Guide d'Exécution sous Docker – Projet PHP / MySQL

📌1. Pré-requis

Avant de commencer, assurez-vous d'avoir installé :

- ✅Docker
- ✅Docker Compose

Pour vérifier l'installation :

```
docker --version
docker compose version
```

📂2. Structure du Projet

Votre projet doit ressembler à ceci :

```
/mon-projet/
│
├── docker-compose.yml
├── Dockerfile
├── BD.sql
├── db.php
├── index.php
├── signup.php
└── login.php
```

📠3. Lancer le Projet avec Docker

Placez-vous dans le dossier du projet :

```
cd mon-projet
```

Puis lancez Docker :

```
docker compose up --build
```

Ce que fait cette commande :

- # **Construit** l'image PHP depuis votre Dockerfile
- 🚫**Démarre MySQL 8.0**
- 🔯**Importe automatiquement BD.sql** (premier démarrage uniquement)
- 🌐**Démarre phpMyAdmin**
- 🚷Lance votre site PHP sur le port 8080

🩺4. Vérifier que les services fonctionnent

🕐Accéder au site PHP :

http://localhost:8080

🕐Accéder à phpMyAdmin :

http://localhost:8081

Identifiants :

- Serveur : `db`
- User : `root`
- Password : `root`

Si vous voyez la base `grade_management` → tout fonctionne.

🔌5. Tester la connexion MySQL dans PHP

Créez un fichier `test_db.php` :

```
<?php
$host = "db";
$user = "user";
$pass = "userpass";
$db = "grade_management";

$conn = new mysqli($host, $user, $pass, $db);

if ($conn->connect_error) {
    die("❌ Échec de connexion : " . $conn->connect_error);
}

echo "✅ Connexion réussie à MySQL depuis Docker !";
?>
```

Puis ouvrez : http://localhost:8080/test_db.php

🕐6. Réinitialiser la Base de Données

Si vous voulez ré-importer BD.sql :

```
docker compose down -v
docker compose up --build
```

Le `-v` supprime les volumes → MySQL est remis à zéro.

⚙️7. Arrêter les Conteneurs

```
docker compose down
```

📦8. Persistance des Données

Votre fichier docker-compose utilise :

```
volumes:
  db_data:
```

➡️Cela permet de **garder les données MySQL** même si les conteneurs sont arrêtés.

Elles sont stockées dans un volume Docker persistant.

🔧9. Gestion des Logs

Voir les logs en temps réel :

```
docker compose logs -f
```

Conteneur particulier :

```
docker compose logs -f php
docker compose logs -f db
```

📥10. Mise à jour du code sans rebuild

Le volume :

```
- .:/var/www/html
```

fait que toute modification des fichiers PHP est **mise à jour instantanément**.

Pas besoin de rebuild.

🧪11. Tests à effectuer

- ✔️Test inscription : Inscrire un utilisateur → voir s'il apparaît dans phpMyAdmin.
- ✔️Test connexion : Essayer de se connecter avec le nouvel utilisateur.
- ✔️Test notes : Ajouter / modifier des notes → vérifier en base.
- ✔️Test calcul de moyenne : Calculer et afficher le résultat.

👙Conclusion

Ce guide vous permet de :

- Lancer votre projet sous Docker
- Vérifier les services
- Tester DB + PHP
- Réinitialiser la base
- Maintenir votre environnement propre et stable