1) Sur la VM qui héberge la base de données

a) Laisser MySQL/MariaDB écouter sur toutes les interfaces

Vérifiez que MySQL (ou MariaDB) est configuré pour écouter les connexions réseau externes. Dans le fichier de config /etc/mysql/mysql.conf.d/mysqld.cnf ou /etc/mysql/mariadb.conf.d/50-server.cnf, cherchez la ligne :

bind-address = 127.0.0.1

et remplacez-la par :

bind-address = 0.0.0.0

(ou bien spécifiez l’IP de la machine si vous préférez lier MySQL/MariaDB à une IP particulière.)

b) Créer (ou modifier) l’utilisateur MySQL qui a le droit de se connecter à distance

Depuis la VM hébergeant la base, connectez-vous à MySQL en root, puis créez/ajustez l’utilisateur :

CREATE USER 'web\_user'@'%' IDENTIFIED BY 'progtr00';

GRANT ALL PRIVILEGES ON sae303.\* TO 'web\_user'@'%';

FLUSH PRIVILEGES;

web\_user est ici un exemple (celui de votre docker-compose).

@'%' signifie que l’utilisateur pourra se connecter depuis n’importe quelle machine. Vous pouvez être plus restrictif en mettant l’IP de la VM Docker à la place.

sae303.\* donne accès à toutes les tables de la base sae303.

c) Ouvrir le port 3306 sur la VM de la base

Si un pare-feu (ufw, firewalld, iptables, etc.) est activé, assurez-vous d’ouvrir le port 3306 :

sudo ufw allow 3306/tcp

(La commande dépend du pare-feu que vous utilisez.)

2) Dans votre configuration Docker (VM Apache/PHP)

a) Arrêter d’appeler le service “db” local si vous ne l’utilisez plus

Dans votre docker-compose.yml, vous avez un service :

services:

db:

image: mysql:5.7

...

qui tourne localement dans un conteneur. Si vous souhaitez remplacer cette base locale par la base de l’autre VM, il faut désactiver (ou supprimer) ce service db et sa référence dans depends\_on:.

b) Changer l’hôte dans votre code PHP

Dans votre index.php, vous utilisez :

$servername = "db"; // Nom du service Docker

$username = "web\_user";

$password = "progtr00";

$dbname = "sae303";

Au lieu de "db", mettez l’IP (ou le nom DNS) de la VM qui héberge MySQL, par exemple :

$servername = "10.0.0.12"; // IP de la VM MySQL

Le reste (username/password/dbname) doit correspondre aux informations créées sur la base distante.

c) Vérifiez la connectivité réseau depuis la VM Docker

Votre conteneur Docker doit pouvoir “voir” l’IP de la VM MySQL.

En général, si les deux VM sont dans le même réseau local ou la même plage d’adresses, un simple ping 10.0.0.12 (depuis la VM Docker, voire dans le conteneur) doit répondre.

Si vous êtes sur des réseaux différents, il faudra configurer la route ou VPN nécessaire.

d) (Optionnel) Ajuster docker-compose pour passer la variable d’environnement

Vous pouvez définir dans votre docker-compose.yml (pour le service Apache) une variable :

services:

apache:

build: .

container\_name: apache

ports:

- "5000:80"

environment:

- DB\_HOST=10.0.0.12

- DB\_USER=web\_user

- DB\_PASS=progtr00

- DB\_NAME=sae303

...

Et dans votre index.php :

$servername = getenv('DB\_HOST');

$username = getenv('DB\_USER');

$password = getenv('DB\_PASS');

$dbname = getenv('DB\_NAME');

Ainsi, vous séparez la configuration du code, ce qui est plus propre.

3) Résumé

Sur la VM qui héberge MySQL :

Configurer bind-address = 0.0.0.0,

Créer un utilisateur avec droit d’accès distant,

Ouvrir le port 3306 sur le pare-feu.

Dans Docker (VM Apache/PHP) :

Supprimer ou désactiver le conteneur MySQL local si vous ne l’utilisez plus,

Modifier l’adresse du serveur MySQL dans votre code PHP (ou via des variables d’environnement) pour pointer vers l’IP (ou DNS) de la VM MySQL,

Vérifier que le trafic réseau est autorisé entre les deux VM.

Après cela, votre conteneur Apache/PHP devrait pouvoir communiquer avec la base de données située dans l’autre VM.