**Docker**

**Partie 1 : Découverte de Docker (Création d’une image Docker basique)**

1. **Quelles sont les étapes pour construire une image Docker à partir d’un fichier Dockerfile ?**

Voici les étapes à suivre :

1. ouvrir un éditeur de texte (VSCode) et créer un fichier Dockerfile
2. lancer la commande FROM qui permet de définir l’image source
3. RUN qui permet d’exécuter des commandes dans le conteneur
4. ADD qui permet d'ajouter des fichiers dans le conteneur
5. WORKDIR qui permet de définir le répertoire de travail
6. EXPOSE qui permet de définir les ports d'écoute par défaut
7. VOLUME qui permet de définir les volumes utilisables
8. CMD qui permet de définir la commande par défaut lors de l’exécution des conteneurs Docker.
9. **Quel est le rôle de la commande docker build ? Donnez un exemple de syntaxe.**

Avec la commande docker build Docker va créer un conteneur pour chaque instruction. Ce résultat va être sauvegardé dans une layer. Le résultat final est un ensemble de layers qui construisent une image Docker complète.

exemple : je viens de créer mon Dockerfile :

FROM alpine:3.14

CMD ["echo", "Hello World !"]

je lance ensuite via mon terminal, docker build -t mon\_image .

-t permet de donner un nom à mon image Docker.

Pour lancer mon conteneur j’exécute la commande : docker run mon\_image

1. **Quelle est la différence entre les instructions FROM, WORKDIR et CMD dans un fichier Dockerfile ?**

FROM : défini l’image source

WORKDIR : équivalent à cd, permet de définir le répertoire de travail

CMD : correspond à la commande par défaut au lancement du conteneur

1. **Pourquoi est-il important de spécifier une image de base dans le fichier Dockerfile ?**

La première instruction du Dockerfile spécifie l'image de base sur laquelle nous ajoutons de nouvelles couches pour notre application. Le choix de l'image de base est assez important car les fonctionnalités qu'elle embarque peuvent avoir un impact sur la qualité des couches construites dessus.

Dans la mesure du possible, nous devrions toujours utiliser des images officielles qui sont en général fréquemment mises à jour et peuvent présenter moins de problèmes de sécurité.

Le choix d'une image de base peut avoir un impact sur la taille de l'image finale. Si nous privilégions la taille par rapport à d'autres considérations, nous pouvons utiliser certaines des images de base de très petite taille et à faible surcharge. Ces images sont généralement basées sur la distribution alpine et sont étiquetées en conséquence. Cependant, pour les applications Python, la variante slim de l'image officielle Docker Python fonctionne bien dans la plupart des cas (par exemple, python:3.8-slim).

1. **Une fois l’image construite, comment pouvez-vous exécuter un conteneur basé sur cette image ?**

Pour exécuter un conteneur il faut lancer la commande Docker run le\_nom\_de\_limage

1. **Si vous modifiez le fichier source Python utilisé dans le conteneur, que devez-vous faire pour refléter ces modifications dans votre conteneur ?**

Si le fichier source est modifié je dois reconstruire l’image docker en relançant la commande : docker build -t mon\_image .

**Partie 2 : Découverte de Docker-Compose (Architecture multi-conteneurs)**

Vous devez orchestrer une architecture comprenant deux conteneurs :

* Un service PostgreSQL.
* Une API Flask permettant de communiquer avec la base de données.

1. **Quelles informations doivent être définies dans le fichier docker-compose.yml pour configurer un conteneur PostgreSQL ?**

**version: "3"**

**services:**

**db:**

**image: postgres:15**

**container\_name: postgres\_db**

**environment:**

**POSTGRES\_USER: myuser**

**POSTGRES\_PASSWORD: mypassword**

**POSTGRES\_DB: mydatabase**

**volumes:**

**-postgres\_data:/var/lib/postgresql/data**

**ports:**

**- "5432:5432"**

1. **À quoi sert la directive depends\_on dans un fichier docker-compose.yml ?**

depends\_on permet de créer une dépendance entre deux conteneurs. Docker démarrera le service db avant de démarrer le service api

1. **Comment pouvez-vous vous assurer que votre API Flask est correctement connectée à la base de données PostgreSQL ?**

api:

build: ./api

container\_name: flask\_api

environment:

DATABASE\_URL: postgresql://myuser:mypassword@db:5432/mydatabase

ports:

- "5000:5000"

depends\_on:

-db

1. **Que fait la commande docker-compose up --build ?**

Cette commande exécute deux actions principales :

* Reconstruction des images : Si vous modifiez le code ou le Dockerfile d'un service, cette commande reconstruit l'image.
* Démarrage de l'architecture : Elle démarre tous les services définis dans docker-compose.yml et les connecte ensemble.

1. **Une fois votre architecture en marche, comment vérifier que l’API Flask renvoie la bonne réponse depuis le navigateur ou via un outil comme curl ?**

Pour vérifier que Flask fonctionne correctement depuis un navigateur on peut accéder à l’url <http://localhost:5000>.

Si on veut utiliser un outil comme curl, on tape la commande : curl <http://localhost:5000>.

1. **Pourquoi est-il recommandé d’utiliser Docker-Compose au lieu de démarrer manuellement chaque conteneur avec Docker ?**

Avantages de Docker-Compose :

* Simplification : Avec un seul fichier YAML, vous pouvez orchestrer plusieurs services, au lieu de gérer chaque conteneur individuellement.
* Reproductibilité : Les configurations des conteneurs (ports, variables d'environnement, réseaux, etc.) sont centralisées et versionnées.
* Gestion des réseaux : Docker-Compose configure automatiquement un réseau pour que vos services puissent communiquer facilement.
* Facilité de maintenance : Ajouter, modifier ou supprimer des services est beaucoup plus simple.
* Commandes puissantes : docker-compose up, down, ps, logs, etc., permettent de gérer l'ensemble de l'architecture efficacement.

**Partie 3 : Projet final (Chaîne complète de traitement de données)**

Vous devez créer une chaîne complète de traitement de données comprenant :

* Un conteneur PostgreSQL pour stocker les données.
* Un conteneur Python exécutant un script ETL (Extraction, Transformation, Chargement).
* Un conteneur Streamlit pour afficher les données stockées dans la base de données.