TP Docker

1. Application Web de base avec Docker

- **Conteneuriser une application Flask**
- **Écrire un Dockerfile simple**
- **Exécuter une application conteneurisée**

2. Application multi-conteneurs avec Docker Compose

- **Mise en place de plusieurs services (Web + Base de données)**
- **Ecriture d'un fichier docker-compose.yml**
- **Mise en réseau entre conteneurs**

1. Application Web conteneurisée

Objectif : apprendre les bases de Docker en conteneurisant une application Web simple.

Pile technologique :

Python (Flask)

Étapes de mise en œuvre :

1. Configurez votre application Web

Créez une simple application Flask.

Exemple (application Flask dans app.py) :

```
from flask import Flask
app = Flask(__name__)
@app.route("/")
def home():
```

```
        return "Hello, Docker!"
if __name__ == "__main__":
        app.run(host="0.0.0.0", port=5000)
```

2. Écrivez un fichier Docker

Créez un Dockerfile à la racine du projet :

```
FROM python:3.9
WORKDIR /app
COPY requirements.txt .
RUN pip install -r requirements.txt
COPY . .
CMD ["python", "app.py"]
```

3. Créez et exécutez l'image Docker

```
docker build -t flask-app .
```

Exécutez le conteneur :

```
docker run -p 5000:5000 flask-app
```

4. Vérifiez l'application

Ouvrez http://localhost:5000 dans votre navigateur.

2. Application multi-conteneurs avec Docker Compose

Objectif : Apprendre à gérer des applications multi-conteneurs à l'aide de Docker Compose.

Pile technologique :

Flask/Node.js

PostgreSQL/MySQL

Docker Compose

Étapes de mise en œuvre :

1. Configurer l'application Web et la base de données

Créez une application Web simple avec une connexion à une base de données (Flask+PostgreSQL).

Exemple d'application.py :

```
import psycopg2
from flask import Flask
app = Flask(__name__)
def connect_db():
        return psycopg2.connect(
                dbname="mydb",
                user="user",
                password="password",
                host="db"
        )
@app.route("/")
def home():

        conn = connect_db()
        return "Connected to DB!"
if __name__ == "__main__":
        app.run(host="0.0.0.0", port=5000)
```

2. Créez un fichier Docker

```
FROM python:3.9
WORKDIR /app
COPY requirements.txt .
RUN pip install -r requirements.txt
COPY . .
CMD ["python", "app.py"]
```

3. Créez docker-compose.yml

```
version: '3.8'
    services:
        web:
            build: .
            ports:
                - "5000:5000"
            depends_on:
                - db
        db:
            image: postgres
            environment:
                POSTGRES_USER:user
                POSTGRES_PASSWORD: password
                POSTGRES_DB: mydb
```

4. Exécuter avec Docker Compose

docker-compose up –build

5. Vérifiez la connexion

Ouvrez http://localhost:5000 dans votre navigateur.