Formation : Création et Administration des conteneurs Docker

I- Installation et configuration de l'environnement de travail

1) Télécharger et Installer Docker Desktop :
   - ❖ Pour Mac/Windows, depuis le site Web Officiel :
     https://www.docker.com/products/docker-desktop
   - ❖ Pour Linux, utiliser les guides suivants :
     Ubuntu : https://docs.docker.com/engine/install/ubuntu/
     CentOS : https://docs.docker.com/engine/install/centos/
2) Démarrer Docker et attendre jusqu'à la fin de chargement
3) Ouvrir une fenêtre de terminal, et lancer les commandes suivantes :
   - **`docker --version`**
   - **`docker info`**

   Que constatez-vous ?
4) Lancer un conteneur docker d'exemple nommé « hello-world » construis par les développeurs Docker pour tester votre installation.
   - **`docker run hello-world`**

   Que constatez-vous ?
5) Lister les images et conteneurs présents sur le système :
   - **`docker images`**
   - **`docker ps`**

   Que constatez-vous ? Comment afficher tous les conteneurs (ceux arrêté inclus)

II- Images et Conteneurs Docker

A- La commande docker image

1) Télécharger l'image du célèbre serveur Web nginx :
   - **`docker image pull nginx`**

   Lister les images disponibles sur le système, que constatez-vous ?
2) Sauvegarder l'image sous format d'archive tar en utilisant la commande :
   - **`docker image save [ID_IMAGE] –o file.tar`**

   Vérifier que le fichier a été bien créé dans le répertoire courant
3) Supprimer l'image sauvegardée en utilisant la commande :
   - **`docker image rm [ID_IMAGE]`**

   Quel est le résultat ?
4) Charger l'image depuis le fichier file.tar grâce à la commande :
   - **`docker image load –i file.tar`**

   Vérifier que l'image a été correctement chargée depuis le fichier.
5) Qu'il est le repository/tag de l'image chargé ?

Les images chargées depuis un fichier ne continent plus de méta-informations. Pour les assigner à une image, on utilise la commande :

- **`docker image tag [ID_IMAGE] nginx:latest`**

B- La commande docker container

1) Démarrer un container issu depuis l'image **nginx** en utilisant la commande :
   - **`docker container run nginx`**
2) Afficher la liste des conteneurs disponible, et vérifier la présence du conteneur nouvellement crée.
3) Arrêter le conteneur en utilisant la commande :
   - **`docker container stop [ID_CONTAINER]`**
4) Supprimer les conteneurs arrêtés en utilisant la commande :
   - **`docker container prune`**

   Afficher à nouveau la liste et vérifier les changements
5) Lancer un nouveau conteneur en utilisant toujours la même image (nginx) et connecter le à la machine locale (localhost) sur le port 8081 en utilisant la commande :
   - **`docker container run –d –p 8081:80 nginx`**

   Visiter l'URL http://localhost:8081, que constatez-vous ?
6) Lancer un nouveau conteneur depuis une image linux (**ubuntu** par exemple) et connecter son terminal au terminal local en utilisant la commande :
   - **`docker container run –it ubuntu`**

   Que Constatez-vous ?
7) Lancer des commandes linux sur la machine (**ls** par exemple). Que se passe-t-il ?
8) Fermer le terminal du conteneur (en utilisant les touches CTRL + P puis CTRL +Q), et essayer d'exécuter une commande (afficher le contenu du répertoire **usr** du conteneur) à l'intérieur du conteneur depuis notre terminal en utilisant la commande :
   - **`docker container exec [ID_CONTAINER] ls /usr`**
9) Créer un dossier nommée **test** dans le répertoire [**/home**] du conteneur, et exécuter une deuxième commande pour confirmer la création du répertoire.
10) Connecter la console du conteneur **nginx** à la console local pour afficher les messages et les logs du conteneur en utilisant la commande :
    - **`docker container logs[ID_CONTAINER]`**

    Que constatez-vous ?
11) Créer un ficher HTML **index.html** avec du contenu personnalisé, et le copier sur le conteneur nginx en utiliant la commande :
    - **`docker cp [LOCAL_PATH] [ID_CONTAINER]:/usr/share/nginx/html`**
12) Visitez l'URL http://localhost:8081/ et vérifier la console. Noter le résultat
13) Supprimer le conteneur créé.

Exercice Pratique

On dispose d'une image d'un projet Springboot sur l'adresse suivante :

https://hub.docker.com/r/jadliaissam/springboot

Cette image contient une API simple contenant une seule route à la racine qui retourne le message : Hello World Spring Boot !!

1) Lancer un conteneur C1 contenant l'API issu de cette image et attachez le sur l'Hôte au port 8080. Visiter l'adresse http://localhost:8080/ . que constatez-vous ?
2) Lister les images et les conteneurs disponibles sur la machine.
3) Est-il possible de changer le port sur l'hôte même si le port est fixe dans le container ? Essayer de changer le port sur l'hôte a 9999 et tester à nouveau.
4) Cette application est conçue de tel façon qu'elle récupère le numéro de port depuis son environnement sous la forme d'une variable au nom de **PORT**.
   Démarrer un nouveau container C2 avec PORT ayant 4500 comme valeur et attacher le au port 5000 dans l'hôte. Vérifier que l'API est accessible au port sélectionné (5000)
5) Lister les conteneurs démarrés et le port occupé par chacun d'eux
6) Supprimer le conteneur C2 qui utilise le port 5000
7) Accéder å l'intérieur du conteneur C1 et ajouter un fichier texte a la racine du système de fichier en utilisant la command linux : **touch**
8) Arrêter le conteneur C1.
9) Redémarrer le conteneur C1 et vérifier que le fichier texte ajouté dans la question précédente est toujours présent.
10) Exécuter le conteneur C1 en mode détaché (en arrière-plan) et observer les journaux (logs) en temps réel pour vérifier le comportement de l'application Spring Boot.
11) Créer une nouvelle image à partir du conteneur C1 avec le fichier texte ajouté.
12) Lister les détails du conteneur C1, tels que l'utilisation des ressources, les ports exposés, etc.
13) Comment spécifiez-vous les ressources (CPU et mémoire) lors du démarrage d'un conteneur Docker ? Donnez un exemple de commande Docker qui limite le conteneur à utiliser une certaine quantité de CPU (1 cpu) et de mémoire (128 mb) lors de son exécution. Vérifier que ces limites sont respectées.
14) Exécuter une commande Shell à l'intérieur du conteneur C1 et afficher les informations système (par exemple, les informations CPU, mémoire, etc.).