlencoded :
        - Name : String
        - Price : Int
- DELETE : {BaseURL}/products/{id}/delete
- HEALTHCHECK : {baseURL}/products/healthcheck

L'application a été testé en local par les développeurs, elle est fonctionnelle. Cependant nous voulons la mettre en production avec la techno Docker.

Les informations sont stockées sur une base de données **MongoDB.** *Pour information, MongoDB est une base de données orientée Documents (https://docs.mongodb.com/manual/introduction/).*

La première partie consiste à faire un schéma d'architecture sur papier sur la façon dont vous allez mettre en place l'application. Pour rappel :

- Un conteneur NodeJS
- Un conteneur MongoDB

Cependant le chef de projet a besoin d'accéder à la base MongoDB depuis le réseau de l'entreprise (ici le réseau host). Dans le schéma, nous voulons les conteneurs, le réseau ainsi que les ports exposés. Et pour finir les volumes afin de garder la BD persistante.

(Schéma papier ou sur un logiciel type http://draw.io)

Faites valider le schéma par le chef de projet avant de passer à l'étape 2.

Partie 2

Le schéma a été validé par le chef de projet, à vous de jouer!
La première itération consiste à créer l'image de notre application. Avec un Dockerfile à la racine du projet, vous devez créer une image Docker en copiant le code source dans un dossier "project" sur le conteneur, puis installer les dépendances (npm install). N'oubliez pas d'exposer le port !

Attention : Le projet a peu de budget pour le moment, nous voulons obligatoirement une image < 100 Mo. Regardez les bonnes pratiques et la bonne image de base à utiliser !

- L'équipe de dév nous a fourni aucune information sur le port de base du serveur, avec vos notions ou avec internet, Recherchez le port exposé du projet.

Une fois l'image créée, vous effectuerez un healthcheck sur {baseURL}/products/healthcheck, appelez le chef de projet afin de valider cette étape.
Attention à bien respecter la taille de l'image sinon le chef de projet refusera votre livraison.

Partie 3

Nous avons notre image de base ainsi que le conteneur qui fonctionne. Pour le bon fonctionnement du CRUD nous allons avoir besoin de notre BD MongoDB.
Le chef de projet veut :

- Une version de type alpine de MongoDB
- Utilisateur : ISEN et mot de passe : ISEN
- Nom de la DB : IsenTP
- En mode daemon
- Avec un lien entre le conteneur applicatif et celui-ci afin d'y accéder, ainsi qu'un port exposé sur l'host afin de se connecter hors du conteneur.

Partie 4 :

Nous voulons livrer au plus vite les nouvelles fonctionnalités, vous connecterez le projet sur un logiciel de déploiement continu de votre choix. Nous vous laissons réfléchir à la meilleure solution, le seul point que je veux c'est lors d'un push, l'image est uploadée sur DockerHub.
Attention, nous voulons 2 tags sur votre image :

- Le tag latest
- le tag numéro de build

(Attention à ne jamais stocker vos secrets sur votre code source : utilisez les secrets du service de CI)

Partie 5 :

Maintenant que nous avons une base solide de notre projet, nous voulons que le Product Owner qui a Docker sur son poste puisse déployer l'application facilement. Grâce à votre image qui est sur DockerHub, fournissez-lui un Docker-Compose qui permet d'avoir les 2 conteneurs (MongoDB et l'application). Attention nous voulons que la DB et l'application soient sur le même réseau, et nous voulons un volume docker pour la data de la BD. Fournissez le Docker-Compose au chef de projet afin qu'il vérifie sur son poste si celui-ci fonctionne bien.

(Félicitations, vous avez conteneurisé une application inconnue. Les notions sont plus avancées par la suite du TP, essayez sans vous décourager ! Avec vos recherches internet et les notions du dernier cours, essayez de faire le maximum sur l'exercice bonus.)

Kubernetes (BONUS)

Vous avez réussi votre mission, cependant au café un collègue vous parle de Kubernetes. Vous vous souvenez qu'avec le module Docker du master à l'ISEN vous avez vu rapidement Kubernetes, et que c'est le bon moment d'essayer de le mettre en place sur le projet !

Dans un fichier config.yml, vous configurez le conteneur NodeJs afin de prendre la dernière version de votre build.
Vous décrivez dans le config.yml :

- Le nom du conteneur
- Le port
- La version de l'image à prendre
- Le label env=dev
- Le label appname=IsenTP
- 3 réplicats du conteneur

Vous vérifierez avec le healthcheck que votre application est fonctionnelle.

Dernière étape : MLab fourni des BD MongoDB gratuitement : https://mlab.com/
Faites une BD, modifiez le fichier "app.js" avec les bonnes informations, poussez la nouvelle version.

Grâce à l'objet deployment de Kubernetes, faites une mise à jour de type "Rolling Update" avec la nouvelle version de votre application (numéro de build) ou un conteneur est toujours disponible.

Félicitations ! Vous avez en moins de 3 heures :

- Conteneuriser une application
- Créer une architecture Microservice
- Utiliser une grande partie des notions de Docker (Env, Port, Daemon, etc …)
- Utiliser Kubernetes afin de deployer votre application et utiliser l'objet "deployment" pour faire une rolling update.