



Universidad
Internacional
de Valencia

Actividad 2:

Biblioteca de series en Laravel

Titulación:

Máster Universitario en Desarrollo de
Aplicaciones y Servicios Web

Curso académico: 2022-2023

Alumno/a: Gómez Olivencia, Rubén

D.N.I.: 78910013-A

Asignatura: Desarrollo de aplicaciones
web I: Lado del servidor (*back-end*)

Índice general

1. Introducción	3
2. Laravel	3
3. Despliegue de la aplicación	4
3.1. Servicios Docker	4
3.2. Despliegue del código fuente	4
3.3. Creación de la base de datos	4
3.4. Despliegue de datos	5
4. Desarrollo realizado	6
4.1. Configuración de la aplicación	6
4.2. Relación entre modelos	6
4.3. Validación de datos en formularios	7
4.4. Autenticación y autorización	8
4.5. Localización del proyecto	9
4.6. Búsqueda de datos	9
5. Dificultades del proyecto	10
6. Conclusiones	11

1. Introducción

A lo largo de este documento se van a explicar las decisiones tomadas, tanto en el ámbito de programación como de diseño, durante el desarrollo de una aplicación web para gestionar una biblioteca que contiene series, sus episodios y los actores que aparecen en ellos.

Para la realización de esta web se ha hecho uso del *framework* [Laravel](#), junto con **javascript** para la programación **front-end**.

2. Laravel

Laravel es un *framework* desarrollado en el año 2011 que hace uso del lenguaje de programación [PHP](#) para la creación de desarrollos basados en la tecnología web.

La idea de todo *framework* es la de crear un entorno de trabajo que unifique distintos conceptos, prácticas y criterios que sirvan como referencia para la creación de un proyecto.

Añadido a eso, los *frameworks* nos van a otorgar distintas librerías que podremos usar durante el desarrollo del proyecto que nos simplificará ciertas tareas muy habituales. Por poner sólo unos pocos ejemplos:

- Gestión de **conexión a la base de datos**.
- Creación de **modelos** que manejan datos e información.
- Generación de **vistas** y plantillas que se pueden reutilizar
- Generación de **controladores** que interactúan con los modelos para pasar la información a las vistas.
- Sistemas de **autenticación y autorización**.



Laravel es un *framework* que hace uso de la arquitectura conocida como “[modelo-vista-controlador](#)”, que nos permite separar la lógica de negocio de la representación visual mostrada.

3. Despliegue de la aplicación

Antes de entrar en detalle en cómo se ha desarrollado la aplicación es importante conocer cómo podemos realizar el despliegue de la aplicación, ya sea para utilizarla o para realizar modificaciones sobre la misma.


3.1. Servicios Docker

Para realizar el desarrollo del proyecto se ha utilizado servicios a través de contenedores **Docker**, los cuáles pueden ser levantados gracias al fichero


 `compose.yaml` y el comando  `docker-compose up`

Al levantar los servicios con el comando **docker-compose up** se hará uso de los siguientes puertos:



- **80**: Para el entorno web, usando Nginx como servidor web.
- **3306**: Para la base de datos.
- **3307**: Para el acceso web a phpmyadmin.

Para el correcto funcionamiento del contenedor se hace uso del fichero  `vhost.conf` que modifica la configuración del servicio Nginx para que funcione de manera correcta con Laravel.

3.2. Despliegue del código fuente

Para realizar el despliegue del código fuente sobre el contenedor del servicio web, el directorio  `viudb` debe estar situado a la misma altura del fichero

 `compose.yaml`.

De esta manera, a la hora de levantar el servicio se crea un volumen compartiendo el directorio local  `viudb` con otro dentro del contenedor, en la ruta  `/app`, que es de donde se nutre Nginx.

3.3. Creación de la base de datos

El servicio de MySQL se encarga de crear una base de datos llamada **actividad2** en el momento en el que el servicio se levanta. También crea los siguientes credenciales de acceso a dicha base de datos:

- usuario: **actividad2**
- contraseña: **4ct1v1d4d2**

3.4. Despliegue de datos

Para realizar el despliegue de datos, se va a utilizar dos características que tiene Laravel y que tienen que ver con el despliegue en la base de datos:

- **Migrate:** Es una forma de tener un sistema de control de versiones del esquema de base de datos. De esta manera, se puede hacer evolucionar el esquema (tablas, columnas, índices, ...) a lo largo del tiempo y también volver a un punto anterior del mismo.
- **Seeds:** También conocido como “semillas”, nos posibilita añadir datos a las tablas que hemos creado. Normalmente se utiliza para crear datos al inicio del proyecto con datos que deben existir (o también datos de pruebas) para que se pueda utilizar el proyecto.

Para realizar el despliegue debemos conectarnos al contenedor donde tenemos el proyecto y lanzar los comandos que realizan el “migrate” y crean los datos. A continuación aparecen los comandos:

Acceder al contenedor Docker para hacer el despliegue

```
#nos conectamos al contenedor
ruben@vega:~$ docker exec -it actividad_2_php_1 /bin/bash

#vamos al directorio donde está el desarrollo
root@a7913ac7b97a:/# cd /app

#lanzamos el migrate
root@a7913ac7b97a:/app# php artisan migrate
Migration table created successfully.
Migrating: 2014_10_12_000000_create_users_table
...

#lanzamos el seed
root@a7913ac7b97a:/app# php artisan db:seed
```

```
Seeding: UserSeeder
Seeded:  UserSeeder (0.06 seconds)
...
```

4. Desarrollo realizado

Una vez tenemos realizado el despliegue, se va a profundizar en el desarrollo realizado, destacando los siguientes aspectos:

4.1. Configuración de la aplicación

Otro de los aspectos en los que Laravel nos facilita el desarrollo es en la creación de un fichero que podemos utilizar para realizar la configuración de la aplicación:



Este fichero se crea con configuración que se debe modificar para la gestión de distintas configuraciones, y lo hemos utilizado para:

- **Acceso a base de datos:** Para configurar el acceso a base de datos tenemos las opciones: servidor, puerto, usuario, contraseña y nombre de la base de datos. **Estas opciones deben ser configuradas para el servidor correspondiente.**
- **Paginación:** Para configurar el número de items por página que se van a visualizar. Opción “VIEW_PAGINATE”.
- Las distintas **opciones que un celebrity** puede participar en un episodio. Dado que en la base de datos lo hemos añadido como “enum”, lo hemos configurado como “CELEBRITY_EPISODE_FUNCTIONS”.
- Similar al caso anterior, pero para los **idiomas y los episodios** en la variable “LANGUAGE_EPISODE_TYPES”.

4.2. Relación entre modelos

A la hora de crear la base de datos se ha tenido en cuenta ciertas relaciones en los datos (teniendo en cuenta el esquema Entidad-Relación del proyecto anterior), y por ello los modelos también se tienen que relacionar entre sí.

Laravel nos permite crear dicha relación a través de los modelos, en donde se indica si un modelo pertenece a otro, o si tiene varios elementos de otro modelo.

Vamos a poner como ejemplo el modelo de los episodios, que pertenece a una serie, y a su vez contiene varias celebrities:

</> Parte del modelo "Episode"

```
// devuelve la serie a la que pertenece
public function tvshow(){
    return $this->belongsTo('App\TVShow');
}

// devuelve las celebrities que tiene el capítulo a través de
// la tabla celebrity_episode
// hay que indicar la columna extra que tiene la tabla.
public function celebrities() {
    return $this->belongsToMany('App\Celebrity')
        ->withPivot(['perform_as']);
}
```

De esta manera, a través de un episodio podremos obtener a qué serie pertenece y qué *celebrities* aparecen en ese episodio y qué función realizan en el mismo.

4.3. Validación de datos en formularios

A la hora de introducir o actualizar datos es importante que exista una validación previa, para que así no llegue el error hasta la base de datos.

A lo largo del proyecto, para todos los modelos, se ha realizado la validación en todos los formularios en donde se ha tenido en cuenta alguna (o varias) de las siguientes validaciones, entre otras:

- Existencia de datos en un campo.
- Número mínimo de caracteres introducidos.
- No duplicidad en alguno de los campos.

Crear plataforma

- El campo name ya ha sido tomado.

Nombre

En caso de que la validación no sea correcta y devuelva un error, la aplicación redirigirá al formulario y aparecerá el error en el campo del formulario que debe ser corregido, tal como aparece en la imagen.

4.4. Autenticación y autorización

Como parte del proyecto, era necesario crear un sistema de autenticación y autorización.

- **Autenticación:** Proceso que se encarga de confirmar que un usuario es quien dice ser. Normalmente se valida con un usuario o e-mail y contraseña.
- **Autorización:** Proceso que permite a un usuario autenticado visitar, visualizar o utilizar ciertos recursos del proyecto.

Login

E-Mail Address

Password

☐ Remember Me

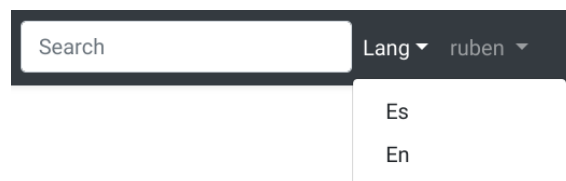
[Login](#) [Forgot Your Password?](#)

En este proyecto sólo el apartado de “plataformas” es público, por lo que para poder utilizar cualquier otro es necesario haberse autenticado. Usando el sistema de despliegue de datos se ha añadido un usuario para poder acceder todos a los apartados, siendo los credenciales:

- **e-mail:** example@example.com
- **Password:** password

4.5. Localización del proyecto

Se denomina “localización” (o internacionalización) al diseño de software que tiene en cuenta que el proyecto realizado será utilizado en distintos idiomas. Este diseño permite que la traducción a otros idiomas que no sea el original sea más sencillo de realizar y que mediante una directiva de configuración se pueda cambiar.



En la barra de navegación, junto al sistema de búsqueda y de acceso se ha añadido un menú desplegable con los dos idiomas que se pueden utilizar actualmente en el desarrollo.

4.6. Búsqueda de datos

Para facilitar el acceso a los datos por parte de los usuarios se han añadido distintas cajas de búsqueda:

- **Búsqueda general:** Se encuentra en la barra de navegación superior, para realizar una búsqueda entre todos los datos guardados de la aplicación
- **Búsqueda simple:** Dentro de cada apartado de la web hay otro buscador, pero este sólo busca en dicho apartado.



Si se usa el buscador general los resultados de búsqueda nos llevará a una página en la que aparecerán los resultados de la misma, resaltando el término buscado, tal como aparece en la imagen.

Las funciones de búsqueda, ya que se utilizan desde distintos apartados de la web, se encuentran en el fichero `app/helpers.php`.

5. Dificultades del proyecto

Teniendo en cuenta que este proyecto es la continuación de uno anterior, se puede determinar que se han subsanado ciertas dificultades expuestas en el anterior proyecto.

Ahora bien, eso no quita que hayan existido otras, que se exponen a continuación:

- **Desconocimiento al usar el *framework*:** A la hora de hacer uso de un *framework* hay que conocer sus particularidades, pero bien es cierto que al haber usado otros (como Ruby on Rails), la curva de aprendizaje es menor. Aunque bien es cierto que el proceso de adaptación, conocer dónde se sitúan los ficheros, ver cómo se traspasa el conocimiento que se tiene de uno a otro, ... lleva su tiempo.
- **Falta de utilidades:** Teniendo en cuenta el punto anterior, se han echado de menos utilidades que otros frameworks incorporan de serie, como es el sistema de *scaffolding* que permite crear el modelo pudiendo pasar los campos por línea de comandos (o a través de un asistente, como sucede en Symfony), creación de funciones y vistas CRUD para los modelos y controllers, ...

Algunos de estos aspectos no se entienden ya que otros frameworks los tienen implementados desde hace muchos años, por lo que es fácil coger esas ideas.

De todas maneras, el único inconveniente ha sido tener que dedicar tiempo a esas particularidades que en otros frameworks ya vienen hechas.

- **Uso de las validaciones:** Al crear el sistema de validación teniendo en cuenta que no se repitan la combinación de varios campos (como el nombre y apellidos de los *celebrities*) no es demasiado intuitivo, y en la

documentación oficial no aparece.

6. Conclusiones

A la hora de desarrollar una aplicación web es importante conocer cómo se gestiona la información en el lado de servidor. Este proyecto ha permitido profundizar en diferentes ámbitos utilizados en el desarrollo **back-end** como son: el patrón **MVC**, la validación de datos obtenidos en peticiones, la creación de una aplicación multi-idioma, ...

El uso de Laravel como framework ha facilitado la labor de crear el proyecto en un tiempo ajustado, gracias a las implementaciones propias que tiene, lo que hace que también haya surgido un interés en comprobar las novedades que tienen las últimas versiones, y hacer un seguimiento del mismo para el futuro.