

ACTIVIDAD #2

04MASW - Desarrollo de aplicaciones Web I:
Lado del Servidor (back-end)

22/02/2022

Alberto Mateo Martínez

Índice de contenido

1. Introducción	3
2. Requisitos Funcionales.....	4
3. Desarrollo conceptual	5
4.Implementación	7
Instalación del entorno	7
Creación de la Base de datos (vacía).....	7
Verificación / instalación de composer y Laravel.....	8
Creación del Proyecto” series”.....	10
Migraciones – Creación de los datos de las tablas.....	15
Creación de los modelos	22
Creación de los Controladores:	24
Creación de las rutas	27
Creación de vistas utilizando Blade.....	30
Autentificación	31
Seeder -” sembradores ” de datos.....	35
Paginación	40
SoftDeletes	41
Uso del Layout.....	43
Código empleado en los controladores (Platfom, Language, Actor, Director, Serie, Cast)	44
Método Index.....	44
Método Create	45
Método Store	45
Método Edit.....	45
Método Update.....	45
Vista de Listar (y buscar)	46
Vista de Crear Actualizar	47
5. Conclusiones.....	48
5. Bibliograffía	49
6. Anexos	50
Anexo1: Vistas de la Aplicación.....	50
Pantalla inicial	50

Creación, Listado, Edición y Borrado de Plataformas	51
Creación, Listado, Edición y Borrado de Idiomas	53
Creación, Listado, Edición y Borrado de directores	56
Creación, Listado, Edición y Borrado de Actores	57
Creación, Listado, Edición y Borrado de series	58
Creación, Listado, Edición y Borrado de la relación Actores - Series	59

1. Introducción

El objetivo de esta actividad es la programación de un sitio web desarrollado con el Framework LARAVEL 6 de PHP del lado del servidor.

El sitio web solicitado es una biblioteca de Series que contendrá datos de Series, Plataforma, Idioma, Directores, Actores, etc.

Se acompaña dicha aplicación web con el presente documento. De forma breve quedan desarrollados los Requisitos, Diseño conceptual y la Implementación. Más que constituir un documento formal, aprendido en otras asignaturas, se trata de una presentación de la aplicación web, de los fundamentos y bases de Back-End con las que se ha construido.

Por la facilidad de uso, por integración con **Visual Studio Code**, y sobre todo por el conocimiento del conjunto de herramientas se elige realizar la práctica con **MySQL** (Mariadb) sobre el paquete o pila de aplicaciones de código abierto **XAMP. Versión 3.3.0** (Compilación 6 de abril de 2021)

2. Requisitos Funcionales

La actividad establece unos requisitos generales. (similares a los de la Actividad 1) Se trata de realizar una aplicación Web para almacenar una biblioteca de series. El sistema realizado ha de almacenar las series de manera que sean recuperables. Se fijan otros condicionantes para asegurar la integridad de los datos como control de duplicados, control de claves foráneas, etc.

Las funcionalidades posibles son muy numerosas y todas muy interesantes. Se podrían crear más tablas, vistas, etc.. Debido al tiempo limitado, se ha optado por una simplificación del sistema al objeto de poder experimentar las características con mayor profundidad. Por todo ello se realizan las siguientes simplificaciones:

- Una serie solo puede estar en una plataforma
- Una serie solo puede tener un idioma
- Una serie solo puede tener un director.

En cambio, un actor puede participar “Cast” en varias series y una Serie tiene varios Actores. Por ello, la relación “n a m”, muchos a muchos, se transforma en una tabla “Cast” para dar soporte a las relaciones entre Actores y Series.

Las funcionalidades definitivas son:

- a. Creación y listado de **Plataformas**. Desde el listado se pueden borrar y editar las plataformas existentes.
- b. Creación y listado de **Idiomas**. Desde el listado se pueden borrar y editar los idiomas existentes.
- c. Creación y listado de **Directores**. Desde el listado se pueden borrar y editar los directores existentes.
- d. Creación y listado de **Actores**. Desde el listado se pueden borrar y editar los Actores existentes.
- e. Creación y listado de **Series**. Desde el listado se pueden borrar y editar los directores existentes.
- f. Creación y listado de **Relaciones entre Actores y Series**. Desde el listado se pueden borrar y editar los directores existentes.

Así mismo se establecen como requisitos no funcionales:

- Las **fechas se muestran en formato español**.
- Se ha de evitar que la creación de **registros duplicados**
- Se ha de evitar que se borren registros de entidades con **claves foráneas**.

3. Desarrollo conceptual

Igual que en la Actividad 1, Para comprender e ilustrar mejor los requisitos funcionales se realiza un diagrama de casos de uso:

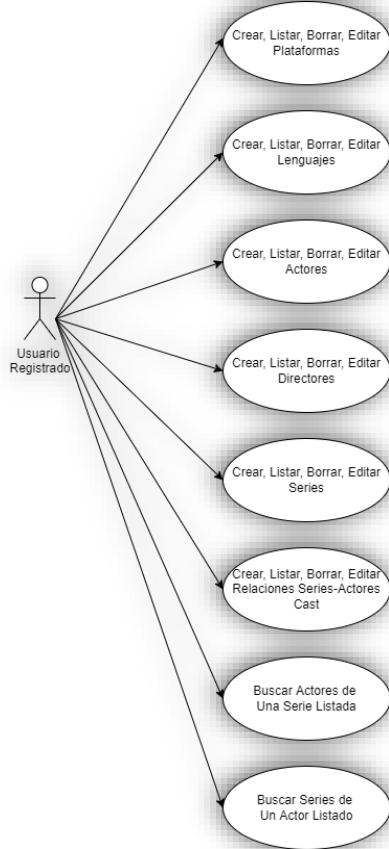


Figura 1. Diagrama de casos de uso.

CASO DE USO

- Mantenimiento de Plataformas.** El usuario puede crear nuevas plataformas, listar las existentes. Sobre el listado de las existentes puede editar o eliminar.
- Mantenimiento de Lenguajes.** El usuario puede crear nuevos lenguajes, listar los existentes. Sobre el listado de los existentes puede editar o eliminar.
- Mantenimiento de Actores.** El usuario puede crear nuevos Actores, listar los existentes. Sobre el listado de los existentes puede editar o eliminar.
- Mantenimiento de Directores.** El usuario puede crear nuevos Directores, listar los existentes. Sobre el listado de los existentes puede editar o eliminar.
- Mantenimiento de Series.** El usuario puede crear nuevas Series, listar las existentes. Sobre el listado de las existentes puede editar o eliminar.
- Mantenimiento de Relaciones Series-Actor.** El usuario puede crear nuevas Relaciones, listar las existentes. Sobre el listado de las existentes puede editar o eliminar.
- Búsqueda de Plataformas, Actores, etc.**

Se elabora también un **diagrama Entidad-Relación** con el fin de organizar y representar los datos de contenido relativos a cada una de las entidades principales y las relaciones existentes entre ellos:

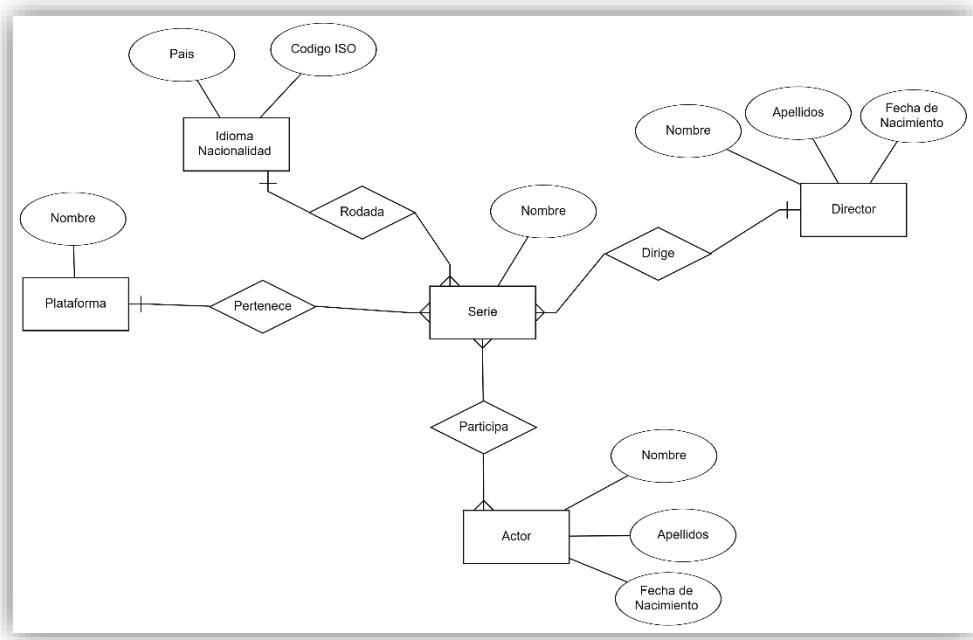
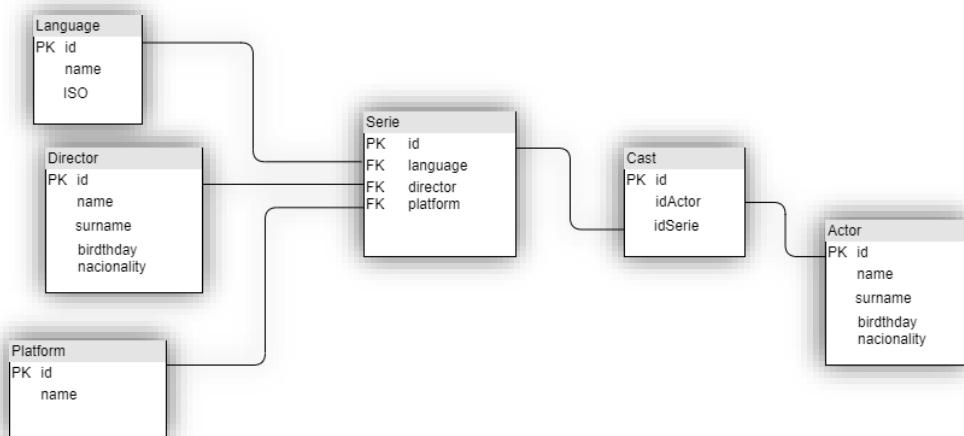


Figura 2. Diagrama Entidad – Relación.

El diagrama de clases de la solución adoptada es:



4.Implementación

Se incluye los detalles del proceso de implementación con Laravel 6 considerados más interesantes. Se ha desarrollado en entorno Windows, no obstante, se podría haber desarrollado perfectamente también en un entorno Linux.

El proceso de instalación se explica de manera simplificada en el fichero [readme.txt](#). En este documento se desarrolla más detalladamente con el apoyo de imágenes y con razonamientos más completos de las decisiones realizadas.

Instalación del entorno

Creación de la Base de datos (vacía)

Se ha creado una base de datos llamada " [actividad2](#)" .

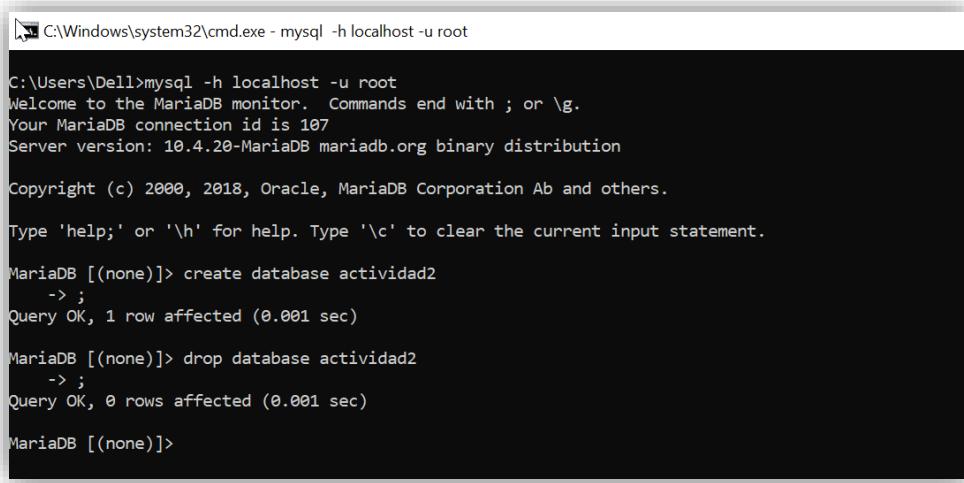
Se ha probado a crearla de dos formas:

1.- Por línea de comandos

```
mysql -h localhost -u root
```

Y después

```
create database actividad2
```



```
C:\Windows\system32\cmd.exe - mysql -h localhost -u root
C:\Users\Dell>mysql -h localhost -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 107
Server version: 10.4.20-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> create database actividad2
      -> ;
Query OK, 1 row affected (0.001 sec)

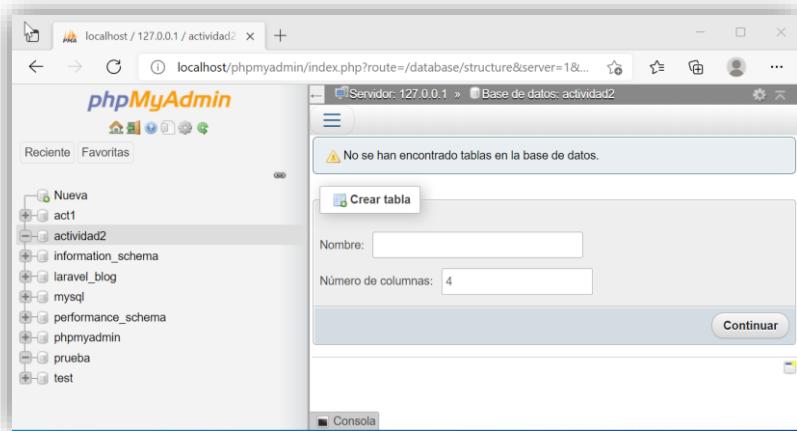
MariaDB [(none)]> drop database actividad2
      -> ;
Query OK, 0 rows affected (0.001 sec)

MariaDB [(none)]>
```

2.- Desde phpmyadmin



El resultado es una base de datos "actividad2" vacía



Verificación / instalación de composer y Laravel

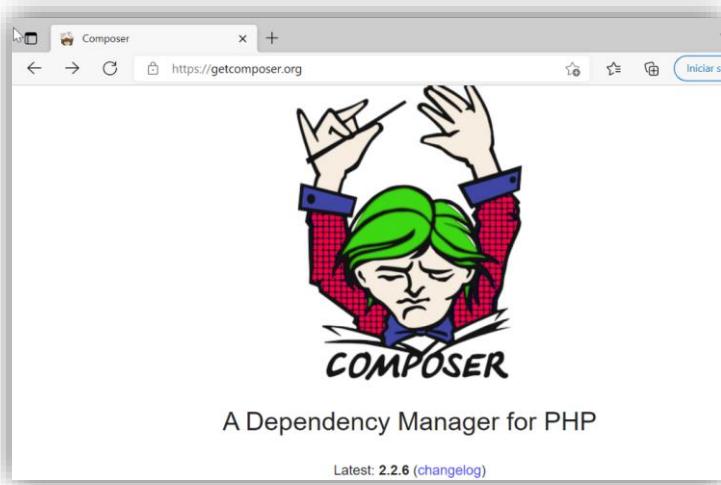
Compruebo que tengo instalado composer en el equipo donde estamos desarrollando la actividad.

Composer -V

```
C:\Windows\system32\cmd.exe
C:\Users\DELL>composer -V
Composer version 2.2.5 2022-01-21 17:25:52
```

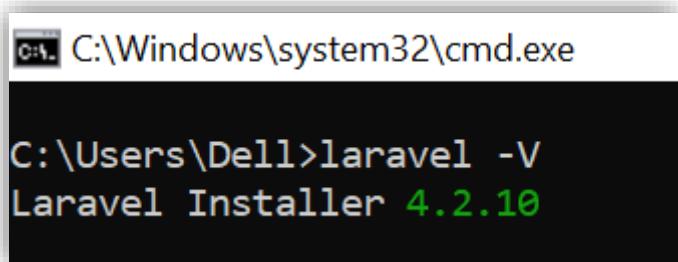
Si no lo tuviésemos, o quisiéramos actualizarlo iríamos a la web de composer:

<https://getcomposer.org>



Verifico la versión de Laravel que tengo instalado Laravel en el equipo de desarrollo

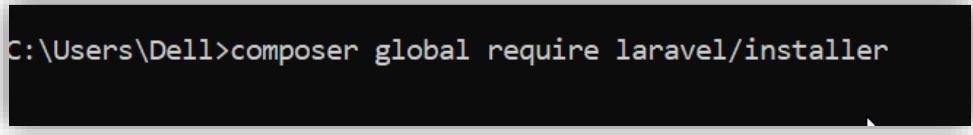
Laravel -V



```
C:\Windows\system32\cmd.exe
C:\Users\Dell>laravel -V
Laravel Installer 4.2.10
```

Si no lo tuviésemos, procederíamos a instalarlo vía composer

composer global require laravel/installer



```
C:\Users\Dell>composer global require laravel/installer
```

Creación del Proyecto” series”

Se puede utilizar de dos formas para crear el proyecto: por medio del comando

`laravel new nombredeproyecto`

```
C:\Users\DELL\laravel>laravel new series
```

O también a través de composer. Utilizamos el siguiente comando:

`composer create-project --prefer-dist laravel/laravel series "6.*"`

```
C:\Users\DELL\laravel>composer create-project --prefer-dist laravel/laravel series "6.*"
```

Se ha elegido realizarlo a través de composer tal y como se muestra en las siguientes figuras

```
C:\Users\DELL\laravel>composer create-project --prefer-dist laravel/laravel series "6.*"
Creating a "laravel/laravel" project at "./series"
Installing laravel/laravel (v6.20.1)
  - Installing laravel/laravel (v6.20.1): Extracting archive
Created project in C:\Users\DELL\laravel\series
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
Updating dependencies
Lock file operations: 93 installs, 0 updates, 0 removals
  - Locking doctrine/inflector (2.0.4)
  - Locking doctrine/instantiator (1.4.0)
  - Locking doctrine/lexer (1.2.2)
  - Locking dragonmantank/cron-expression (v2.3.1)
  - Locking egulias/email-validator (2.1.25)
  - Locking facade/flare-client-php (1.9.1)
  - Locking facade/ignition (1.18.0)
  - Locking facade/ignition-contracts (1.0.2)
  - Locking fakerphp/faker (v1.19.0)
  - Locking fideloper/proxy (4.4.1)
  - Locking filp/whoops (2.14.5)
  - Locking hamcrest/hamcrest-php (v2.0.1)
  - Locking laravel/framework (v6.20.44)
  - Locking laravel/tinker (v2.7.0)
  - Locking league/commonmark (1.6.7)
  - Locking league/flysystem (1.1.9)
  - Locking league/mime-type-detection (1.9.0)
  - Locking mockery/mockery (1.5.0)
  - Locking monolog/monolog (2.3.5)
  - Locking myclabs/deep-copy (1.10.2)
  - Locking nesbot/carbon (2.57.0)
  - Locking nikic/php-parser (v4.13.2)
  - Locking nunomaduro/collision (v3.2.0)
  - Locking opis/closure (3.6.3)
```

Termina satisfactoriamente

```
- Installing phar-io/version (3.1.1): Extracting archive
- Installing phar-io/manifest (2.0.3): Extracting archive
- Installing myclabs/deep-copy (1.10.2): Extracting archive
- Installing phpunit/phpunit (9.5.14): Extracting archive
83 package suggestions were added by new dependencies, use 'composer suggest' to see details.
Package swiftmailer/swiftmailer is abandoned, you should avoid using it. Use symfony/mail in instead.
Generating optimized autoload files
> Illuminate\Foundation\ComposerScripts::postAutoloadDump
> @php artisan package:discover --ansi
Discovered Package: facade/ignition
Discovered Package: fideloper/proxy
Discovered Package: laravel/tinker
Discovered Package: nesbot/carbon
Discovered Package: nunomaduro/collision
Package manifest generated successfully.
58 packages you are using are looking for funding.
Use the 'composer fund' command to find out more!
> @php artisan key:generate --ansi
Application key set successfully.

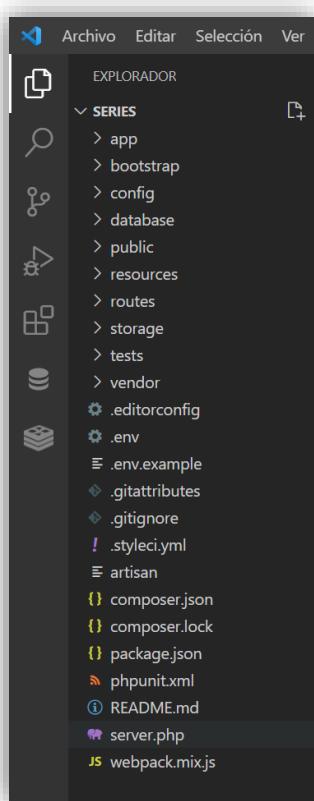
C:\Users\Dell\laravel>
```

Para saber la versión exacta de mi proyecto Laravel ejecuto en una terminal el siguiente comando

`php artisan -V`

```
C:\Users\Dell\laravel\series>php artisan -V
Laravel Framework 6.20.44
```

Abrimos con visual studio code el árbol de carpetas del proyecto para ver que todo esta correcto



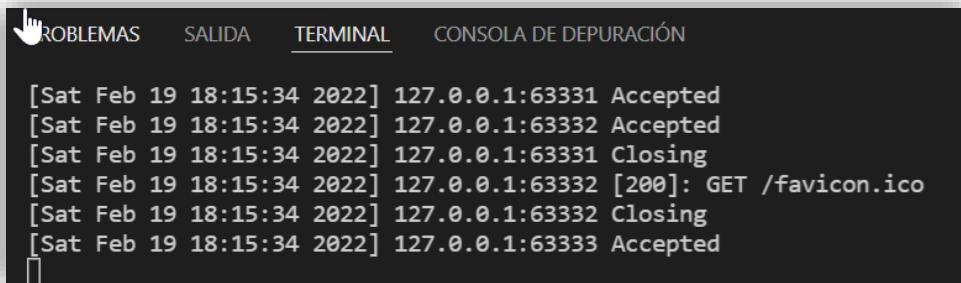
Lanzamos el servidor, y comprobamos que la página de bienvenida se carga correctamente

`php artisan serve`

```
PROBLEMAS SALIDA TERMINAL CONSOLA DE DEPURACIÓN
PS C:\Users\DELL\laravel\series> php artisan serve
Laravel development server started: http://127.0.0.1:8000
[Sat Feb 19 17:53:47 2022] PHP 8.0.9 Development Server (http://127.0.0.1:8000) started
```



En la terminal en la que levanto el servidor puedo observar las llamadas de la página



A screenshot of a terminal window titled "TERMINAL". The window contains the following log entries:

```
[Sat Feb 19 18:15:34 2022] 127.0.0.1:63331 Accepted
[Sat Feb 19 18:15:34 2022] 127.0.0.1:63332 Accepted
[Sat Feb 19 18:15:34 2022] 127.0.0.1:63331 Closing
[Sat Feb 19 18:15:34 2022] 127.0.0.1:63332 [200]: GET /favicon.ico
[Sat Feb 19 18:15:34 2022] 127.0.0.1:63332 Closing
[Sat Feb 19 18:15:34 2022] 127.0.0.1:63333 Accepted
```

Configuracion del Proyecto Series

En el fichero `.env` configuramos el nombre de la aplicación "series" y el nombre de la base de datos:"actividad2"

`DB_DATABASE=nombredelabasededatos.`

Dejo el entorno como viene por defecto: `APP_ENV=local` más adelante se deberá de cambiar a `APP_ENV=production`

```
⚙ .env
1 APP_NAME=series
2 APP_ENV=local
3 APP_KEY=base64:KiC0cMpd9GeRqUjIBU3pvORWskj2FGTII+teTebpqw0=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8
9 DB_CONNECTION=mysql
10 DB_HOST=127.0.0.1
11 DB_PORT=3306
12 DB_DATABASE=actividad2
13 DB_USERNAME=root
14 DB_PASSWORD=
15
```

También debo configurar el fichero `database.php` dentro del directorio `/config`

Cambio 'database' =>('DB_DATABASE', 'actividad2')

Cambio 'username' =>('DB_USERNAME', 'root')

```
SERIES          config > database.php
> app           35
> bootstrap     36   'connections' => [
  < config        37     'sqlite' => [
    < config        38       'driver' => 'sqlite',
    < config        39       'url' => env('DATABASE_URL'),
    < config        40       'database' => env('DB_DATABASE', database_path('database.sqlite')),
    < config        41       'prefix' => '',
    < config        42       'foreign_key_constraints' => env('DB_FOREIGN_KEYS', true),
    < config        43   ],
    < config        44   'mysql' => [
      < config        45     'driver' => 'mysql',
      < config        46     'url' => env('DATABASE_URL'),
      < config        47     'host' => env('DB_HOST', '127.0.0.1'),
      < config        48     'port' => env('DB_PORT', '3306'),
      < config        49     'database' => env('DB_DATABASE', 'actividad2'),
      < config        50     'username' => env('DB_USERNAME', 'root'),
      < config        51     'password' => env('DB_PASSWORD', ''),
      < config        52     'unix_socket' => env('DB_SOCKET', ''),
      < config        53     'charset' => 'utf8mb4',
      < config        54     'collation' => 'utf8mb4_unicode_ci',
      < config        55     'prefix' => '',
      < config        56     'prefix_indexes' => true,
      < config        57     'strict' => true,
      < config        58     'engine' => null,
      < config        59     'options' => extension_loaded('pdo_mysql') ? array_filter([
        < config        60       PDO::MYSQL_ATTR_SSL_CA => env('MYSQL_ATTR_SSL_CA')
      ]) : [],
      < config        61   ],
    < config        62   ],
    < config        63   ],
    < config        64   ],
    < config        65 ]
```

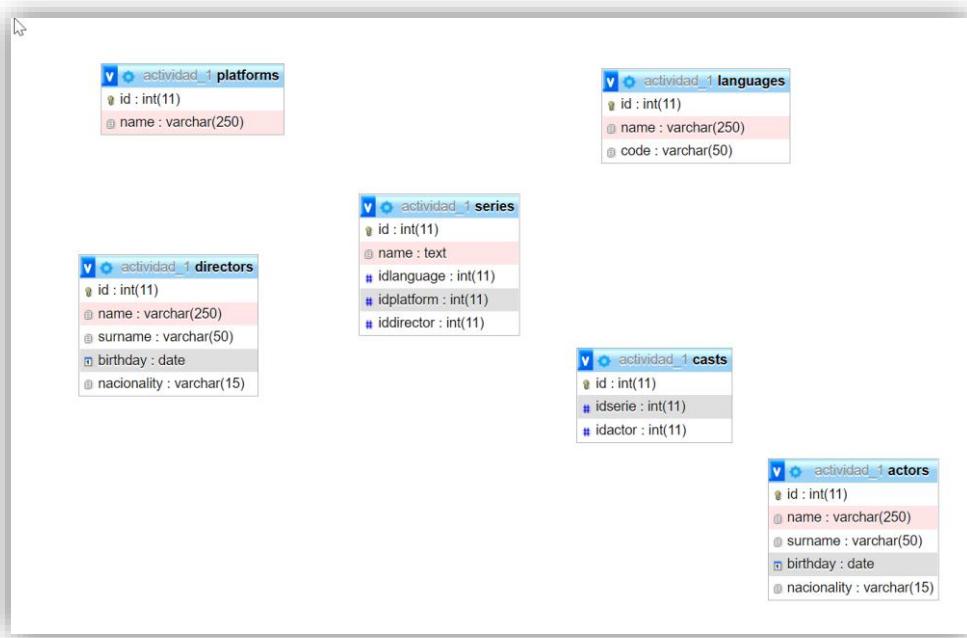
Migraciones – Creación de los datos de las tablas

Todo el proceso de creación de tablas se ha realizado través de "migraciones" incluso la creación del usuario por defecto.

Se crea un fichero de migración para una de las tablas.

Al lanzar las migraciones puedo comprobar si se han creado bien las conexiones en el apartado anterior. En caso de error, la aplicación nos lo mostrará.

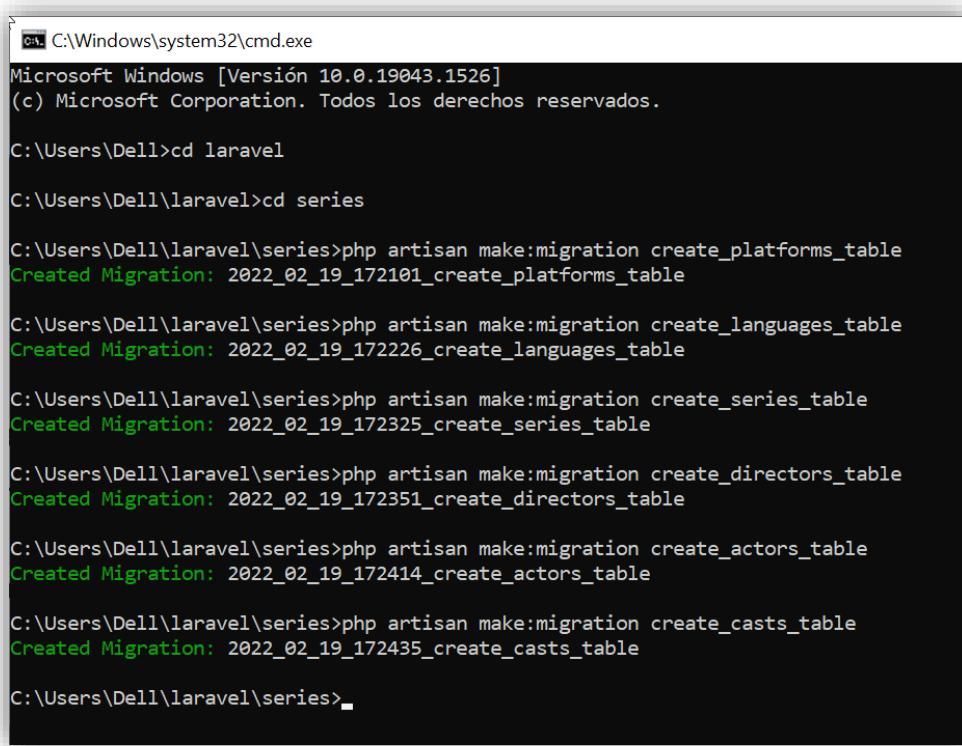
Para la creación de las tablas se ha usado como plantilla la misma estructura utilizada en la actividad 1. Se observa en la figura siguiente.



Se crean los archivos de migración con el siguiente comando de artisan:

```
php artisan make:migration Nombredelamigracion
```

Como son clases deben empezar el nombre debe empezar por letras mayúsculas y el fichero ha de llamarse igual que la clase



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Versión 10.0.19043.1526]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\DeLL>cd laravel
C:\Users\DeLL\laravel>cd series

C:\Users\DeLL\laravel\series>php artisan make:migration create_platforms_table
Created Migration: 2022_02_19_172101_create_platforms_table

C:\Users\DeLL\laravel\series>php artisan make:migration create_languages_table
Created Migration: 2022_02_19_172226_create_languages_table

C:\Users\DeLL\laravel\series>php artisan make:migration create_series_table
Created Migration: 2022_02_19_172325_create_series_table

C:\Users\DeLL\laravel\series>php artisan make:migration create_directors_table
Created Migration: 2022_02_19_172351_create_directors_table

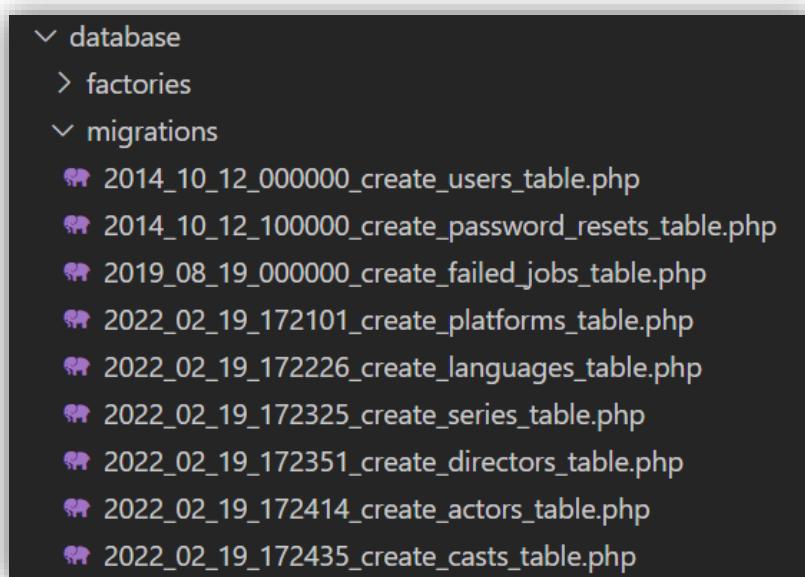
C:\Users\DeLL\laravel\series>php artisan make:migration create_actors_table
Created Migration: 2022_02_19_172414_create_actors_table

C:\Users\DeLL\laravel\series>php artisan make:migration create_casts_table
Created Migration: 2022_02_19_172435_create_casts_table

C:\Users\DeLL\laravel\series>_
```

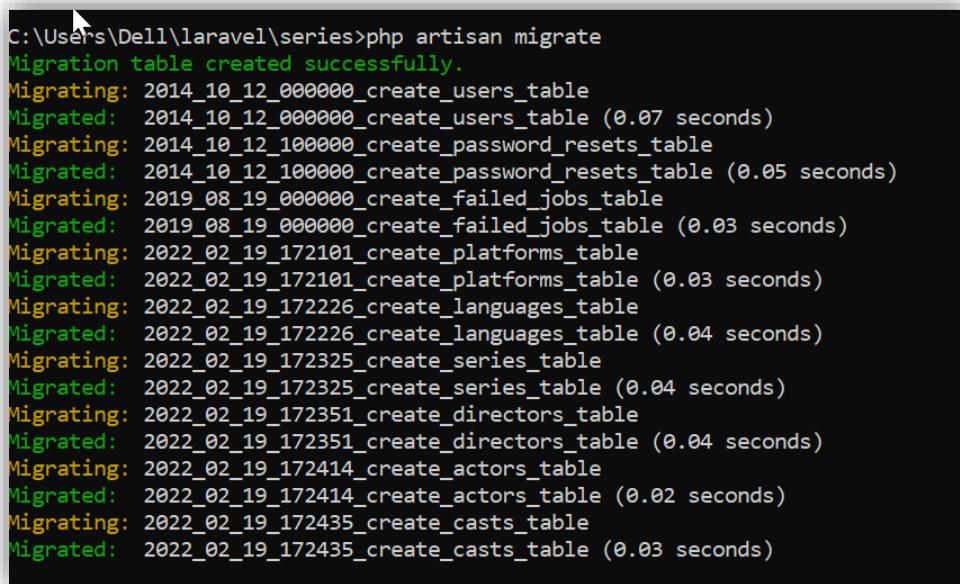
Los ficheros son creados automáticamente en la ruta [/database/migrations](#).

Los tres primeros se habían creado automáticamente junto a la creación del proyecto, los otros siguientes, son los que se acaban de generar con el comando anterior



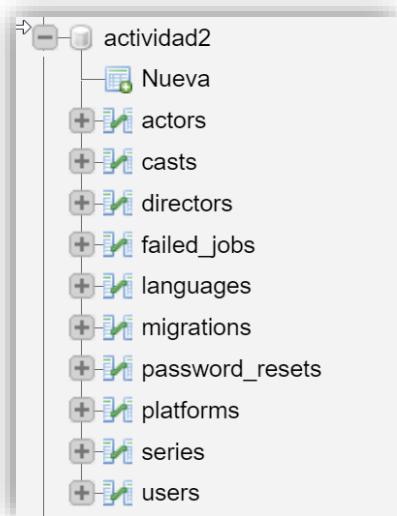
Ejecuto las migraciones (Todas juntas) observando que la conexión con la base de datos es satisfactoria con el siguiente comando de artisan:

```
php artisan migrate
```



```
C:\Users\DELL\laravel\series>php artisan migrate
Migration table created successfully.
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table (0.07 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: 2022_02_19_172101_create_platforms_table
Migrated: 2022_02_19_172101_create_platforms_table (0.03 seconds)
Migrating: 2022_02_19_172226_create_languages_table
Migrated: 2022_02_19_172226_create_languages_table (0.04 seconds)
Migrating: 2022_02_19_172325_create_series_table
Migrated: 2022_02_19_172325_create_series_table (0.04 seconds)
Migrating: 2022_02_19_172351_create_directors_table
Migrated: 2022_02_19_172351_create_directors_table (0.04 seconds)
Migrating: 2022_02_19_172414_create_actors_table
Migrated: 2022_02_19_172414_create_actors_table (0.02 seconds)
Migrating: 2022_02_19_172435_create_casts_table
Migrated: 2022_02_19_172435_create_casts_table (0.03 seconds)
```

Compruebo el resultado viendo las tablas creadas en la base de datos con phpmyadmin



Observo también el contenido de la tabla `migrations` para asegurarme que todo está correctamente.

SELECT * FROM `migrations`						
<input type="checkbox"/> Perfilando [Editar en línea] [Editar] [Explicar SQL] [Crear código PHP] [Actualizar]						
<input type="checkbox"/> Mostrar todo	Número de filas:	25	<input type="button" value="▼"/>	Filtrar filas:	Buscar en esta tabla	Sort by key
+ Opciones						
<input type="checkbox"/>	<input type="button" value="←"/>	<input type="button" value="→"/>	<input type="button" value="▼"/>	id	migration	batch
<input type="checkbox"/>	<input type="button" value="Editar"/>	<input type="button" value="Copiar"/>	<input type="button" value="Borrar"/>	10	2014_10_12_000000_create_users_table	1
<input type="checkbox"/>	<input type="button" value="Editar"/>	<input type="button" value="Copiar"/>	<input type="button" value="Borrar"/>	11	2014_10_12_100000_create_password_resets_table	1
<input type="checkbox"/>	<input type="button" value="Editar"/>	<input type="button" value="Copiar"/>	<input type="button" value="Borrar"/>	12	2019_08_19_000000_create_failed_jobs_table	1
<input type="checkbox"/>	<input type="button" value="Editar"/>	<input type="button" value="Copiar"/>	<input type="button" value="Borrar"/>	13	2022_02_19_172101_create_platforms_table	1
<input type="checkbox"/>	<input type="button" value="Editar"/>	<input type="button" value="Copiar"/>	<input type="button" value="Borrar"/>	14	2022_02_19_172226_create_languages_table	1
<input type="checkbox"/>	<input type="button" value="Editar"/>	<input type="button" value="Copiar"/>	<input type="button" value="Borrar"/>	15	2022_02_19_172325_create_series_table	1
<input type="checkbox"/>	<input type="button" value="Editar"/>	<input type="button" value="Copiar"/>	<input type="button" value="Borrar"/>	16	2022_02_19_172351_create_directors_table	1
<input type="checkbox"/>	<input type="button" value="Editar"/>	<input type="button" value="Copiar"/>	<input type="button" value="Borrar"/>	17	2022_02_19_172414_create_actors_table	1
<input type="checkbox"/>	<input type="button" value="Editar"/>	<input type="button" value="Copiar"/>	<input type="button" value="Borrar"/>	18	2022_02_19_172435_create_casts_table	1

Deshago las migraciones. Para ello, como únicamente se ha lanzado un paso de migrate, batch=1 con un solo paso de rollback elimino la totalidad de las tablas creadas.

```
php artisan migrate:rollback
```

```
C:\Users\DELL\laravel\series>php artisan migrate:rollback
Rolling back: 2022_02_19_172435_create_casts_table
Rolled back: 2022_02_19_172435_create_casts_table (0.02 seconds)
Rolling back: 2022_02_19_172414_create_actors_table
Rolled back: 2022_02_19_172414_create_actors_table (0.01 seconds)
Rolling back: 2022_02_19_172351_create_directors_table
Rolled back: 2022_02_19_172351_create_directors_table (0.01 seconds)
Rolling back: 2022_02_19_172325_create_series_table
Rolled back: 2022_02_19_172325_create_series_table (0.01 seconds)
Rolling back: 2022_02_19_172226_create_languages_table
Rolled back: 2022_02_19_172226_create_languages_table (0.01 seconds)
Rolling back: 2022_02_19_172101_create_platforms_table
Rolled back: 2022_02_19_172101_create_platforms_table (0.01 seconds)
Rolling back: 2019_08_19_000000_create_failed_jobs_table
Rolled back: 2019_08_19_000000_create_failed_jobs_table (0.01 seconds)
Rolling back: 2014_10_12_100000_create_password_resets_table
Rolled back: 2014_10_12_100000_create_password_resets_table (0.01 seconds)
Rolling back: 2014_10_12_000000_create_users_table
Rolled back: 2014_10_12_000000_create_users_table (0.01 seconds)
```

Compruebo el resultado en phpmyadmin. Únicamente ha quedado la tabla migrate pero sin registros.



Modifico desde visual studio los archivos de migración creados en la carpeta **database** para introducir los campos requeridos de cada tabla.

Este proceso de crear y borrar, lo repetiré varias veces hasta que no haya errores y todas las tablas queden con la estructura que deseada.

Depurando los errores que se cometan. Por ejemplo este:

```
Migrated: 2022_02_19_172226_create_languages_table (0.03 seconds)
Migrating: 2022_02_19_172305_create_genres_table (0.02 seconds)
Migrated: 2022_02_19_172325_create_series_table (0.02 seconds)
Migrating: 2022_02_19_172351_create_directors_table (0.03 seconds)
Migrated: 2022_02_19_172351_create_genres_table (0.03 seconds)
Migrated: 2022_02_19_172414_create_actors_table (0.04 seconds)
Migrated: 2022_02_19_172435_create_casts_table (0.04 seconds)

BadMethodCallException : Method Illuminate\Database\Schema\Blueprint::int does not exist.

at C:\Users\williamelserie\vendor\laravel\framework\src\Illuminate\Support\Traits\Macroable.php:103
  99|     / *
100|     public function __call($method, $parameters)
101|     {
102|         if (! static::hasMacro($method))
103|             throw new BadMethodCallException(sprintf(
104|                 "Method %s::%s does not exist.", static::class, $method
105|             ));
106|     }
107| }

Exception trace:
  1  Illuminate\Database\Schema\Blueprint::__call("int")
      C:\Users\williamelserie\series\database\migrations\2022_02_19_172435_create_casts_table.php:18
  2  CreateCastsTable::(closure)(Object(Illuminate\Database\Schema\Blueprint))
      C:\Users\williamelserie\vendor\laravel\framework\src\Illuminate\Database\Builder.php:166

Please use the argument -v to see more details.
```

Se crean muchos campos con el modificador `->nullable()`; (Permitir valores nulos) para poder realizar un desarrollo más rápido. En producción, muchos campos, no estará permitido ni por el código de la aplicación código ni por las condiciones impuestas a la base de datos.

Por igual motivo, tampoco se establecen relaciones entre campos de las tablas. Quedando dicha responsabilidad al principio, en el propio código empleado. Las relaciones quedan establecidas en la Lógica de la propia aplicación. En desarrollos posteriores se pueden definir relaciones en la base de datos y capturar los errores producidos en los borrados y actualizaciones.

Detalle a modo de ejemplo tres de las migraciones creadas

create_series_table

```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateSeriesTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('series', function (Blueprint $table) {
            $table->bigIncrements('id');
            $table->string('name')->nullable();
            $table->biginteger('idlanguage')->nullable();
            $table->biginteger('idplatform')->nullable();
            $table->biginteger('iddirector')->nullable();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('series');
    }
}
```

create directors_table

```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateDirectorsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('directors', function (Blueprint $table) {
            $table->bigIncrements('id');
            $table->string('name')->nullable();
            $table->string('surname')->nullable();
            $table->date('birthday')->nullable();
            $table->string('nacionality')->nullable();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('directors');
    }
}
```

Y create_casts_table

```
<?php

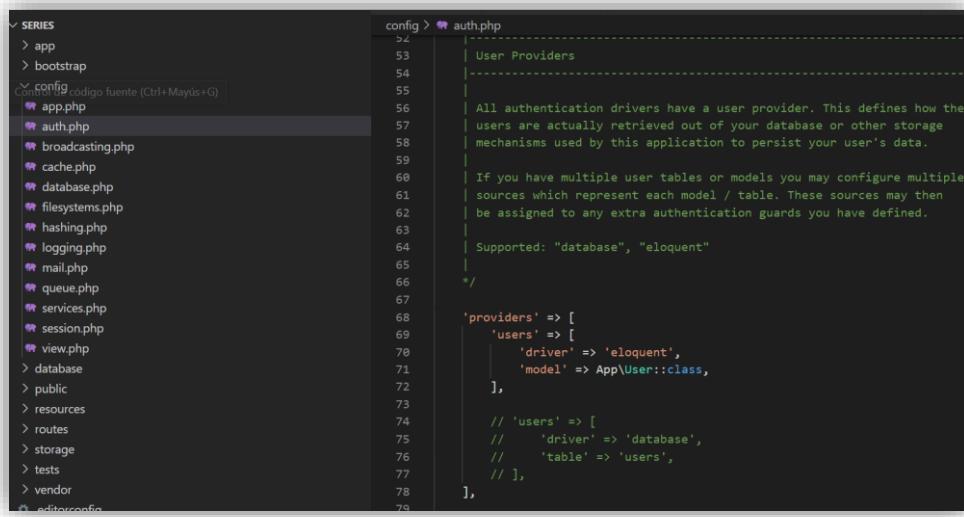
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateCastsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('casts', function (Blueprint $table) {
            $table->bigIncrements('id');
            $table->biginteger('idserie')->nullable();
            $table->biginteger('idactor')->nullable();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('casts');
    }
}
```

Creación de los modelos

En la versión 8 se ubican automáticamente en una carpeta llamada Models. Podríamos definir esta ubicación en Laravel 6 y establecerla en el fichero `auth.php`. No obstante, para evitar problemas inesperados, vamos a dejar que cree los modelos en la carpeta que por defecto establece Laravel 6. Es decir, directamente en la carpeta `app`



```
config > auth.php
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
```

```
User Providers
-----
All authentication drivers have a user provider. This defines how the
users are actually retrieved out of your database or other storage
mechanisms used by this application to persist your user's data.

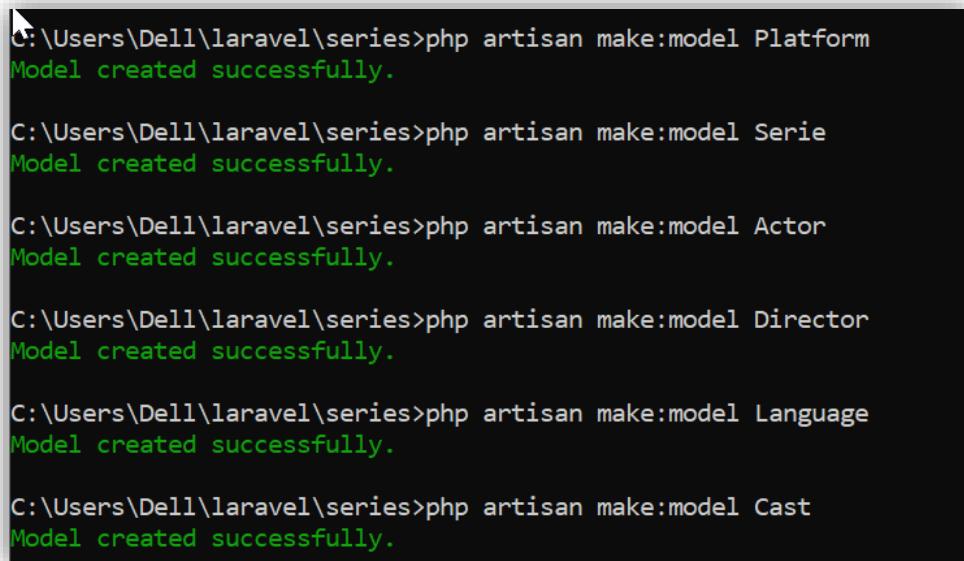
If you have multiple user tables or models you may configure multiple
sources which represent each model / table. These sources may then
be assigned to any extra authentication guards you have defined.

Supported: "database", "eloquent"

providers => [
    'users' => [
        'driver' => 'eloquent',
        'model' => App\User::class,
    ],
    // 'users' => [
    //     'driver' => 'database',
    //     'table' => 'users',
    // ],
],
```

Los modelos, por comodidad, los creamos con el comando de artisan

```
php artisan make:model Nombremodelo
```



```
C:\Users\DELL\laravel\series>php artisan make:model Platform
Model created successfully.

C:\Users\DELL\laravel\series>php artisan make:model Serie
Model created successfully.

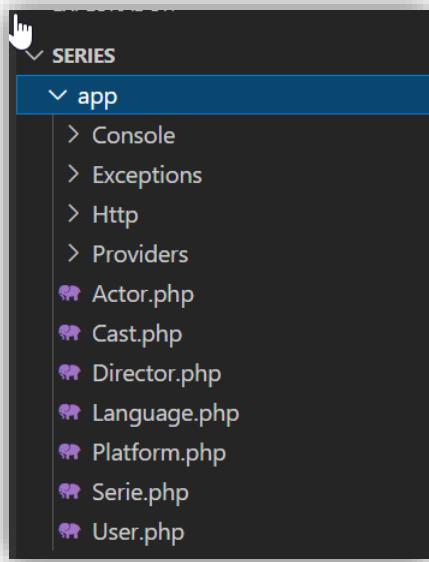
C:\Users\DELL\laravel\series>php artisan make:model Actor
Model created successfully.

C:\Users\DELL\laravel\series>php artisan make:model Director
Model created successfully.

C:\Users\DELL\laravel\series>php artisan make:model Language
Model created successfully.

C:\Users\DELL\laravel\series>php artisan make:model Cast
Model created successfully.
```

Detalle de los seis modelos creados y su ubicación en el proyecto



Dentro de cada modelo creado, tiene el contenido recién creado es similar a este:

```
1 <?php
2
3 namespace App;
4
5 use Illuminate\Database\Eloquent\Model;
6
7 class Platform extends Model
8 {
9     //
10 }
```

Existe una forma de generar a la vez la migracion y el modelo, pero en nuestro caso no la hemos utilizado. Seria así:

```
php artisan make:model Nombredelmodelo --migrate
```

Creación de los Controladores:

Su misión fundamental es poder acceder a los modelos y devolverlos a las vistas.

Existen varias formas de crearlos, desde cero con el IDE empleado o con utilizando diversos comandos Artisan de consola.

Se crean dentro de la carpeta app\http\controllers.

Se pueden organizar de distinta forma agrupando por entidades, por entidades y métodos, etc. En nuestro caso lo he agrupado por entidad. Un controlador para todo lo que se realiza sobre las plataformas, otro para todas las acciones sobre actores, etc.

Hemos usado el comando

Php artisan make: controller NombredelControlador –resource ya que implementa todos los métodos básicos CRUD

```
C:\Users\DELL\laravel\series>php artisan make:controller PlatformController --resource  
Controller created successfully.
```

Esto es lo que nos crea:

```
app > Http > Controllers > PlatformController.php > ...  
1  <?php  
2  
3  namespace App\Http\Controllers;  
4  
5  use Illuminate\Http\Request;  
6  
7  class PlatformController extends Controller  
8  {  
9      /**  
10      * Display a listing of the resource.  
11      *  
12      * @return \Illuminate\Http\Response  
13      */  
14      public function index()  
15      {  
16          //  
17      }  
18  
19      /**  
20      * Show the form for creating a new resource.  
21      *  
22      * @return \Illuminate\Http\Response  
23      */  
24      public function create()  
25      {  
26          //  
27      }  
28  
29      /**  
30      * Store a newly created resource in storage.  
31      *  
32      * @param \Illuminate\Http\Request $request  
33      * @return \Illuminate\Http\Response  
34      */  
35      public function store(Request $request)  
36      {  
37          //  
38      }  
39
```

```
39     /**
40      * Display the specified resource.
41      *
42      * @param int $id
43      * @return \Illuminate\Http\Response
44      */
45     public function show($id)
46     {
47         //
48     }
49
50     /**
51      * Show the form for editing the specified resource.
52      *
53      * @param int $id
54      * @return \Illuminate\Http\Response
55      */
56     public function edit($id)
57     {
58         //
59     }
60
61     /**
62      * Update the specified resource in storage.
63      *
64      * @param \Illuminate\Http\Request $request
65      * @param int $id
66      * @return \Illuminate\Http\Response
67      */
68     public function update(Request $request, $id)
69     {
70         //
71     }
72
73     /**
74      * Remove the specified resource from storage.
75      *
76      * @param int $id
77      * @return \Illuminate\Http\Response
78      */
79     public function destroy($id)
80     {
81         //
82     }
83 }
84
85 }
```

Eliminando comentarios y estableciendo los métodos del controlador para que actúen de forma similar a la práctica 1, quedaría así.

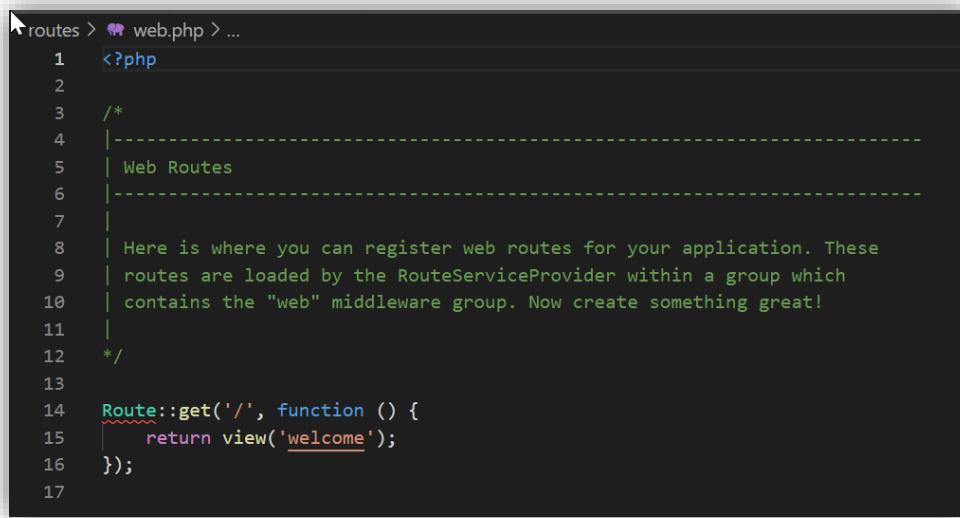
```
app > Http > Controllers > PlatformController.php > PHP Intelephense > PlatformController
  2
  3 namespace App\Http\Controllers;
  4
  5 use App\Platform;
  6 use Illuminate\Http\Request;
  7
  8 class PlatformController extends Controller
  9 {
10
11     public function index()
12     {
13         //
14     }
15
16     public function create()
17     {
18         //
19     }
20
21     public function store(Request $request)
22     {
23         //
24     }
25
26     public function edit(Platform $platform)
27     {
28         //
29     }
30
31     public function update(Request $request, Platform $platform)
32     {
33         //
34     }
35
36     public function delete(Request $request, Platform $platform)
37     {
38         //
39     }
40 }
41
```

Creación de las rutas

Las rutas son utilizadas normalmente para devolver una vista o para llamar a un método de un controlador que realizará a su vez acciones con la base de datos y/o con una vista.

El fichero `web.php` contiene todas las rutas. Al crearse un nuevo proyecto tiene este contenido (Una sola ruta creada):

Por defecto al entrar "/" te lleva a `/welcome`



```
routes > 🌐 web.php > ...
1  <?php
2
3  /*
4  | -----
5  | Web Routes
6  | -----
7  |
8  | Here is where you can register web routes for your application. These
9  | routes are loaded by the RouteServiceProvider within a group which
10 | contains the "web" middleware group. Now create something great!
11 |
12 */
13
14 Route::get('/', function () {
15     return view('welcome');
16 });
17
```

Añadiremos rutas a lo largo de la actividad para cada uno de los requisitos a implementar

Para cada tabla, agrupamos las distintas rutas (Listar, crear, guardar, editar y borrar) utilizando prefijos.

Ejemplo de las rutas creadas sobre la entidad Platform:

El Método get o post te lleva al controlador **PlatformController** y dentro de él, al método **index**. (Si hubiésemos creado controladores autoinvocables no necesitaríamos definir el método puesto que se ejecutaría el definido como **__invoke**).

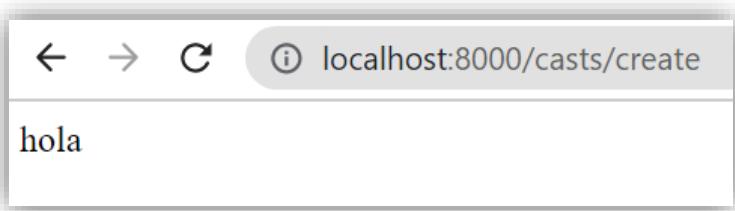
A esta ruta le llamamos **platforms.index**

La cuarta ruta **{platform}/edit** e estoy pasando un parámetro. Igual con update y delete.

```
Route::prefix('platforms')->group(function()
{
    Route::match(['get', 'post'], '/', 'PlatformController@index')->name('platforms.index'); // Listado
    Route::get('/create', 'PlatformController@create')->name('platforms.create'); // creacion
    Route::post('/store', 'PlatformController@store')->name('platforms.store'); // guardado
    Route::get('{platform}/edit', 'PlatformController@edit')->name('platforms.edit');// edicion
    Route::post('{platform}/update', 'PlatformController@update')->name('platforms.update');//guardado
    Route::delete('{platform}/delete', 'PlatformController@delete')->name('platforms.delete'); //borrado
});
```

Para comprobar si están funcionando bien las rutas realizo una prueba sobre el método create del controlador cast

```
public function create()
{
    echo ("hola");
}
```



Esto mismo lo realizo con las seis tablas (entidades) del proyecto.

Para comprobar que se accede a los datos de las tablas utilizando el modelo realizo la siguiente prueba : Introduzco dos actores desde phpmyadmin y coloco el siguiente código en el controlador

```
dd(Actor::all());
```

```
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use App\Actor;
6 use Illuminate\Http\Request;
7
8 class ActorController extends Controller
9 {
10
11     public function index()
12     {
13         dd( Actor::all() );
14     }
15 }
```

El resultado es satisfactorio porque nos devuelve dos objetos (actores)



```
Illuminate\Database\Eloquent\Collection {#303 ▶
  #items: array:2 [▶
    0 => App\Actor {#304 ▶
      #table: "actors"
      #connection: "mysql"
      #primaryKey: "id"
      #keyType: "int"
      +incrementing: true
      #with: []
      #withCount: []
      #perPage: 15
      +exists: true
      +wasRecentlyCreated: false
      #attributes: array:7 [▶
        "id" => 1
        "name" => "antonio"
        "surname" => "banderas"
        "birthday" => null
        "nacionality" => null
        "created_at" => null
        "updated_at" => null
      ]
      #original: array:7 [▶]
      #changes: []
      #casts: []
      #dates: []
      #dateFormat: null
      #appends: []
      #dispatchesEvents: []
      #observables: []
      #relations: []
      #touches: []
      +timestamps: true
      #hidden: []
      #visible: []
      #fillable: []
      #guarded: array:1 [▶]
    ]
    1 => App\Actor {#305 ▶
      #table: "actors"
      #connection: "mysql"
      #primaryKey: "id"
      #keyType: "int"
      +incrementing: true
      #with: []
      #withCount: []
      #perPage: 15
      +exists: true
      +wasRecentlyCreated: false
      #attributes: array:7 [▶
        "id" => 3
        "name" => "Penelope"
        "surname" => "Cruz"
        "birthday" => "1969-01-01"
        "nacionality" => "ES"
        "created_at" => null
        "updated_at" => null
      ]
      #original: array:7 [▶]
      #changes: []
      #casts: []
      #dates: []
      #dateFormat: null
      #appends: []
      #dispatchesEvents: []
      #observables: []
      #relations: []
      #touches: []
      +timestamps: true
      #hidden: []
      #visible: []
      #fillable: []
      #guarded: array:1 [▶]
    }
  ]
}
```

Creación de vistas utilizando Blade

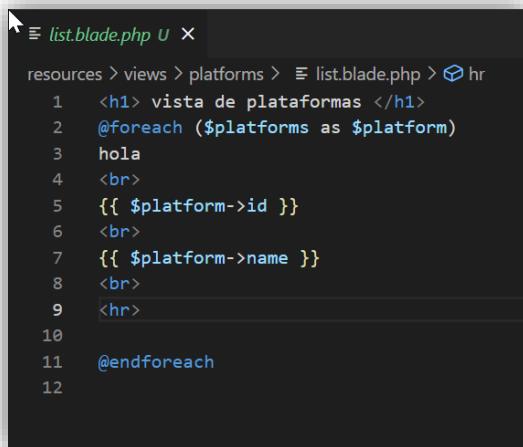
En este caso no se ha utilizado comandos php artisan para que las cree automáticamente

El controlador de plataformas tiene el siguiente método index():

```
public function index()
{
    $platforms = Platform::paginate(10); // podríamos poner Platform::all();
    return view('platforms.list', ['platforms' => $platforms]);
}
```

Dentro de **resources**, **vistas**, creo una carpeta para las vistas de cada tabla.

En dicha carpeta creo el fichero **list.blade.php** con el siguiente código:



```
list.blade.php U x
resources > views > platforms > list.blade.php > hr
1  <h1> vista de plataformas </h1>
2  @foreach ($platforms as $platform)
3      hola
4      <br>
5      {{ $platform->id }}
6      <br>
7      {{ $platform->name }}
8      <br>
9      <hr>
10
11  @endforeach
12
```

Tras introducir desde phpmyadmin un par de plataformas pruebo a ver si la vista me devuelve los registros.

El resultado es satisfactorio



vista de plataformas

hola
1
Netflix

hola
2
Prime

Llegados a este punto, vemos que todo funciona correctamente, modelos, controladores, rutas, vistas.

Autenticación

Hemos usado la autenticación que implementa por defecto Laravel por medio del paquete `laravel/ui`.

Podríamos haberlo realizado al comienzo, junto con la creación del proyecto desde cero con el comando:

```
Laravel new nombredeproyecto -- auth
```

En nuestro caso, con el proyecto ya creado, ejecutamos dos comandos:

```
composer require laravel/ui "^1.0" -- dev
```

Y a continuación el comando

```
php artisan ui vue -- auth.
```

Detenemos primeramente el servidor para evitar posibles problemas

```
C:\Users\DELL\laravel\series>composer require laravel/ui "^1.0" --dev
./composer.json has been updated
Running composer update laravel/ui
Loading composer repositories with package information
Updating dependencies
Lock file operations: 1 install, 0 updates, 0 removals
  - Locking laravel/ui (v1.3.0)
Writing lock file
Installing dependencies from lock file (including require-dev)
Package operations: 1 install, 0 updates, 0 removals
  - Downloading laravel/ui (v1.3.0)
  - Installing laravel/ui (v1.3.0): Extracting archive
Package swiftmailer/swiftmailer is abandoned, you should avoid using it. Use symfony/mailer instead.
Generating optimized autoload files
> Illuminate\Foundation\ComposerScripts::postAutoloadDump
> @php artisan package:discover --ansi
Discovered Package: facade/ignition
Discovered Package: fideloper/proxy
Discovered Package: laravel/tinker
Discovered Package: laravel/ui
Discovered Package: nesbot/carbon
Discovered Package: nunomaduro/collision
Package manifest generated successfully.
68 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
```

```
C:\Users\DELL\laravel\series>php artisan ui vue --auth
Vue scaffolding installed successfully.
Please run "npm install && npm run dev" to compile your fresh scaffolding.
Authentication scaffolding generated successfully.
```

Nos solicita que ejecutemos `npm install && npm run dev` y así se ha procedido

```
C:\Users\DELL\laravel\series>npm install && npm run dev
npm WARN deprecated source-map-url@0.4.1: See https://github.com/lydell/source-map-url#deprecated
npm WARN deprecated urix@0.1.0: Please see https://github.com/lydell/urix#deprecated
npm WARN deprecated source-map-resolve@0.5.3: See https://github.com/lydell/source-map-resolve#deprecated
npm WARN deprecated chokidar@2.1.8: Chokidar 2 does not receive security updates since 2019. Upgrade to chokidar 3 with
15x fewer dependencies
npm WARN deprecated resolve-url@0.2.1: https://github.com/lydell/resolve-url#deprecated
npm WARN deprecated querystring@0.2.0: The querystring API is considered Legacy. new code should use the URLSearchParams
API instead.
npm WARN deprecated uid@3.4.0: Please upgrade to version 7 or higher. Older versions may use Math.random() in certain
circumstances, which is known to be problematic. See https://v8.dev/blog/math-random for details.
npm WARN deprecated axios@0.19.2: Critical security vulnerability fixed in v0.21.1. For more information, see https://gi
thub.com/axios/axios/pull/3410
npm WARN deprecated svgo@1.3.2: This SVGO version is no longer supported. Upgrade to v2.x.x.
npm WARN deprecated popper.js@1.16.1: You can find the new Popper v2 at @popperjs/core, this package is dedicated to the
legacy v1
```

```
DONE Compiled successfully in 10887ms
0:32:00
Asset      Size  Chunks     Chunk Names
/css/app.css 180 Kib  /js/app [emitted]  /js/app
/js/app.js   1.4 MiB  /js/app [emitted]  /js/app
Installing shortcut: "C:\Users\DELL\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\SnoreToast\0.7.0\SnoreToast.lnk" "C:\Users\DELL\laravel\series\node_modules\node-notifier\vendor\snoreToast\snoretoast-x64.exe" Snore.DesktopToasts.0.7.0
C:\Users\DELL\laravel\series>
```

A través de GIT podemos ver todo lo que ha cambiado en el proyecto al añadir la autenticación.

File	Type	Status
composer.json	{} composer.json	M
composer.lock	{} composer.lock	M
package-lock.json	{} package-lock.json	U
package.json	{} package.json	M
webpack.mix.js	JS webpack.mix.js	M
HomeController.php	PHP HomeController.php	U
mix-manifest.json	{} mix-manifest.json	U
app.css	# app.css	U
app.js	JS app.js	U
app.js	JS app.js	M
bootstrap.js	JS bootstrap.js	M
ExampleComponent.vue	ExampleComponent.vue	U
_variables.scss	? _variables.scss	U
app.scss	? app.scss	M
home.blade.php	!! home.blade.php	U
login.blade.php	!! login.blade.php	U
register.blade.php	!! register.blade.php	U
verify.blade.php	!! verify.blade.php	U
confirm.blade.php	!! confirm.blade.php	U
email.blade.php	!! email.blade.php	U
reset.blade.php	!! reset.blade.php	U
app.blade.php	!! app.blade.php	U
web.php	!! web.php	M
routes	routes	

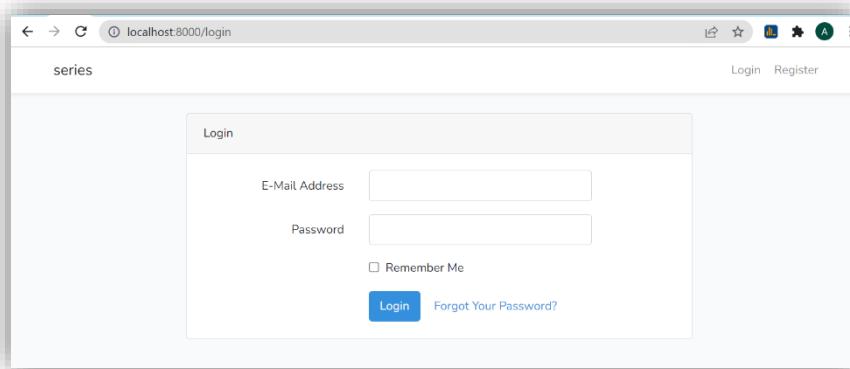
Entre otras modificaciones, ha creado vistas y ha añadido nueva ruta al fichero web.php

```
Auth::routes();  
  
Route::get('/home', 'HomeController@index')->name('home');
```

Y en el fichero de rutas api.php ya estaba creado

```
Route::middleware('auth:api')->get('/user', function (Request $request) {  
    return $request->user();  
});
```

Pruebo la nueva vista de autentificación: localhost:8000/home



Pruebo a registrarme con éxito

Los datos de prueba con los que me registro son:

Nombre: no lo usa en la validación

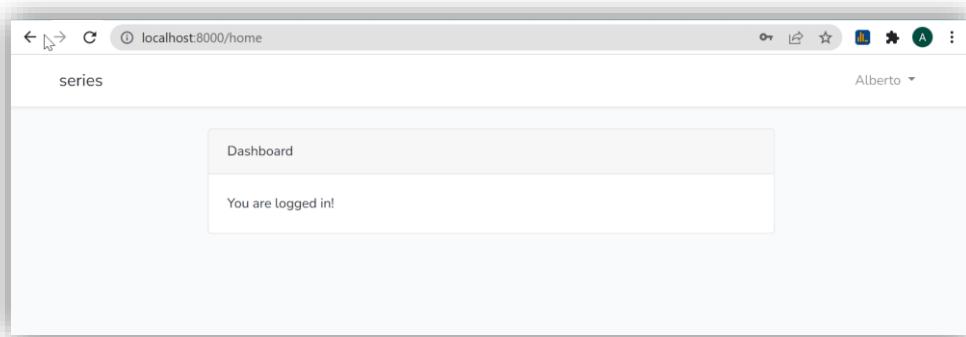
Email: correo informatico11540@gmail.com

Contraseña: abcdefghi

Estos datos se introducen utilizando un seeder

Register

Name	Alberto
E-Mail Address	informatico11540@gmail.com
Password	***** ⓘ The password must be at least 8 characters.
Confirm Password	*****
Register	



A través de Phpmyadmin comprobamos que no guarda la contraseña sino el hash de la contraseña.

	id	name	email	email_verified_at	password	remember_token	created_at	updated_at
□	1	Alberto	informatico11540@gmail.com	NULL	\$2y\$10\$gwJls2GHIJU.8aarkdJtL0xtSz2JqPr2hC5fO1feRi4...	NULL	2022-02-19 23:47:11	2022-02-19 23:47:11

Como requisito se establece bloquear el acceso a todas las entidades menos la de plataformas. Esto lo realizamos a través del enrutado modificando el fichero web.php

Se añade `->middleware('auth');` a las rutas en las que tenemos que estar autenticados

```
Route::prefix('languages')->group(function()
{
    Route::get('/', '_LanguageController@index')->name('languages.index')->middleware('auth'); // Listado
    Route::get('/create', '_LanguageController@create')->name('languages.create')->middleware('auth'); // creacion
    Route::post('/store', '_LanguageController@store')->name('languages.store')->middleware('auth'); // guardado
    Route::get('/{language}/edit', '_LanguageController@edit')->name('languages.edit')->middleware('auth');// edicion
    Route::post('/{language}/update', '_LanguageController@update')->name('languages.update')->middleware('auth');//guardado
    Route::delete('/{language}/delete', '_LanguageController@delete')->name('languages.delete')->middleware('auth'); // eliminacion
});
```

Muchos parámetros relacionados con la autentificación pueden modificarse en el fichero

Auth.php dentro de la carpeta **app/config**

A través de un comando puedo ver también un resumen de todas las rutas establecidas hasta el

Momento.

Domain	Method	URI	Name	Action	Middleware
	GET HEAD	/		Closure	web
	GET HEAD	actors	actors.index	App\Http\Controllers\ActorController@index	web,auth
	GET HEAD	actors/create	actors.create	App\Http\Controllers\ActorController@create	web,auth
	POST	actors/store	actors.store	App\Http\Controllers\ActorController@store	web,auth
	DELETE	actors/{actor}/delete	actors.delete	App\Http\Controllers\ActorController@delete	web,auth
	GET HEAD	actors/{actor}/edit	actors.edit	App\Http\Controllers\ActorController@edit	web,auth
	POST	actors/{actor}/update	actors.update	App\Http\Controllers\ActorController@update	web,auth
	GET HEAD	api/user		Closure	api,auth:api
	GET HEAD	casts	casts.index	App\Http\Controllers\CastController@index	web,auth
	GET HEAD	casts/create	casts.create	App\Http\Controllers\CastController@create	web,auth
	POST	casts/store	casts.store	App\Http\Controllers\CastController@store	web,auth
	DELETE	casts/{cast}/delete	casts.delete	App\Http\Controllers\CastController@delete	web,auth
	GET HEAD	casts/{cast}/edit	casts.edit	App\Http\Controllers\CastController@edit	web,auth
	POST	casts/{cast}/update	casts.update	App\Http\Controllers\CastController@update	web,auth
	GET HEAD	directors	directors.index	App\Http\Controllers\DirectorController@index	web,auth
	GET HEAD	directors/create	directors.create	App\Http\Controllers\DirectorController@create	web,auth
	POST	directors/store	directors.store	App\Http\Controllers\DirectorController@store	web,auth
	DELETE	directors/{director}/delete	directors.delete	App\Http\Controllers\DirectorController@delete	web,auth
	GET HEAD	directors/{director}/edit	directors.edit	App\Http\Controllers\DirectorController@edit	web,auth
	POST	directors/{director}/update	directors.update	App\Http\Controllers\DirectorController@update	web,auth
	GET HEAD	home	home	App\Http\Controllers\HomeController@index	web,auth
	GET HEAD	indice		Closure	web
	GET HEAD	languages	languages.index	App\Http\Controllers\LanguageController@index	web,auth
	GET HEAD	languages/create	languages.create	App\Http\Controllers\LanguageController@create	web,auth
	POST	languages/store	languages.store	App\Http\Controllers\LanguageController@store	web,auth
	DELETE	languages/{language}/delete	languages.delete	App\Http\Controllers\LanguageController@delete	web,auth
	GET HEAD	languages/{language}/edit	languages.edit	App\Http\Controllers\LanguageController@edit	web,auth
	POST	languages/{language}/update	languages.update	App\Http\Controllers\LanguageController@update	web,auth
	GET HEAD	login	login	App\Http\Controllers\Auth\LoginController@login	web,guest
	POST	login		App\Http\Controllers\Auth\LoginController@showLoginForm	web,guest
	POST	logout	logout	App\Http\Controllers\Auth\LoginController@logout	web
	GET HEAD	password/confirm	password.confirm	App\Http\Controllers\Auth\ConfirmPasswordController@showConfirmForm	web,auth
	POST	password/confirm		App\Http\Controllers\Auth\ConfirmPasswordController@confirm	web,auth
	POST	password/email	password.email	App\Http\Controllers\Auth\ForgotPasswordController@sendResetLinkEmail	web
	GET HEAD	password/reset	password.request	App\Http\Controllers\Auth\ForgotPasswordController@requestResetLinkForm	web
	POST	password/reset	password.update	App\Http\Controllers\Auth\ResetPasswordController@reset	web
	GET HEAD	password/reset/{token}	password.reset	App\Http\Controllers\Auth\ResetPasswordController@showResetForm	web
	GET HEAD	platforms	platforms.index	App\Http\Controllers\PlatformController@index	web
	GET HEAD	platforms/create	platforms.create	App\Http\Controllers\PlatformController@create	web
	POST	platforms/store	platforms.store	App\Http\Controllers\PlatformController@store	web
	DELETE	platforms/{platform}/delete	platforms.delete	App\Http\Controllers\PlatformController@delete	web
	GET HEAD	platforms/{platform}/edit	platforms.edit	App\Http\Controllers\PlatformController@edit	web
	POST	platforms/{platform}/update	platforms.update	App\Http\Controllers\PlatformController@update	web
	GET HEAD	prueba		Closure	web
	GET HEAD	register	register	App\Http\Controllers\Auth\RegisterController@showRegistrationForm	web,guest
	POST	register		App\Http\Controllers\Auth\RegisterController@register	web,guest
	GET HEAD	series	series.index	App\Http\Controllers\SerieController@index	web,auth
	GET HEAD	series/create	series.create	App\Http\Controllers\SerieController@create	web,auth
	POST	series/store	series.store	App\Http\Controllers\SerieController@store	web,auth
	DELETE	series/{serie}/delete	series.delete	App\Http\Controllers\SerieController@delete	web,auth
	GET HEAD	series/{serie}/edit	series.edit	App\Http\Controllers\SerieController@edit	web,auth
	POST	series/{serie}/update	series.update	App\Http\Controllers\SerieController@update	web,auth

Seeder -" sembradores " de datos.

Los seeder se utilizan para crear datos de prueba. Se crean distintos seeder

ActorsTableSeeder

DirectorsTableSeeder

LanguagesTableSeeder

PlatformsTableSeeder

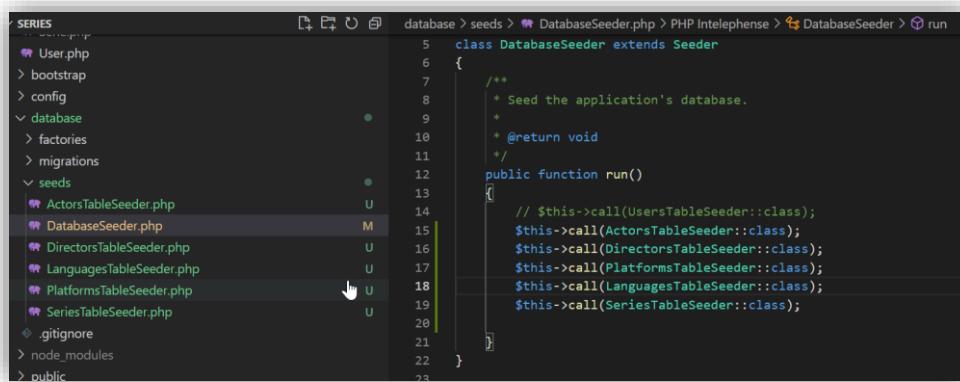
SeriesTableSeeder

UsuariopordefectoSeeder

```
php artisan make:seeder nombredelseeder
```

```
C:\Users\DELL\laravel\series>php artisan make:seeder ActorsTableSeeder  
Seeder created successfully.  
  
C:\Users\DELL\laravel\series>php artisan make:seeder DirectorsTableSeeder  
Seeder created successfully.  
  
C:\Users\DELL\laravel\series>php artisan make:seeder LanguagesTableSeeder  
Seeder created successfully.  
  
C:\Users\DELL\laravel\series>php artisan make:seeder SeriesTableSeeder  
Seeder created successfully.  
  
C:\Users\DELL\laravel\series>php artisan make:seeder PlatformsTableSeeder  
Seeder created successfully.
```

También hay que modificar el fichero `DatabaseSeeder.php` para añadir los seeder que se ejecutarán



```
class DatabaseSeeder extends Seeder
{
    /**
     * Seed the application's database.
     *
     * @return void
     */
    public function run()
    {
        // $this->call(UsersTableSeeder::class);
        $this->call(ActorsTableSeeder::class);
        $this->call(DirectorsTableSeeder::class);
        $this->call(PlatformsTableSeeder::class);
        $this->call(LanguagesTableSeeder::class);
        $this->call(SeriesTableSeeder::class);
    }
}
```

También es necesario ejecutar `composer dump-autoload` (paro el servidor primero)

```
C:\Users\DELL\laravel\series>composer dump-autoload
Generating optimized autoload files
> Illuminate\Foundation\ComposerScripts::postAutoloadDump
> @php artisan package:discover --ansi
Discovered Package: facade/ignition
Discovered Package: fideloper/proxy
Discovered Package: laravel/tinker
Discovered Package: laravel/ui
Discovered Package: nesbot/carbon
Discovered Package: nunomaduro/collision
Package manifest generated successfully.
Generated optimized autoload files containing 4399 classes
```

Así es como los seeder quedan generados:

```
 ActorsTableSeeder.php U ●
database > seeds > ActorsTableSeeder.php > PHP Intelephense > ActorsTableSeeder > run
1  <?php
2
3  use Illuminate\Database\Seeder;
4
5  class ActorsTableSeeder extends Seeder
6  {
7      /**
8      * Run the database seeds.
9      *
10     * @return void
11    */
12    public function run()
13    [
14
15    ]
16 }
17
```

Si queremos que genere 100 actores, este es el código que se utiliza:

(hay que añadir las librerías)

`Illuminate\Support\Facades\DB;`

`Illuminate\Support\Str`

```
 database > seeds > ActorsTableSeeder.php > PHP Intelephense > ActorsTableSeeder > run
1  <?php
2
3  use Illuminate\Database\Seeder;
4  use Illuminate\Support\Facades\DB;
5  use Illuminate\Support\Str;
6
7  class ActorsTableSeeder extends Seeder
8  {
9      /**
10     * Run the database seeds.
11     *
12     * @return void
13    */
14    public function run()
15    [
16        for ($i = 0; $i < 100; $i++) {
17            DB::table('actors')->insert([
18                'name' => Str::random(10),
19                'surname' => Str::random(7)
20            ]);
21        }
22    }
23 }
24
```

Tambien creamos un Seeder para crear el usuario por defecto para pruebas:

admin, informatico11540@gmail.com, contraseña:abcdefghijkl (guardo su hash)

```
c:\Users\DELL\laravel\series>php artisan make:seeder usuariordefectoSeeder
Seeder created successfully.
```

```
database > seeds > 🏃 usuariordefectoSeeder.php > ...
 4  use Illuminate\Support\Facades\DB;
 5  use Illuminate\Support\Str;
 6  use Illuminate\Support\Facades\Hash;
 7  class usuariordefectoSeeder extends Seeder
 8  {
 9      /**
10      * Run the database seeds.
11      *
12      * @return void
13      */
14      public function run()
15      {
16          DB::table('users')->insert([
17              'name' =>"admin",
18              'email' =>"informatico11540@gmail.com",
19              'password' => Hash::make('abcdefghijkl')
20          ]);
21      }
}
```

Ahora voy especificando los campos de cada tabla a llenar

Después ejecuto el comando:

```
php artisan migrate:fresh --seed
```

Este comando elimina los datos y tablas y los vuelve a crear con el seeder.

Es un comando que he usado en muchas ocasiones. Por ejemplo, al realizar cambios en datos o estructura

```
C:\Users\DELL\laravel\series>php artisan migrate:fresh --seed
Dropped all tables successfully.
Migration table created successfully.
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table (0.04 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 (0.05 seconds)
Migrating: 2022_02_19_172101_create_platforms_table
Migrated: 2022_02_19_172101_create_platforms_table (0.03 seconds)
Migrating: 2022_02_19_172226_create_languages_table
Migrated: 2022_02_19_172226_create_languages_table (0.04 seconds)
Migrating: 2022_02_19_172325_create_series_table
Migrated: 2022_02_19_172325_create_series_table (0.03 seconds)
Migrating: 2022_02_19_172351_create_directors_table
Migrated: 2022_02_19_172351_create_directors_table (0.02 seconds)
Migrating: 2022_02_19_172414_create_actors_table
Migrated: 2022_02_19_172414_create_actors_table (0.03 seconds)
Migrating: 2022_02_19_172435_create_casts_table
Migrated: 2022_02_19_172435_create_casts_table (0.04 seconds)
Seeding: ActorsTableSeeder
Seeded: ActorsTableSeeder (0.28 seconds)
Seeding: DirectorsTableSeeder
Seeded: DirectorsTableSeeder (0 seconds)
Seeding: PlatformsTableSeeder
Seeded: PlatformsTableSeeder (0.29 seconds)
Seeding: LanguagesTableSeeder
Seeded: LanguagesTableSeeder (0 seconds)
Seeding: SeriesTableSeeder
Seeded: SeriesTableSeeder (0 seconds)
Database seeding completed successfully.
```

Paginación

Se realiza en varios pasos:

Primeramente, en el controlador se recuperan los registros en bloques determinados en la constante PAGINATE_SIZE, por ejemplo, de diez en diez

```
$platforms = Platform::paginate(self::PAGINATE_SIZE);
```

A continuación, el controlador llama a la vista y a la vez que la llama, le pasa el nombre de la variable.

```
return view('platforms.list', ['platforms' => $platforms, 'platformName' => $platformName]);
```

A continuación en la vista hay que poner el siguiente código:

```
@endif  
{{ $languages->links() }}  
@endsection
```

Y ya está funcionando como podemos ver la barra inferior correctamente

The screenshot shows a table with two columns: 'ID' and 'NAME'. The first row contains the value '10' in the 'ID' column and 'uk9pj9fWwh' in the 'NAME' column. Below the table is a navigation bar with page numbers 1, 2, and 3, where '1' is highlighted. At the bottom of the page, there is a footer with the text '2021 Actividad 2 Backend'.

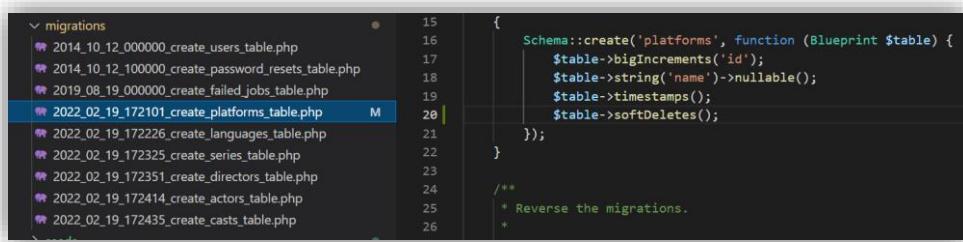
ID	NAME
10	uk9pj9fWwh

SoftDeletes

Es de gran utilidad. Permite una mejor gestión de los borrados no permanentes. Se ha implementado en todas las tablas. Se requieren dos pasos:

1.- Creación del campo en la migracion

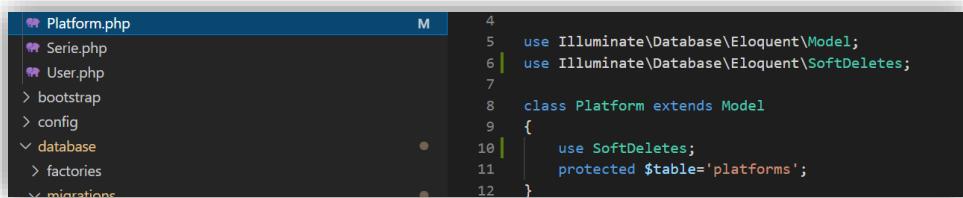
`$table->softDeletes()`



```
15 |     Schema::create('platforms', function (Blueprint $table) {
16 |         $table->bigIncrements('id');
17 |         $table->string('name')->nullable();
18 |         $table->timestamps();
19 |         $table->softDeletes();
20 |     });
21 |
22 | }
23 |
24 | /**
25 |  * Reverse the migrations.
26 | *
```

2.- Modificación del modelo de cada tabla, añadiendo "use SoftDeletes" y la librería:

"use Illuminate\Database\Eloquent\SoftDeletes;"



```
4
5     use Illuminate\Database\Eloquent\Model;
6     use Illuminate\Database\Eloquent\SoftDeletes;
7
8     class Platform extends Model
9     {
10        use SoftDeletes;
11        protected $table='platforms';
12    }
```

Después vuelvo a ejecutar el comando:

```
php artisan migrate:fresh -- seed
```

Y compruebo el resultado satisfactorio en phpmyadmin:

The screenshot shows the phpMyAdmin interface with the following details:

- Query bar: `SELECT * FROM `platforms``
- Toolbar buttons: Perfilando [Editar en línea] [Editar] [Explicar SQL] [C]
- Table navigation: Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla | Sort
- Table header: id, name, created_at, updated_at, deleted_at
- Table data:

	id	name	created_at	updated_at	deleted_at
<input type="checkbox"/>	1	FoPgSf1zka	NULL	NULL	NULL
<input type="checkbox"/>	2	O8a6xRdfSV	NULL	NULL	NULL
<input type="checkbox"/>	3	z7kOsyTzDG	NULL	NULL	NULL
<input type="checkbox"/>	4	H7Bc4pfI3r	NULL	NULL	NULL
<input type="checkbox"/>	5	CsweacmHXI	NULL	NULL	NULL
<input type="checkbox"/>	6	JxfzStydCe	NULL	NULL	NULL

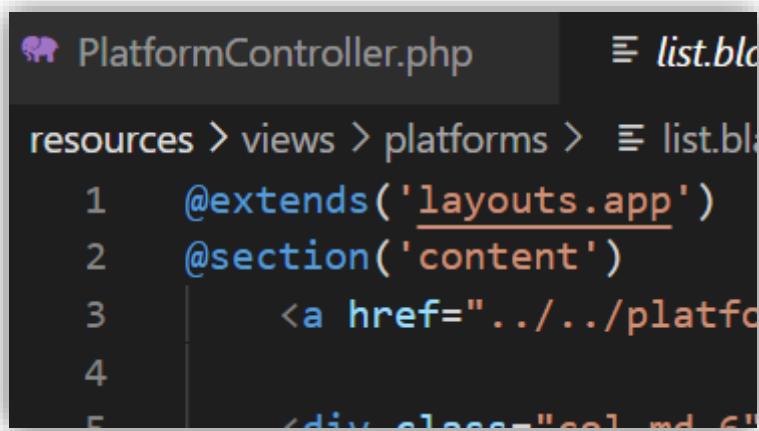
Uso del Layout

Se ha usado la herencia de plantillas de blade.

Se consigue de forma fácilmente tener un aspecto homogéneo en toda la aplicación.

Una cabecera, un footer común y además poder usar Bootstrap fácilmente sin tener que poner enlaces de Bootstrap en cada pagina

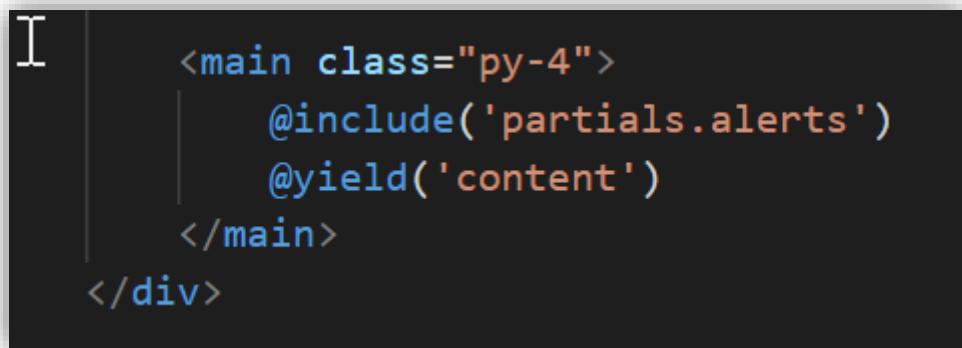
```
@extends('layouts.app')  
@section('content')  
...  
@endsection
```



The screenshot shows a code editor with a dark theme. The file path is 'resources > views > platforms > list.blade.php'. The code is:

```
1 @extends('layouts.app')  
2 @section('content')  
3     <a href=".../.../platforms">  
4         ...  
5     </div>
```

También incluyo un @include ('partials.alerts') arriba del content para mostrar los distintos mensajes



The screenshot shows a code editor with a dark theme. The code is:

```
<main class="py-4">  
    @include('partials.alerts')  
    @yield('content')  
</main>  
</div>
```

Código empleado en los controladores (Platform, Language, Actor, Director, Serie, Cast)

Cada entidad tiene su vista, su controlador y su modelo.

La estructura es similar a empleada en la actividad 1.

Las diferencias son:

Se incorpora una función de búsqueda por nombre.

Se junta en una misma vista la función de creación y actualización

El borrado se hace directamente en el controlador sin necesidad de usar una vista.

En cada controlador se definen los siguientes métodos:

- Index() para listar todas las plataformas. Posteriormente se cambia para poder hacer búsquedas
- Create ()
- Store (Request \$request)
- Edit(Platform \$platform) nos indican la plataforma a modificar
- Update (Request \$request, Platform \$platform)
- Delete (Platform \$platform) Recibe la plataforma a borrar

Método Index

```
public function index(Request $request)
{
    $platformName = null;
    if ($request->has('platformName')) {
        $platformName = $request->platformName;
        $platforms = Platform::where('name', 'like', '%' . $platformName . '%')->paginate(self::PAGINATE_SIZE);
    } else {
        $platforms = Platform::paginate(self::PAGINATE_SIZE);
    }
    // $platforms = Platform::paginate(10); // podríamos poner Platform::all(); para devolver todos los registros
    return view('platforms.list', ['platforms' => $platforms, 'platformName' => $platformName]);
}
```

Explicación:

Si el parámetro con el que se llama este relleno con caracteres de búsqueda, entonces utilizando el modelo Platform recupera todas las plataformas cuyo nombre contenga esos caracteres (like% ..%).

Si no recibe nada, entonces recupera todas las plataformas.

En ambos casos recupera paginando con el número de registros establecido en la constante PAGINATE_SIZE.

Por último, llama a la vista a través del nombre definido en las rutas del fichero de rutas web.php y le pasa como parámetro el array de plataformas para que puedan ser visualizadas.

Método Create

```
public function create()
{
    return view('platforms.create');
```

Este método solo llama a la vista

Método Store

```
public function store(Request $request)
{
    $this->validatePlatform($request)->validate();
    $platform = new Platform(); You, 2 days ago • Listado y creacion de platform
    $platform->name = $request->platformName;
    $platform->save();
    return redirect()->route('platforms.index')->with('success', 'Plataforma Creada correctamente');
```

Recibe como parámetro la plataforma a crear. Crea un nuevo objeto a partir del modelo y le asigna uno a uno los atributos. Después lo almacena.

Por último, llama a la vista de listado pasándole un mensaje a la vista en forma de variable.

Método Edit

```
public function edit(Platform $platform)
{
    return view('platforms.create', ['platform' => $platform]);
```

Llama a la vista definida en el fichero de rutas como plataforma.create.

Le pasa como parámetro la plataforma recibida.

Método Update

```
public function update(Request $request, Platform $platform)
{
    $this->validatePlatform($request)->validate();
    $platform->name = $request->platformName;
    $platform->save();
    return redirect()->route('platforms.index')->with('success', 'Plataforma Actualizada correctamente');
```

Recibe como parámetros el array request con los nuevos datos y el objeto platforms.

Actualizo el nombre y guardo.

Por último, llamo al listado de plataformas y le paso la variable con el mensaje de operación realizada con éxito.

Vista de Listar (y buscar)

Colapsando el código se aprecia mejor las distintas partes de la vista:

```
1  @extends('layouts.app')
2  @section('content')
3  <a href="/platforms/create" class="btn btn-primary">Crear Plataforma</a>
4
5  > <div class="col-md-6"> ...
6  </div>
7
8
9
10 <@if (count($platforms) > 0)
11 > <table class="table table-bordered table-hover table-sm" style="background-color:white">
12 </table>
13 > <div class="container-fluid h-100">...
14 </div>
15 <@else
16 > <div class="alert alert-warning" role="alert">
17     Aun no existen plataformas.
18 </div>
19 <@endif
20 <@endsection
```

Se utiliza la herencia de `layouts.app`

Se incluye todo el código dentro de la sección `content` de la plantilla `layouts.app`.

Otra característica es que en función de si el campo de búsqueda este lleno o no,

Se envía el formulario con datos o vacío. búsqueda o listado

```
value="@isset($platformName) {{ $platformName }} @endisset"
```

Contiene la llamada a edición.

```
<a href="{{ route('platforms.edit', $platform) }}" class="btn btn-success btn-sm">
    editar
</a>
```

Llamada al borrado

```
<form id="delete-form-{{ $platform->id }}" action="{{ route('platforms.delete', [$platform]) }}" method="post" style="display: inline-block;">
    {{ method_field('delete') }} You, seconds ago * Uncommitted changes
    {{ csrf_field() }}
    <a class="btn btn-danger btn-sm" href="javascript:void(0); onclick=" document.getElementById('delete-form-{{ $platform->id }}').submit()">
        borrar
    </a>
</form>
```

Hay que especificarlo expresamente al no ser ni post ni get `{{ method_field('delete') }}`

Hay que poner el token para verificar que el usuario autenticado es quien está haciendo la petición a la aplicación. Si no se pone da error 419. {{ csrf_field() }}

Vista de Crear Actualizar

```
1 @isset($platform)
  <form name="edit_platform" action="{{ route('platforms.update',$platform) }}" method="POST">
    @csrf
  @else
    <form name="create_platform" action="{{ route('platforms.store') }}" method="POST">
      @csrf
    @endisset| You, seconds ago • Uncommitted changes
```

Según se reciban o no datos dentro de la variable \$platform, se establece la propiedad **action** para que llame a la actualización o a la creación de plataformas.

- Tiene datos. -> Actualizacion (update)
- No tiene datos -> Creacion (store)

5. Conclusiones

A través de esta Actividad se ha puesto en práctica muchos de los conocimientos teóricos aprendidos en la asignatura sobre Laravel.

Se ha realizado sobre la base de los mismos requisitos funcionales de la Actividad 1 y por ello, se ha podido comparar las diferencias de realizar el proyecto con y sin framework.

El patrón Modelo-Vista-Controlador se emplea de forma distinta. En la Actividad 1, las vistas llamaban al controlador que a su vez llamaban al modelo. En esta Actividad 2, con Laravel, es distinto. Las llamadas, aunque también pueden realizarse a las vistas, normalmente se realizan al controlador, el cual, tras utilizar el modelo, llama a las vistas pasándole a la vez los datos oportunos. Es definitiva, con Laravel, el controlador es la piedra angular de la aplicación.

La utilización del framework en general me ha parecido muy interesante por todo lo que aporta al desarrollo. Ha facilitado muchísimo la creación y conexión a la base de datos a través de los ficheros de configuración .env, database.php, las migraciones y los seeders. El mecanismo de rutas de Laravel me parece fácil y útil a la vez que muy potente. La autentificación que crea a través de un middleware es muy transparente y completa.

Se ha utilizado mucho los comandos de artisan para experimentar y crear todo lo posible: modelos, controladores, etc. (Las vistas no se puede)

He usado decenas de veces el comando superútil que regenera la información y tablas a partir de migraciones y seeder.

El código con Laravel es mucho más legible que utilizando únicamente PHP. La interpolación que aporta Blade es muy útil ya que permite no tener que estar todo el tiempo saltando de código HTML a PHP y viceversa como ocurría en la Actividad 1.

El uso del Layout también me ha parecido muy útil añadiendo facilidad en el desarrollo mantenimiento y aportando homogeneidad al proyecto.

Me ha parecido también muy útil el orden que aporta el framework al desarrollo.

La integración con el IDE Visual Studio es muy buena y tras instalar extensiones, el autocompletado ha funcionado muy bien.

Se ha realizado labor de investigación de desarrollo haciendo llamadas de métodos a métodos dentro del mismo controlador, paso de información de controladores a vistas, ejecución de php dentro de las vistas, y realizando todo tipo de generación de consultas con mayor o menor éxito debido al tiempo limitado.

Los posibles desarrollos futuros pueden ser: Una mejor presentación de las vistas, elaboración de artefactos para mostrar mensajes y ampliación de los sistemas de búsqueda de datos.

5. Bibliografía

Documentación Oficial de Laravel. Recuperado de <https://laravel.com/docs/6.x> con fecha 18 de febrero de 2022

Documentación de Laravel 6 en español. recuperado de: <https://styde.net/documentacion-de-laravel-6/> con fecha 18 de febrero de 2022

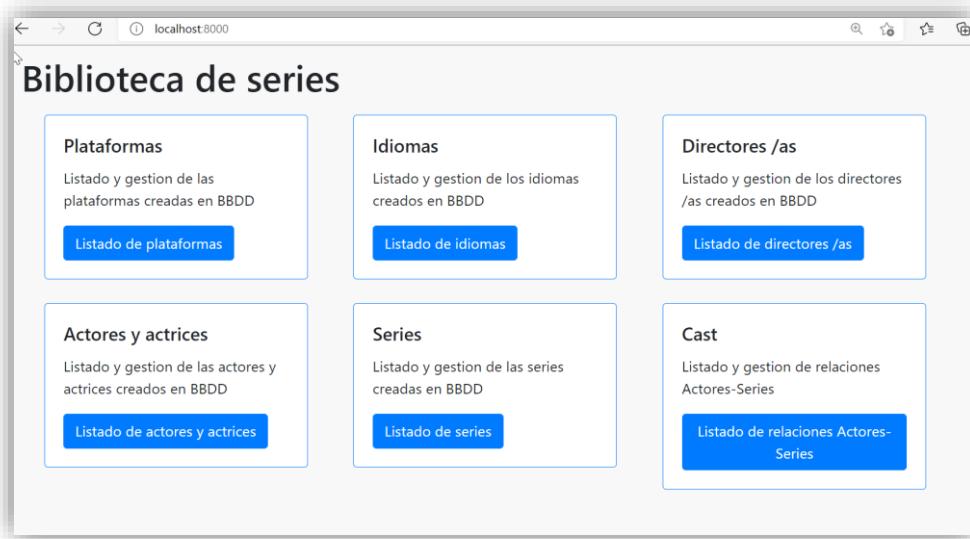
Apuntes de la Asignatura Desarrollo de Aplicaciones WEB I, lado del servidor (Backend) 2022. Universidad Internacional de Valencia.

Cabezas Granado, L.M. et González Lozano, F.J. Cursos de PHP 8 y MySql 8. Anaya Multimedia.

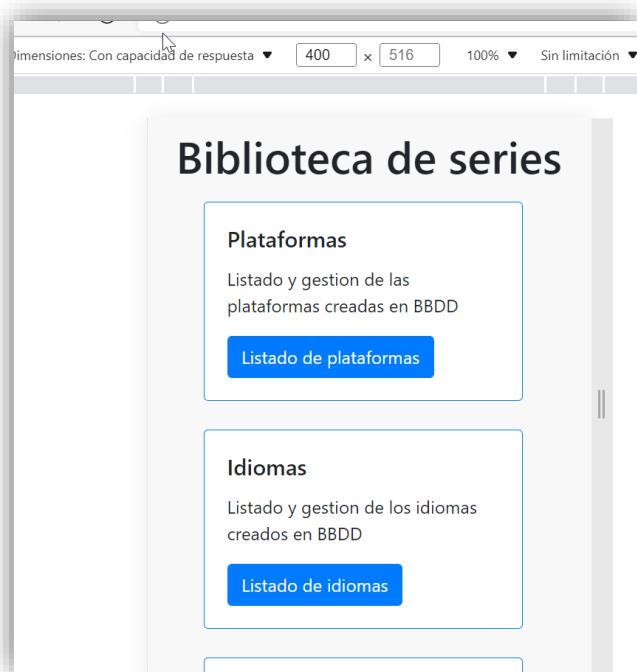
6. Anexos

Anexo1: Vistas de la Aplicación

Pantalla inicial



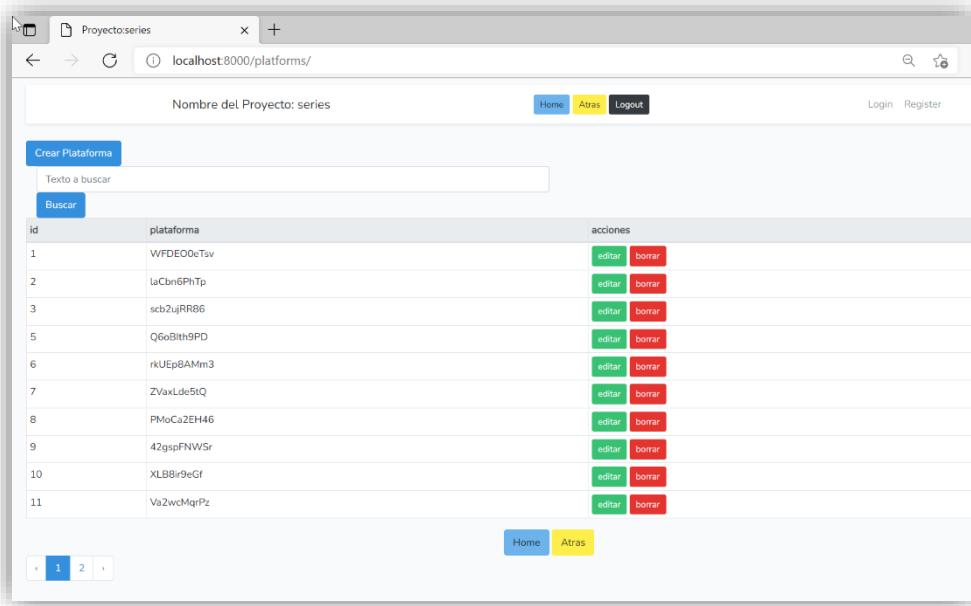
Prueba del diseño Responsive



Creación, Listado, Edición y Borrado de Plataformas

Entidad Plataformas. Se entra sin necesidad de identificación

Listado



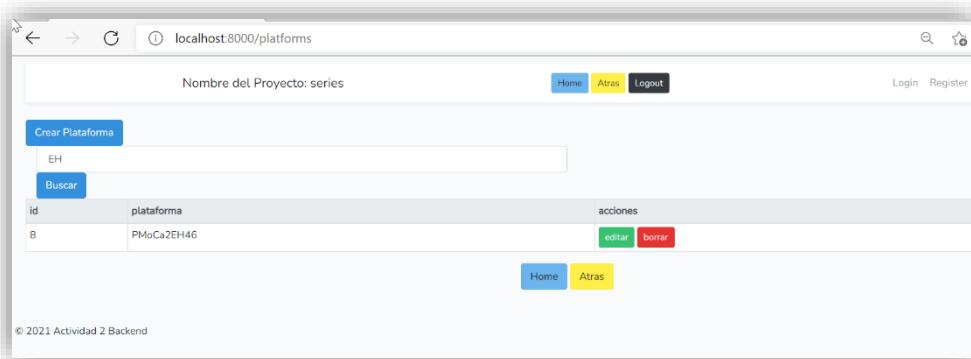
A screenshot of a web browser displaying a list of platforms. The title bar says "localhost:8000/platforms/". The page header includes "Nombre del Proyecto: series", "Home", "Atras", "Logout", "Login", and "Register". A search bar and a "Buscar" button are present. Below is a table with columns "id", "plataforma", and "acciones". The "acciones" column contains green "editar" and red "borrar" buttons. The table shows 11 rows of data.

id	plataforma	acciones
1	WFDEOOctsv	<button>editar</button> <button>borrar</button>
2	laCbn6PHTp	<button>editar</button> <button>borrar</button>
3	scb2ujRR86	<button>editar</button> <button>borrar</button>
5	Q6oBith9PD	<button>editar</button> <button>borrar</button>
6	rkUEp8AMm3	<button>editar</button> <button>borrar</button>
7	ZVaxLd6t5Q	<button>editar</button> <button>borrar</button>
8	PMoCa2EH46	<button>editar</button> <button>borrar</button>
9	42gspiFNW5r	<button>editar</button> <button>borrar</button>
10	XLB8ir9eGf	<button>editar</button> <button>borrar</button>
11	Va2wcMqrPz	<button>editar</button> <button>borrar</button>

Home Atras

1 2

Búsqueda de Plataformas



A screenshot of a web browser displaying a list of platforms after a search. The search term "EH" is entered in the search bar. The search results show one entry: row 8 with plataforma "PMoCa2EH46". The "acciones" column contains green "editar" and red "borrar" buttons. The page footer includes "© 2021 Actividad 2 Backend".

id	plataforma	acciones
8	PMoCa2EH46	<button>editar</button> <button>borrar</button>

Home Atras

© 2021 Actividad 2 Backend

Actualización

A screenshot of a web browser window titled "localhost:8000/platforms/1/edit". The page header includes "Nombre del Proyecto: series", "Home", "Atras", "Logout", "Login", and "Register". The main content area has a label "Nombre de la plataforma" and a text input field containing "WFDEO0eTsv". Below the input is a blue "Actualizar" button. At the bottom left, there is a copyright notice: "© 2021 Actividad 2 Backend".

Creación

A screenshot of a web browser window titled "localhost:8000/platforms/create". The page header includes "Nombre del Proyecto: series", "Home", "Atras", "Logout", "Login", and "Register". The main content area has a label "Nombre de la plataforma" and a text input field with placeholder text "Introduce el nombre de la plataforma". Below the input is a blue "Crear" button. At the bottom left, there is a copyright notice: "© 2021 Actividad 2 Backend".

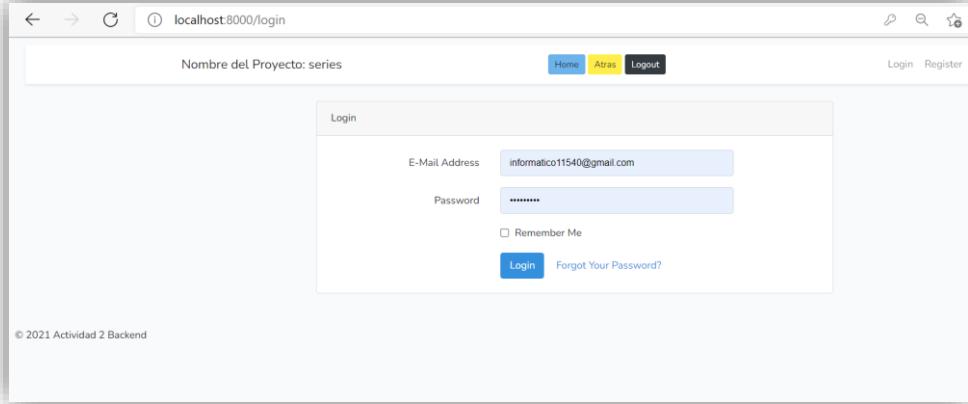
Borrado

A screenshot of a web browser window titled "localhost:8000/platforms". The page header includes "Nombre del Proyecto: series", "Home", "Atras", "Logout", "Login", and "Register". The main content area features a search bar with "Crear Plataforma" and "Buscar" buttons. Below is a table with columns "id", "plataforma", and "acciones". The table contains the following data:

id	plataforma	acciones
1	WFDEO0eTsv	<button>editar</button> <button>borrar</button>
3	scb2ujRR86	<button>editar</button> <button>borrar</button>
5	Q6oBith9PD	<button>editar</button> <button>borrar</button>
6	rkUEp8AMm3	<button>editar</button> <button>borrar</button>
7	7VwuldeFO	<button>editar</button> <button>borrar</button>

Creación, Listado, Edición y Borrado de Idiomas

Entidad Idioma. Me lleva automáticamente a Identificarme si no lo estoy

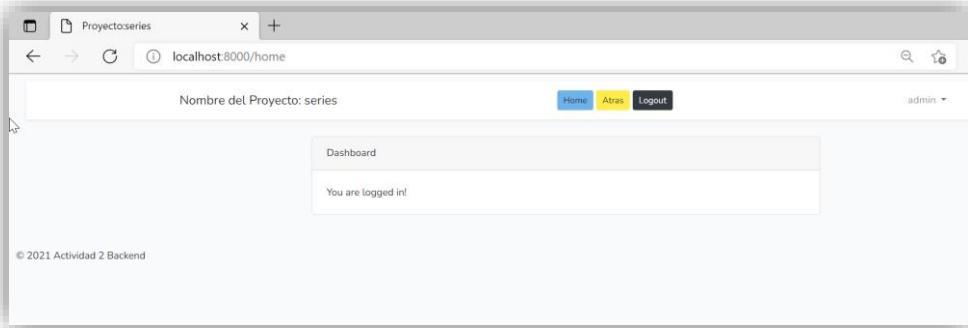


Nombre del Proyecto: series

E-Mail Address: informatico11540@gmail.com

Password:
 Remember Me

Login Forgot Your Password?



Nombre del Proyecto: series

Dashboard

You are logged in!

admin ▾

Registro de usuario

Nombre del Proyecto: series

Home Atras Logout

Login Register

Register

Name

E-Mail Address

Password

Confirm Password

Register

Nombre del Proyecto: series

Home Atras Logout

Login Register

Register

Name

E-Mail Address

Password

Confirm Password

Register

© 2021 Actividad 2 Backend

Identificado como Admin me lleva a las distintas pantallas de aplicación

Nombre del Proyecto: series

Home Atras Logout

Logout

admin

Crear Idioma

id	idioma	acciones
1	AFV5HAJvUO	<button>editar</button> <button>borrar</button>
2	lj2Kr29Sq6	<button>editar</button> <button>borrar</button>
4	zCCceRb3	<button>editar</button> <button>borrar</button>
5	oVfp8IPFOS	<button>editar</button> <button>borrar</button>
6	6wtYPP9rwy	<button>editar</button> <button>borrar</button>
7	OyyCHIUAYo	<button>editar</button> <button>borrar</button>
8	sq1t6nkz2g	<button>editar</button> <button>borrar</button>
9	XIWWMWD7ew9	<button>editar</button> <button>borrar</button>
10	uLJ9E2H4U	<button>editar</button> <button>borrar</button>
11	Yj4uIPW2p	<button>editar</button> <button>borrar</button>

Home Atras

1 2

The screenshot shows a web browser window titled "Proyectoseries" with the URL "localhost:8000/languages". The page displays a table of language entries. At the top left is a blue button labeled "Crear idioma". The table has columns for "id", "idioma", and "acciones". The "acciones" column contains two buttons: "editar" (green) and "borrar" (red). The "idioma" column lists 11 entries, each starting with a unique identifier (e.g., 1, 2, 4, 5, 6, 7, 8, 9, 10, 11) followed by a random string of letters and numbers. The "acciones" column for each row contains two buttons: "editar" and "borrar". Below the table is a navigation bar with "Home" (blue), "Atras" (yellow), and "Logout" (black). On the right, there is a user profile with the name "admin" and a dropdown arrow. At the bottom, there is a page navigation with a blue "1" button, a white "2" button, and a white "3" button.

Nombre del Proyecto: series		
Crear idioma		
id	idioma	acciones
1	AFV5HAjUO	<button>editar</button> <button>borrar</button>
2	lj2kr29Sg6	<button>editar</button> <button>borrar</button>
4	zCCceRb3	<button>editar</button> <button>borrar</button>
5	oVfpBHPFOS	<button>editar</button> <button>borrar</button>
6	6wtYPp9rwy	<button>editar</button> <button>borrar</button>
7	OvvCHIUAYo	<button>editar</button> <button>borrar</button>
8	sq1t6nkz2g	<button>editar</button> <button>borrar</button>
9	XIWWMWD7ew9	<button>editar</button> <button>borrar</button>
10	uLJ8E2H4U	<button>editar</button> <button>borrar</button>
11	Yj4ufPfW2p	<button>editar</button> <button>borrar</button>

Creación, Listado, Edición y Borrado de directores

Nombre del Proyecto: series

Home Atras Logout admin

Crear Director					
id	nombre	apellidos	nacionalidad	fecha de nacimiento	acciones
1	IPZgDPjLb6	tD55jE		22-02-2022	<button>editar</button> <button>borrar</button>
2	UnWtRtEriqb	mvtfpzP		22-02-2022	<button>editar</button> <button>borrar</button>
3	2JNnViBkC	dLlxC1g		22-02-2022	<button>editar</button> <button>borrar</button>
4	aMi5O2KqRZ	JxNDHKU		22-02-2022	<button>editar</button> <button>borrar</button>
5	rGDb5k50ri	DGYDan7		22-02-2022	<button>editar</button> <button>borrar</button>
6	W59eQxhG0a	xaFgRs9		22-02-2022	<button>editar</button> <button>borrar</button>
7	rsiteGTw2Z	rcOhLjj		22-02-2022	<button>editar</button> <button>borrar</button>
8	Vvos9UWu0N	XDFHyD		22-02-2022	<button>editar</button> <button>borrar</button>
9	I22btzjh9	WPrgaVq		22-02-2022	<button>editar</button> <button>borrar</button>
10	KmGyZxtKGE	WLAcpfq		22-02-2022	<button>editar</button> <button>borrar</button>

Home Atras

1 2 3 4 5 6 7 8 9 10

Creación, Listado, Edición y Borrado de Actores

Nombre del Proyecto: series

Home Atras Logout admin ▾

Crear Actor

id	nombre	apellidos	nacionalidad	fecha de nacimiento	acciones
1	hDgkWoB6t3	0Ez2IMj		22-02-2022	<button>editar</button> <button>borrar</button>
2	qUJHAk526e	wzLl3hM		22-02-2022	<button>editar</button> <button>borrar</button>
3	6YDUqThCjm	V9jvsx1		22-02-2022	<button>editar</button> <button>borrar</button>
4	AizgB6gzFw	OmaH7jQ		22-02-2022	<button>editar</button> <button>borrar</button>
5	IKIDivdWfI	B4WznVi		22-02-2022	<button>editar</button> <button>borrar</button>
6	lFuYt5r7sL	dcysKWB		22-02-2022	<button>editar</button> <button>borrar</button>
7	cVwttm9Qc7	JFlwXW6		22-02-2022	<button>editar</button> <button>borrar</button>
8	erycv9DzIE	zyFLH49		22-02-2022	<button>editar</button> <button>borrar</button>
9	Vx34bE1Yue	6gagKNz		22-02-2022	<button>editar</button> <button>borrar</button>
10	s0mlFp4acB	DikLQNk		22-02-2022	<button>editar</button> <button>borrar</button>

Home Atras

1 2 3 4 5 6 7 8 9 10

Creación, Listado, Edición y Borrado de series

Crear Serie		
id	nombre	acciones
1	huulzBOMvw	<button>editar</button> <button>borrar</button>
2	wqEkH3bhKu	<button>editar</button> <button>borrar</button>
3	yQ64cDBIbn	<button>editar</button> <button>borrar</button>
4	jWxF9tsAEW	<button>editar</button> <button>borrar</button>
5	AziVUNjc6O	<button>editar</button> <button>borrar</button>
6	spAxjy4Cis	<button>editar</button> <button>borrar</button>
7	hQ2hLPytS	<button>editar</button> <button>borrar</button>
8	ccYzC1AIsk	<button>editar</button> <button>borrar</button>
9	t95thpyOn3	<button>editar</button> <button>borrar</button>
10	Pjp2kW3zax	<button>editar</button> <button>borrar</button>

Creación, Listado, Edición y Borrado de la relación Actores - Series

The screenshot shows a web application window titled "Proyectoseries". The URL in the address bar is "localhost:8000/casts/". The main content area has a header "Nombre del Proyecto: series". Below this is a search bar with placeholder text "Texto a buscar" and a blue "Buscar" button. A yellow banner at the bottom states "Aun no existen relaciones.". At the top right, there are links for "Home", "Atras", and "Logout", and a user account dropdown labeled "admin". The footer contains the copyright notice "© 2021 Actividad 2 Backend".