MEDIOUNA Younouss
FRANCESCO Alexis
ENGELS Lou

Documentation du Projet Docker

1. Introduction

Ce projet utilise Docker pour containeriser une application composée de trois services principaux : un frontend, un backend, et une base de données MySQL. Docker Compose est utilisé pour orchestrer ces conteneurs et gérer les réseaux et volumes nécessaires.

2. Architecture Docker de l'Application

L'architecture Docker de l'application est définie dans le fichier `docker-compose.yml`. Cette section décrit en détail chaque composant de l'architecture.

2.1 Services

Les services sont les composants principaux de l'application. Voici une description de chaque service configuré dans Docker Compose.

2.1.1 Service `frontend`

- **Image**: `younouss1/partiel_dock2:frontend`
- **Ports**: Le frontend écoute sur le port 80 à l'intérieur du conteneur et est mappé sur le port 8082 de l'hôte.
- **Réseaux**: Le frontend est connecté au réseau `app-network`.
- **Dependencies**: Le frontend dépend du service backend.

2.1.2 Service `backend`

- **Image**: `younouss1/partiel_dock2:backend`
- **Ports**: Le backend écoute sur le port 80 à l'intérieur du conteneur et est mappé sur le port 8083 de l'hôte.
- **Réseaux**: Le backend est connecté au réseau `app-network`.
- **Variables d'environnement**:
  - `DB_HOST`: db
  - `DB_DATABASE`: bdd_docker
  - `DB_USER`: root
  - `DB_PASSWORD`: root
- **Dependencies**: Le backend dépend du service db.

2.1.3 Service `db`

- **Image**: `younouss1/partiel_dock2:mysql-5.7`
- **Réseaux**: Le service db est connecté au réseau `app-network`.
- **Variables d'environnement**:
  - `MYSQL_ROOT_PASSWORD`: root
  - `MYSQL_DATABASE`: bdd_docker
- **Volumes**:
  - `db_data`: Monté sur `/var/lib/mysql` pour persister les données de la base de données.
  - `./bdd_docker.sql`: Monté sur `/docker-entrypoint-initdb.d/bdd_docker.sql` pour initialiser la base de données.

2.2 Volumes

Les volumes sont utilisés pour persister les données de la base de données même si le conteneur est supprimé ou redémarré.

- **partiel_dock2_hub_db_data**: Un volume nommé pour stocker les données MySQL.

2.3 Réseaux

Les réseaux permettent une communication sécurisée entre les services Docker.

- **app-network**: Un réseau Docker personnalisé utilisé pour connecter tous les services de l'application.

3. Fichier `docker-compose.yml`

Voici le contenu du fichier `docker-compose.yml` qui définit l'architecture Docker de l'application :

```
yaml
Copier le code
version: '3.8'

services:
  frontend:
    image: younouss1/partiel_dock2:frontend-latest
    ports:
      - "8082:80"
    networks:
      - app-network
    depends_on:
      - backend

  backend:
    image: younouss1/partiel_dock2:backend-latest
    ports:
      - "8083:80"
    networks:
      - app-network
    environment:
      DB_HOST: db
      DB_DATABASE: bdd_docker
      DB_USER: root
      DB_PASSWORD: root
    depends_on:
      - db

  db:
    image: younouss1/partiel_dock2:mysql-5.7
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: bdd_docker
    volumes:
      - db_data:/var/lib/mysql
      - ./bdd_docker.sql:/docker-entrypoint-initdb.d/bdd_docker.sql
    networks:
      - app-network

volumes:
  db_data:

networks:
  app-network:
```

4. Instructions pour Déployer le Projet

4.1 Prérequis

- Docker installé sur votre machine : Installer Docker
- Docker Compose installé sur votre machine : Installer Docker Compose

4.2 Étapes

1. **Authentification sur Docker Hub**

   Assurez-vous que vous êtes connecté à votre compte Docker Hub en utilisant la commande suivante. Remplacez `<your-username>` et `<your-password>` par vos informations de connexion Docker Hub.

   ```
   sh
   Copier le code
   docker login -u <your-username> -p <your-password>
   ```

2. **Puller les images Docker**

   Utilisez les commandes suivantes pour tirer les images nécessaires à partir de Docker Hub sur le dossier du projet.

   ```
   sh
   Copier le code
   docker pull younouss1/partiel_dock2:backend
   docker pull younouss1/partiel_dock2:frontend
   docker pull younouss1/partiel_dock2:mysql-5.7
   ```

3. **Construire et démarrer les conteneurs**

   Utilisez les commandes suivantes pour construire et démarrer les conteneurs en utilisant Docker Compose.

   ```
   sh
   Copier le code
   docker-compose build
   docker-compose up -d
   ```

4. **Accéder aux applications**
   - Front-end : http://localhost:8082/
   - Back-end : http://localhost:8083/back.php

5. Teste de la communication entre les conteneurs et la persistance des données

Pour tester la communication entre les conteneurs, entrez du texte dans l'interface backend via http://localhost:8083/back.php Ce texte doit être stocké dans la base de données et affiché sur le frontend à http://localhost:8082/ Pour vérifier la persistance des données, redémarrez les pages web et les conteneurs en utilisant docker-compose down puis docker-compose up -d. Ensuite, supprimez les conteneurs et les images avec docker-compose down --rmi all et recréez-les. Assurez-vous que les volumes conservent la base de données en vérifiant que le texte est toujours présent après ces opérations.