**Identificación:** Anexo03-Artisan   
**Versión:** 1.0

Tabla de contenido

[Estructura de un Proyecto de Laravel 8 2](#_Toc63100304)

[¿Artisan? 7](#_Toc63100305)

[Bibliografia 12](#_Toc63100306)

# **Estructura de un Proyecto de Laravel 8**

Al crear un nuevo proyecto de Laravel se nos generará una estructura de carpetas y ficheros para organizar nuestro código. Es importante que conozcamos para que vale cada elemento y donde tenemos que colocar nuestro código

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

|  |  |
| --- | --- |
| Interfaz de usuario gráfica, Aplicación  Descripción generada automáticamente  Interfaz de usuario gráfica, Aplicación  Descripción generada automáticamente | * app – Contiene el código principal de la aplicación. Esta carpeta a su vez está dividida en muchas subcarpetas que analizaremos en la siguiente sección.   + **app/Exceptions:** Manejo de las excepciones de la aplicación.   + **app/Http/Controllers:** Esta carpeta contiene todos los archivos con las clases de los controladores.   + **app/Http/Middleware**: Los archivos contenidos en esta carpeta proporcionan un mecanismo conveniente para filtrar peticiones HTTP que ingresan a su aplicación; Por ejemplo Laravel incluye un middleware que verifica que el usuario de su aplicación esté autenticado; si por acaso el usuario no está autenticado, el middleware redirigirá al usuario a la pantalla para que inicie sesión. Sin embargo, en caso el usuario esté autenticado permitirá que el usuario continue en la aplicación. Y por supuesto, se pueden crear más archivos MiddleWare para realizar más tareas además la autenticación.   + **app/Mail:** Esta contiene las clases para la interacción con los correos electronicos.   + **app/Models:** Esta contiene los modelos creados en el proyecto “presente desde laravel 8”   + **app/Providers:** Esta contiene las clases de los diferentes proveedores de servicio para la aplicación * **config** – Aquí se encuentran todos los archivos de configuración de la aplicación: base datos, cache, correos, sesiones o cualquier otra configuración general de la aplicación. * **database** – En esta carpeta se incluye todo lo relacionado con la **definición de la base de datos** de nuestro proyecto. Dentro de ella podemos encontrar a su vez tres carpetas: factores, migrations y seeds. * **public** – Es la única carpeta pública, la única que debería ser **visible** en nuestro servidor web. Todo las peticiones y solicitudes a la aplicación pasan por esta carpeta, ya que en ella se encuentra el index.php, este archivo es el que inicia todo el proceso de ejecución del framework. En este directorio también se alojan los archivos CSS, Javascript, imágenes y otros archivos que se quieran hacer públicos. * **resources** – Esta carpeta contiene a su vez tres carpetas: assets, views y lang:   + **resources/views** – Este directorio contiene las vistas de nuestra aplicación. En general serán plantillas de HTML que usan los controladores para mostrar la información. Hay que tener en cuenta que en esta carpeta no se almacenan los Javascript, CSS o imágenes, ese tipo de archivos se tienen que guardar en la carpeta public.   + **resources/lang** – En esta carpeta se guardan archivos PHP que contienen arrays con los textos de nuestro sitio web en diferentes lenguajes, solo será necesario utilizarla en caso que se desee que la aplicación se pueda traducir.   + **resources/assets** – Se utiliza para almacenar los fuentes de los assets tipo less o sass que se tendrían que compilar para generar las hojas de estilo públicas. No es necesario usar esta carpeta ya que podemos escribir directamente las las hojas de estilo dentro de la carpeta public. * **bootstrap** – En esta carpeta se incluye el código que se carga para procesar cada una de las llamadas a nuestro proyecto. Normalmente no tendremos que modificar nada de esta carpeta. * **storage** – En esta carpeta Laravel almacena toda la información interna necesarios para la ejecución de la web, como son los archivos de sesión, la caché, la compilación de las vistas, meta información y los logs del sistema. Normalmente tampoco tendremos que tocar nada dentro de esta carpeta, unicamente se suele acceder a ella para consultar los logs. * **tests** – Esta carpeta se utiliza para los ficheros con las pruebas automatizadas. Laravel incluye un sistema que facilita todo el proceso de pruebas con PHPUnit. * **vendor** – En esta carpeta se alojan todas las librerías y dependencias que conforman el framework de Laravel. Esta carpeta tampoco la tendremos que modificar, ya que todo el código que contiene son librerías que se instalan y actualizan mediante la herramienta Composer.   Además en la carpeta raíz también podemos encontrar dos ficheros muy importantes y que también utilizaremos:   * **.env** – Este fichero ya lo hemos mencionado en la sección de instalación, se utiliza para almacenar los valores de configuración que son propios de la máquina o instalación actual. Lo que nos permite cambiar fácilmente la configuración según la máquina en la que se instale y tener opciones distintas para producción, para distintos desarrolladores, etc. Importante, este fichero debería estar en el .gitignore. * **composer.json** – Este fichero es el utilizado por Composer para realizar la instalación de Laravel. En una instalación inicial únicamente se especificará la instalación de un paquete, el propio framework de Laravel, pero podemos especificar la instalación de otras librerías o paquetes externos que añadan funcionalidad a Laravel. |

# **¿Artisan?**

Entre las herramientas que Laravel nos proporciona para el desarrollo de aplicaciones se encuentra Artisan, la interfaz de línea de comandos (CLI por sus siglas en inglés de *Command-line interface*), la cual es un medio para la interacción con la aplicación donde los usuarios (en este caso los desarrolladores) dan instrucciones en forma de línea de texto simple o línea de comando. Artisan está basado en el componente Console de Symfony y nos ofrece un conjunto de comandos que nos pueden ayudar a realizar diferentes tareas durante el desarrollo e incluso cuando la aplicación se encuentra en producción. En el tutorial de hoy veremos las características de esta gran herramienta.

Van a realizar cada una de las pruebas dentro el proyecto previamente creado y verificar los cambios en el proyecto

Para conocer el listado completo de los comandos disponibles ejecutamos en consola, en el directorio raíz de un proyecto de Laravel:

**php artisan list**

Texto

Descripción generada automáticamente

**Ver listado de rutas de la app:**

*php artisan route:list*

*Texto

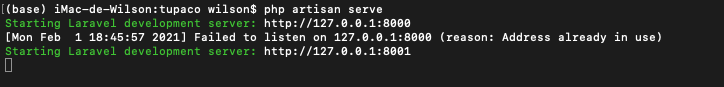
Descripción generada automáticamente con confianza media*

**Cambiar nombre de namespace: OJO**

*php artisan app:name <nombre>*

**Levantar servidor:**

*php artisan serve*



**Consultar todas las funcionalidades de artisan:**

*php artisan -h*

**Ver comandos de los modelos:**

*php artisan make:model –h*

*Texto

Descripción generada automáticamente*

**Crear modelos:**

*php artisan make:model <modelo>*

*Interfaz de usuario gráfica

Descripción generada automáticamente*

**Crear modelos con migración:**

*php artisan make:model <modelo> -m*

*Texto

Descripción generada automáticamente*

**Crear modelos con migración y factories:**

*php artisan make:model <modelo> -mf*

*Texto

Descripción generada automáticamente*

Crear modelos con migración, factories y controladores:

*php artisan make:model <modelo> -crmf*

*php artisan make:model <modelo> --all*

*Texto

Descripción generada automáticamente*

*Resumen*

*Texto

Descripción generada automáticamente*

**Ver comandos de los controladores:**

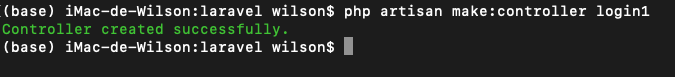
*php artisan make:controller - h*

Texto

Descripción generada automáticamente

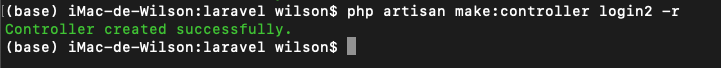
**Crear controlador vacío:**

*php artisan make:controller <Carpeta/ClaseController>*

**

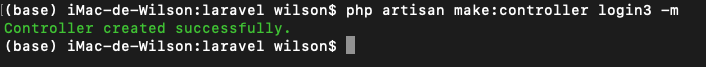
Crear controlador con recursos (métodos por default de laravel):

*php artisan make:controller <Carpeta/Clasecontroller> -r*

**

**Crear controlador con recursos con inyección de dependencias de modelos:**

*php artisan make:controller <Carpeta/Clasecontroller> -m <Modelo>*



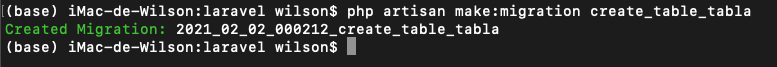
*Resumen*

Texto

Descripción generada automáticamente

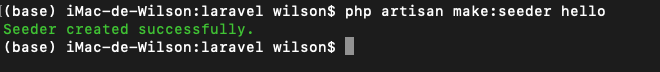
**Crear tabla como migración sin modelo:**

*php artisan make:migration create\_table\_tabla*

****

**Crear factories**

*php artisan make:seeder <seeder>*



**Hacer migraciones:**

*php artisan migrate*

**Hacer migraciones de factories:**

*php artisan db:seed*

**Hacer migraciones y seeders**

*php artisan migrate --seed*

**Retroceder una migración:**

*php artisan migrate:rollback*

**Retroceder pasos de migración:**

*php artisan migrate:rollback –step=5*

**Rehacer una migración:**

*php artisan migrate:fresh*

**Rehacer una migración y factories:**

*php artisan migrate:fresh –seed*

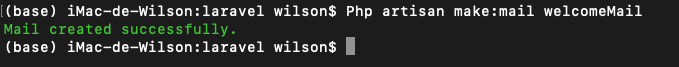
**Crear recursos**

*php artisan make:resource ModelResource*

**

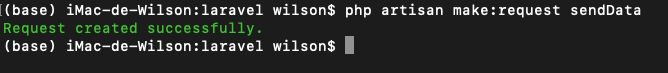
**Crear clase para correo electrónico**

*Php artisan make:mail <Clase>*

**

**Crear Request para validaciones**

*php artisan make:request <ModelRequest>*

**

# Bibliografia

https://laravel.com/docs/8.x