**Procédure de déploiement**

**Environnement**   
Un serveur Linux qui gère le serveur web nginx et mes fichier de mes application  
Et un serveur Windows 2022 qui gère la base de données MySQL.

**Sur le serveur Windows:**

Configurer la connexion à la base de données MySQL depuis le serveur Linux

Pour permettre à mes applications du serveur Linux de communiquer avec MySQL sur le serveur Windows, il faut que MySQL soit configuré pour accepter les connexions distantes. Sur le serveur Windows, il faut donc ajuster les paramètres de MySQL et du pare-feu pour autoriser les connexions sur le port 3306 (le port par défaut de MySQL).

Ouvrir le fichier de configuration de MySQL (my.ini ou my.cnf) et vérifier que la ligne bind-address est configurée pour écouter sur une adresse IP accessible ou commentée pour écouter sur toutes les interfaces.

## Étape 1: Accéder à MySQL depuis l’invite de commande du serveur Windows 2022

Ajouter répertoire bin de MySQL à votre variable d'environnement PATH

Ouvrez CMD ou PowerShell.

$ mysql -u root -)

Saisir le mot de passe de l'utilisateur MySQL.

## Étape 2: Créer la base de données sur votre serveur Windows

Créer la base de données en exécutant la commande SQL suivante :

$ CREATE DATABASE VACA\_MEET;

$ SHOW DATABASES ; // pour vérifier

## Étape 3 : Créez un utilisateur MySQL spécifique pour l’application.

L’utilisateur doit avoir droit de se connecter depuis l'adresse IP du serveur Linux ou depuis n’importe quelle adresse ip (%), et tous les droits nécessaires sur la base de données.

Commande SQL pour créer un utilisateur:

$ USE VACA\_MEET; // pour être sur la base de données VACA\_MEET

$ CREATE USER 'utilisateur\_vaca'@'10.14.3.1' IDENTIFIED BY 'adrien';

$ GRANT ALL PRIVILEGES ON VACA\_MEET.\* TO 'utilisateur \_vaca @'10.14.3.1';

$ FLUSH PRIVILEGES; // pour valider les nouveaux paramètres des privilèges

$ SHOW GRANTS FOR 'utilisateur \_vaca@'%';

Ensuite copier le script SQL de la base à importer puis vérifier :  
$ SHOW TABLES;  
Plus faire des requête SELECT pour voir si les données sont-elles aussi bien importées

**Sur le serveur Linux**

## Étape 1: Cloner vos projets Git sur le serveur Linux

$ sudo apt install git

Ensuite, naviguer vers le dossier où l’on souhaite cloner le projet (par exemple, /var/www/) et exécutez la commande de clonage :

$ cd /var/www/

$ git clone url\_de\_votre\_projet\_github

## Étape 2 : Configurer Nginx pour servir votre application

Naviguer vers le dossier /etc/nginx/sites-available/.

Créez un fichier de configuration pour votre site. Utiliser le nom du projet

Dans ce fichier, configurer le serveur pour pointer vers le dossier du projet cloné.

server {

listen 80;

server\_name Vaca-Meet-Application-Web;

location / {

root /var/www/ Vaca-Meet-Application-Web

index index.php;

}

}

Activer le site en créant un lien symbolique de votre fichier de configuration dans le dossier /etc/nginx/sites-enabled/.

$ sudo ln -s /etc/nginx/sites-available/monFichierDeConfig /etc/nginx/sites-enabled/

Testez la configuration de Nginx et redémarrez le service :

$ sudo systemctl reload nginx

$ sudo systemctl status nginx

## Étape 3. Modifier le fichier /etc/hosts pour la résolution locale

Si Vaca-Meet-Application-Web est un nom de domaine local (c'est-à-dire, qu'il n'est pas enregistré publiquement), I faut ajouter une entrée dans le fichier /etc/hosts de votre machine cliente (et potentiellement sur le serveur si vous y accédez localement) pour faire correspondre ce nom de domaine à l'adresse IP du serveur.

Ouvrit le fichier /etc/hosts sur la machine cliente et ajoutez une ligne comme suit :

10.14.3.1 Vaca-Meet-Application-Web

## Étape 4 : Vérifier les écoutes de Nginx

$ sudo netstat -tuln | grep :80

Normalement ça nous affiche une ligne indiquant que nginx écoute sur le port 80.

Maintenant il faut modifier le fichier config.ph et revoir tous les chemins de tous les fichiers de mon application pour les faire correspondre

## Configurer le fichier config.php

<?php

$host = '10.14.3.5'; // Adresse IP de votre serveur MySQL Windows

$dbName = 'VACA\_MEET'; // Le nom de votre base de données

$user = 'utilisateur\_vacar'; // L'utilisateur MySQL

$password = 'xxxxxx'; // Le mot de passe de l'utilisateur MySQL

try {

$pdo = new PDO("mysql:host=$host;dbname=$dbName", $user, $password);

// Définir le mode d'erreur PDO sur exception

$pdo->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

echo "Connexion réussie";

} catch(PDOException $e) {

echo "Erreur de connexion : " . $e->getMessage();

}

?>

## Tester la connexion

Après avoir configuré votre config.php, tester la connexion à la base de données en accédant à un script PHP qui utilise ce fichier de configuration sur votre serveur Linux. Si la connexion est réussie, le message "Connexion réussie" devrait s'afficher. Sinon, vérifiez les messages d'erreur pour identifier le problème (par exemple, problème de connexion réseau, informations d'identification incorrectes, etc.).