	Partie 2 Docker La conteneurisation	Enseignante Nour CHAKER
Matière Virtualisation et cloud computing	Travaux pratiques 6	A.U. 2022-2023

I. Objectifs

- L'objectif de ce travail pratique est de vous familiariser avec les fonctions de base de Docker sur les images. Vous allez apprendre à créer, exécuter et gérer des images Docker en utilisant les commandes de base de Docker.

1. **Introduction aux images Docker**

- Qu'est-ce qu'une image Docker ?

Une image Docker est un fichier binaire qui contient toutes les informations nécessaires pour exécuter une application, y compris

- Code
- Bibliothèques
- Dépendances
- Variables d'environnement
- Fichiers de configuration

Les images Docker sont créées à partir de fichiers appelés "**Dockerfiles**" qui contiennent des instructions pour construire l'image.

- Pourquoi utiliser des images Docker ?

Les images Docker sont légères, portables et peuvent être utilisées pour exécuter des applications sur n'importe quel système qui prend en charge Docker. Les images Docker sont stockées dans des registres Docker, tels que Docker Hub, et peuvent être téléchargées et utilisées par d'autres utilisateurs.

- Comment fonctionnent les images Docker ?

Les images Docker sont utilisées pour créer des conteneurs Docker, qui sont des instances en cours d'exécution d'une image Docker. Chaque conteneur Docker est isolé des autres conteneurs et du système hôte, ce qui garantit que les applications s'exécutent de manière cohérente et prévisible, indépendamment de l'environnement.

- Comment gérer les images Docker ?

Les images Docker peuvent être gérées à l'aide de la ligne de commande Docker, qui fournit des commandes pour rechercher, télécharger, créer, modifier, supprimer et publier des images. Dans ce TP, nous allons explorer les fonctions de base de Docker pour travailler avec des images Docker, y compris la recherche, le téléchargement, l'exécution, la gestion et la publication d'images.

Tâche 1 : Installer Docker

Pour vérifier que Docker est bien installé, ouvrez un terminal et tapez la commande suivante :

```
docker --version
```

Si Docker est installé correctement, vous devriez voir la version de Docker affichée à l'écran.

Tâche 2 : Créer une image Docker

La deuxième tâche consiste à créer une image Docker en utilisant un Dockerfile. Un Dockerfile est un fichier texte qui contient les instructions nécessaires pour construire une image Docker. Dans cette tâche, vous allez créer une image Docker qui contient le serveur Web Nginx et un fichier HTML de base.

1. Ouvrez un éditeur de texte et créez un nouveau fichier nommé **Dockerfile**.
2. Dans le fichier **Dockerfile**, ajoutez les lignes suivantes :

```
FROM ubuntu:latest

RUN apt-get update && apt-get install -y nginx

COPY index.html /var/www/html/

CMD ["nginx", "-g", "daemon off;"]
```

La première ligne spécifie l'image de base que nous allons utiliser, dans ce cas **ubuntu:latest**. La deuxième ligne met à jour la liste des paquets et installe le serveur Web Nginx.

La troisième ligne copie le fichier **index.html** dans le répertoire **/var/www/html/** de l'image. La quatrième ligne définit la commande par défaut qui sera exécutée lorsque le conteneur sera lancé, c'est-à-dire le démarrage de Nginx.

3. Enregistrez le fichier **Dockerfile**.
4. Ouvrez un terminal et déplacez-vous dans le répertoire contenant le fichier **Dockerfile**.
5. Pour créer l'image Docker, exécutez la commande suivante :

```
docker build -t my-nginx-image .
```

Cette commande utilise le Dockerfile pour créer une nouvelle image Docker nommée **my-nginx-image**.

- L'option **-t** permet de donner un nom à l'image.
- Le point **.** à la fin de la commande indique que le Dockerfile se trouve dans le répertoire courant.

Tâche 3 : Exécuter une image Docker

La troisième tâche consiste à exécuter l'image Docker que vous venez de créer à l'aide de la commande **docker run**.

1. Pour exécuter l'image Docker, exécutez la commande suivante :

```
docker run -d -p 80:80 --name my-nginx-container my-nginx-image
```

Cette commande exécute le conteneur Docker à partir de l'image **my-nginx-image**.

- L'option **-d** permet de lancer le conteneur en arrière-plan (mode détaché).
- L'option **-p** permet de mapper le port de votre ordinateur avec le port utilisé par le conteneur. Dans cet exemple, le port 80 du conteneur est mappé sur le port 80 de votre ordinateur.
- L'option **--name** permet de donner un nom au conteneur. Ici, nous avons nommé le conteneur **my-nginx-container**.

2. Pour vérifier que le conteneur est en cours d'exécution, exécutez la commande suivante :

```
docker ps
```

Cette commande affiche la liste des conteneurs en cours d'exécution.

3. Pour accéder au serveur Web Nginx dans le conteneur, ouvrez un navigateur Web et accédez à l'adresse **http://localhost**. Vous devriez voir la page d'accueil de Nginx.
4. Pour arrêter le conteneur, exécutez la commande suivante :

```
docker stop my-nginx-container
```

Cette commande arrête le conteneur nommé **my-nginx-container**.

5. Pour supprimer le conteneur, exécutez la commande suivante :

docker rm my-nginx-container

Cette commande supprime le conteneur nommé **my-nginx-container**. Si vous souhaitez supprimer l'image, vous pouvez utiliser la commande **docker rmi my-nginx-image**.

Tâche 4 : Gérer les images Docker

La quatrième tâche consiste à gérer les images Docker en utilisant différentes commandes telles que **docker images**, **docker tag** et **docker rmi**.

1. Pour afficher la liste des images Docker présentes sur votre système, exécutez la commande suivante :

docker images

Cette commande affiche la liste de toutes les images Docker présentes sur votre système.

2. Pour taguer une image Docker, utilisez la commande **docker tag**. Par exemple, pour ajouter un tag **v1.0** à l'image **my-nginx-image**, exécutez la commande suivante :

docker tag my-nginx-image my-nginx-image:v1.0

Cette commande ajoute un tag **v1.0** à l'image **my-nginx-image**.

3. Pour supprimer une image Docker, utilisez la commande **docker rmi**. Par exemple, pour supprimer l'image **my-nginx-image**, exécutez la commande suivante :

docker rmi my-nginx-image

Cette commande supprime l'image **my-nginx-image** de votre système. Si l'image est utilisée par un conteneur, vous devez d'abord arrêter et supprimer le conteneur avant de pouvoir supprimer l'image.

Si vous avez plusieurs images portant le même nom, vous pouvez utiliser l'option **-f** pour forcer la suppression de toutes les images. Par exemple :

docker rmi -f my-nginx-image

Cette commande supprime toutes les images portant le nom **my-nginx-image**.

4. Vous pouvez également rechercher des images Docker sur Docker Hub, qui est un registre d'images Docker publiques. Pour rechercher une image Docker sur Docker Hub, utilisez la commande **docker search**. Par exemple, pour rechercher une image Docker contenant le serveur Web Apache, exécutez la commande suivante :

docker search apache

Cette commande affiche la liste des images Docker contenant le terme "apache".

Pour télécharger une image Docker à partir de Docker Hub, utilisez la commande **docker pull**. Par exemple, pour télécharger l'image **httpd** (qui contient le serveur Web Apache) à partir de Docker Hub, exécutez la commande suivante :

docker pull httpd

Cette commande télécharge l'image **httpd** à partir de Docker Hub et l'ajoute à votre système.

Tâche 5 : Modifier une image Docker existante

Dans cette tâche, nous allons apprendre à modifier une image Docker existante en utilisant la commande **docker commit**.

La commande **docker commit** permet de créer une nouvelle image Docker à partir d'un conteneur en cours d'exécution. Cette nouvelle image est basée sur l'état actuel du conteneur, c'est-à-dire sur tous les fichiers, paramètres, variables d'environnement, etc. qui ont été modifiés depuis la création de l'image d'origine.

Voici les étapes à suivre pour modifier une image Docker existante :

1. Créez un nouveau conteneur à partir de l'image Docker existante en utilisant la commande **docker run**. Par exemple, pour créer un conteneur à partir de l'image **my-nginx-image**, exécutez la commande suivante :

```
docker run -it my-nginx-image /bin/bash
```

Cette commande crée un nouveau conteneur à partir de l'image **my-nginx-image** et ouvre un shell interactif dans le conteneur.

2. Apportez les modifications souhaitées au conteneur. Par exemple, vous pouvez installer un nouveau logiciel, modifier un fichier de configuration ou ajouter des fichiers supplémentaires.
3. Fermez le shell interactif du conteneur en tapant **exit**.
4. Utilisez la commande **docker ps -a** pour trouver l'ID du conteneur que vous venez de créer.
5. Utilisez la commande **docker commit** pour créer une nouvelle image Docker à partir du conteneur modifié. Par exemple, pour créer une nouvelle image **my-nginx-image-modified** à partir du conteneur dont l'ID est **CONTAINER_ID**, exécutez la commande suivante :

```
docker commit CONTAINER_ID my-nginx-image-modified
```

Cette commande crée une nouvelle image **my-nginx-image-modified** à partir du conteneur modifié.

6. Vérifiez que la nouvelle image a été créée en utilisant la commande **docker images**. Vous devriez voir la nouvelle image **my-nginx-image-modified** dans la liste.

7. Vous pouvez maintenant utiliser la nouvelle image pour créer de nouveaux conteneurs avec les modifications que vous avez apportées. Par exemple, pour créer un nouveau conteneur à partir de la nouvelle image **my-nginx-image-modified**, exécutez la commande suivante :

```
docker run -it my-nginx-image-modified /bin/bash
```

Cette commande crée un nouveau conteneur à partir de la nouvelle image **my-nginx-image-modified**.

Tâche 6 : Publier une image Docker sur Docker Hub

Docker Hub est un service d'hébergement d'images Docker qui vous permet de stocker et de partager des images Docker avec d'autres utilisateurs. Dans cette tâche, nous allons apprendre à publier une image Docker sur Docker Hub.

Voici les étapes à suivre pour publier une image Docker sur Docker Hub :

1. Créez un compte Docker Hub en vous rendant sur le site Docker Hub et en cliquant sur le bouton "Sign up" dans le coin supérieur droit de la page.
2. Connectez-vous à votre compte Docker Hub en exécutant la commande suivante :

```
docker login
```

Cette commande vous demande votre nom d'utilisateur et votre mot de passe Docker Hub.

3. Ajoutez une étiquette à l'image Docker que vous souhaitez publier en utilisant la commande **docker tag**. L'étiquette doit inclure votre nom d'utilisateur Docker Hub et le nom que vous souhaitez donner à l'image. Par exemple, si vous avez une image nommée **my-nginx-image**, vous pouvez ajouter l'étiquette suivante :

```
docker tag my-nginx-image monnomutilisateur/my-nginx-image:latest
```

Cette commande ajoute l'étiquette **monnomutilisateur/my-nginx-image:latest** à l'image **my-nginx-image**.

4. Publiez l'image sur Docker Hub en utilisant la commande **docker push**. Par exemple, pour publier l'image **my-nginx-image** sur Docker Hub, exécutez la commande suivante :

```
docker push monnomutilisateur/my-nginx-image
```

Cette commande télécharge l'image sur Docker Hub et la rend disponible pour tous les utilisateurs.

5. Vérifiez que l'image a été publiée en vous connectant à votre compte Docker Hub et en recherchant l'image dans la liste des images que vous avez publiées.

Félicitations, vous avez réussi à publier une image Docker sur Docker Hub ! Veuillez noter que si vous souhaitez publier une image privée, vous devrez mettre à niveau votre compte Docker Hub vers un compte payant pour pouvoir stocker des images privées.