1. **README SPECIALE WORD**

**Résumé de mes recherches liées aux divers exercices**

**Recherche rappelle définition « Docker, Docker Compose, Docker Hub » :**

Docker est un outil de conteneurisation très populaire qui permet d'encapsuler une application et ses dépendances dans un conteneur isolé. Cela facilite le déploiement et la gestion des applications, car le conteneur peut fonctionner de manière cohérente sur divers environnements.

Docker Compose est un outil pour définir et exécuter des applications Docker multi-conteneurs. Avec Docker Compose, vous pouvez utiliser un fichier YAML pour configurer les services de votre application. Cela permet de démarrer et d'arrêter l'ensemble de l'application avec des commandes simples.

Docker Hub est un service de registre pour stocker et partager des images Docker. Les utilisateurs peuvent y pousser leurs propres images ou télécharger des images publiées par d'autres. Cela simplifie le partage et la distribution d'applications conteneurisées.

**Exercice 1**

On a une image docker hub disponible dans dockerhub via ce lien https://hub.docker.com/r/docsseo91/stardocker  
  
Nous devons la récupéré via une commande (terminal)

Pour télécharger l'image Docker depuis Docker Hub j’ai trouvé sur internet et sur le site de docker hub la commandes « docker pull » suivantes

Donc :

**- docker pull docsseo91/stardocker**

Ca ne marche pas

J’ai remarqué un tag spécifique donc je l’ajoute dans le lien   
V1

**- docker pull docsseo91/stardocker:v1  
  
Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquement**

Ça marche

Maintenant j’essaye de lancer un conteneur docker a partir ce cette image

J’ai trouvé cette commande de base   
 **-docker run [OPTIONS] IMAGE [COMMAND] [ARG...]**

Pour notre image :

**-** **docker run -d -p 8001:80 docsseo91/stardocker:v1  
  
j’ai utilisé le port 8001 car 8080 et 8081 déjà utilisé**

Une erreur apparait concernant l’environement linux mais s’est exécuté correctement   
on peut le voir dans l’application docker ou avec **docker ps**

Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Description générée automatiquement

on peut faire **« docker ps »** pour voir tous les conteneurs actifs.

Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Description générée automatiquement

**Exercice 2 :**

1. Cloné le repo

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquement

Une image contenant texte, Appareils électroniques, capture d’écran, logiciel

Description générée automatiquement

# Une image contenant texte, Appareils électroniques, capture d’écran, logiciel Description générée automatiquement

# Une image contenant capture d’écran, texte, logiciel, Logiciel multimédia Description générée automatiquement

**Fichier DOCKER FILE disponible dans ce git**

**« Nom du fichier Dockerfile »**

**Exercice 3**

**Publication de mon image sur DOCKERHUB**

Une image contenant texte, Appareils électroniques, capture d’écran, logiciel

Description générée automatiquementUne image contenant texte, Appareils électroniques, capture d’écran, logiciel

Description générée automatiquement

Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Description générée automatiquement

**Exercice 4**

**Avec Sever Node le docker file ressemble a ceci :**

# j'utilise l'image de base officielle Node.js

FROM node:14-alpine

# Ici je créer un répertoire pour l'application dans le conteneur

WORKDIR /usr/src/app

# Ensuite on copie les fichiers 'package.json' et 'package-lock.json' (s'il existe)

# Car ses fichiers définissent les dépendances de mon projet Node.js

COPY package.json ./

COPY package-lock.json ./

# Puis j'installe les dépendances du projet

RUN npm install

# je copie les fichiers et dossiers de l'application dans le conteneur

COPY . .

# enfin j'expose le port sur lequel l'application va tourner

EXPOSE 3000

# Cette commande me permet de démarrer l'application

CMD ["node", "server.js"]

On build et run le docker

Une image contenant texte, logiciel, Logiciel multimédia, Logiciel de graphisme

Description générée automatiquement

Une image contenant texte, logiciel, Logiciel multimédia, Icône d’ordinateur

Description générée automatiquement

On peut check les conteneurs en cours d’exécution

Une image contenant texte, logiciel, Logiciel multimédia, Icône d’ordinateur

Description générée automatiquement

L’affichage sur le local host sur le port 3000 ressemble à ceci :

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquement

J’ai aussi essayé de lancer avec ngnix comme l’exo 2

Une image contenant texte, logiciel, Logiciel multimédia, Icône d’ordinateur

Description générée automatiquement

Une image contenant texte, logiciel, Logiciel multimédia, Icône d’ordinateur

Description générée automatiquement

**Tout marche bien aussi**

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquement

**Exercice 5**

Étape 1 : Préparons le Dockerfile

FROM postgres:latest

# connexions

ENV POSTGRES\_DB=linus\_db

ENV POSTGRES\_USER=linus

ENV POSTGRES\_PASSWORD= « linusdb1000 »

# On Copie le script SQL dans le conteneur pour l'exécuter au démarrage

COPY db.sql /docker-entrypoint-initdb.d/

Étape 2 : Construisons et lançons notre conteneur

docker build -t mon-postgres .

docker run -d -p 5432:5432 mon-postgres

Étape 3 : Connectons-nous à PostgreSQL

docker ps

docker exec -it [2da6f5e5308e] psql -U linus

Requête : SELECT COUNT(\*) FROM personnes WHERE age > 30;

54 personne sont âgée de plus de 30 ans

**Exercice 6**

Pour créer le réseau je fais la commande

docker network create reseau\_de\_linus

Ensuite j’essaye de lancer 2 le conteneur PostgreSQL sur le réseau Docker

docker run --name mon-postgres --network reseau\_de\_linus -e POSTGRES\_USER=linus -e POSTGRES\_PASSWORD=linusdb1000 -e POSTGRES\_DB=linuss\_db -d postgres

**Exercice 7**