PRÁCTICA 3 DAW

Daniel Rodríguez Ventosa

David Retamal Rojas

David López Higueras

Alejandro García de Marina Martín

ÍNDICE

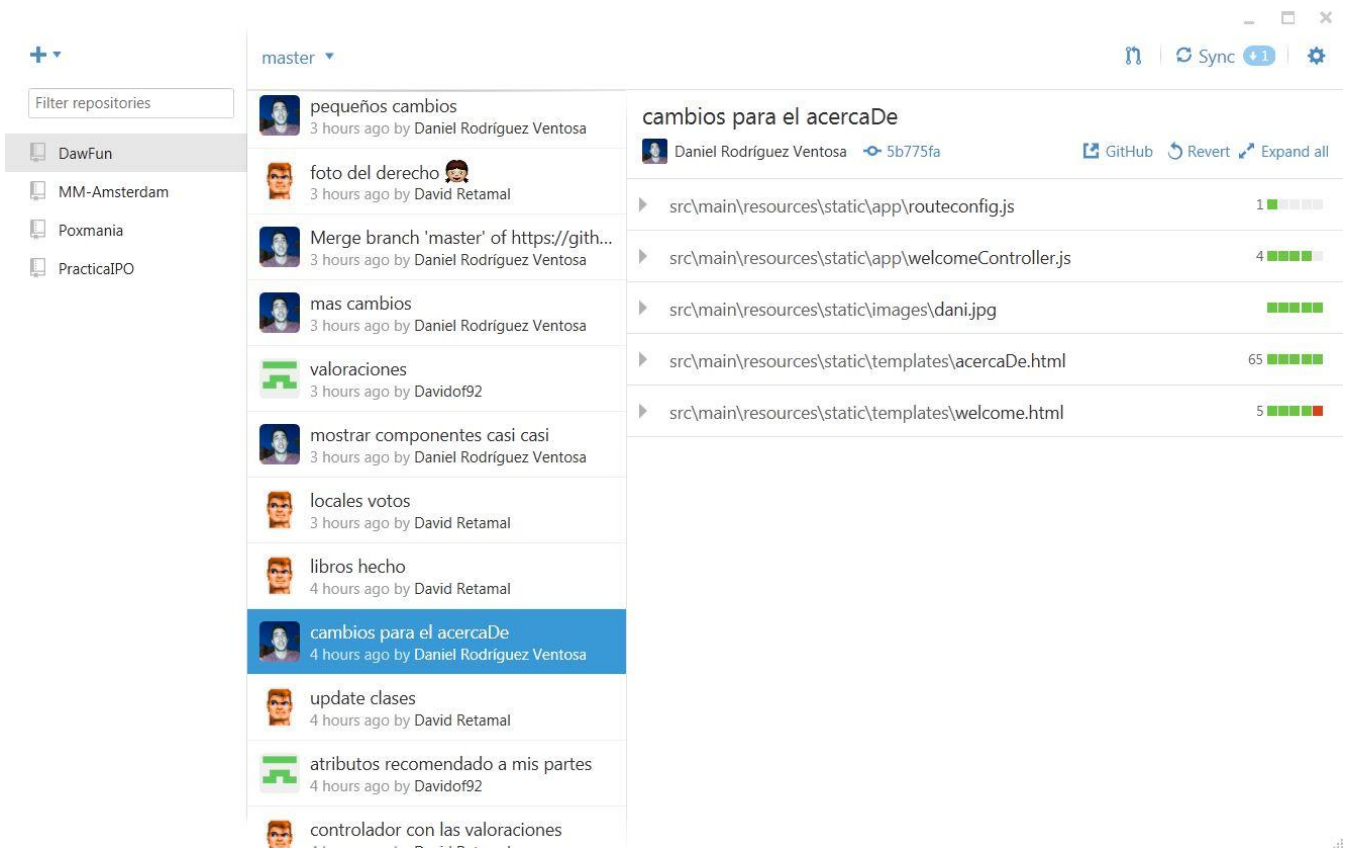
Motivación

La aplicación “DAW always Fun” está diseñada como un sistema de sugerencias en el cual los usuarios pueden valorar, tanto positiva como negativamente, los juegos, películas, videojuegos, locales de ocio , actividades al aire libre y parques temáticos que el administrador de la aplicación ha decidido incluir. La finalidad de la aplicación es que otros usuarios de la comunidad puedan interesarse en dichas actividades y puedan ponerse en contacto con las empresas externas que realizan las actividades o extraer información acerca de los productos en los que puede estar interesado.

https://youtu.be/1urTckrK5_U

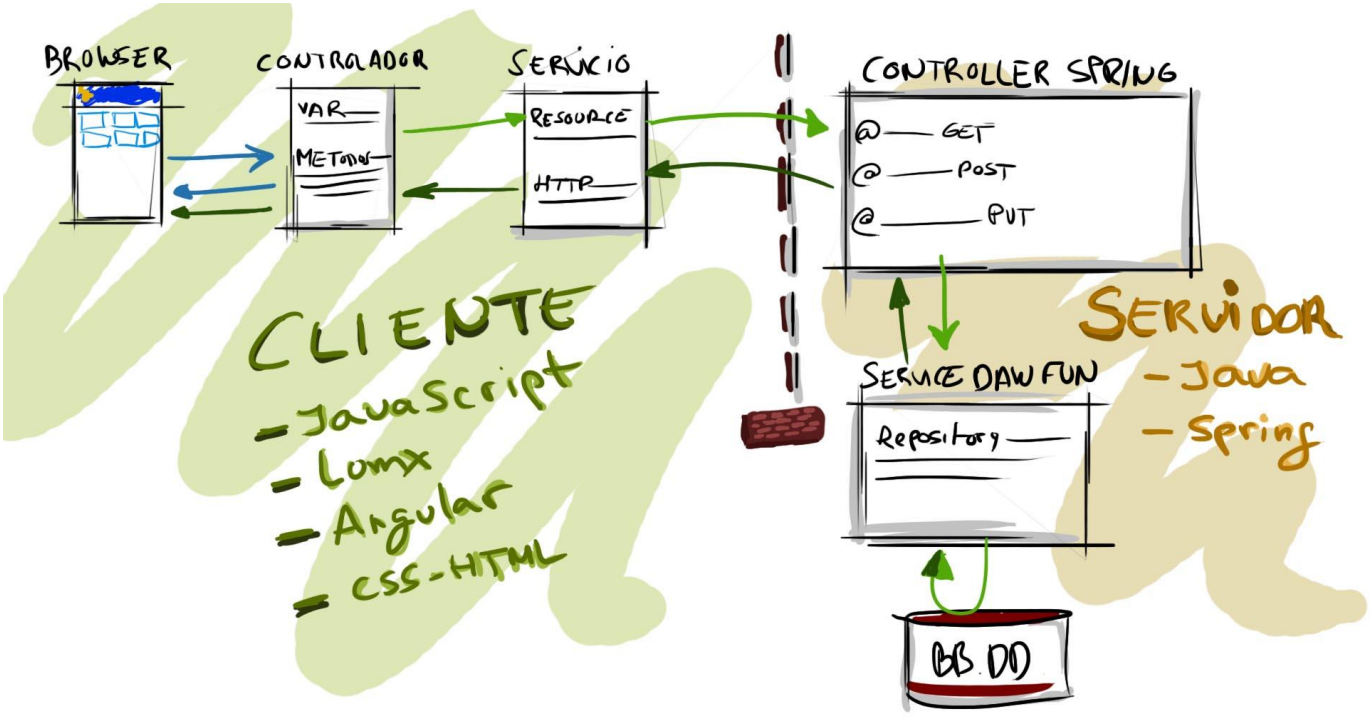
Antes de entrar en materia, debíamos tener instalada la infraestructura necesaria para poder empezar a programar. Hablamos del control de versiones, obligatorio si no queremos hacer de la práctica un infierno. El C.V. elegido es **GitHub**, debido a su facilidad de instalación, popularidad y eficacia. Si desea acceder a nuestro repositorio público de **MMAmsterdam** en **GitHub**, haga clic en el siguiente enlace:

https://github.com/danirodriguezv/DawFun



Tecnologías usadas

En el back-end (servidor) se ha utilizado la tecnología Spring MVC, la cual está basada en java, para poder desarrollar una API REST que permite la conexión con bases de datos relacionales (H2 y MySQL). La estructura de la API REST sería de la siguiente manera, el controlador será el encargado de recibir las peticiones web, el controlador se comunica con el servicio mediante la inclusión de @AutoWired serviceDawFun y el servicio hace de intermediario entre el controlador y el acceso a la base de datos. El servicio deberá tener @Autowired "nombre del repositorio" por cada repositorio creado y que se acceda desde este servicio. En nuestro caso son siete repositorios (BookRepository(almacenar los libros), LocalRepository(almacenar los locales), MovieRepository(almacenar las películas), OutdoorRepository(almacenar las actividades), TheParkRepository(almacenar los parques), UserRepository(almacenar los usuarios) y VideogamesRepository(almacenar los videojuegos).



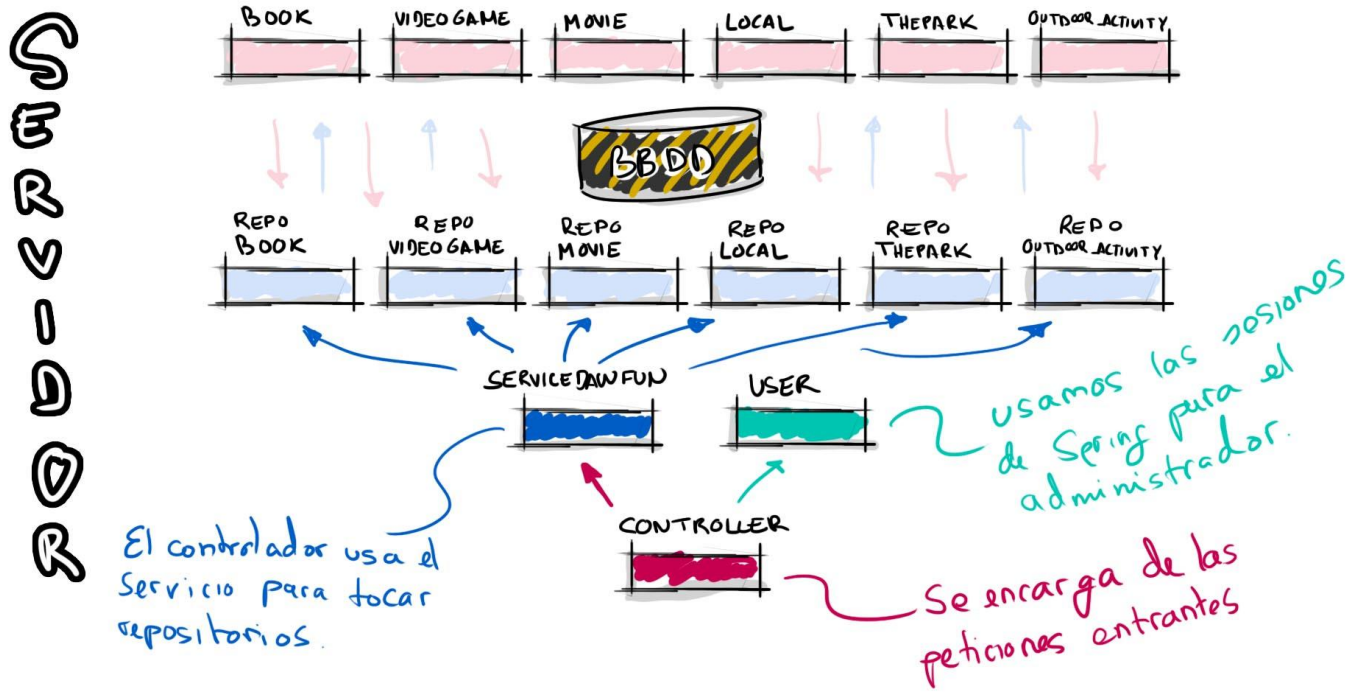
En nuestra aplicación web no tenemos usuarios porque no los vimos necesarios para este planteamiento pero tenemos una base de datos para guardarlos en caso de ampliar la funcionalidad más adelante.

Para las sesiones utilizamos la clase User la cual tiene un atributo llamado "admin" que indica mediante la devolución de un booleano si es administrador o no lo es.

Para añadir productos tenemos que hacer una petición por POST que se encuentra en cada uno de los métodos del controlador en los que queremos añadir un producto.

Para editar los productos hemos tenido que utilizar una petición mediante PUT para poder modificar los valores.

Cada clase tiene un atributo class_type que indica a que clase pertenecen para posteriormente hacer la búsqueda en la parte del administrador según el tipo que sea.



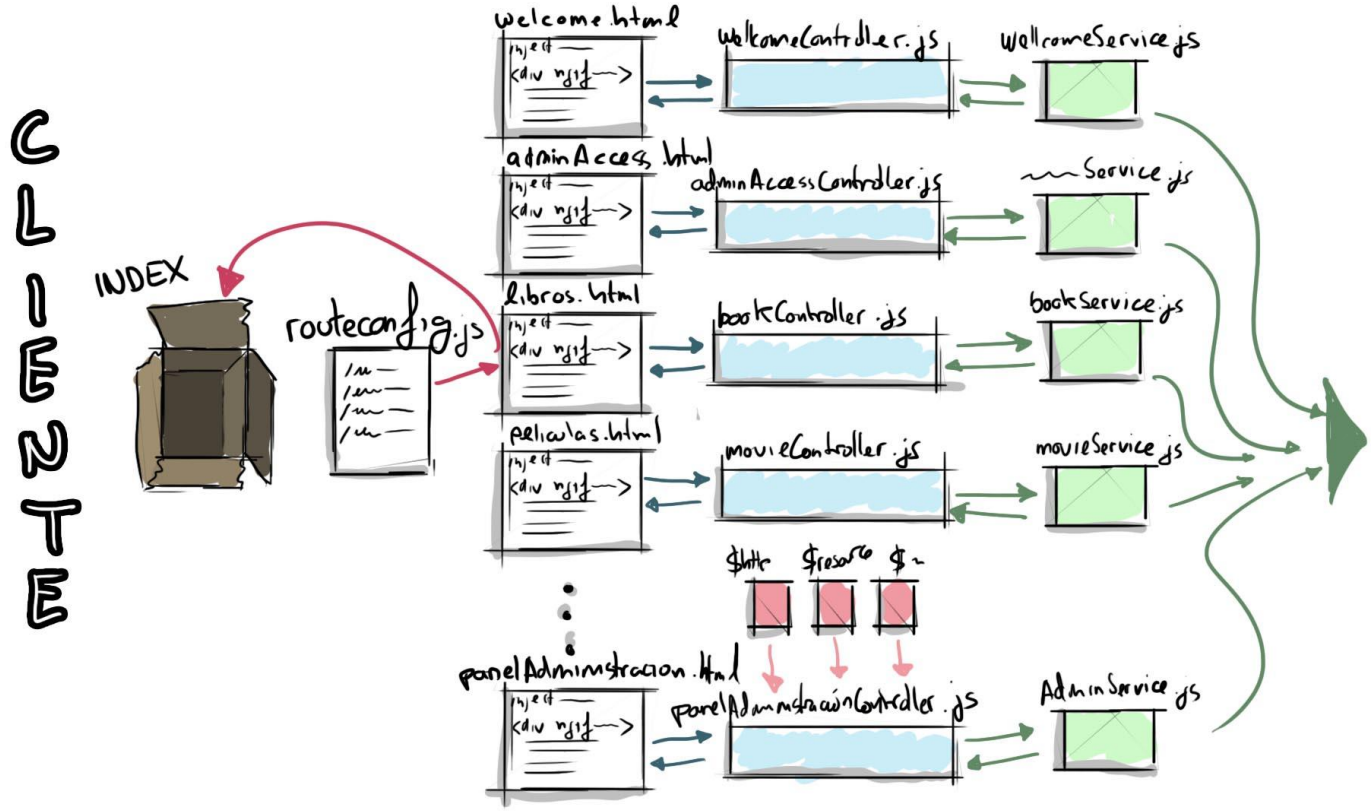
En cuanto al front-end (cliente), se ha utilizado el framework basado en javascript AngularJs. Por medio de este framework se realizan las peticiones necesarias a la API REST y se muestra la información al usuario final con ayuda del framework de diseño LumX, el cual dota a la aplicación de una apariencia familiar e intuitiva.

La tecnología Angular se basa en tener un index.html en el cual hay una parte en la que se mostrará el contenido dinámicamente y esa parte se la indicaremos con la directiva ng-view. Se trata una aplicación SPA(Single Page Aplication) ya que dentro de index.html se mostrarán las distintas plantillas de forma dinámica. Las peticiones al ser javascript se realizan en segundo plano dotando a la aplicación de una fluidez y tiempos de espera en recarga menores.

La parte mágica de Angular JS viene cuando tenemos que comunicar la parte del cliente con la parte servidor. Ahí entra en juego $resource el cual “mágicamente” se comunica con la parte del back-end mediante las URLs. En alguna ocasión hemos necesitado que el controlador java nos devolviera un tipo primitivo. En estos casos hemos usado el servicio $http, ya que $resource no trabaja a tan bajo nivel.

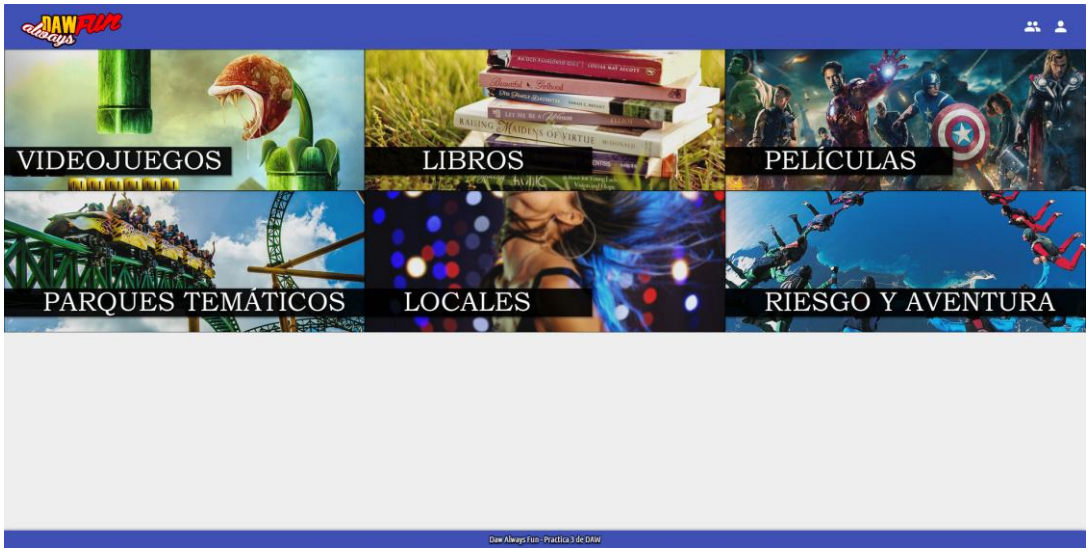
En la parte del back-end el funcionamiento es similar, el controlador se comunica con el servicio. Después el servicio será el encargado de hacer las correspondientes Querys.

El lenguaje Javascript es un lenguaje de tipado dinámico, esto quiere decir que una variable puede tomar distintos valores en tiempo de ejecución teniendo que estar muy pendiente del determinado valor que toma.

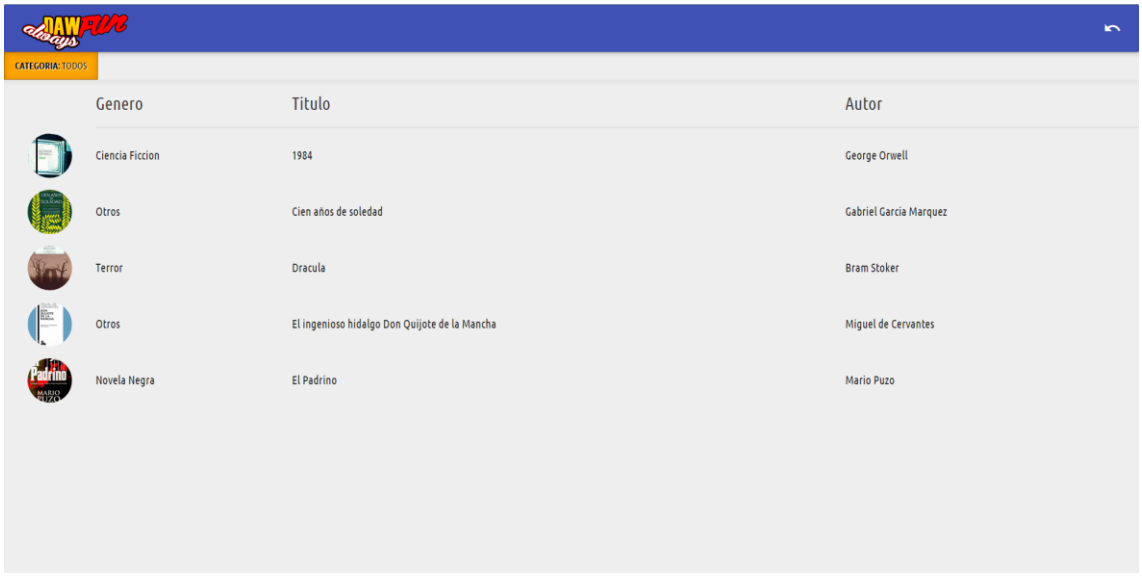


Navegación por la aplicación

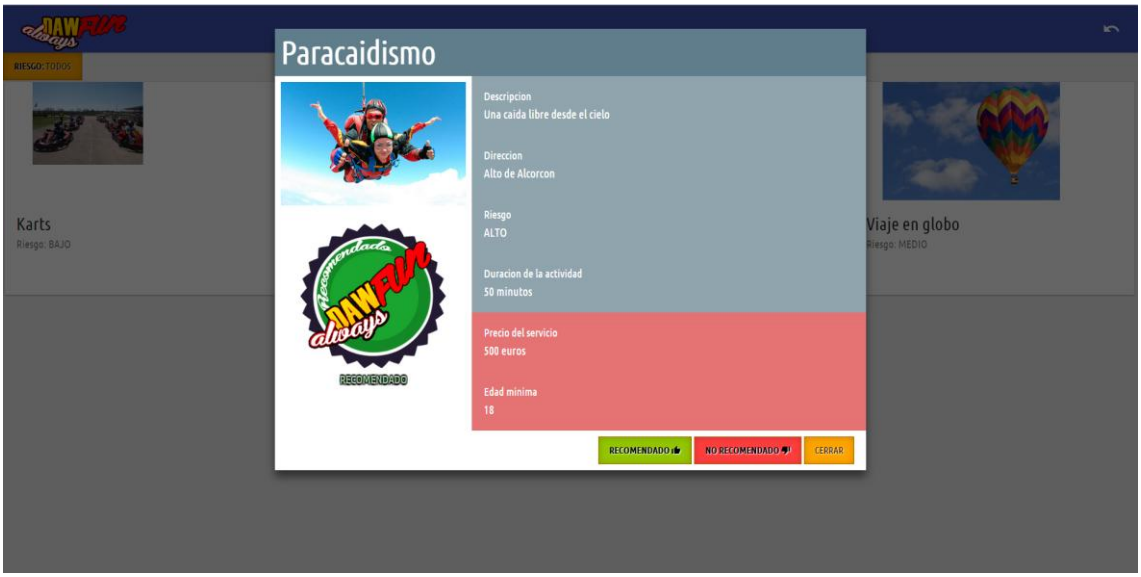
En la pantalla principal de la aplicación “DAW always Fun” se muestra una lista de categorías por las que el usuario puede filtrar los distintos tipos de sugerencias. Además en la esquina superior derecha se encontraran a lo largo de toda la aplicación botones con funcionalidad útil, como por ejemplo volver a la pantalla anterior o permitir la identificación y la desconexión de los usuarios.



Cuando el usuario hace click sobre cualquier categoría se le presenta una nueva lista con todas las sugerencias concretas referidas a dicha categoría. Dentro de esta pantalla el usuario podrá filtrar los resultados mediante los distintos atributos de interés que se muestran en una barra horizontal y ordenarlos alfabéticamente haciendo click sobre la columna sobre la cual quiere ordenar.



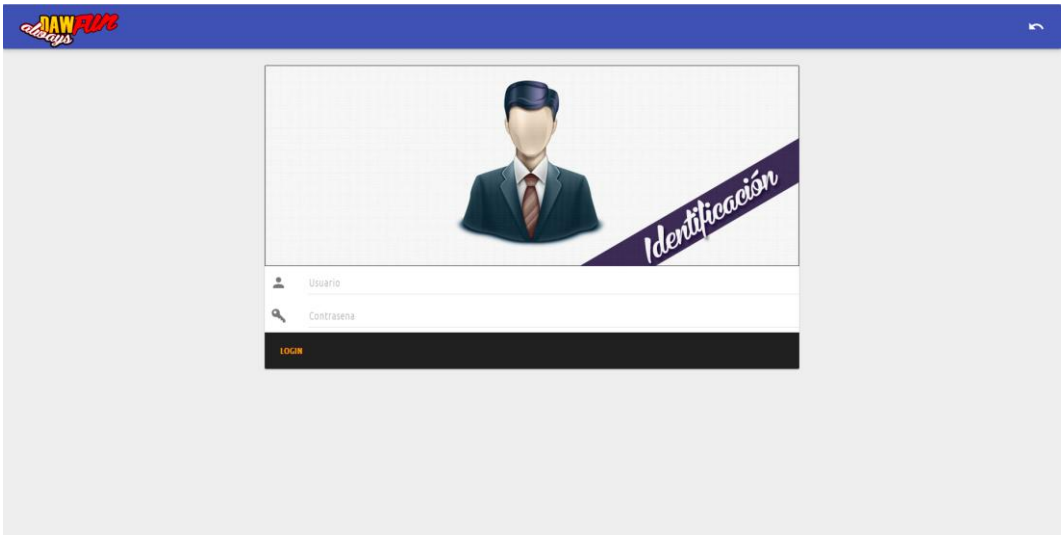
Para que el usuario pueda ver más información sobre alguna sugerencia en concreto deberá hacer click sobre ella. A continuación se le mostrara en forma de dialogo toda la información disponible y el sistema de votos que determinará si una sugerencia es recomendada o no, teniendo en cuenta la cantidad de usuarios que han votado positiva y negativamente.



En cuanto a la parte de administración, el usuario deberá identificarse introduciendo su usuario y contraseña. Esta sección de la aplicación es restringida, es decir, no es accesible para todos los tipos de usuarios. Para poder realizar las pruebas necesarias y poder acceder se ha puesto por defecto como usuario administrador, “admin”, y como contraseña “1234”. Este usuario se deberá eliminar al finalizar el desarrollo.

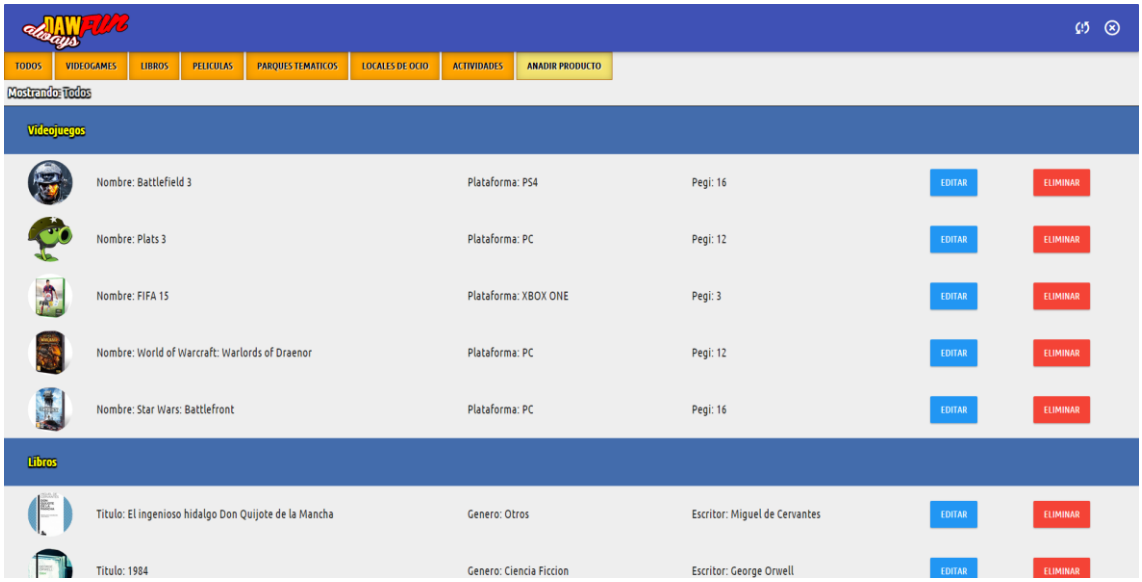
La validación del administrador funciona de la siguiente manera:

- El usuario escribe su identificador y contraseña, al darle al botón, se recogen los dos campos y se envían al servidor mediante una petición POST (en el cuerpo de la petición, oculta).
- Esta información es recogida por el Rest-Controller y procesada. Si el acceso es válido, pone a true el booleano de la sesión (sesiones de Spring) y devuelve una confirmación.
- Este booleano que confirma si es admin o no, es recogido mediante el servicio $http, ya que $resource no recoge tipos primitivos.
- Si la respuesta es verdadera, usamos el servicio $location para “navegar” a la plantilla de administración.
- En esta plantilla de administración, siempre comprobaremos si el usuario es admin cada vez que se recargue la página. Si no lo fuera, sería expulsado a la página principal.

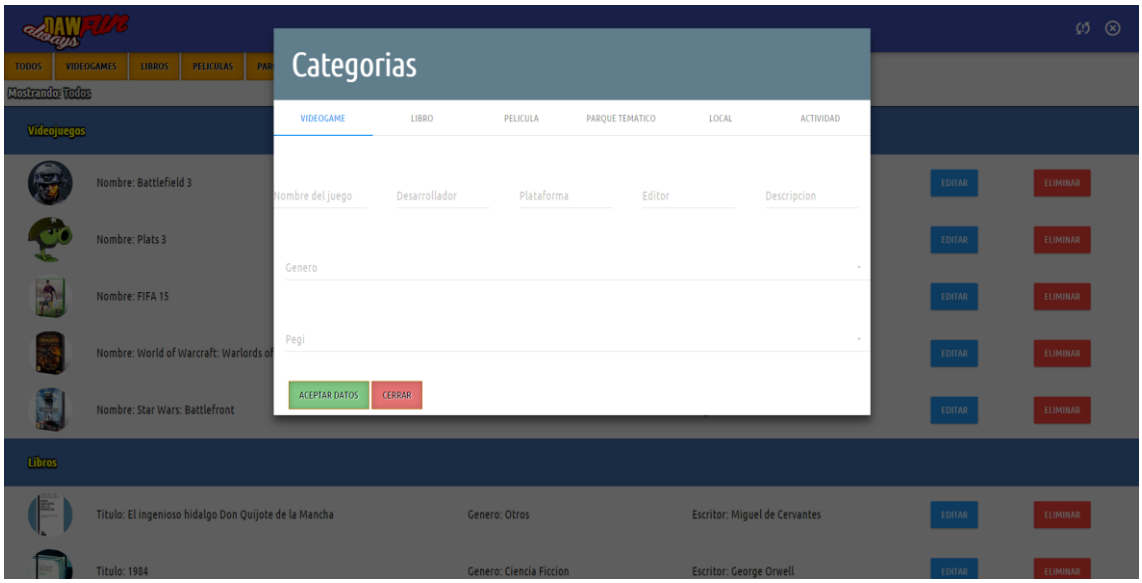


Una vez el usuario se ha identificado correctamente se le presentara una pantalla con todas las sugerencias disponibles que se han almacenado en la base de datos. Además se le permitirá realizar toda la gestión de dichas sugerencias, pudiendo así tanto añadir nuevas sugerencias, como modificar o eliminar las ya existentes. Para agilizar la labor de gestión de sugerencias, se ha implementado un sistema de búsqueda similar al que tienen los usuarios normales permitiendo que el administrador pueda filtrar las sugerencias por categorías.

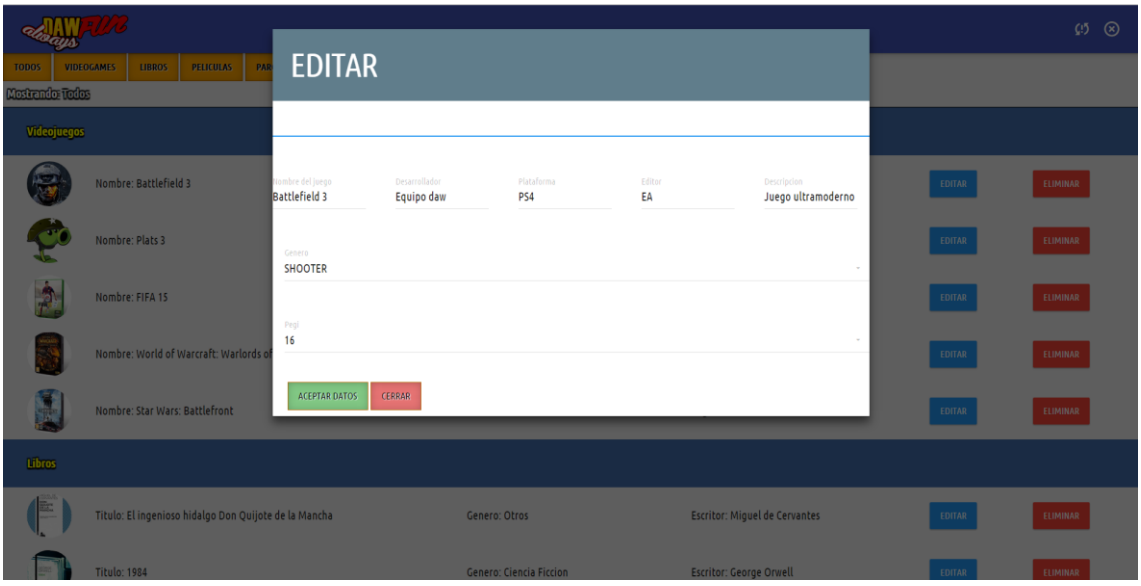
El administrador durante toda la interacción tiene visible dos botones en la parte superior derecha, mediante los cuales podrá refrescar la página y desconectarse.



Al pinchar en añadir producto, se abrirá un dialog en el que seleccionaremos la categoría deseada y rellenaremos la información necesaria.



Si pulsamos en editar, editaremos el elemento correspondiente de la tabla. Todas estas operaciones críticas, tienen su correspondiente validación en el servidor en el remoto caso de que un usuario no autentificado consiguiera colarse en el panel de admin. Para evitar esto además, se hacen comprobaciones de administrador cada vez que se recarga el panel.



EXTRA

A lo largo del desarrollo de DawAlwaysFun, nos hemos topado con bastantes contratiempos que hemos podido solventar. Algunos de ellos fueron:

- Conseguir la respuesta del servidor si el usuario es administrador. Era necesario $http con querys personalizadas.
- Conectar correctamente Lumx. Tuvimos al principio algunos problemas con las versiones antiguas.
- Pensar el sistema de clasificación y ordenamiento.
- Desconexiones de controladores y servicios por falta de las inyecciones necesarias.

A pesar de esto, hemos podido adentrarnos un poco en la parte optativa, por lo menos, en lo referente a incluir un video de youtube en la aplicación web. Algo que a priori es trivial, resulta que no lo es tanto porque Angular identifica las urls y todas aquellas que son externas, las marca de inseguras. Para solucionar este problema debemos inyectar los servicios $sce , $sceDelegateProvider y $sceDelegate y cambiar la configuración de la “app” para añadir a la “whitelist” la dirección de Youtube.

Con el siguiente código hemos podido embeber videos en nuestra aplicación.

```
angular.module("app").config(function($sceDelegateProvider) {
        $sceDelegateProvider.resourceUrlWhitelist(['self',
          "https://www.youtube.com/embed/**"
        ]);
});
```