

DECOUVERTE DU FRAMEWORK LARAVEL

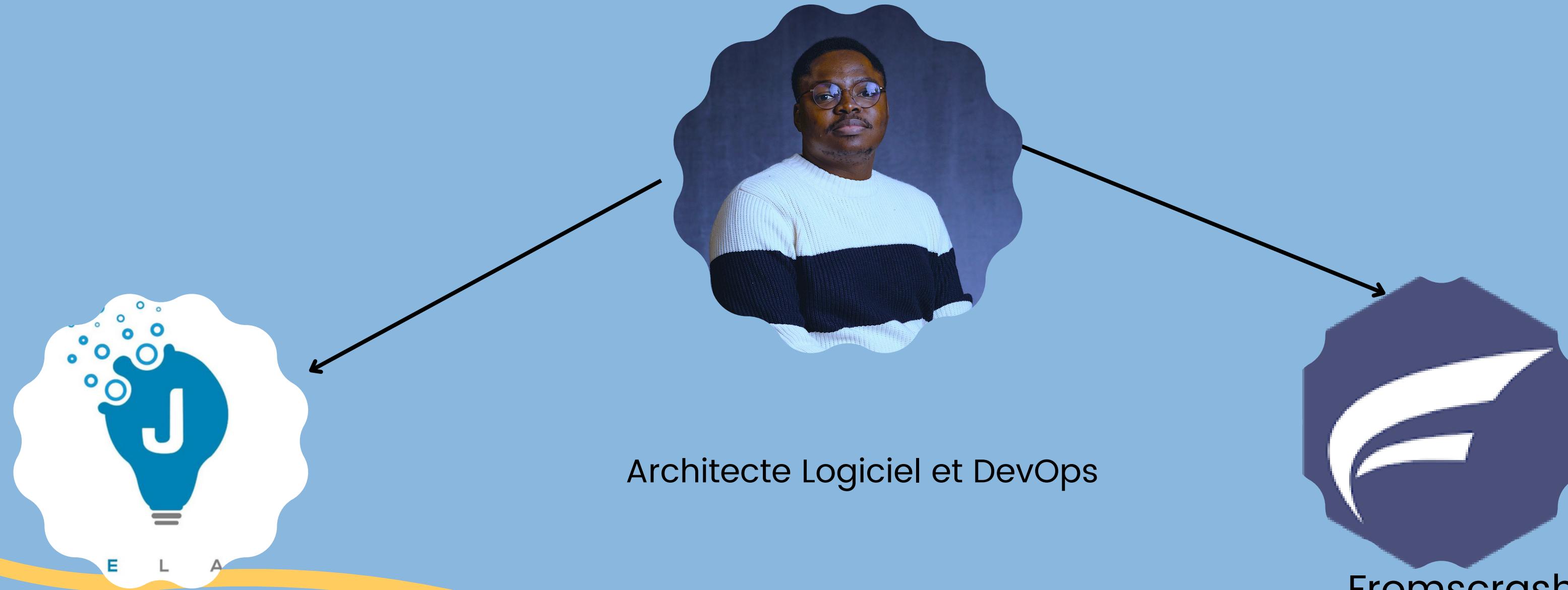


Moufid ALAOFE

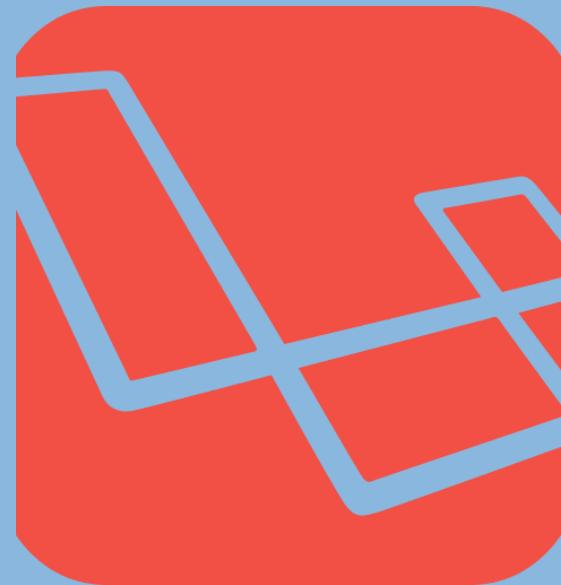
Architect Logiciel et DevOps , CEO
fromscrash



PRÉSENTATION



Formation sur les divers technologies



PRÉ-REQUIS

Les éléments à maîtriser

De bonnes
connaissances en HTML,
en CSS et/ou en SCSS et
PHP

STRUCTURE DE L'ATELIER



- Introduction: Histoire/Avantages/Popularité de Laravel
- Fonctionnalité: Routage/Migration/Gestion de données/authentification/Validation/Modèles
- Création d'un projet Laravel
- Structure : Architecture MVC
- Moteur de template : Présentation de Blade
- Base de donnée : Comment ça marche ?
- Conclusion

LARAVEL , LE FRAMEWORK WEB OPEN SOURCE



Laravel est un framework web open-source écrit en PHP. Il a été créé par Taylor Otwell en 2011 et a connu une croissance rapide en popularité dans la communauté des développeurs PHP

- L'histoire de Laravel commence en 2011 lorsque Taylor Otwell, alors développeur web freelance, cherchait un framework PHP pour créer des applications web. Cependant, il n'était pas satisfait des options disponibles à l'époque, car elles étaient soit trop compliquées, soit manquaient de fonctionnalités clés.
- Il a donc décidé de créer son propre framework web, qui deviendra plus tard Laravel. Otwell a travaillé sur le framework pendant plusieurs mois, en s'appuyant sur les principes de développement web modernes et les meilleures pratiques pour créer une plateforme flexible, facile à utiliser et extensible

LARAVEL , LE FRAMEWORK WEB OPEN SOURCE



La première version de Laravel, la version 1, est sortie en juin 2011. Cependant, elle n'a pas connu un grand succès en raison de certains problèmes de performance et de fiabilité. Otwell a donc travaillé dur pour améliorer le framework et a sorti la version 2 en 2012, qui a connu un grand succès.

- Depuis lors, Laravel est devenu l'un des frameworks PHP les plus populaires au monde, avec une large communauté de développeurs et une abondance de ressources disponibles pour l'apprentissage et le développement
- Laravel est connu pour sa facilité d'utilisation, sa prise en charge de nombreuses fonctionnalités clés telles que la gestion de base de données, l'authentification, la validation et la sécurité, ainsi que pour son architecture MVC (Modèle-Vue-Contrôleur) facile à comprendre et à utiliser

LES AVANTAGES DE LARAVEL

- **Architecture MVC**

Laravel suit l'architecture MVC (Modèle-Vue-Contrôleur) qui facilite la séparation des préoccupations et la gestion des différentes couches d'une application

- **Syntaxe élégante**

Laravel est connu pour sa syntaxe élégante et expressive, ce qui facilite la compréhension du code pour les développeurs.

- **Système de routage flexible**

Laravel offre un système de routage flexible qui facilite la définition de routes et de leurs actions correspondantes.

- **Facilité de développement**

Laravel est livré avec de nombreuses fonctionnalités prêtes à l'emploi, telles que l'authentification, la validation de formulaire, la gestion des sessions, la gestion des erreurs, etc. Cela permet aux développeurs de se concentrer sur la logique métier plutôt que sur les aspects techniques.



LES AVANTAGES DE LARAVEL

- Base de données

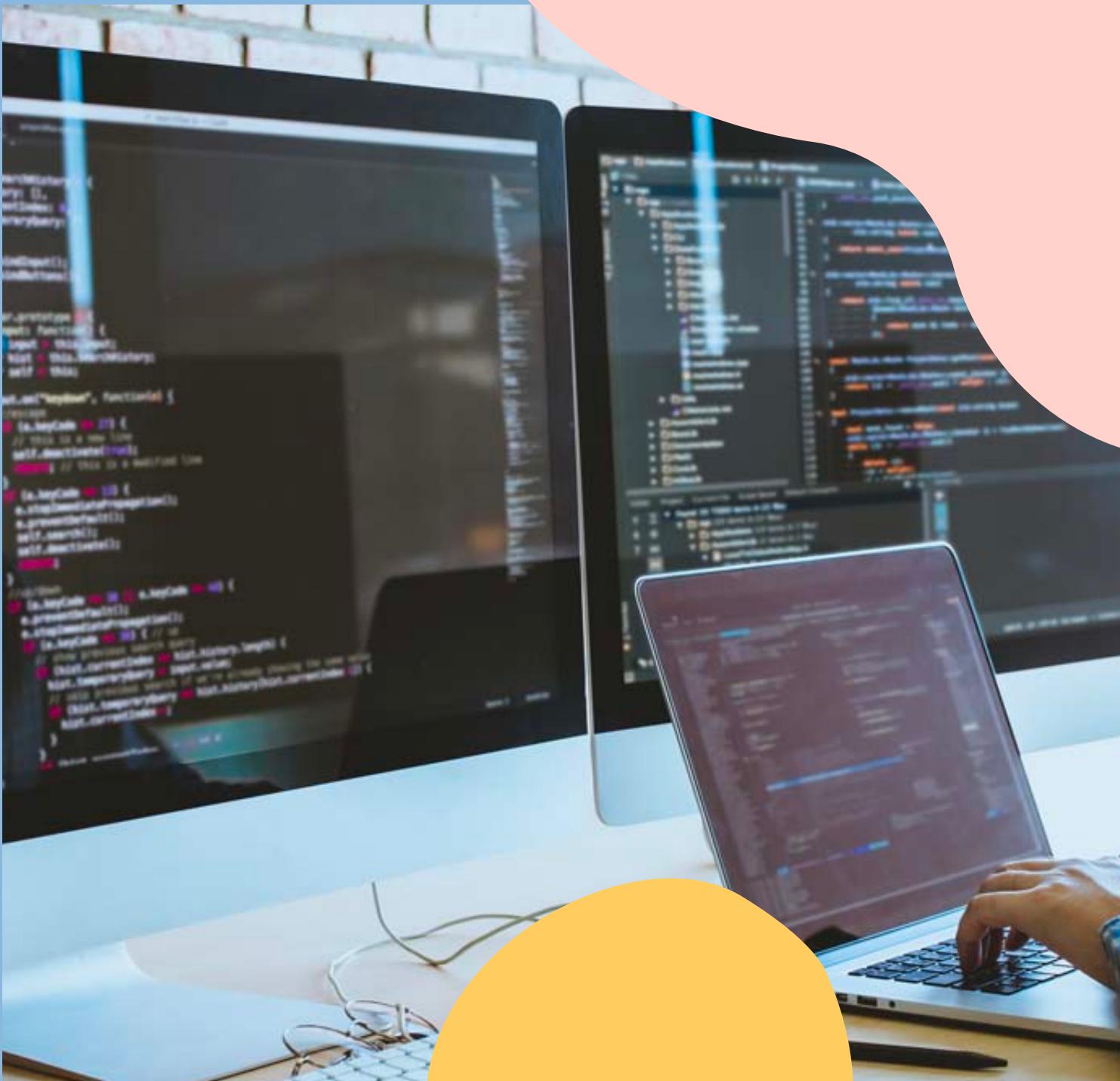
Laravel propose une prise en charge de plusieurs bases de données, ainsi que des fonctionnalités telles que la migration de schéma et les ORM (Object-Relational Mapping), qui simplifient la gestion des données.

- Sécurité

Laravel intègre des fonctionnalités de sécurité avancées telles que la protection CSRF (Cross-Site Request Forgery), la protection XSS (Cross-Site Scripting), la gestion des accès et des autorisations

- Communauté active

Laravel dispose d'une communauté active et d'une documentation complète, ce qui facilite l'apprentissage et la résolution des problèmes.



CRÉATION D'UN PROJET LARAVEL

INSTALLATION

- PHP

Laravel nécessite PHP 7.4 ou version ultérieure pour fonctionner. Vous devez donc installer PHP sur votre ordinateur si ce n'est pas déjà fait (<https://www.php.net/manual/fr/install.php>)

- Composer

Laravel utilise Composer comme gestionnaire de dépendances. Vous devez donc installer Composer sur votre ordinateur si ce n'est pas déjà fait (<https://getcomposer.org/>)

- Serveur Web

Laravel nécessite un serveur web pour fonctionner. Vous pouvez utiliser Apache ou Nginx pour servir votre application Laravel.

(<https://ubuntu.com/tutorials/install-and-configure-nginx#2-installing-nginx>)

- Base de données :

Laravel prend en charge plusieurs bases de données, notamment MySQL, PostgreSQL, SQLite et SQL Server. Vous devez donc installer la base de données que vous souhaitez utiliser sur votre ordinateur.

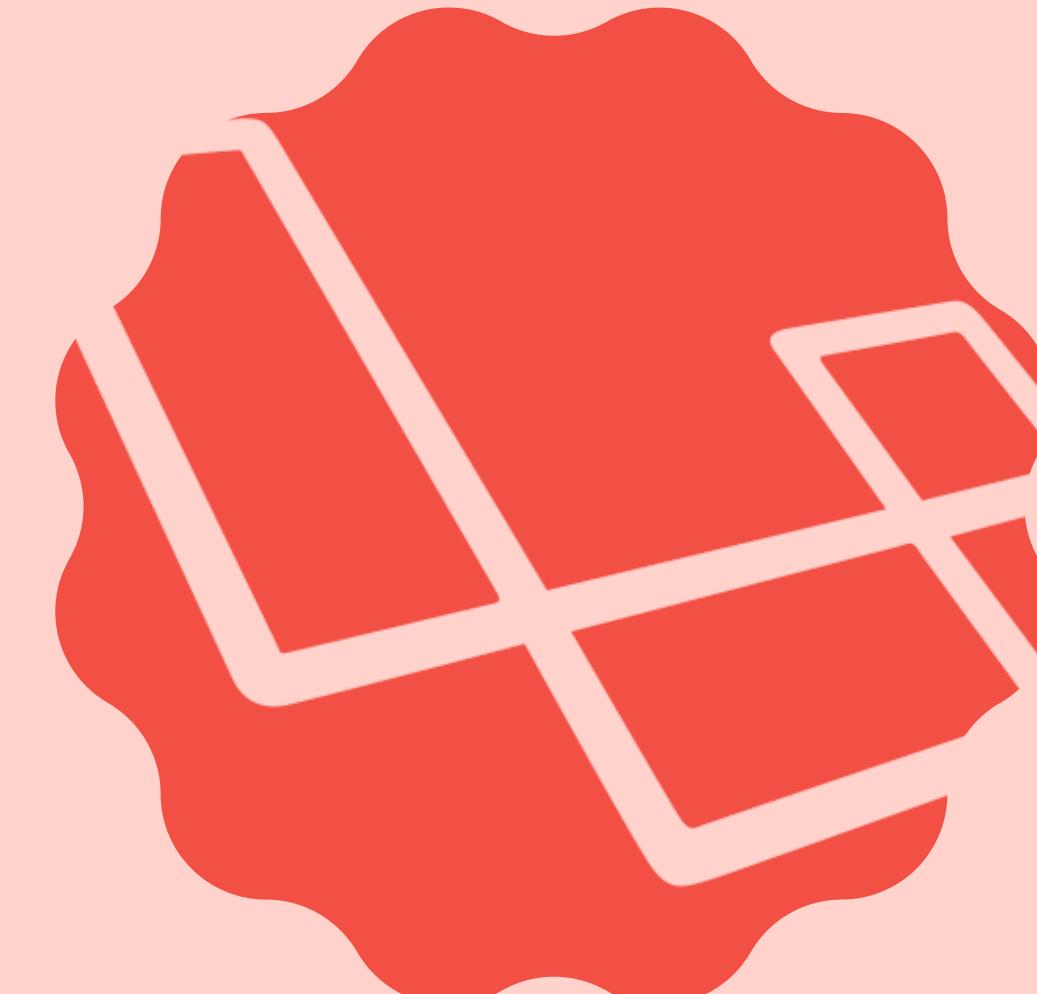


CRÉATION D'UN PROJET LARAVEL

La commande pour créer un nouveau projet Laravel à l'aide de Composer est la suivante :

```
composer create-project --prefer-dist laravel/laravel nom-du-projet
```

Cette commande crée un nouveau projet Laravel dans un dossier nommé "nom-du-projet". Elle utilise l'option "**--prefer-dist**" pour télécharger les fichiers du framework à partir d'une archive plutôt qu'à partir du code source complet, ce qui peut accélérer le processus de création du projet.



```
admin@SSSIT MINGW64 /c/xampp/htdocs
$ composer create-project laravel/laravel firstproject
Installing laravel/laravel (v5.4.30)
- Installing laravel/laravel (v5.4.30): Downloading (connect
Downloaded (100%)
Created project in firstproject
> php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
Updating dependencies (including require-dev)
```

STRUCTURE D'UN PROJET LARAVEL

- Le dossier **app**:

Ce dossier contient le code de l'application, y compris les modèles, les contrôleurs, les vues et les classes utilitaires.

- Le dossier **bootstrap**

Ce dossier contient les fichiers nécessaires pour démarrer l'application, y compris les fichiers d'amorçage, les fichiers de configuration et les fichiers de chargement automatique.

- Le dossier **config**

Ce dossier contient les fichiers de configuration de l'application, y compris les fichiers de configuration de la base de données, les fichiers de configuration de l'authentification et les fichiers de configuration de l'environnement

- Le dossier **database**

Ce dossier contient les fichiers de migration de la base de données, les fichiers de configuration de la base de données et les fichiers de semences de données.

- Le dossier **public**

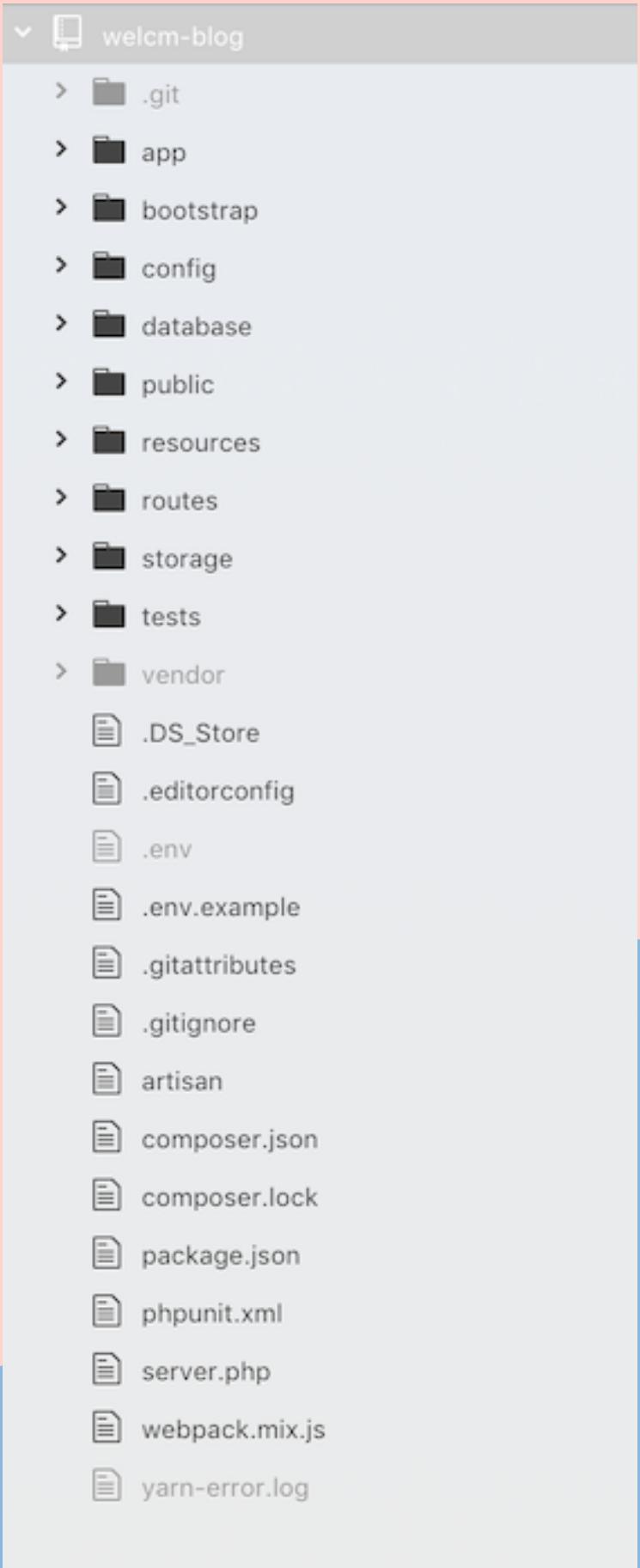
Ce dossier contient les fichiers accessibles au public, y compris le fichier index.php qui est le point d'entrée de l'application.

- Le dossier **resources**

Ce dossier contient les fichiers de ressources tels que les fichiers de vue, les fichiers de langues, les fichiers de style et les fichiers de script.

- Le dossier **routes**

Ce dossier contient les fichiers de routage qui définissent les points de terminaison de l'application et leurs actions correspondantes.



STRUCTURE D'UN PROJET LARAVEL

- Le dossier **storage** :

Ce dossier contient les fichiers générés par l'application, tels que les fichiers de cache, les fichiers journaux, les fichiers de session et les fichiers téléchargés

- Le dossier **tests** :

Ce dossier contient les fichiers de test de l'application.

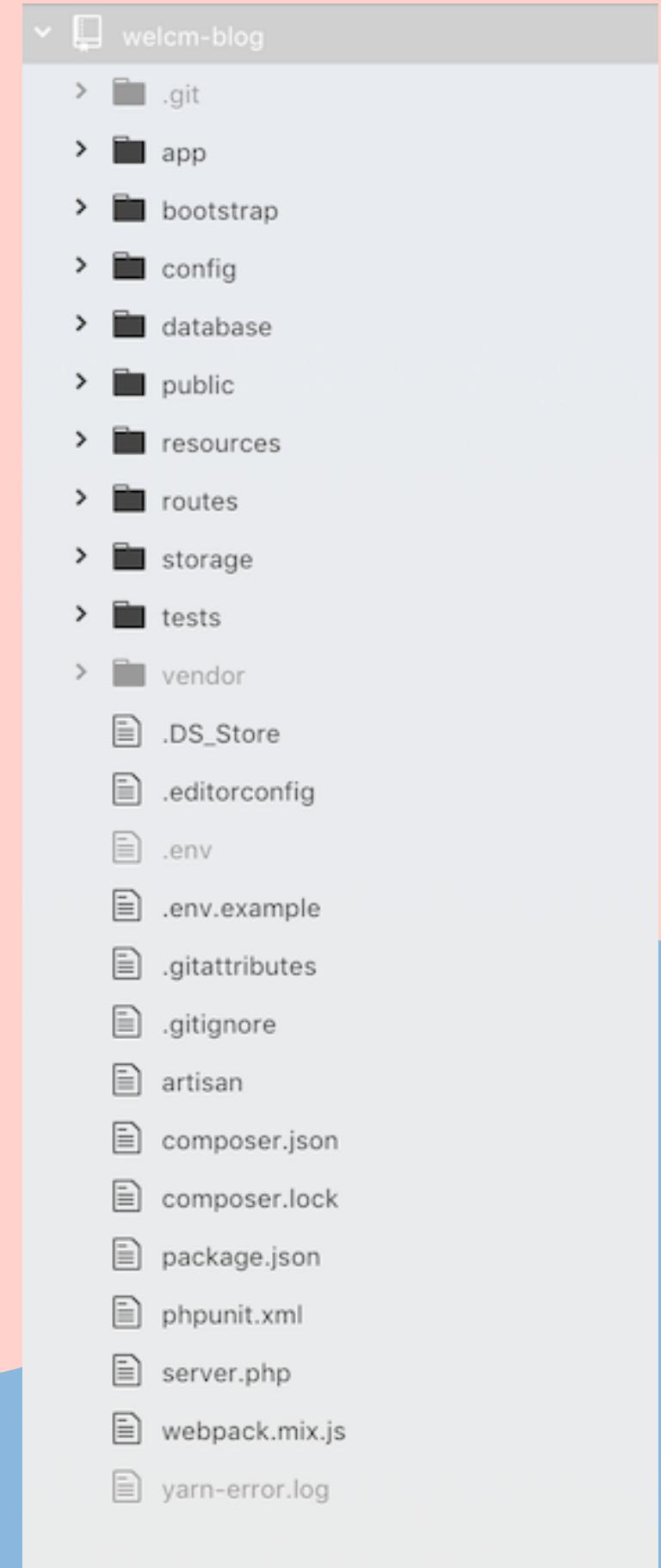
Le dossier **vendor** : Ce dossier contient les dépendances de l'application installées via Composer.

- Le fichier **.env** :

Ce fichier contient les variables d'environnement de l'application, telles que les informations de connexion à la base de données et les clés secrètes.

- Le fichier **artisan** :

Ce fichier est l'outil de ligne de commande de Laravel qui permet d'exécuter des commandes telles que la génération de code, la migration de base de données et le démarrage du serveur de développement.



LE MOTEUR DE TEMPLATE BLADE

Le moteur de template de Laravel est une fonctionnalité qui permet de générer des vues HTML dynamiques à partir de modèles réutilisables. Les moteurs de templates sont des outils qui facilitent la création d'interfaces utilisateur pour les applications Web.

Dans Laravel, le moteur de template par défaut est **Blade**.

Blade offre un syntaxe simple et expressive qui facilite la création de vues réutilisables pour votre application. Il utilise des fichiers de template (fichiers .blade.php) pour définir la structure de vos pages HTML, et permet également d'intégrer du code PHP pour générer du contenu dynamique.

```
@if ($users->count() > 0)
    @foreach ($users as $user)
        <li>{{ $user->name }}</li>
    @endforeach
@else
    <p>No users</p>
@endif

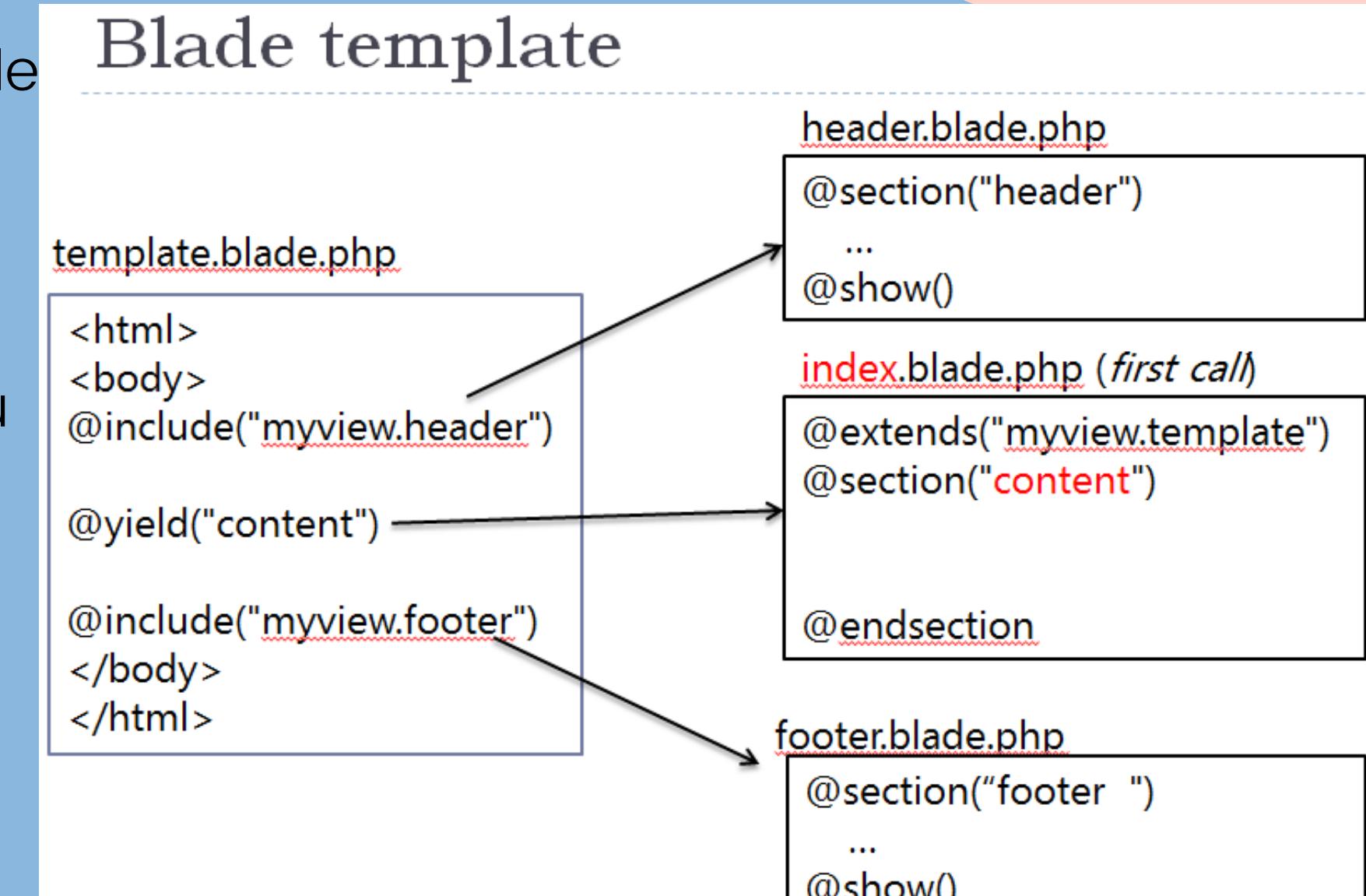
// Translates to

@foreach ($users as $user)
    <li>{{ $user->name }}</li>
@endforeach
@empty
    <p>No users</p>
@endforeach
```

LE MOTEUR DE TEMPLATE BLADE

Voici quelques-unes des fonctionnalités clés de Blade :

- **Héritage de template:** permet de définir un layout de base pour votre application, qui peut ensuite être étendu par d'autres vues spécifiques.
- **Sections:** permet de définir des sections de contenu dans vos vues, qui peuvent ensuite être remplies avec du contenu dynamique à partir du contrôleur de votre application.
- **Directives:** permet de définir des directives personnalisées pour effectuer des actions spécifiques dans vos vues, telles que l'inclusion de fichiers partiels, la définition de variables ou l'affichage de messages d'erreur.



LE MOTEUR DE TEMPLATE BLADE

header.blade.php

```
<ul class="nav">
    <li>Home</li>
    <li>Contact us</li>
    <li>About</li>
</ul>
```

sidebar.blade.php

```
<ul class="sidebar">
    <li>Links</li>
    <li>Archive</li>
    <li>Search</li>
</ul>
```

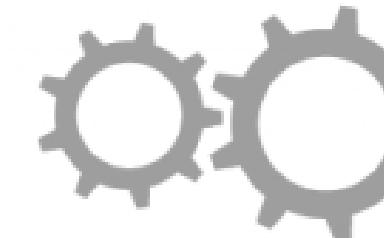
content.blade.php

```
<div class="container">
    <div class="row">
        <div class="col-md-12">
            @foreach($posts as $post)
                <h1>{{ $post->title }}</h1>
                <p>{{ $post->content }}</p>
            @endforeach
        </div>
    </div>
</div>
```

footer.blade.php

```
<div id="footer">
    Copyright. {{ date('Y') }}</div>
```

Blade



```
<ul class="nav">
    <li>Home</li>
    <li>Contact us</li>
    <li>About</li>
</ul>
```

```
<div class="container">
    <div class="row">
        <div class="col-md-12">
            <h1>Laravel and templates</h1>
        </div>
    </div>
```

Laravel allows the use of templates to separate application's views into different parts

```
</p>
```

```
<h1>Blade layouts</h1>
```

Layouts make it possible to use a single layout for all application's views

```
</p>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
<ul class="sidebar">
    <li>Links</li>
    <li>Archive</li>
    <li>Search</li>
</ul>
```

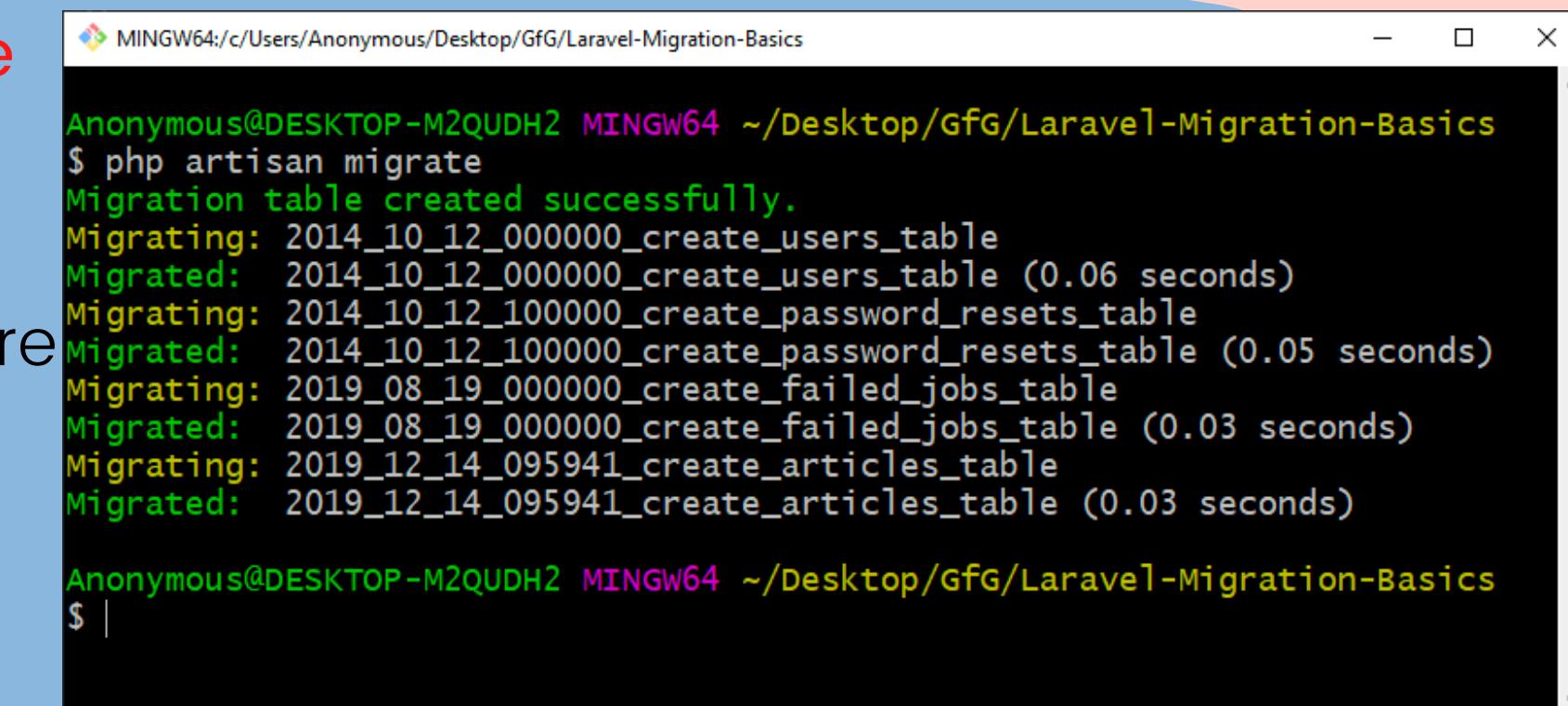
```
<div id="footer">
    Copyright. {{ date('Y') }}</div>
```

LES BASES DE DONNÉES

Laravel utilise la notion de migration qui est un outil qui permet de gérer la structure de la base de données de votre application. Elle vous permet de versionner votre schéma de base de données et de le mettre à jour en toute sécurité.

Les migrations sont des fichiers PHP stockés dans le répertoire database/migrations de votre application Laravel. Chaque fichier de migration contient des instructions pour créer ou modifier des tables ou des colonnes dans votre base de données.

Lorsque vous exécutez une migration, Laravel applique automatiquement les modifications à la base de données en fonction des instructions contenues dans le fichier. Cela permet de gérer facilement les modifications de la structure de la base de données au fil du temps.



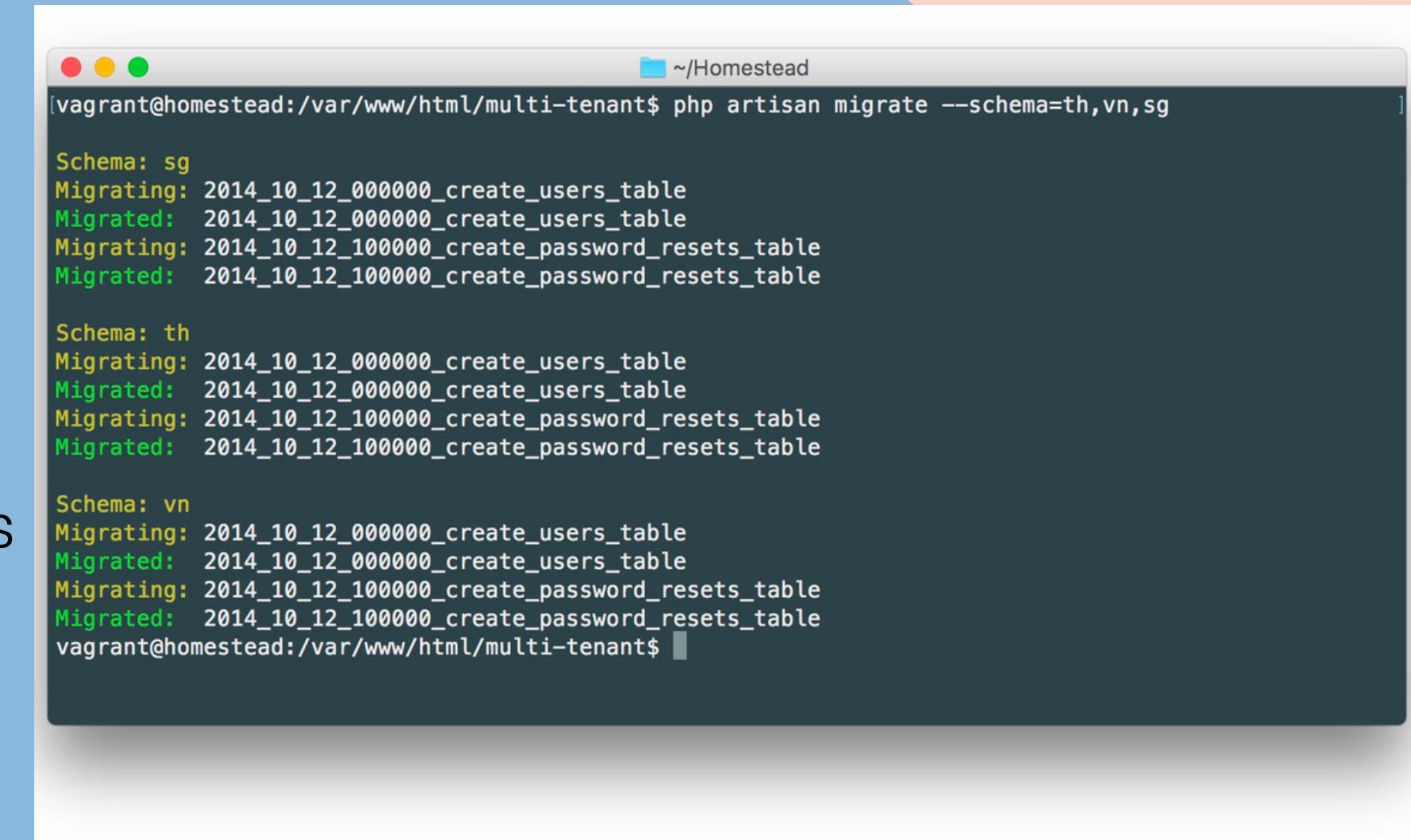
```
MINGW64:/c/Users/Anonymous/Desktop/GfG/Laravel-Migration-Basics
Anonymous@DESKTOP-M2QUDH2 MINGW64 ~/Desktop/GfG/Laravel-Migration-Basics
$ php artisan migrate
Migration table created successfully.
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table (0.06 seconds)
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated: 2014_10_12_100000_create_password_resets_table (0.05 seconds)
Migrating: 2019_08_19_000000_create_failed_jobs_table
Migrated: 2019_08_19_000000_create_failed_jobs_table (0.03 seconds)
Migrating: 2019_12_14_095941_create_articles_table
Migrated: 2019_12_14_095941_create_articles_table (0.03 seconds)

Anonymous@DESKTOP-M2QUDH2 MINGW64 ~/Desktop/GfG/Laravel-Migration-Basics
$ |
```

LES BASES DE DONNÉES

Voici quelques exemples de ce que vous pouvez faire avec les migrations :

- Créer des tables dans la base de données.
- Ajouter ou supprimer des colonnes dans une table.
- Modifier le type de données d'une colonne.
- Ajouter ou supprimer des index ou des clés étrangères.



```
vagrant@homestead:/var/www/html/multi-tenant$ php artisan migrate --schema=th,vn,sg
Schema: sg
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated: 2014_10_12_100000_create_password_resets_table

Schema: th
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated: 2014_10_12_100000_create_password_resets_table

Schema: vn
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated: 2014_10_12_100000_create_password_resets_table
vagrant@homestead:/var/www/html/multi-tenant$
```

LES COMMANDES DE BASE DE LARAVEL

1. `composer create-project --prefer-dist laravel/laravel monprojet`: cette commande permet de créer un nouveau projet Laravel en utilisant Composer.
2. `php artisan serve`: cette commande permet de lancer un serveur de développement local pour votre application Laravel.
3. `php artisan make:controller NomDuController`: cette commande permet de créer un nouveau contrôleur dans votre application.
4. `php artisan make:model NomDuModel`: cette commande permet de créer un nouveau modèle dans votre application.
5. `php artisan make:migration NomDeLaMigration`: cette commande permet de créer une nouvelle migration de base de données.
6. `php artisan migrate`: cette commande permet d'exécuter toutes les migrations en attente pour mettre à jour votre base de données.
7. `php artisan tinker`: cette commande permet de lancer un shell interactif de Laravel pour tester rapidement du code.
8. `php artisan route:list`: cette commande permet d'afficher la liste de toutes les routes enregistrées dans votre application..

MERCI DE VOTRE ATTENTION !

N'hésitez pas à envoyer vos questions
à rejoindre la communauté Jelab sur
Youtube.