

TP 2 – Déployer une application PHP avec MySQL sous Docker

Objectif :

- Déployer une application PHP simple avec Apache
- Connecter cette application à une base de données MySQL via Docker
- Travailler avec un réseau Docker et des variables d'environnement

1. Etape 1 : Préparation des fichiers

Créez un dossier nommé `tp2-php-mysql` et ajoutez les fichiers suivants dans un sous-dossier `app/` :

- a. `index.php` :

```
<?php
```

```
echo "<h1>Hello World from Docker + PHP!</h1>";
```

```
try {
```

```
    $pdo = new PDO("mysql:host=mysql-container;dbname=dockerdb", "root", "root1234");
```

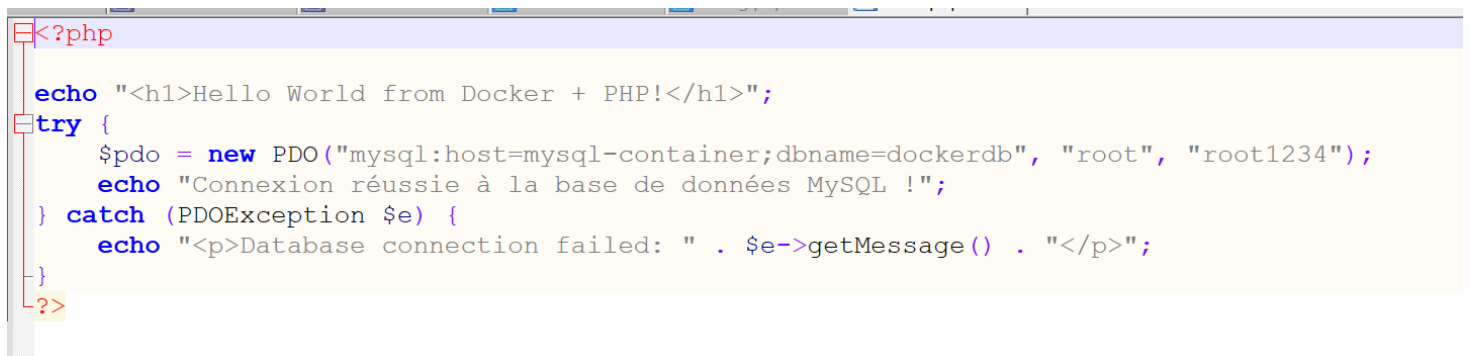
```
    echo "Connexion réussie à la base de données MySQL !";
```

```
} catch (PDOException $e) {
```

```
    echo "<p>Database connection failed: " . $e->getMessage() . "</p>";
```

```
}
```

```
?>
```



```
<?php
echo "<h1>Hello World from Docker + PHP!</h1>";
try {
    $pdo = new PDO("mysql:host=mysql-container;dbname=dockerdb", "root", "root1234");
    echo "Connexion réussie à la base de données MySQL !";
} catch (PDOException $e) {
    echo "<p>Database connection failed: " . $e->getMessage() . "</p>";
}
?>
```

- b. `Dockerfile` :

```
FROM php:8.2-apache
COPY . /var/www/html/
RUN docker-php-ext-install pdo pdo_mysql
```

```
FROM php:8.2-apache

# Copier les fichiers de l'application
COPY app/ /var/www/html/

# Installer l'extension PDO MySQL
RUN docker-php-ext-install pdo pdo_mysql
```

2. Création du réseau Docker

Cette commande permet de créer un réseau **Docker personnalisé** appelé my-network, que tu peux utiliser pour **connecter plusieurs conteneurs entre eux** de manière privée et isolée :

```
docker network create my-network
```

```
PS C:\Users\salah_bentalba\Desktop\EMSI\Docker\TP\tp2 docker-php-app> docker network create my-network
0f33b93b1e264b392ce17aa45c62920ec79308a7729c48952098e8ec9273fac5
```

- **`docker network`** : commande pour gérer les réseaux Docker.
- **`create`** : indique qu'on veut créer un nouveau réseau.
- **`my-network`** : nom que tu choisis pour ton réseau.

Cela aussi évite d'utiliser les adresses IP et permet d'appeler un conteneur par son nom.

3. Lancement du conteneur MySQL

On lance la base de données MySQL :

```
docker run -d --name mysql-container --network my-network \  
-e MYSQL_ROOT_PASSWORD=root1234 \  
-e MYSQL_DATABASE=dockerdb \  
mysql:8.0
```

```
PS C:\Users\salah bentalba\Desktop\EMSI\docker\TP\tp2 docker-php-app> docker run -d --name mysql-container --network my-network -e MYSQL_ROOT_PASSWORD=root1234 -e MYSQL_DATABASE=dockerdb mysql:8.0  
Unable to find image 'mysql:8.0' locally  
8.0: Pulling from library/mysql  
fb10e0b9f49b: Pull complete  
142bf0fa8655: Pull complete  
c2eb5d06bfea: Pull complete  
253ce0d09858: Pull complete  
26b53dc04f70: Pull complete  
8609c4bc4339: Pull complete  
cc6ed31dccc2a: Pull complete  
e0211b440535: Pull complete  
6c1e029c2325: Pull complete  
c0468773f197: Pull complete  
a48ef6c0e15d: Pull complete  
Digest: sha256:51d7ec709cde798d2b5cb3f402a4873166be70302214a85ab55be2188d6728fc  
Status: Downloaded newer image for mysql:8.0  
8cc155c633326976e5853f73868eb651b34f6ccb827ae0a49129b798188f10ce  
PS C:\Users\salah bentalba\Desktop\EMSI\docker\TP\tp2 docker-php-app>
```

- `--name mysql-container` : nom du conteneur
- `--network my-network` : le conteneur rejoint le réseau créé
- `-e` : définit les variables d'environnement (mot de passe root, base testdb)

4. Vérifier la création du conteneur My SQL et la création de la base de données Dockerdb

Exécutez la commande suivante pour lancer le conteneur my SQL

```
docker exec -it mysql-container mysql -u root -p
```

Vous serez amené à saisir le mot de passe root par la suite :

```
PS C:\Users\salah bentalba\Desktop\EMSI\docker\TP\tp2 docker-php-app> docker exec -it mysql-container mysql -u root -p  
Enter password:  
Welcome to the MySQL monitor.  Commands end with ; or \g.  
Your MySQL connection id is 8  
Server version: 8.0.42 MySQL Community Server - GPL  
  
Copyright (c) 2000, 2025, Oracle and/or its affiliates.  
  
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

- **``docker exec``** : Sert à exécuter une commande à l'intérieur d'un conteneur Docker déjà en cours d'exécution.
- **``-it``** : Combine deux options :
 - `-i` = interactif (le terminal reste ouvert)
 - `-t` = alloue un pseudo-terminal (comme si tu étais dans une console Linux)
- **``mysql-container``** : Le nom du conteneur dans lequel on veut exécuter la commande.
- **``Mysql``** : La commande qu'on veut exécuter dans le conteneur (ici, le client MySQL).
- **``-u root``** : Spécifie l'utilisateur MySQL à utiliser pour se connecter (root ici).
- **``-p``** : Indique qu'on souhaite fournir un mot de passe (il sera demandé juste après).

Vous êtes maintenant à l'intérieur du moteur sql. Exécutez la commande suivante pour voir les bases de données et confirmer la création de « dockerd » :

Show databases ;

```
mysql> show databases
->
-> ;
+-----+
| Database |
+-----+
| dockerd  |
| information_schema |
| mysql    |
| performance_schema |
| sys      |
+-----+
5 rows in set (0.01 sec)

mysql> quit
Bye
```

Quittez.

5. Construction de l'image PHP

Depuis le dossier ``tp2-php-mysql/app/``, exécutez :

docker build -t hello-php .

```

C:\Users\salah_bentalba\Desktop\EMSI\Dockert\T\tp2 docker-php-app> docker build -t hello-php .
[+] Building 2.1s (9/9) FINISHED
-> [internal] load build definition from Dockerfile
-> [internal] load metadata for docker.io/library/php:8.2-apache
[auth] library/php:pull token for registry-1.docker.io
-> [internal] load .dockerignore
-> [internal] load build context
-> [internal] load build context
-> [internal] load build context: 58B
-> [1/3] FROM docker.io/library/php:8.2-apache@sha256:e2408924aac97ed8dce0ba54adff30443fe7a940a87d7bd083b36941d8aa431
-> resolve docker.io/library/php:8.2-apache@sha256:e2408924aac97ed8dce0ba54adff30443fe7a940a87d7bd083b36941d8aa431
-> CACHED [2/3] COPY /var/www/html/
-> CACHED [3/3] RUN docker-php-ext-install pdo pdo_mysql
-> exporting to image
-> exporting layers
-> exporting manifest sha256:ea575bc8a3f9e2c928e664f0d08a4f01baefaf556a1c6ab58592c45bbf67996
-> exporting config sha256:434cf5e04bbb764a10c6744ea350fe430cc7b6e9ca8c49afbf5c5520c229282b
-> exporting attestation manifest sha256:70a7be2a9bd296c661b9d071366a37d48d46c5b45f1779dcf05100ee4e84ee7b
-> exporting manifest list sha256:7b42b77e5a2daaf462ee6d44151df636d499e944f071229ac64b5fe9a877d0c
-> naming to docker.io/library/hello-php:latest
-> unpacking to docker.io/library/hello-php:latest
View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/5xx8agud0egsxnw90jpe6cnbi

```

- Cette commande crée une image à partir du `Dockerfile` disponible dans le dossier courant.

6. Lancement du conteneur PHP

Une fois l'image créée, on exécute le conteneur avec :

```
docker run -d --name container-hello-php --network my-network -p 8080:80 hello-php
```

```
PS C:\Users\salah_bentalba\Desktop\EMSI\Docker\TP\tp2 docker-php-app> docker run -d --name container-hello-php --network my-network -p 2020:80 hello-php
8c013901f8ea1e27a02d079195e4f059f9c79401173dc976e584e1a51e0d2158
```

- `-d`` : mode détaché (en arrière-plan)
- `--name container-hello-php` : nom du conteneur
- `-p 8080:80` : redirige le port 80 interne vers le port 8080 de l'hôte
- `--network my-network` : rejoint le même réseau que MySQL
- `hello-php` : Image de notre app php

7. Tester l'application

Ouvrez votre navigateur et accédez à : <http://localhost:8080>

Si tout est correct, le message "Connexion réussie à la base de données MySQL !" s'affiche.



Hello World from Docker + PHP!

Connexion réussie à la base de données MySQL !

8. Nettoyage (optionnel)

Pour arrêter et supprimer les conteneurs :

```
docker stop php-app mysql-container
```

```
PS C:\Users\salah bentalba\Desktop\EMSI\Docker\TP\tp2 docker-php-app> docker stop container-hello-php mysql-container  
container-hello-php  
mysql-container
```

```
docker rm php-app mysql-container
```

```
PS C:\Users\salah bentalba\Desktop\EMSI\Docker\TP\tp2 docker-php-app> docker rm container-hello-php mysql-container  
container-hello-php
```

```
docker network rm my-network
```

```
PS C:\Users\salah bentalba\Desktop\EMSI\Docker\TP\tp2 docker-php-app> docker network rm my-network  
my-network
```

```
docker rmi rm mysql:8.0
```

```
PS C:\Users\salah bentalba\Desktop\EMSI\Docker\TP\tp2 docker-php-app> docker rmi mysql:8.0  
Untagged: mysql:8.0  
Deleted: sha256:51d7ec709cde798d2b5cb3f402a4873166be70302214a85ab55be2188d6728fc
```

Fin du TP – Bravo !