Laravel es un framework de código abierto para el desarrollo de aplicaciones web en PHP que posee una sintaxis simple y elegante.

Características:

- Creado en 2011 por Taylor Otwell.
- Está inspirado en **Ruby on rails** y **Symfony**, de quien posee muchas dependencias.
- Está diseñado para desarrollar bajo el patrón **MVC**
- Posee un sistema de mapeado de datos relacional llamado **Eloquent ORM**.
- Utiliza un sistema de procesamiento de plantillas llamado **Blade**, el cual hace uso de la caché para darle mayor velocidad

Para la utilización de Laravel en primer lugar necesitamos tener instalado lo siguiente:

- Un servidor web Apache, PHP y MySQL, (XAMPP)
- Composer Enlace a la página oficial

  Permite descargar y gestionar las dependencias del Framework.

  Es un administrador de dependencias para PHP que nos permite descargar paquetes desde un repositorio para agregarlo a nuestro proyecto.

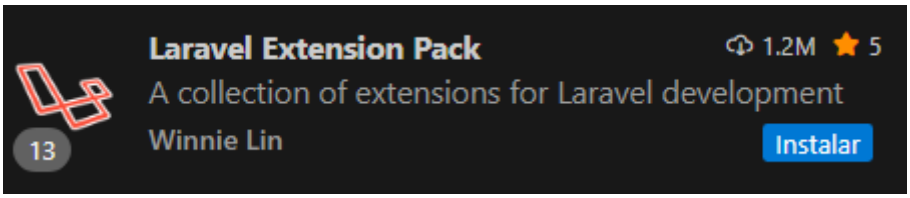
  Por defecto, se agregan a una carpeta llamada /vendor. De esta manera evitamos hacer las búsquedas manualmente y el mismo Composer se puede encargar de actualizar las dependencias que hayamos descargado por una nueva versión.

- Adicionalmente también es recomendable instalar Node.js. Se instala la herramienta NPM (Node Package Manager), herramienta que permite instalar librerías de JavaScript, como BootStrap o jQuery. La página oficial de Node.js.

Comprobamos la versión de node.js instalada

```
node -v
```

Como entorno de desarrollo vamos a utilizar **Visual Studio code** e instalar la extensión Laravel extension pack que a su vez nos instala 13 plugins.

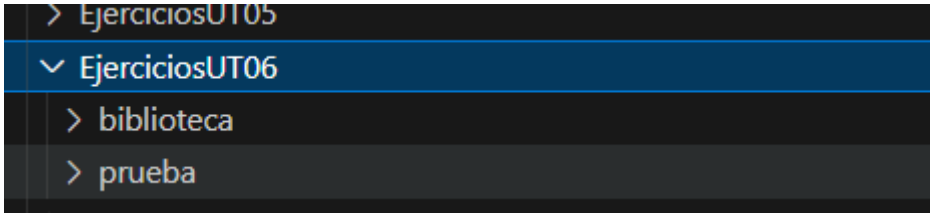



### Instalación y primer proyecto

Instalación de Laravel y la creación del proyecto, poniendo en **la carpeta de publicación** por ejemplo, c:\xampp\htdocs\EjerciciosUT06:

```
PS C:\xampp\htdocs\EjerciciosUT06> composer create-project laravel/laravel prueba
Creating a "laravel/laravel" project at "./prueba"
Installing laravel/laravel (v12.11.0)
  - Downloading laravel/laravel (v12.11.0)
```

Ahora nos ha creado una carpeta de nombre prueba, como nuestro proyecto



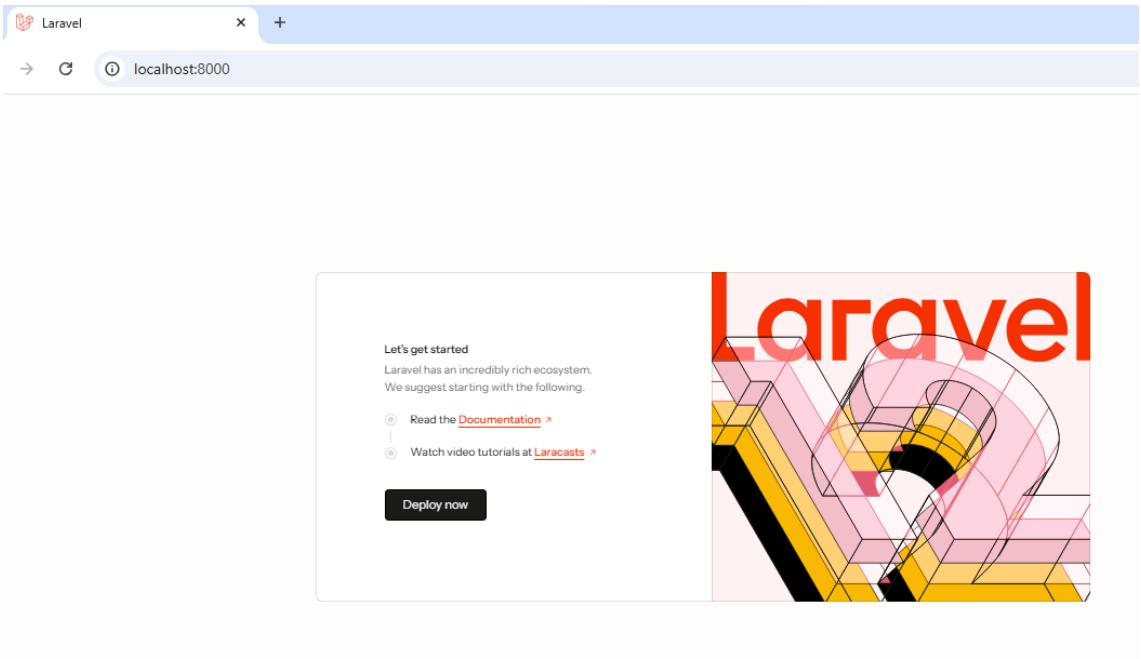

Vamos a terminal y nos ponemos en la ruta del proyecto y ejecutamos el siguiente comando de artisan, que lanza nuestro proyecto al servidor en modo desarrollo.

```
PS C:\xampp\htdocs\EjerciciosUT06> cd prueba
PS C:\xampp\htdocs\EjerciciosUT06\prueba> php artisan serve

  INFO  Server running on [http://127.0.0.1:8000].

  Press Ctrl+C to stop the server
```

En el navegador:




## Patrón de Laravel

Laravel se basa en un diseño MVC

# Estructura de Laravel

Vamos a ver la estructura de nuestro proyecto, para así entender que hay dentro de las principales carpetas:

**app** - contiene los controladores, modelos, vistas y configuraciones de la aplicación. En esta carpeta escribiremos la mayoría del código para que nuestra aplicación funcione. En la carpeta Models tenemos un solo modelo: "User.php" que se crea por defecto.

**app/Http**

> **Controllers** son los que interaccionan con los modelos
>
> **Midleware** son filtros de seguridad cuando se envía una ruta, un formulario….

**bootstrap** son archivos del sistema. 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.

**config** todos los archivos de configuración del sistema

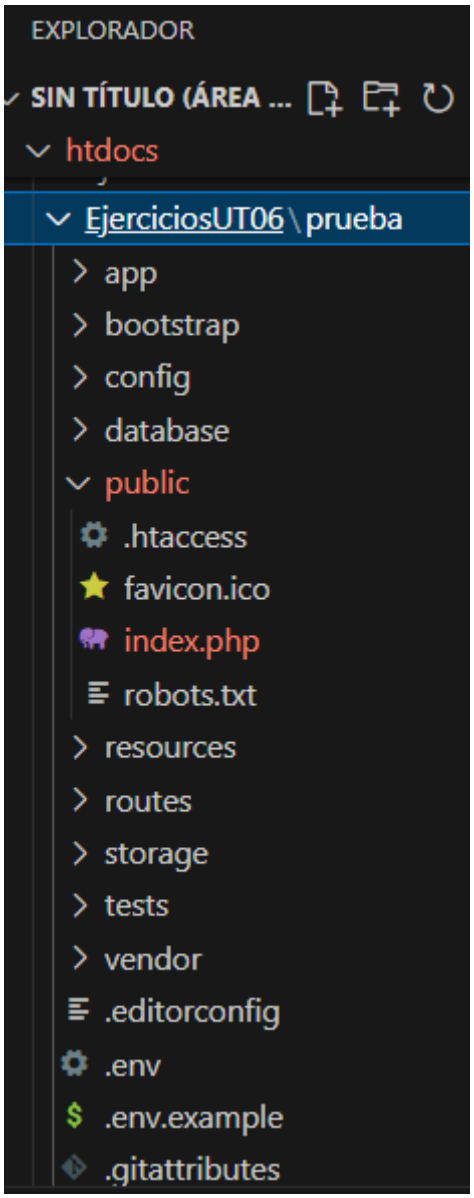
**app.php** tenemos los namespace para acceder a las librerías internas de Laravel,si descargamos algo nuevo para nuestra aplicación hay que instalar los namespace aquí

**database.php** se configura la B.D. que ya contiene mysql, pero también tiene sqlite y otras

**filesystems.php** maneja discos internos en laravel: imágenes, videos,…

**database**

**migrations** se crea la estructura para la BD,tablas…




En Laravel 10, el directorio lang que en versiones anteriores aparecía en la raíz del proyecto ya no se incluye por defecto, para que aparezca, es necesario ejecutar el siguiente comando:

```
php artisan lang:publish
```

**lang** en esta carpeta se guardan archivos PHP que contienen arrays con los textos de nuestro sitio web en diferentes lenguajes; solo será necesario utilizarlo en caso de que se desee que la aplicación se pueda traducir

**public** es la única carpeta a la que los usuarios de la aplicación pueden acceder. Todas 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**

**views** las vistas. Welcome.blade.php que es la página de inicio

**routes** las rutas. web.php es la más importante, aquí se definen las rutas para interpretar las solicitudes que el usuario hace al sistema

**storage** discos internos de laravel. En esta carpeta almacena toda la información interna necesaria para la ejecución de la web, como los archivos de sesión, la caché, la compilación de las vistas, metainformación y logs del sistema.

**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 que conforman el framework y sus dependencias
Además, en la carpeta raíz también encontramos tres ficheros importantes que utilizaremos:

**.env** 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.

**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.

**package.json** en este fichero se encuentran algunas dependencias por parte cliente( Bootstrap o jQuery), y se encuentran preinstaladas en la carpeta node_modules.

Estos tres archivos no deben subirse a ningún repositorio(GitHub) incluir en el **.gitignore**, porque si importamos un proyecto en laravel, podemos regenerar las dependencias de PHP con **composer install** y las dependencias de JavaScript con **npm install**, es decir, los archivos composer.json y package.json actúan como índice de dependencias de PHP y JavaScript, respectivamente.

## Nuestro primer “Hola Mundo!”

Para ellos vamos a crear una vista en el proyecto, en resources/views nos creamos un archivo holamundo.blade.php, simplemente código HTML.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Hola Mundo</title>
</head>
<body>
    <h1>¡Hola Mundo!</h1>
</body>
</html>
```

Ahora nos vamos a routes/web.php, donde inicialmente tenemos el siguiente contenido:

```
<?php

use Illuminate\Support\Facades\Route;

Route::get('/', function () {
    return view('welcome');
});
```

Muestra que para una petición get a nuestro directorio raíz, va a devolver una vista de nombre ‘welcome’, que la hemos visto en resources/views-> welcome.blade.php

Route nos va a facilitar entre otros los siguiente métodos estáticos, que se corresponden con peticiones HTTP

```
// Define routes for various HTTP methods
Route::get();
//Lógica para manejar la solicitud GET
Route::post();
//Lógica para manejar la solicitud POST
Route::put();
//Lógica para manejar la solicitud PUT
Route::delete();
//Lógica para manejar la solicitud DELETE
Route::patch();
//Lógica para manejar la solicitud PATCH
```

Además de los anteriores nos va a dar otro método estático como view(), que nos servirá para mostrar vistas estáticas (no va a obtener datos de ningún modelo y por tanto no necesita controlador).

```php
<?php

use Illuminate\Support\Facades\Route;

Route::view('/', 'holamundo ');
```

El primer parámetro de Route es la ruta propiamente dicha y el segundo la vista o controlador.