Sujet : Conteneurisation d'une Application Full Stack (VueJS, Flask, MariaDB)

Contexte

Des développeurs ont créé une application full stack utilisant Vue.js pour le front-end, Flask pour le back-end, et MariaDB pour la base de données. Le projet a été développé dans un environnement de développement intégré mais doit maintenant être déployé en utilisant Docker pour assurer la portabilité et la scalabilité. Vous travaillerez en groupe de deux pour conteneuriser chaque composant de l'application **sans utiliser Docker Compose**.

Structure du Projet

- **Dossier back** : Contient le code source de l'application Flask.
- **Dossier front** : Contient le code source de l'application Vue.js.
- **Fichier init_db.sql** : Script SQL pour initialiser la structure de la base de données.
- **Fichier nginx.conf** : Configuration pour le serveur NGINX.

Stack Technologique

- **Front-end** : Vue.js, compilé avec Yarn, servi par NGINX sur le port 80 (HTTP).
- **Back-end** : Flask, accessible sur le port 5000.
- **Base de données** : MariaDB, écoutant sur le port 3306.

Instructions Détaillées

1. Conteneur MariaDB

- Créez un conteneur MariaDB.
- Utilisez la version 11 de MariaDB.
- Initialisez la base de données en montant init_db.sql au chemin /docker-entrypoint-initdb.d/init_db.sql dans le conteneur.
- Configurez un volume pour la persistance des données de MariaDB.

2. Dockerfile pour Flask

- Créez un Dockerfile pour construire une image contenant l'application Flask.
- Nommez le conteneur "flask-app" pour la résolution DNS.
- Incluez le code de l'application dans l'image.
- Installer les dépendances via le fichier requirements.txt
- Lancer le serveur python.

3. Front-end Vue.js

- Compilez l'application Vue.js avec Yarn. Commandes à exécuter :
    - yarn install
    - yarn build
- Le code compilé se trouve dans le répertoire /app/dist.
- Créez une image Docker basée sur NGINX pour servir l'application.
- Copiez le code compilé dans /usr/share/nginx/html/ du conteneur.
- Copiez nginx.conf dans /etc/nginx/conf.d/default.conf du conteneur.
- Exposez le port 80.

4. Instructions Complémentaires

- Utilisez Node.js version 16.
- Les images Docker doivent être poussées sur un dépôt DockerHub public.
- Le code source, Dockerfiles et autres fichiers doivent être poussés sur un dépôt Git public.

Livrables

Un document PDF contenant :

- L'URL du dépôt DockerHub.
- L'URL du dépôt Git.
- Toutes commandes Docker à exécuter pour configurer l'application.

Exigences Techniques

- L'application doit fonctionner en exécutant uniquement des commandes Docker SANS DOCKER COMPOSE sur un système Debian 11 avec uniquement Docker installé. Aucune interface graphique ne sera utilisée pour exécuter les commandes Docker, uniquement un terminal.

Note Importante

- Vous ne devez pas utiliser Docker Compose, seulement Docker.

Ce projet évaluera votre capacité à travailler avec Docker pour conteneuriser une application en respectant des directives spécifiques, ainsi que votre compréhension des concepts de persistance des données et de configuration réseau entre les conteneurs.