**Conteneurisation**

**Présenté par:**

**Mathis Douangpanya**

**Scénario choisi : Application Web avec Base de Données**

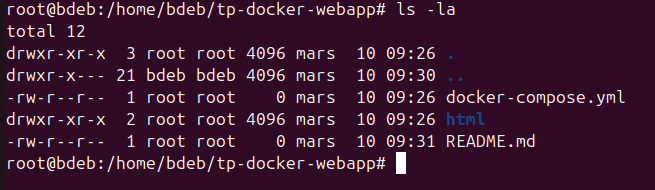
Dans ce projet, on va mettre en place une application web simple fonctionnant avec une base de données en utilisant Docker. L'objectif est de simuler un environnement infonuagique où un serveur web et une base de données communiquent via un réseau personnalisé, avec une gestion sécuritaire.

**Composants nécessaires**

1. **Conteneurs**
   * Un conteneur NGINX qui servira de serveur web pour héberger une application statique ou dynamique.
   * Un conteneur MySQL ou pour gérer la base de données de l’application.
2. **Volume**
   * Un volume Docker pour stocker les fichiers de l’application web afin qu’ils persistent même après l’arrêt du conteneur.
   * Un volume pour stocker les données de la base de données et éviter la perte d’informations en cas de suppression du conteneur.
3. **Réseau**
   * Un réseau Docker personnalisé pour permettre aux conteneurs de communiquer entre eux tout en isolant l’environnement du reste du système.
   * Le serveur web devra être en mesure de se connecter au conteneur de base de données via ce réseau interne.
4. **Sécurité**
   * Utilisation d’images officielles et maintenues pour éviter les vulnérabilités.
   * Interdiction de l’utilisation de l’utilisateur root dans les conteneurs pour limiter les risques de sécurité.
   * Configuration de variables d’environnement pour sécuriser les identifiants de la base de données.
   * Limitation des ressources CPU et mémoires alloués aux conteneurs pour éviter toute surcharge du système.
5. **Test**
   * Nous avons effectué des tests d’insertions avec mon conteneur MYSQL et par la suite on les a affichées dans mon serveur NGINX dans mon conteneur nginx-secure.

**Partie 1 Activité 2 :**

-Création des fichiers nécessaire :

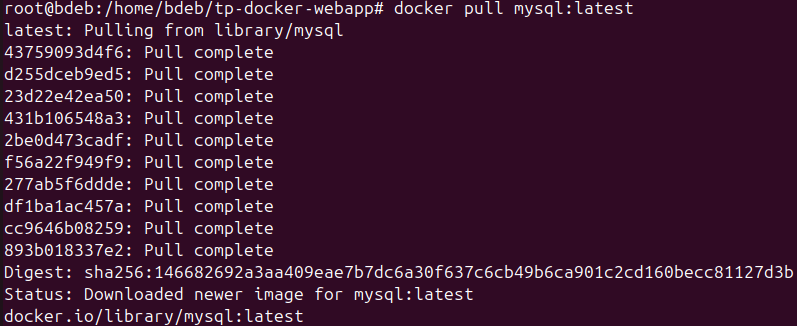
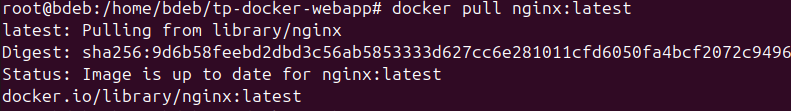


Docker-compose.yml : Fichier de configuration Docker Compose (pas utilisé)

README.md : Documentation du projet

Dossier html: Pour le code de l’application web

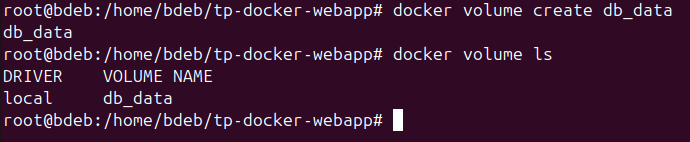
-Télécharger les images Docker requises:

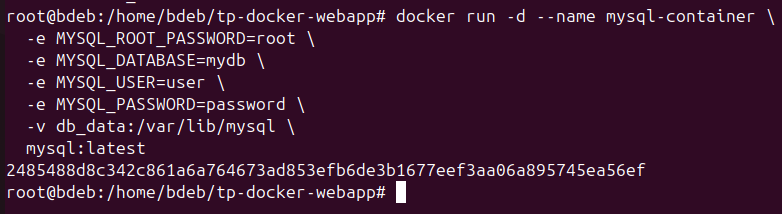


Installation de NGINX : Le serveur web qui va servir notre application web aux utilisateurs.

Installation de MYSQL : Système de gestion de bases de données (SGBD) qui permet de stocker et gérer les données de notre application.

**Partie 2 Activité 3 :**

-Création d’un volume pour MYSQL : 

-Exécution du conteneur:

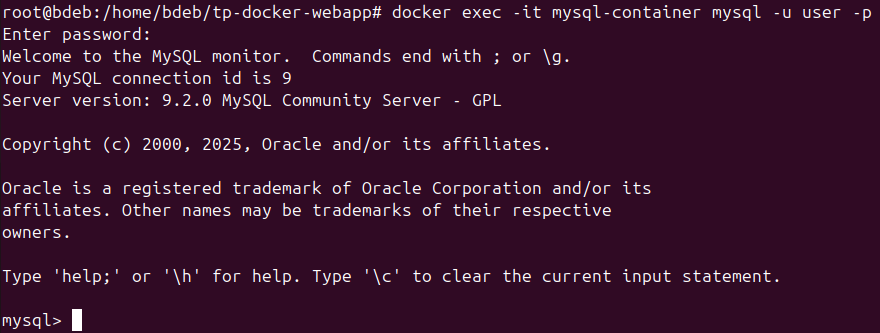
Explication des commandes :  
-d : Exécute le conteneur en arrière-plan.

-name mysql-container : Donne un nom au conteneur.

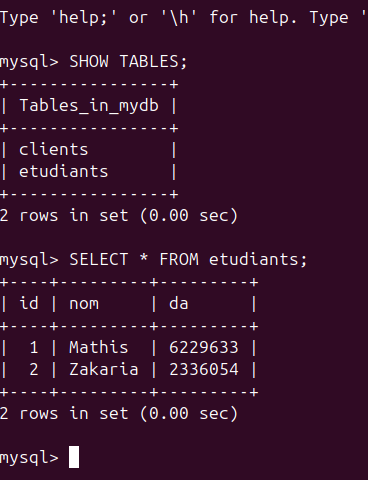
-e MYSQL\_... : Définit les variables d’environnement (mot de passe, utilisateur, base de données).

-v db\_data:/var/lib/mysql : Monte le volume db\_data pour stocker les données de MySQL.

mysql:latest : Utilise l’image MySQL.

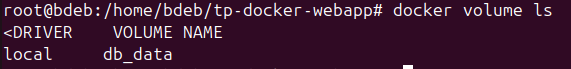
-Connexion MYSQL sur machine Ubuntu :

Test de persistance :



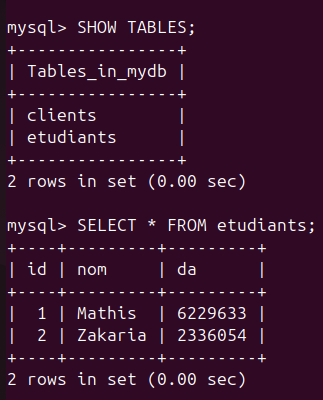
A black background with white text

AI-generated content may be incorrect.



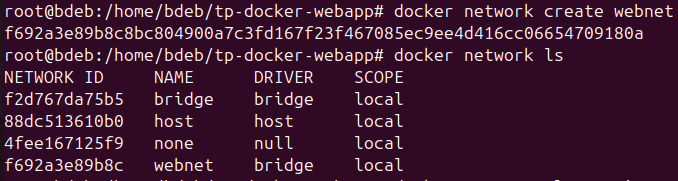
A computer screen shot of a password

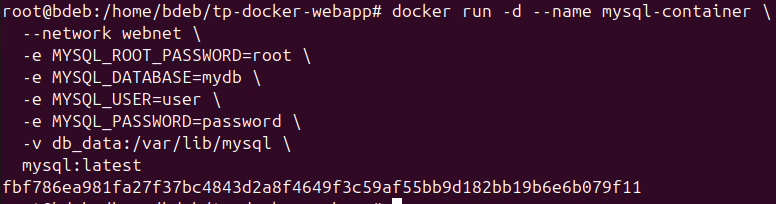
AI-generated content may be incorrect.



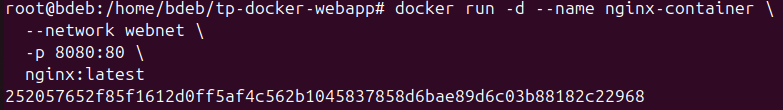
Exécution test persistance: j’ai créé un premier volume, on a par la suite connecté sur MYSQL et j’ai inséré des données. J’ai supprimé et arrêter le container. J’ai ainsi affiché la commande docker volume ls pour être sûr que mon volume ou il y a les données était toujours actif. Par la suite en re créer un nouveau mais cette fois-ci seulement en exécutant SHOW \* FROM users pour voir si les données que j’avais inséré la première fois était encore là. Donc la persistance fonctionne.

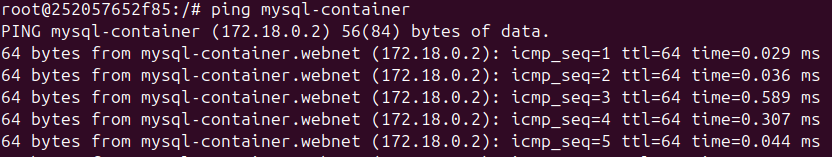
**Partie 2 Activité 4 : Z**

-Création d’un réseau personnalisé :  


-Lancement de MYSQL avec mon réseau personnalisé :

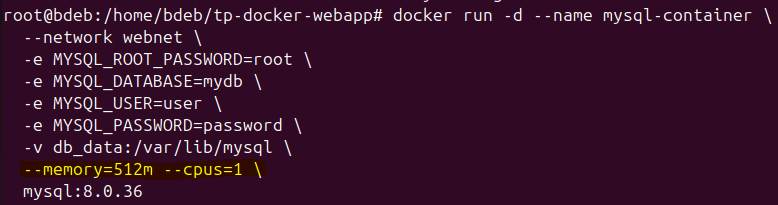
Ici, j’ai supprimé l’ancien conteneur MYSQL que j’avais, et j’en ai relancé un nouveau mais cette fois-ci avec le réseau webnet que j’ai créé.

-Lancement de NGINX avec mon réseau personnalisé :  


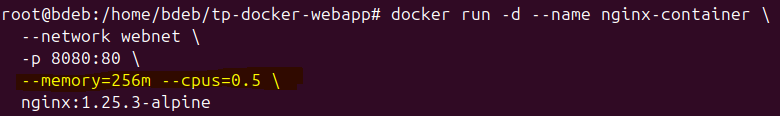
-Test de la connexion entre les deux conteneurs :

Pour la connectivité, on s’est déplacé dans le conteneur NGINX et par la suite, j’ai ping mon conteneur MYSQL.

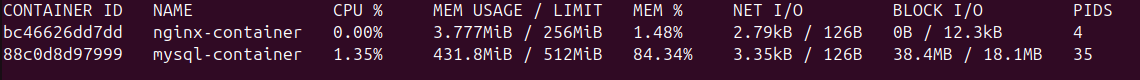
**Partie 2 Activité 5 :**

-Lancement des conteneurs MYSQL et NGINX avec les restrictions CPU/RAM : 

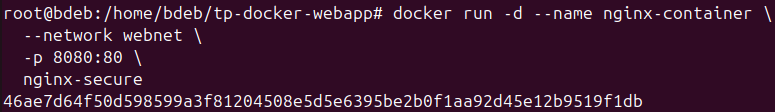
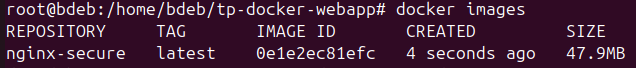
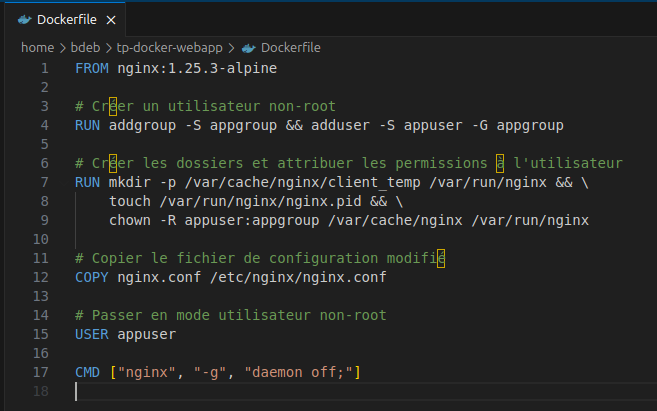
Téléchargement d’une image moin lourde que latest.



Téléchargement d’une image moin lourde que latest.

Voici les stats des conteneurs :

-NGINX secure (pour avoir une sécurité renforcée)



On a créé un Dockerfile personnalisé basé sur NGINX :1.25.3-alpine, en ajoutant un utilisateur non-root (appuser) pour renforcer la sécurité. Ensuite, nous avons construit une nouvelle image Docker nommée nginx-secure avec la commande docker build -t nginx-secure., puis nous avons vérifié sa présence dans la liste des images Docker. Enfin, nous avons relancé le conteneur basé sur cette image, en le connectant au réseau webnet et en exposant le port 80 sur localhost:8080.

Création d’un fichier important :M

Quand on relançait le conteneur NGINX et que on faisait la commande docker logs nginx-container, ça me retournait une erreur qui était :

donc pour réparer cette erreur, il fallait que on créer un fichier nginx.conf :

A computer screen with white text

AI-generated content may be incorrect.Worker\_processes : Définit le nombre de processus de travail, donc 1 procesus.

Error\_logs : Définit le fichier où les erreurs seront enregistrées.

Events : Définit les paramètres liés aux connexions et événements du serveur. Chaque processus de travail peut gérer jusqu'à 1024 connexions simultanées.

Http : Contient toutes les configurations HTTP pour NGINX.

Server : Définit un serveur virtuel qui répond aux requêtes HTTP

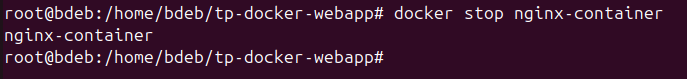
Location : Indique où sont stockés les fichiers à servir

Ce fichier sert à définir comment NGINX va gérer les requêtes, distribuer le trafic, sécuriser les connexions et bien plus encore.

Simulation d’une panne :

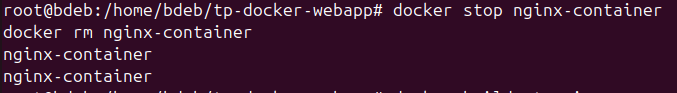


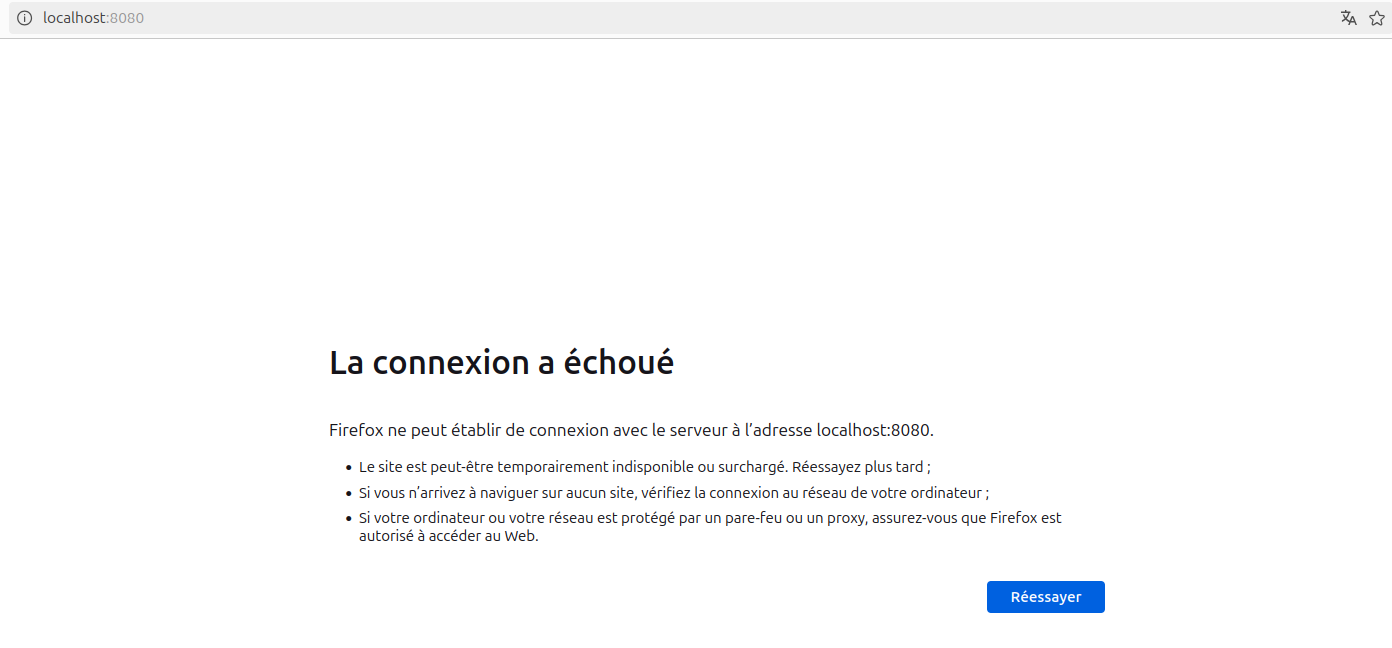
L’arrêt du serveur NGINX :



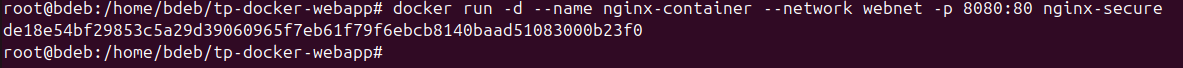


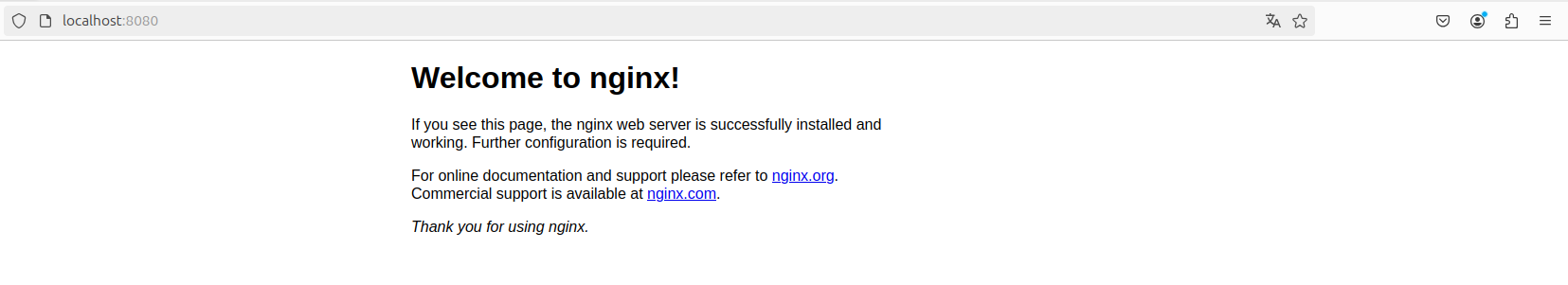
Supprimer le conteneur NGINX :





Relancement du conteneur NGINX :





**Partie 3 Activité 6 :**

**Test d’insertion:**

A screenshot of a computer

AI-generated content may be incorrect.

Dans mon conteneur MYSQL, j’ai inséré des données des clients.

↓↓↓↓↓↓↓↓↓↓↓

Création du fichier script shell :

A screenshot of a computer screen

AI-generated content may be incorrect.A screenshot of a computer code

AI-generated content may be incorrect.

Explication du fichier script shell :

Le fichier update\_html.sh est un script shell qui permet d'automatiser l'extraction des données depuis une base de données MySQL et leur insertion dans un fichier HTML afin qu'elles puissent être affichées par un serveur NGINX. Ce script commence par définir les paramètres de connexion à la base de données, notamment le nom du conteneur MySQL (mysql-container), l'utilisateur, le mot de passe et la base de données ciblée. Ensuite, il crée un fichier HTML (index.html) qui servira de page d'affichage. Il y insère la structure de base d'une page web, comprenant un tableau destiné à recevoir les données de MySQL.

Une requête SQL est ensuite exécutée dans le script, permettant de récupérer les informations des clients enregistrés dans la base de données (SELECT id, nom, email FROM clients). Ces données sont ensuite insérées dynamiquement dans le tableau HTML ligne par ligne, grâce à une boucle while qui lit chaque entrée et la formate en HTML. Une fois toutes les données insérées, le script ferme correctement la structure HTML en ajoutant la fin du tableau et les balises de fermeture.

Pour exécuter ce script, il est nécessaire de le rendre exécutable avec la commande chmod +x update\_html.sh, puis de le lancer avec bash update\_html.sh. Une fois le script exécuté, le fichier index.html mis à jour est accessible via le navigateur en visitant http://localhost:8080/index.html sur le serveur NGINX, affichant ainsi dynamiquement les données de la base MySQL sous forme de tableau sur une page web.

A screenshot of a computer

AI-generated content may be incorrect.

↓↓↓↓↓↓↓↓↓↓↓

Fichier README.md :  
A screenshot of a computer

AI-generated content may be incorrect.

↓↓↓↓↓↓↓↓↓↓↓

A computer screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.