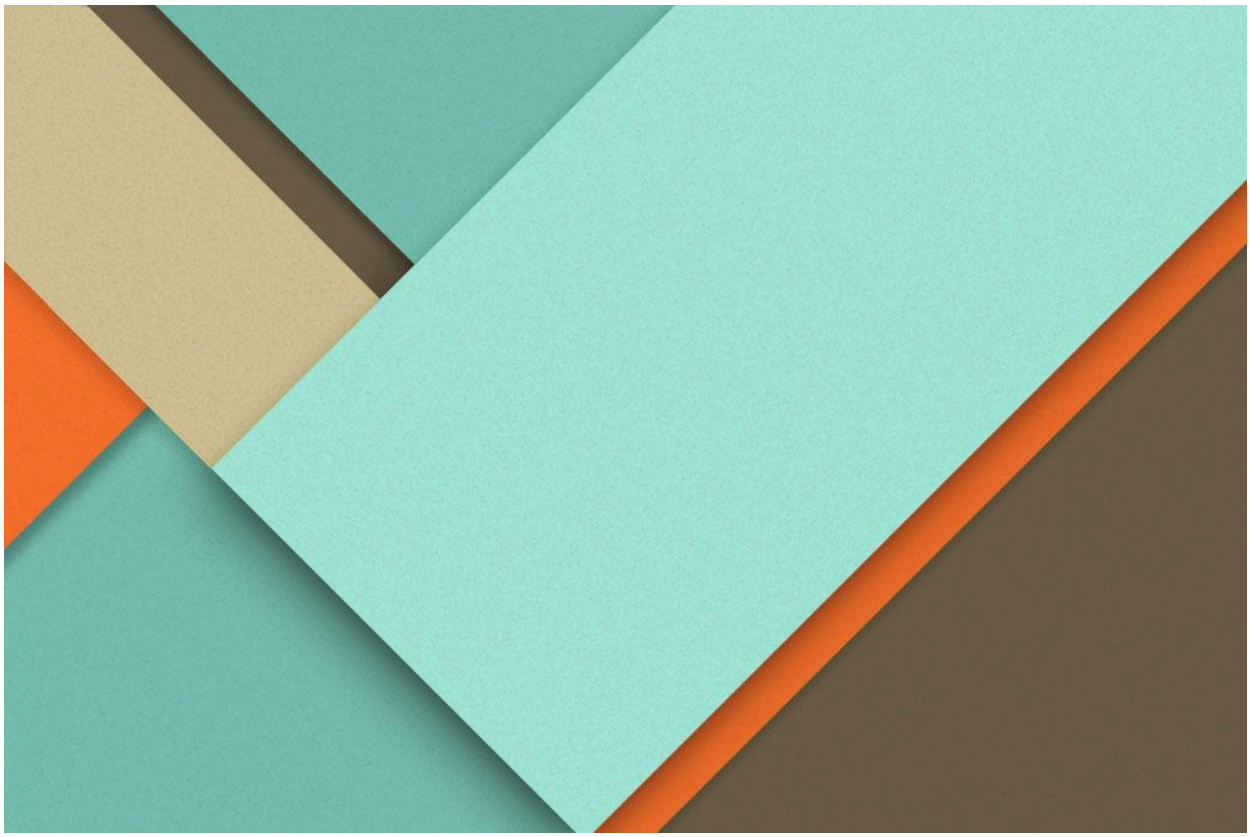


Laravel - Premiers pas

18.03.2025

Riot Bastien

Introduction

Laravel est un Framework PHP, en sources ouvertes, créé en juin 2011 et en constant développement par la communauté.

Ce framework a été conçu comme un Fork du framework PHP appelé Symfony, créé en octobre 2005.

Laravel se distingue dans le domaine du développement web avec PHP grâce à sa forte communauté, services, helpers et implémentations permettant d'ajouter de fortes fonctionnalités à PHP.

Comme la plupart des frameworks et modèles de développement web, aujourd'hui, Laravel suit un modèle en Model - Vue - Controller (MVC).

De plus, Laravel supporte trois types d'applications, permettant un développement parallèle basé sur des ressources communes, nous retrouvons donc :

- Les applications HTTP (Backend + Frontend)
- Les API (Application Programming Interface)
- Les applications CLI (Command Line Interface)

A travers cette conception souple et ouverte, nous pouvons partager des models et entités à travers nos applications et avoir un core commun à nos applications.

Objectifs

1. Installer les dépendances et services nécessaires
2. Créer une base de données et un utilisateur
3. Créer et configurer une application par défaut Laravel
4. Démarrer notre application

I - Installation des dépendances et services

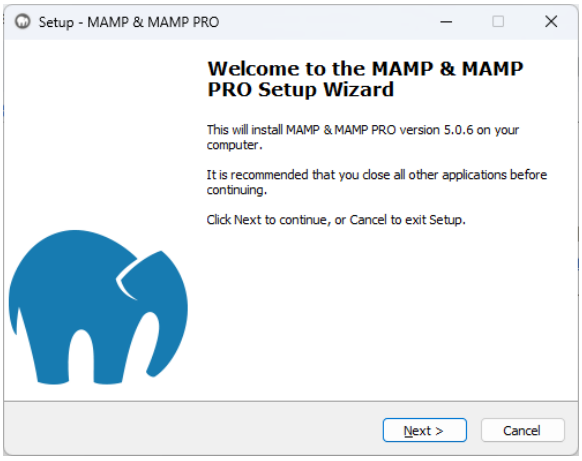
Nous allons donc commencer, sur un environnement Windows avec MAMP, par installer les dépendances et services nécessaires au bon fonctionnement de Laravel.

Tout d'abord, il nous faut avoir un environnement web avec une base de données. Pour cela, nous allons installer MAMP qui permet d'embarquer un serveur Web Apache, un serveur MySQL pour stocker une base de données et un serveur PHP pour exécuter nos scripts.

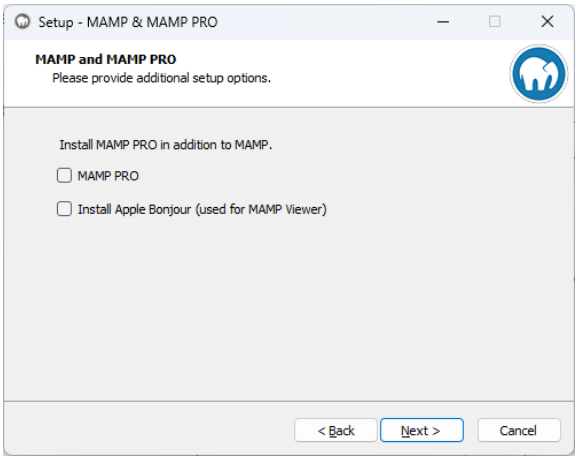
A - Installation de MAMP

Pour installer MAMP, je vous invite à vous rendre sur le site officiel de l'application, afin de récupérer l'installateur pour votre architecture : https://www.mamp.info/

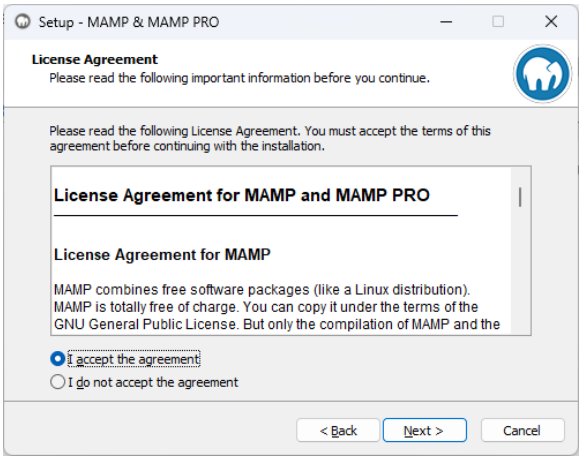
Après avoir téléchargé l'installateur, vous pouvez l'exécuter et démarrer l'installation de MAMP :



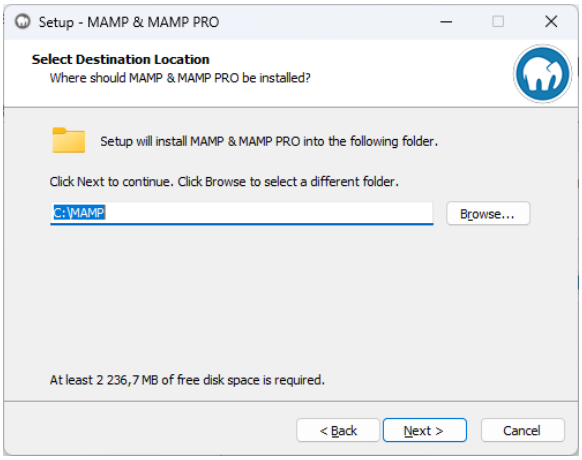
Arrivé à cette étape, vous pouvez décocher les deux cases qui nous seront inutiles pour la configuration de Laravel et de notre programme :



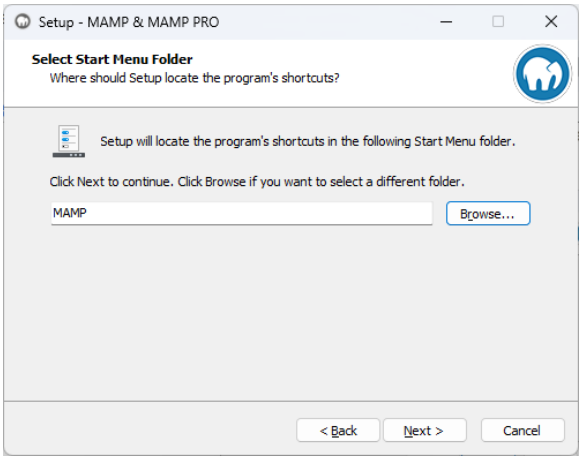
Après avoir lu les conditions d'utilisations de l'application, vous pouvez les accepter et passer à l'étape suivante :



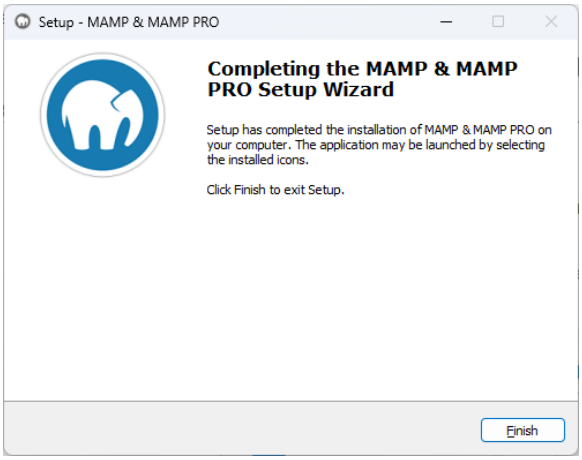
A cette étape, vous pouvez spécifier le lieu où MAMP sera installé et où votre programme sera stocké :



Vous pouvez aussi définir le dossier et le nom du fichier du raccourci, dans la barre d'application de votre Windows :



Votre MAMP est maintenant installée :



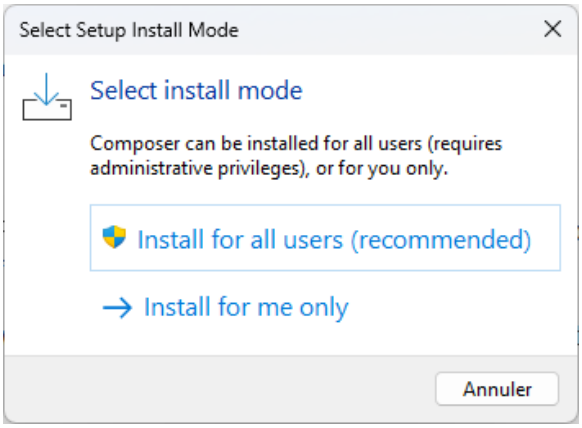
B - Installation de Composer

Composer est le service de gestion de dépendances et librairies pour PHP le plus utilisé dans la communauté des développeurs PHP. Pour installer Laravel, nous allons donc utiliser Composer sous la forme des créateurs. En effet, Composer permet d'installer des dépendances mais il peut aussi permettre d'avoir des structures d'installations complètes, comme pour Laravel.

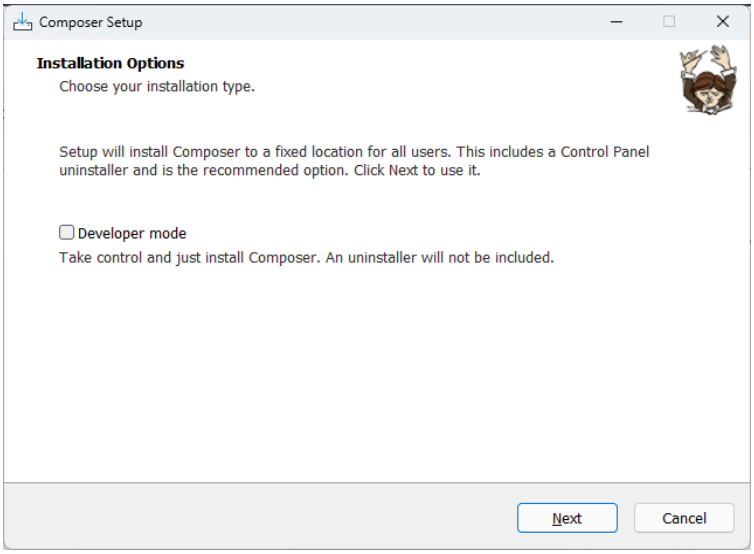
Pour installer Composer avec le serveur PHP de MAMP, nous allons d'abord récupérer l'installateur depuis le site officiel de Composer : https://getcomposer.org/download/

Attention de bien prendre le "Composer-setup.exe"

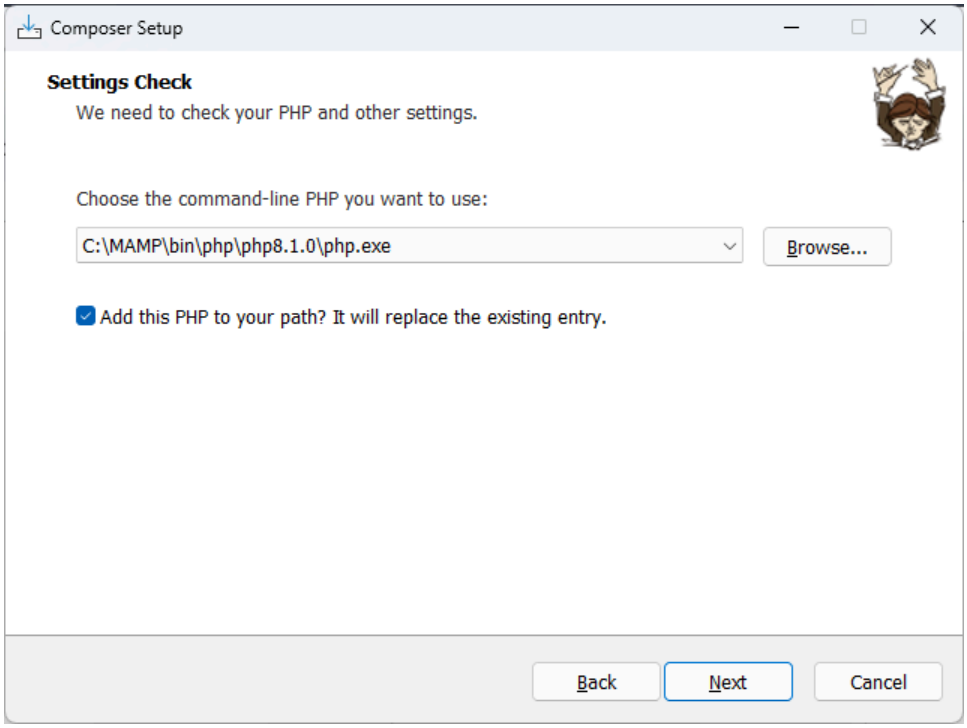
Au début de l'installation, vous pouvez choisir l'option que vous souhaitez parmi ces deux présentes :



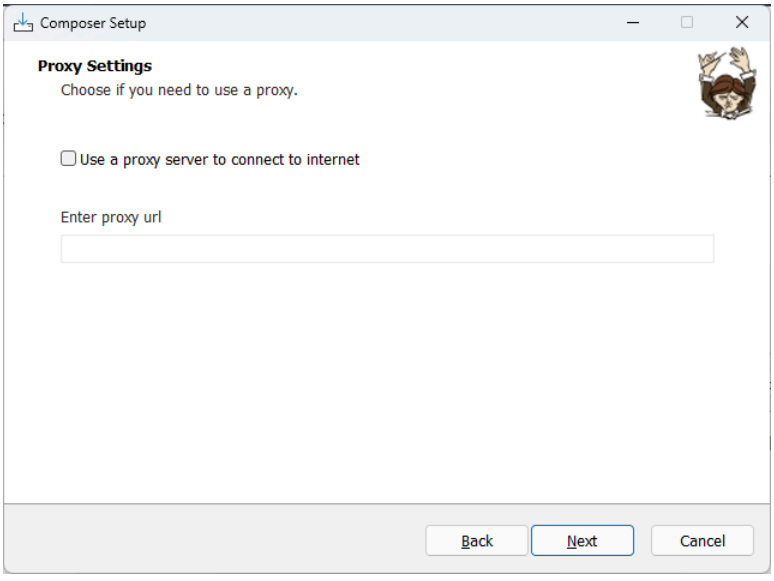
A cette étape, il n'est pas utile de cocher la case :



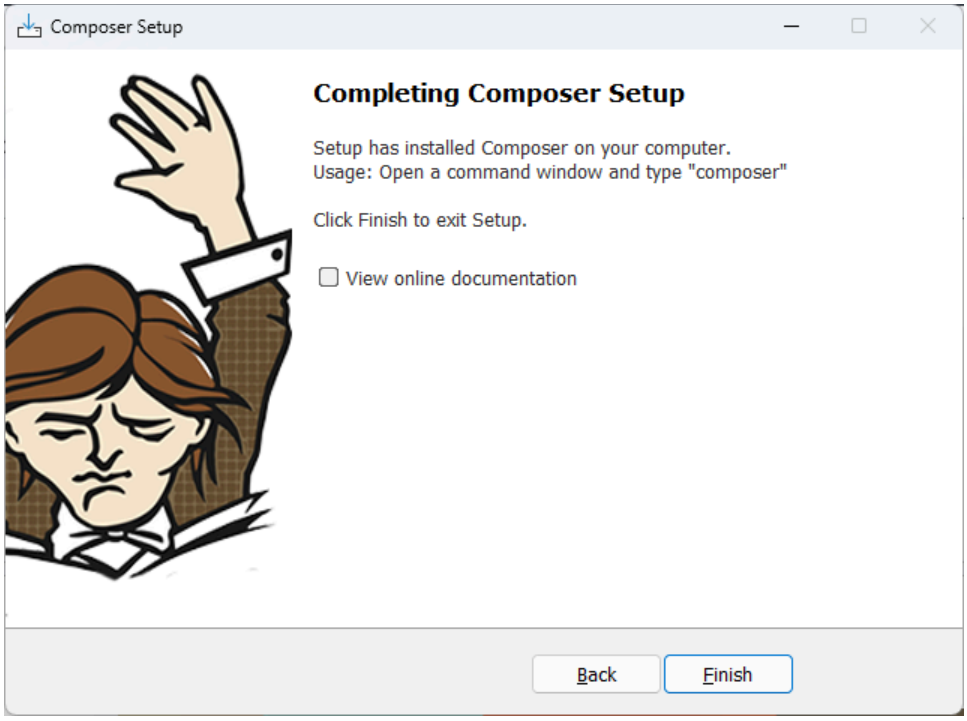
Cette étape est cruciale, attention à bien sélectionner le bon chemin vers l'exécutable de PHP **(VERSION 8.3.1)** afin que Composer puisse s'exécuter (le chemin doit ressembler à celui présenté en-dessous) :



Veillez à ne pas cocher la case à cette étape, nous ne passerons pas par un Proxy dans l'environnement de Composer :



L’installation de Composer est officiellement terminée !



C - Vérifier les installations

Afin de nous assurer que notre serveur MAMP est prêt à démarrer notre application, nous allons le tester.

Pour cela, nous allons démarrer l'application MAMP et cliquer sur le bouton "Start Servers" puis nous rendre sur http://localhost/

Nous voyons ainsi la page suivante :

MAMP est donc bien démarré et accessible en local.

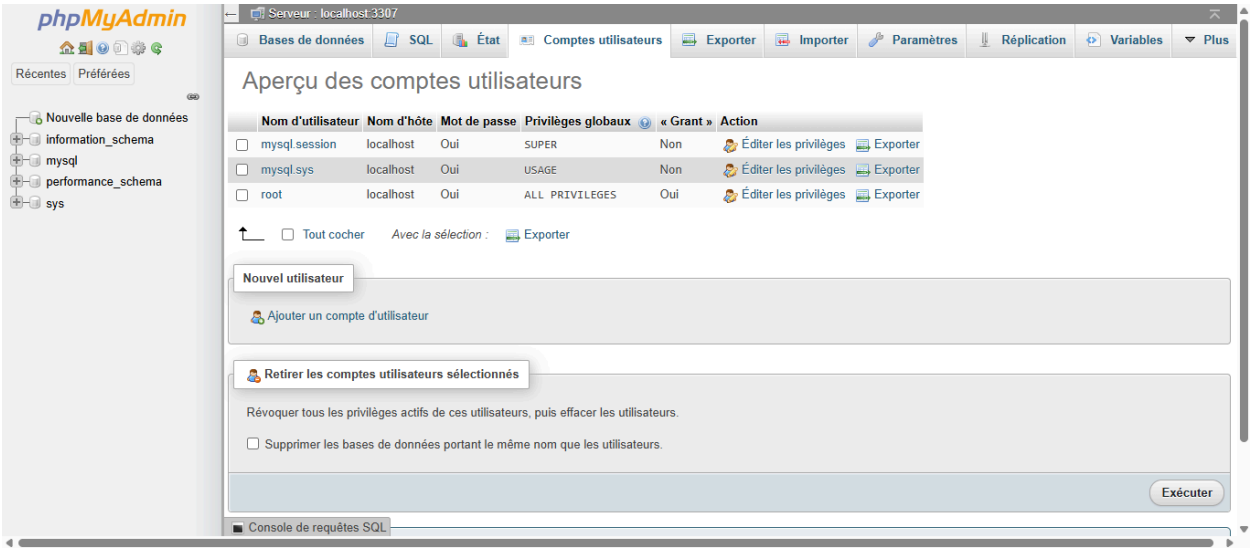
II - Création d'une base de données et d'un utilisateur

A - Connexion a PHPMyAdmin

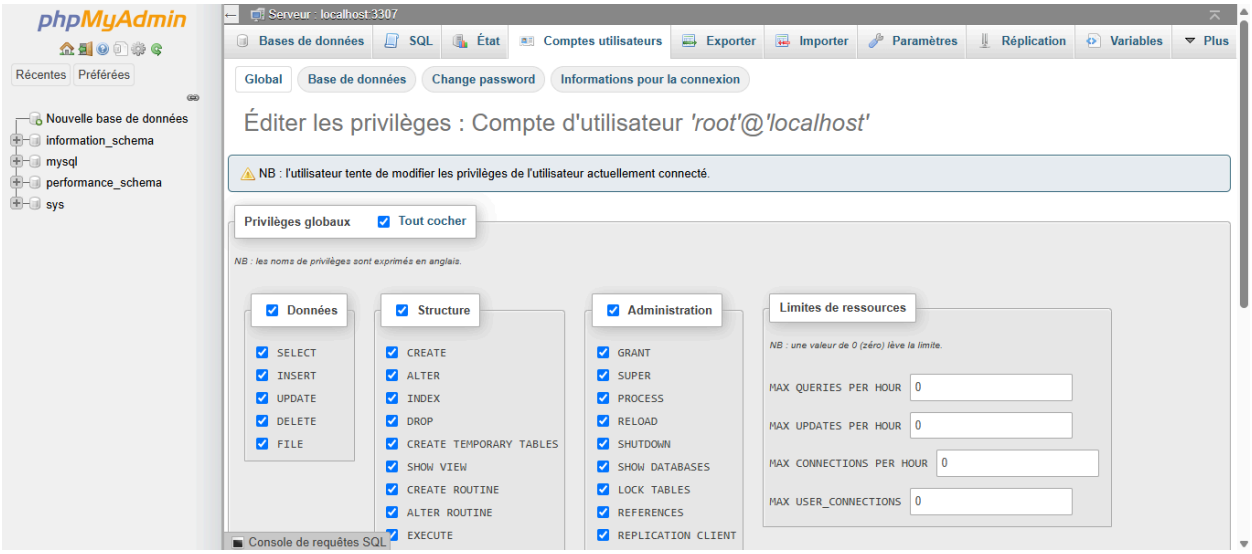
Afin de créer notre base de données et notre utilisateur, nous allons utiliser le panel PHPMyAdmin qui est fourni avec MAMP. Pour se faire, nous allons nous rendre sur http://localhost/phpMyAdmin5/

La connexion est faite automatiquement, ce qui indique que notre utilisateur root de la base de données, celui possédant tous les pouvoirs, ne possède pas de mot de passe. Avant tout, nous allons changer cela afin de renforcer la sécurité.

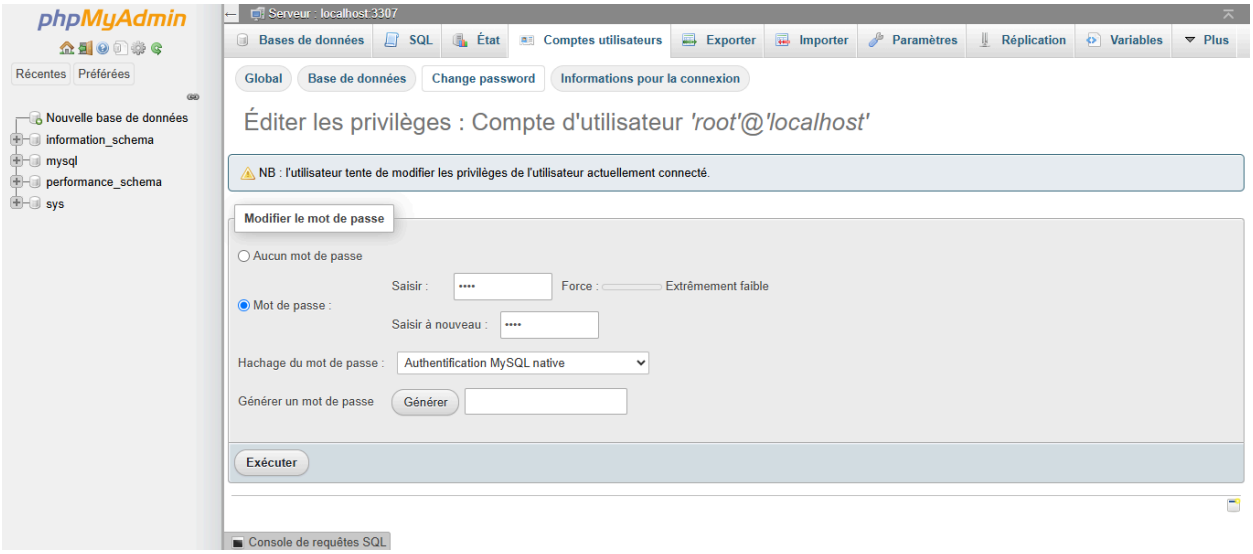
Rendons-nous dans la catégorie "Comptes utilisateurs" :



Ensuite, nous allons cliquer sur "root" :



Puis, nous allons aller dans l'onglet "Change password" et nous pouvons, ici, entrer un nouveau mot de passe et exécuter :

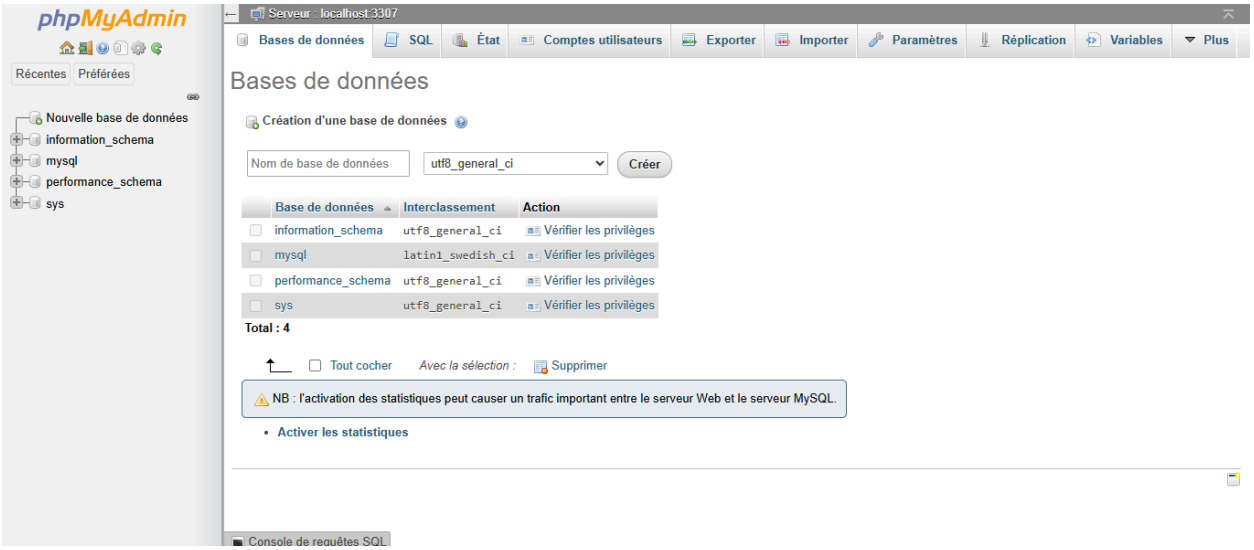


Notre root est maintenant sécurisé !

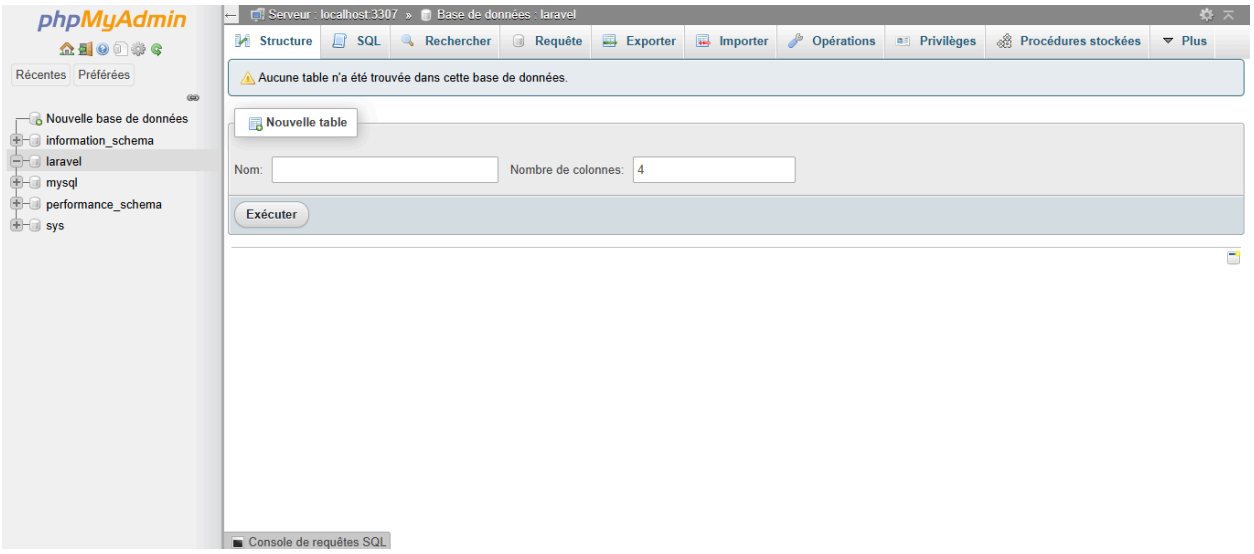
B - Création de la base de données de l'application

Nous allons maintenant procéder à la création d'une base de données pour notre application.

Dans PHPMyAdmin, nous pouvons cliquer sur "Nouvelle base de données" à gauche de la page :



Nous entrons donc le nom de notre base de données dans le champ "Nom de base de données" et ensuite nous appuyons sur "Créer" :

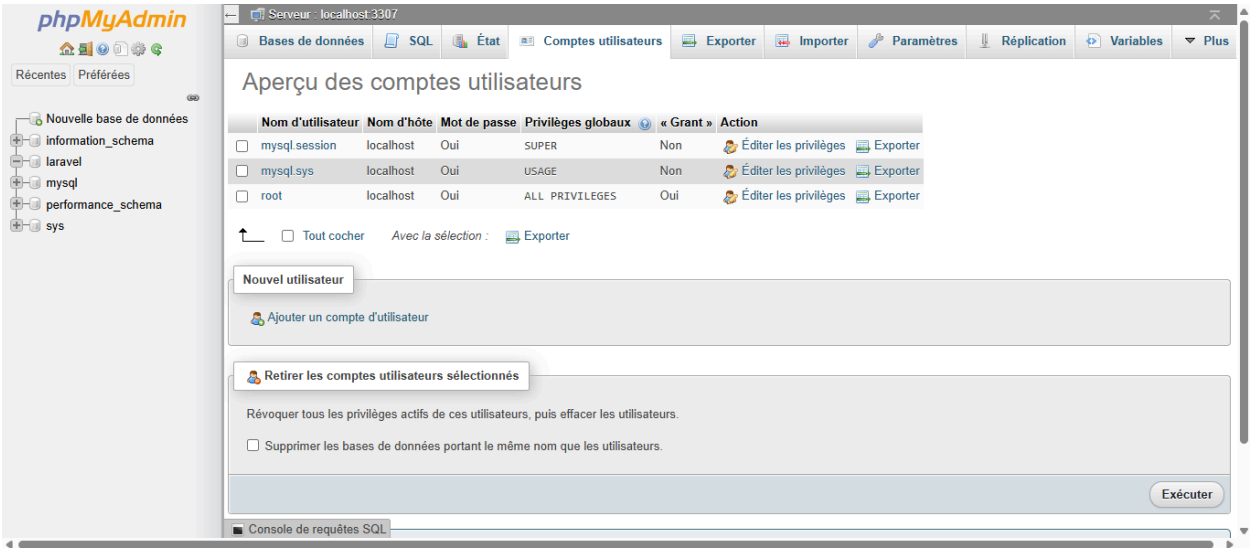


Notre base est créée, PHPMyAdmin nous invite à créer une table mais nous n'allons rien en faire car toutes les tables seront gérées par le système de migration de Laravel.

C - Création d'un utilisateur pour accéder à la base de données

Afin de ne pas utiliser notre utilisateur root, qui possède tous les pouvoirs, nous allons créer un utilisateur sur le serveur de base de données qui n'a de pouvoirs que sur la base de données que nous venons de créer.

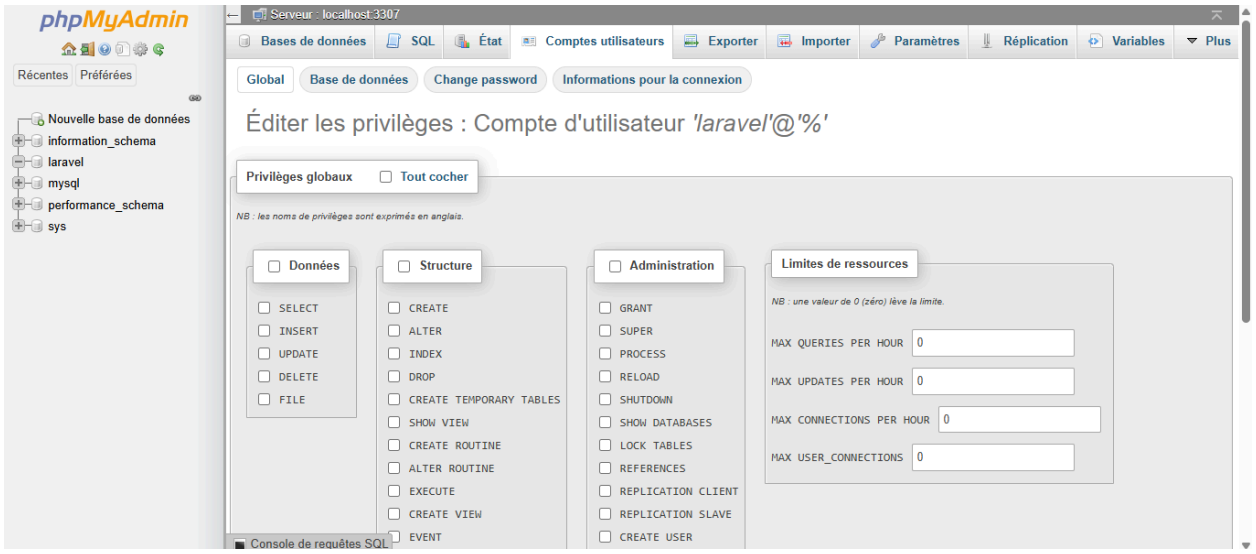
Sur PHPMyAdmin, nous allons retourner dans la catégorie "Comptes utilisateurs" :



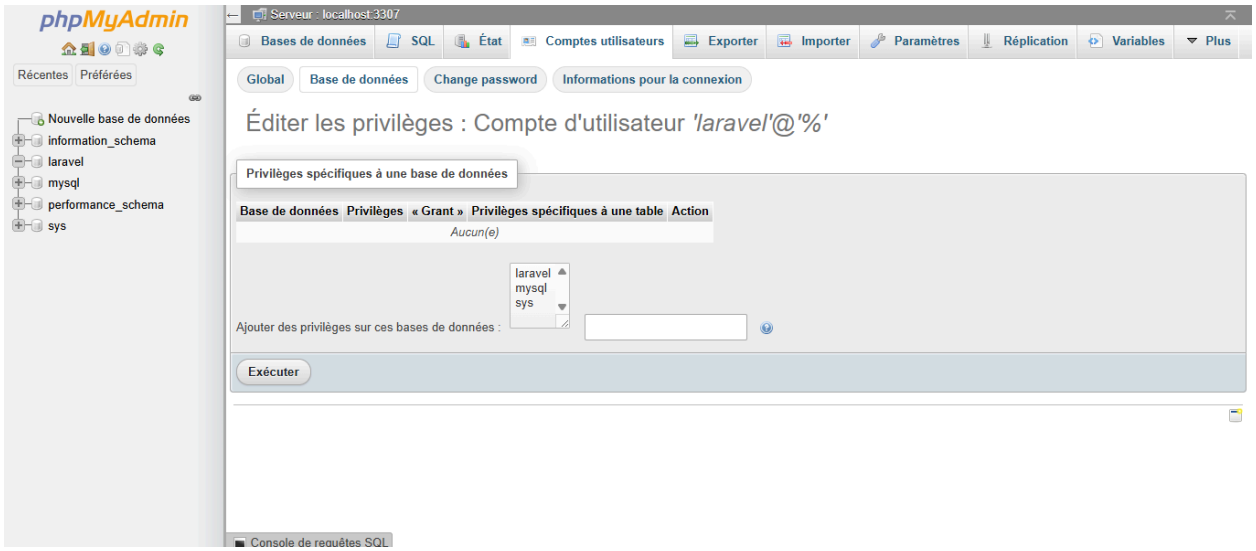
Ici, nous pouvons appuyer sur "Ajouter un compte d'utilisateur" et donc créer notre utilisateur.

Attention, ne touchez à aucune permission ni paramètres sauf le nom de l'utilisateur et son mot de passe.

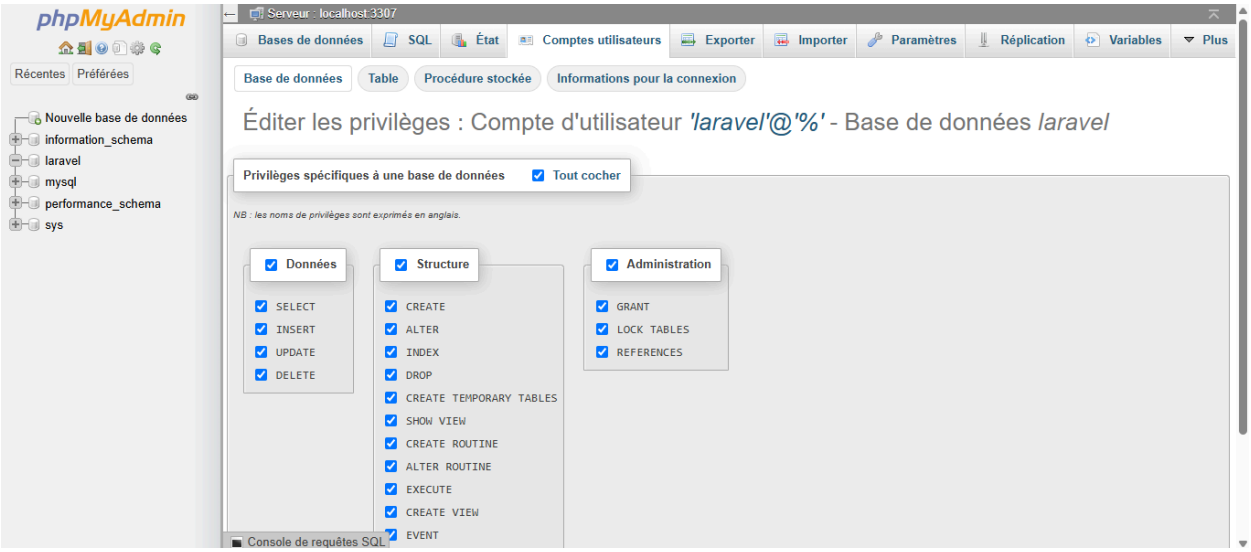
Nous allons donc maintenant accorder toutes les permissions pour notre utilisateur sur la base de données que nous avons créé. Pour cela, nous retournons dans les "Comptes utilisateurs" puis sur notre nouvel utilisateur :



Nous nous rendons dans l'onglet "Base de données" présent en haut :



Nous sélectionnons notre base de données et ensuite nous appuyons sur "Exécuter" :



Ici, nous pouvons cliquer sur “Tout cocher” afin d’accorder toutes les permissions à notre utilisateur. Nous pouvons finalement appuyer sur “Exécuter” tout en bas de la page.

Notre base de données et notre utilisateur sont maintenant prêts !

III - Création et configuration d'une application par défaut Laravel

A - Génération d'une application Laravel par défaut avec Composer

Tout d'abord, nous allons ajouter la librairie d'installation de laravel dans notre Composer global afin de pouvoir, par la suite, lancer un script de génération de l'application.
Pour cela, nous pouvons lancer la commande suivante dans un terminal :

composer global require laravel/installer

Pendant l'installation vous risquez de tomber sur cette erreur :

```
The openssl extension is required for SSL/TLS protection but is not available. If you can not enable the openssl ex
tension, you can disable this error, at your own risk, by setting the 'disable-tls' option to true.
```

Si c'est le cas, rendez-vous dans le dossier CHEMIN_MAMP\bin\php\php8.3.1 puis dans le fichier php.ini-development et remplacez :

;extension=openssl

avec

extension=openssl

Et aussi :

;extension_dir = "ext"

avec

extension_dir = "ext"

Enfin, vous pouvez faire une copie du fichier en php.ini dans le même dossier et relancer la commande.

Nous pouvons maintenant nous déplacer dans le dossier **htdocs** de notre serveur web avec un **cd** et exécutons la commande :

laravel new nom_projet

Lors de l'installation, vous risquez de rencontrer l'erreur suivante :

```
The following PHP extensions are required but are not installed: mbstring
```

Si c'est le cas, vous pouvez vous rendre dans le fichier php.ini que nous avons précédemment créé et remplacer :

;extension=mbstring

avec

extension=mbstring

Et :

;extension=fileinfo

avec

extension=fileinfo

Pendant l'installation vous allez avoir plusieurs questions afin de configurer au mieux votre base de projet Laravel.

Tout d'abord, l'installateur vous propose de choisir un **starter kit** ou **kit de démarrage**, vous pouvez appuyer sur Entrée sans rien inscrire, nous allons utiliser l'application sans kit de démarrage :

L'installation va tourner et vous allez rencontrer une question sur le mécanisme de base de données à utiliser. Par défaut, l'installateur vous propose **SQLite** mais nous allons choisir **mysql** à la place. Vous pouvez donc inscrire **mysql** et appuyer sur Entrée :

Ensuite, vous allez avoir une question au sujet des migrations par défaut. Vous pouvez inscrire **no** afin de ne pas lancer les migrations par défaut, nous les lancerons a posteriori de l'installation, quand vos identifiants de base de données seront renseignés :

Ensuite, l'installateur nous propose de démarrer les commandes **npm install** et **npm run build** qui vont permettre d'installer un ensemble de composants utiles au **frontend**. Vous pouvez appuyer sur Entrée sans rien inscrire, ce qui lancera l'installation :

L'installation de notre premier projet est terminée !

B - Configuration de l'application

Maintenant que notre application est créée et que les dépendances sont toutes installées et disponibles, nous allons pouvoir entrer les identifiants de base de données et configurer notre application.

Tout d'abord, nous allons accéder au fichier .env de notre application afin d'y placer les identifiants de base de données qui seront utilisés par l'application.
Information : Vous remarquerez un champ APP_KEY en haut du fichier .env, cette clé est générée automatiquement par laravel et permet de chiffrer les données. Veillez à ne pas perdre cette clé ni la changer, sous peine de perdre vos données.

Dans le fichier, vous trouverez 6 variables d'environnement en lien à la connexion avec la base de données :

- **DB_CONNECTION** → Moteur de base de données utilisé, nous allons laisser "mysql"
- **DB_HOST** → Hôte de votre base de données
- **DB_PORT** → Port d'accès à la base de données, vous pouvez laisser 3306
- **DB_DATABASE** → Nom de la base de données qui sera utilisée par l'application
- **DB_USERNAME** → Nom de l'utilisateur ayant les droits sur la base de données
- **DB_PASSWORD** → Mot de passe de l'utilisateur ayant les droits sur la base de données

Nous allons aussi paramétrer certaines variables afin de mieux modeler notre application :

- **APP_NAME** → Nom de notre application
- **APP_ENV** → Environnement utilisé, sur votre ordinateur vous allez laisser "local" mais lorsque l'application sera accessible en ligne il faudra le passer en "production".

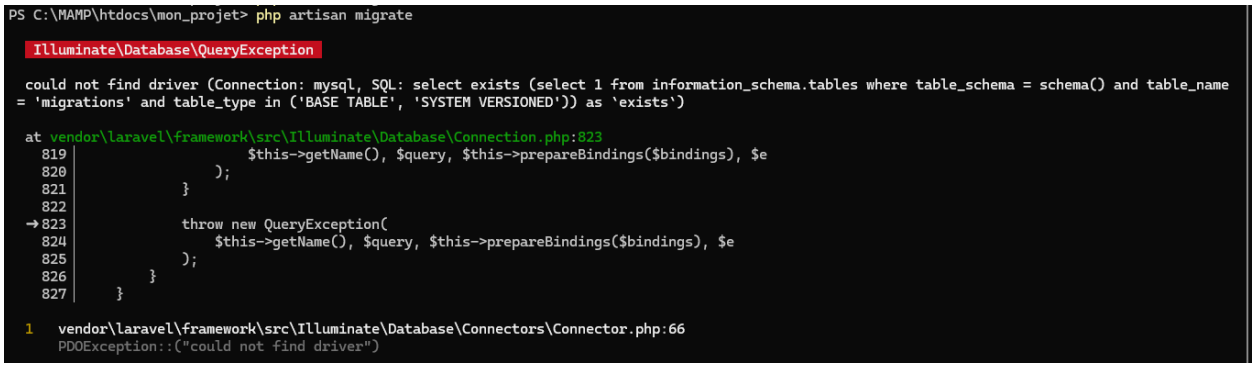
Ce paramètre permettra d'éviter les messages de debug sur votre environnement déployé

- **APP_URL** → URL de votre application, de la même façon, en production, il faudra le changer afin de ne pas avoir de soucis sur les redirections
- **APP_LOCALE** -> Langue de votre application, nous allons le passer en **fr**
- **APP_FALLBACK_LOCALE** -> Même chose qu'au-dessus
- **APP_FAKER_LOCALE** -> Même chose qu'au-dessus mais cette fois ce sera **fr_FR** qu'il faudra inscrire

Nous avons fait le tour des variables d'environnement de notre projet. Nous pouvons maintenant lancer les migrations par défaut afin d'avoir les tables de données utiles au bon fonctionnement de Laravel. Vous pouvez donc faire :

php artisan migrate

Pendant l'exécution, vous risquez de rencontrer cette erreur :



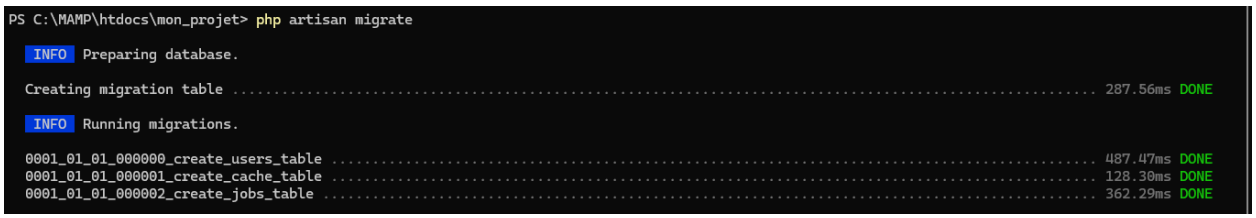
Si c'est le cas, je vous invite à retourner dans le fichier php.ini et effectuer la même manipulation qu'au-dessus mais en changeant :

;extension=pdo_mysql

avec

extension=pdo_mysql

Vous pouvez ainsi relancer la précédente commande :



Notre application est maintenant configurée et les tables par défaut sont installées.

C - Lancement de notre application

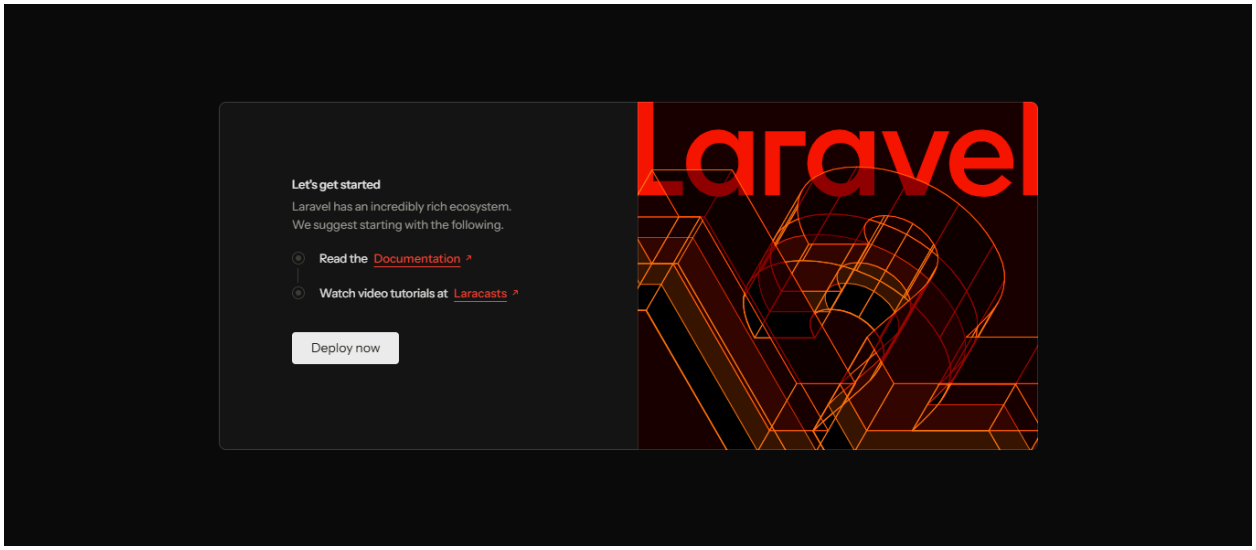
Nous arrivons à la dernière étape !
Afin de vérifier qu'aucun souci de configuration ne réside dans votre installation, nous allons démarrer l'application et l'ouvrir depuis le navigateur.

Pour démarrer l'application, vous pouvez exécuter la commande :

php artisan serve

Votre console affiche donc ceci :

Nous pouvons nous rendre sur l'URL indiquée, http://127.0.0.1:8000 et voir la page par défaut des projets Laravel :



De plus, vous pouvez voir que toutes les requêtes HTTPs transmises au serveur web ouvert par Laravel apparaissent dans la console :

Si vous ne souhaitez pas utiliser le serveur web de Laravel, vous pouvez accéder à votre site depuis l'URL suivante :
http://localhost/dossier_de_votre_projet/public

Or, il faut savoir que certaines redirections risquent de ne pas aboutir car Laravel utilise son propre hôte, en interne, par mesure de sécurité.