

CREACIÓN DE PROYECTO LARAVEL

ALEJANDRO DE LA HUERGA FERNANDEZ

DESARROLLO DE APLICACIONES WEB

10/02/2026

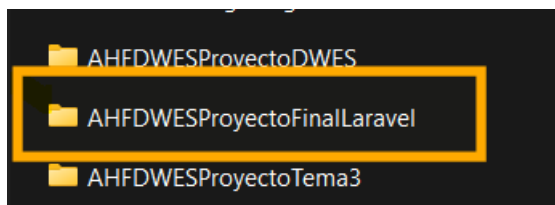
Creación de proyecto en Laravel Framework:

A continuación, vamos a ver una guía paso a paso de la creación de un proyecto Laravel , nos centraremos en la estructura de directorios que nos creara Laravel por defecto, así como sus ficheros de configuración, conexión a la base de datos y manera de trabajar del framework.

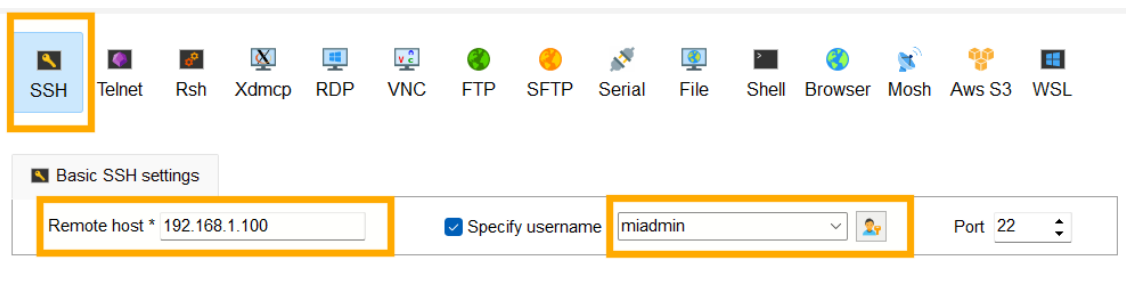
Preparación del cliente de desarrollo.

En primer lugar, nos vamos a centrar en la preparación de nuestro cliente de desarrollo, preparándolo así para poder trabajar de manera local con el futuro proyecto que crearemos con Laravel en nuestro servidor de desarrollo.

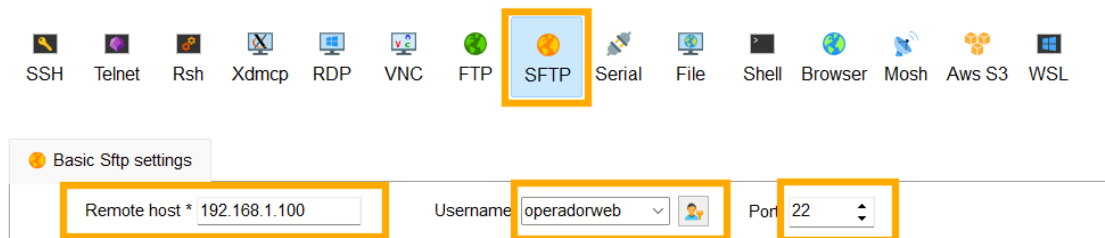
Lo primero que realizaremos será **crear una carpeta local** con el mismo nombre que nuestro proyecto Laravel, esto nos permitirá la transferencia de archivos mediante el **protocolo SFTP** y así poder **actualizar nuestro proyecto de manera continua con el servidor**.



Una vez creada nuestra carpeta de manera local lo que debemos hacer será preparar la **conexión por SSH con el servidor**, esto nos permitirá la ejecución de comandos desde el cliente, para ello utilizaremos el programa **MobaXterm** ya que este mismo programa nos permitirá tanto la conexión por SFTP y SSH.



El siguiente paso que realizaremos será la **conexión por SFTP** con el servidor de desarrollo, que como indicamos anteriormente también lo vamos a realizar desde el **MobaXterm**.

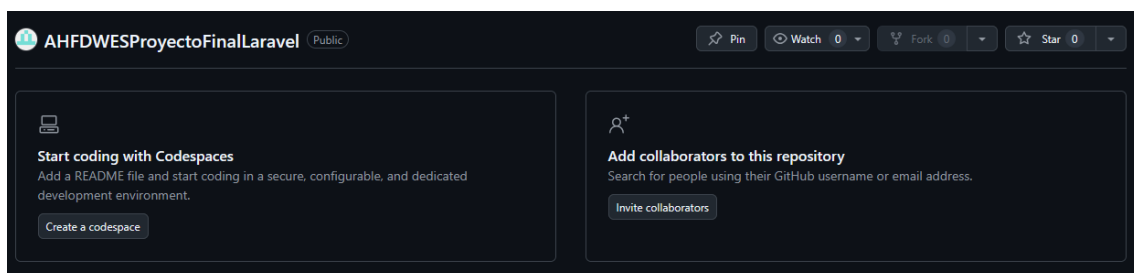


Una vez ya preparadas las conexiones desde el cliente con el servidor y creada la carpeta de manera local, vamos con el siguiente paso.

Creación del repositorio remoto y local.

Para una buena práctica de nuestro proyecto y poder llevar un seguimiento de los cambios que hacemos en el mismo durante su ciclo de vida, vamos a crear un **repositorio de manera remota en GitHub** y enlazarlo con nuestro **repositorio local**.

Esto nos permitirá llevar el control de versiones directamente desde nuestro Entorno de Desarrollo.



Una vez creado nuestro repositorio de manera remota, lo **clonaremos en la carpeta creada anteriormente** para así tenerlo enlazados y sincronizar los cambios desde nuestro Entorno de Desarrollo.

Creación del proyecto Laravel:

Una vez preparado nuestro cliente de desarrollo y nuestro repositorio de GitHub, lo que vamos a realizar es la creación del proyecto Laravel en el Servidor de Desarrollo.

Para ello seguiremos los siguientes pasos:

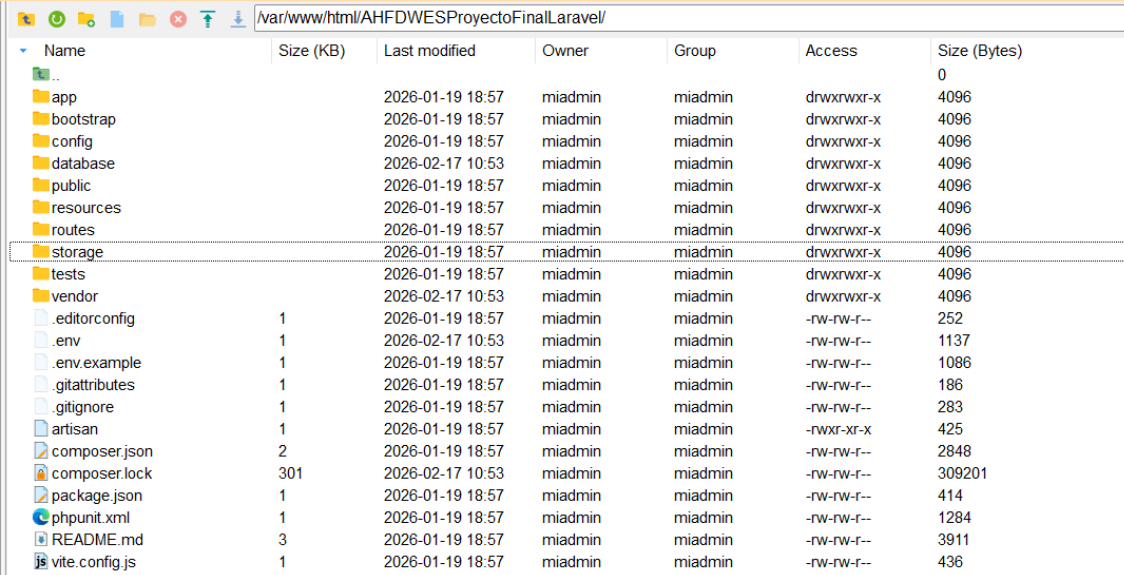
1. Abrimos la **conexión por SSH en el MobaXterm**.
2. Vamos al **directorio** donde queremos crear el proyecto.
3. Ejecutamos el **comando de creación de composer**.

```
miadmin@ahf-casa:~$ cd /var/www/html
miadmin@ahf-casa:/var/www/html$ composer create-project laravel/laravel AHFDWESProyectoFinalLaravel
Creating a 'laravel/laravel' project at './AHFDWESProyectoFinalLaravel'
Installing laravel/laravel (v12.11.2)
- Installing laravel/laravel (v12.11.2): Extracting archive
Created project in /var/www/html/AHFDWESProyectoFinalLaravel
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
Updating dependencies
```

Esto nos creara un proyecto Laravel en la ruta donde hemos ejecutado el comando de creación que se ve a continuación.

composer create-project laravel/laravel AHFDWESProyectoFinalLaravel

Mediante este comando ya hemos realizado la **creación de nuestro proyecto Laravel** en el servidor tal y como se ve a continuación:



Name	Size (KB)	Last modified	Owner	Group	Access	Size (Bytes)
..						0
app		2026-01-19 18:57	miadmin	miadmin	drwxrwxr-x	4096
bootstrap		2026-01-19 18:57	miadmin	miadmin	drwxrwxr-x	4096
config		2026-01-19 18:57	miadmin	miadmin	drwxrwxr-x	4096
database		2026-02-17 10:53	miadmin	miadmin	drwxrwxr-x	4096
public		2026-01-19 18:57	miadmin	miadmin	drwxrwxr-x	4096
resources		2026-01-19 18:57	miadmin	miadmin	drwxrwxr-x	4096
routes		2026-01-19 18:57	miadmin	miadmin	drwxrwxr-x	4096
storage		2026-01-19 18:57	miadmin	miadmin	drwxrwxr-x	4096
tests		2026-01-19 18:57	miadmin	miadmin	drwxrwxr-x	4096
vendor		2026-02-17 10:53	miadmin	miadmin	drwxrwxr-x	4096
.editorconfig	1	2026-01-19 18:57	miadmin	miadmin	-rw-rw-r--	252
.env	1	2026-02-17 10:53	miadmin	miadmin	-rw-rw-r--	1137
.env.example	1	2026-01-19 18:57	miadmin	miadmin	-rw-rw-r--	1086
.gitattributes	1	2026-01-19 18:57	miadmin	miadmin	-rw-rw-r--	186
.gitignore	1	2026-01-19 18:57	miadmin	miadmin	-rw-rw-r--	283
artisan	1	2026-01-19 18:57	miadmin	miadmin	-rwxr-xr-x	425
composer.json	2	2026-01-19 18:57	miadmin	miadmin	-rw-rw-r--	2848
composer.lock	301	2026-02-17 10:53	miadmin	miadmin	-rw-rw-r--	309201
package.json	1	2026-01-19 18:57	miadmin	miadmin	-rw-rw-r--	414
phpunit.xml	1	2026-01-19 18:57	miadmin	miadmin	-rw-rw-r--	1284
README.md	3	2026-01-19 18:57	miadmin	miadmin	-rw-rw-r--	3911
vite.config.js	1	2026-01-19 18:57	miadmin	miadmin	-rw-rw-r--	436

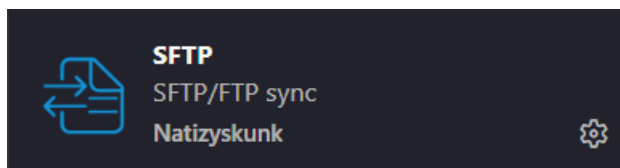
Como podemos comprobar desde el MobaXterm ya tenemos nuestro proyecto Laravel creado en nuestro servidor en la ruta indicada.

Sincronización mediante SFTP del proyecto en el servidor con el proyecto en el cliente.

Una vez ya creado nuestro proyecto en el servidor lo que debemos hacer es realizar la sincronización con la carpeta local que creamos inicialmente en nuestro cliente de desarrollo.

Para ello vamos a utilizar el entorno de desarrollo Visual Studio Code el cual nos permitirá realizar esta sincronización.

1. Abrimos nuestra carpeta local en Visual Studio Code.
2. Debemos tener instalada la extensión: **SFTP**.



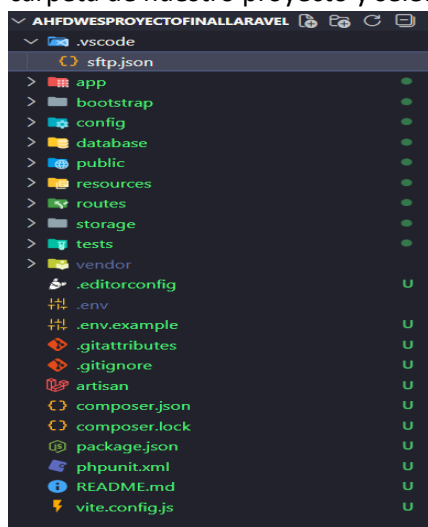
3. **CTRL + SHIFT + P**: Buscamos SFTP, esto nos creara un archivo JSON donde debemos poner nuestros datos de conexión.

```

sftp.json U X
.vscode > sftp.json > ...
1  {
2      "name": "My Server",
3      "host": "192.168.1.100",
4      "protocol": "sftp",
5      "port": 22,
6      "username": "operadorweb",
7      "remotePath": "/var/www/html/AHFDWESProyectoFinalLaravel",
8      "password": "paso",
9      "uploadOnSave": true,
10     "useTempFile": false,
11     "openSsh": false
12 }
13

```

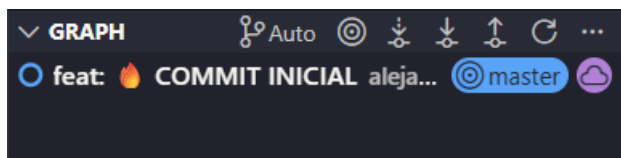
4. **Bajar el proyecto de nuestro servidor a local**: Para ello haremos click derecho en la carpeta de nuestro proyecto y seleccionamos la opción **"Download folder"**.



Sincronización repositorio local con repositorio remoto.

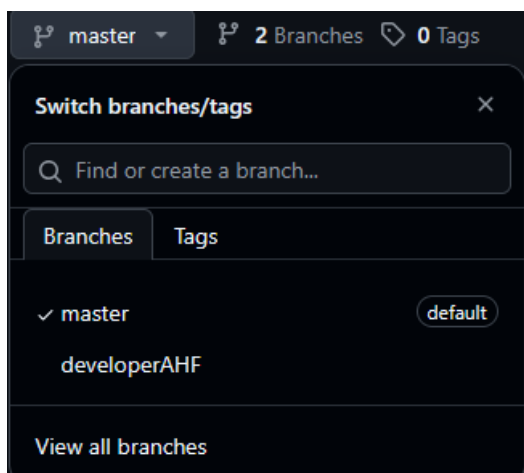
Una vez bajado nuestro proyecto Laravel para poder trabajar con el debemos sincronizar nuestro repositorio local con nuestro repositorio remoto de GitHub.

Para ello haremos un commit inicial y un push con todo el proyecto Laravel para así subirlos a nuestro repositorio remoto.

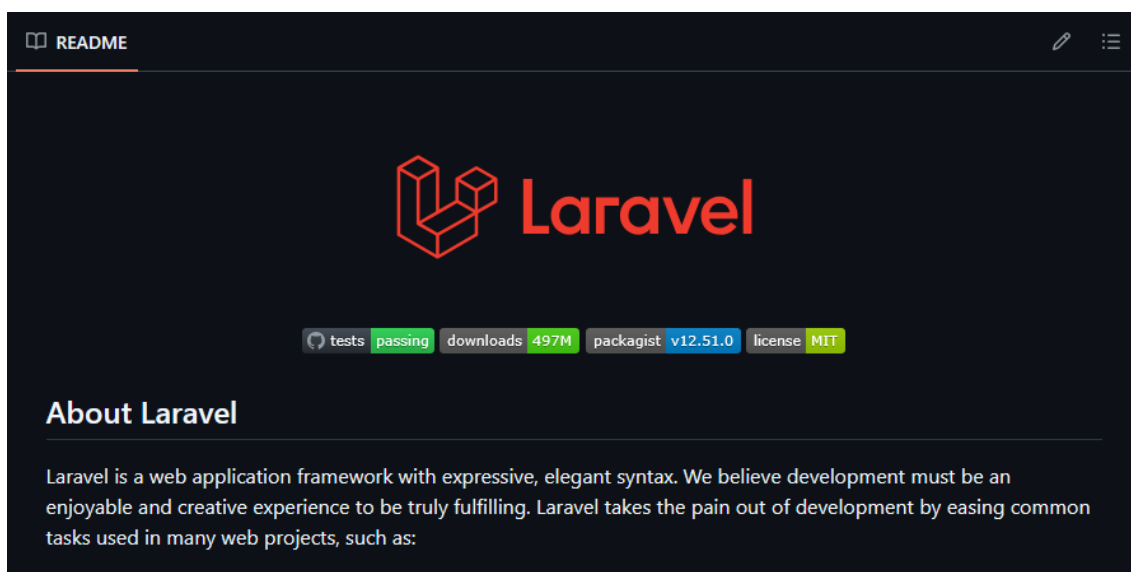


Una vez publicado nuestro primer commit lo que demos hacer es **crear nuestra rama developerAHF** en mi caso **en local**.

Una vez creada **publicamos nuestra nueva rama developer en GitHub**, consiguiendo así tener ambas ramas enlazadas entre nuestro repositorio local y remoto.



Cabe destacar que mediante la creación del proyecto Laravel, este framework ya nos proporciona un **README** estándar para nuestros repositorios.



Directorios y estructura de un proyecto Laravel.

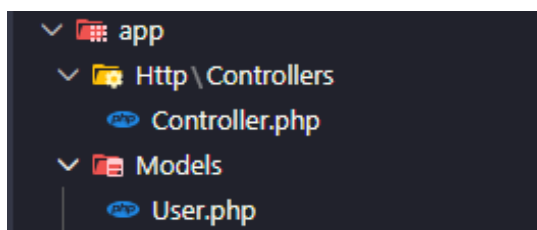
Ya creado nuestro proyecto, bajado a local para poder trabajar con el y versionado mediante el control de versiones de Git, vamos a continuar con **la explicación de los directorios y archivos de un proyecto Laravel así como su funcionalidad.**

Archivo .env.

Este es el **archivo de configuración** de cualquier proyecto creado por Laravel, este archivo nos permitirá configurar los diferentes aspectos de nuestra aplicación web **como la URL, la conexión con la base de datos los LOG, el idioma...**

Directorio app.

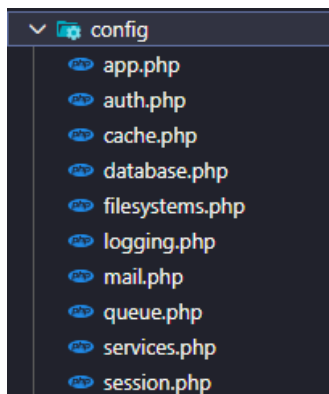
En este directorio es donde se encuentran tanto los controladores como los modelos que representan las tablas de la base de datos.



Directorio config.

El directorio config es el **cerebro de la configuración de la aplicación**, aquí se define el comportamiento de Laravel en cuanto a las bases de datos, correo, sesiones, etc.

Cada archivo de este directorio es un **array el cual devuelve las diferentes configuraciones.**



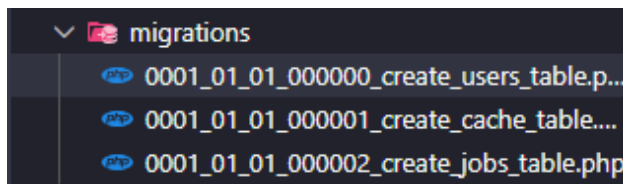
Ejemplo del archivo de configuración app.php

Directorio database.

El directorio “database” almacena diferentes **directorios relacionados con la base de datos** /factories , /migrations, /seeders.

- **Factories:** Contiene los factories de Eloquent , sirven para generar datos de prueba. Muy utilizadas para testing o rellenar tablas con datos en desarrollo.
- **Migrations:** Contiene archivos de migración que son versiones de la base de datos. Cada vez que creamos una tabla con **Artisan**, se **crea automáticamente un archivo en migrations** el cual contiene la estructura de la tabla.

Comando Artisan: **`php artisan make:migration create_productos_table`**



```
/**
 * Run the migrations.
 */
public function up(): void
{
    Schema::create('users', function (Blueprint $table) {
        $table->id();
        $table->string('name');
        $table->string('email')->unique();
        $table->timestamp('email_verified_at')->nullable();
        $table->string('password');
        $table->rememberToken();
        $table->timestamps();
    });
}
```

- **Seeders:** Contiene clases para llenar datos iniciales en la base de datos.

```
class DatabaseSeeder extends Seeder
{
    use WithoutModelEvents;

    /**
     * Seed the application's database.
     */
    public function run(): void
    {
        // User::factory(10)->create();

        User::factory()->create([
            'name' => 'Test User',
            'email' => 'test@example.com',
        ]);
    }
}
```

Directorio public.

El directorio public almacena todo lo relacionado con el **front controller**, todo lo que ve el cliente desde el navegador.

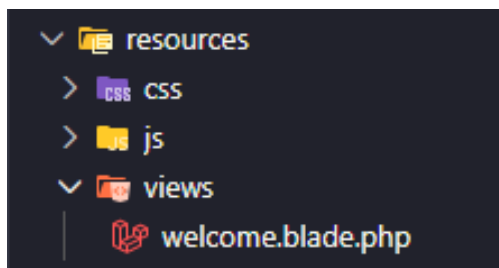
- Index.php: Cada petición HTTP pasa por ese archivo. Es el **front controller del patrón MVC**.

Este directorio almacena todo lo que el navegador puede ver por eso no debemos poner ningún tipo de archivo de configuración aquí.

Directorio resources.

En este directorio se encuentran todos aquellos archivos relacionados con el **cliente o front controller como son el css o los archivos JavaScript**, estos archivos **se compilarán y aparecerán también en la carpeta public para que el navegador pueda acceder a ellos**.

También encontraremos en este directorio el **directorio views o vistas** el cual almacena todas aquellas **vistas generadas con Blade**.



Directorio routes.

Este directorio almacena todos los archivos con las diferentes rutas de las url a lo largo de la aplicación.

El archivo creado por defecto **web.php** define las rutas accesibles desde el navegador y normalmente usa **sesión y cookies**.

```
routes > web.php
1  <?php
2
3  use Illuminate\Support\Facades\Route;
4
5  Route::get('/', function () {
6      return view('welcome');
7  });
8
```

Configuración Hola Mundo Laravel

Una vez vista la estructura de directorios y archivos vamos a visualizar por primera vez nuestro proyecto Laravel.

Para ello vamos a editar el fichero `.env` para realizar la configuración con nuestros datos del servidor.

```

1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:wxFA0FahIAOKKz6kEIV9KbEEeORPJxj3tzWS50ExIR0=
4 APP_DEBUG=true
5 APP_URL=http://192.168.1.100/AHFDWESProyectoFinalLaravel/public
6

```

Las primeras líneas de este archivo tenemos la configuración de nuestra aplicación como el nombre el mode debug o la **url donde queremos desplegar nuestra aplicación en desarrollo**.

Como indicamos anteriormente debemos indicarle la carpeta public ya que es la carpeta a la que tiene acceso nuestro navegador.

También debemos editar la parte de la base de datos ya que por defecto viene sqlite y nos saltara **Exception**, debemos **poner mysql** ya que es la que tenemos instalada en nuestro servidor.

```

DB_CONNECTION=mysql
# DB_HOST=127.0.0.1
# DB_PORT=3306
# DB_DATABASE=laravel
# DB_USERNAME=root
# DB_PASSWORD=

```

Lo siguiente que debemos hacer es crear la base de datos y el usuario el cual indicaremos a Laravel para que el mismo pueda realizar las migraciones.

```

C: > Users > aleja > .dbclient > storage > 1768843093891@@192.168.1.100@3306 > create-db-template.sql > ...
  ▷Run | ▷Run on active connection | ≡Select block
1 CREATE DATABASE IF NOT EXISTS AHFDWESProyectoFinalLaravel;
2
  ▷Run
3 USE AHFDWESProyectoFinalLaravel;
4
  ▷Run
5 CREATE USER 'USERAHFDWESProyectoFinalLaravel'@'%' IDENTIFIED BY 'paso';
6
  ▷Run
7 GRANT ALL PRIVILEGES ON AHFDWESProyectoFinalLaravel.* TO 'USERAHFDWESProyectoFinalLaravel'@'%' ;
8
  ▷Run
✓ 9 FLUSH PRIVILEGES; 3ms

```

Estos datos de creación los utilizaremos para rellenar el archivo de configuración.

```
DB_CONNECTION=mysql
DB_HOST=192.168.1.100
DB_PORT=3306
DB_DATABASE=AHFDWESProyectoFinallaravel
DB_USERNAME=USERAHFDWESProyectoFinallaravel
DB_PASSWORD=paso
```

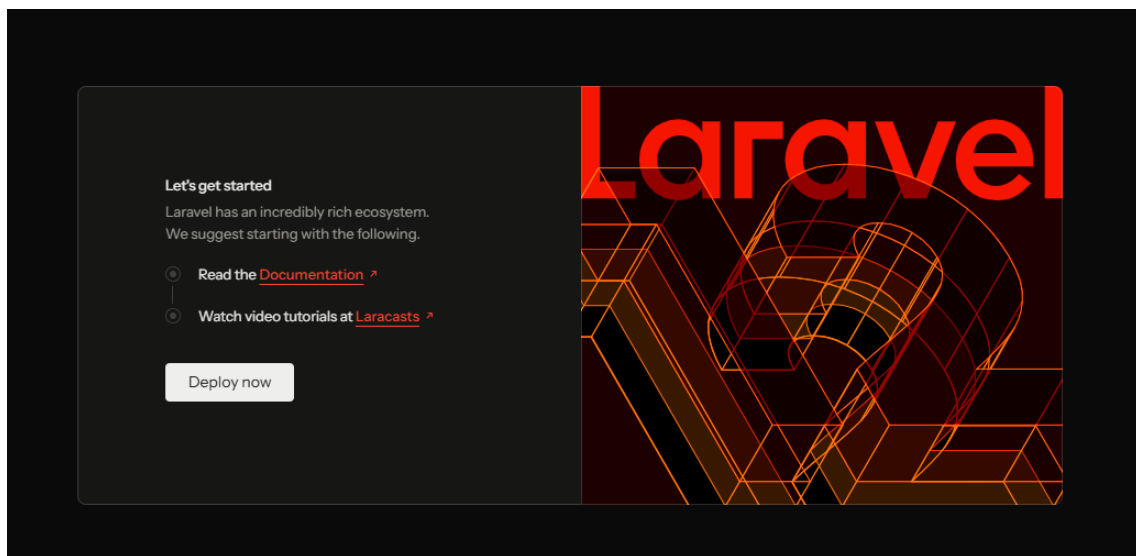
Ahora debemos realizar las migraciones mediante Artisan , ya que sino lo hacemos Laravel no encontrara las tablas y nos lanzara una Excepción.

php artisan migrate

INFO Running migrations.

```
0001_01_01_000000_create_users_table ..... 88.61ms DONE
0001_01_01_000001_create_cache_table ..... 67.49ms DONE
0001_01_01_000002_create_jobs_table ..... 72.90ms DONE
```

Una vez realizadas las migraciones ya podemos ver la página de bienvenida de nuestro proyecto en Laravel.



Documentación

La siguiente documentación ha sido realizada mediante la extracción de información de los siguientes sitios web:

- Documentación Laravel: <https://laravel.com/docs/12.x/configuration>
- Instalación de proyecto Laravel en Apache: <https://youtu.be/v6aE-hk8XoM?si=eGQg-ZTHXArVVo8P>
- Creación de proyecto Laravel entorno y configuración: <https://10code.es/crear-proyecto-laravel/>