C.I.F.P. “JUAN DE COLONIA”

Departamento de Informática

2º D.A.W

Adrián Hernando Gavilán

Curso: 2021/22

LIVEFÚTBOL

MEMORIA

**ÍNDICE**

[***1.*** ***INTRODUCCIÓN*** 2](#_Toc103977082)

[*1.1.* *Descripción* 2](#_Toc103977083)

[*1.2.* *Justificación* 2](#_Toc103977084)

[***2.*** ***PLANIFICACIÓN*** 2](#_Toc103977085)

[*2.1.* *Requisitos* 2](#_Toc103977086)

[2.1.1. Requisitos funcionales 2](#_Toc103977087)

[2.1.2. Requisitos no funcionales 2](#_Toc103977088)

[*2.2.* *Recursos* 4](#_Toc103977089)

[2.2.1. Recursos software 4](#_Toc103977090)

[*2.3.* *Planificación temporal* 6](#_Toc103977091)

[*2.4.* *Planificación económica* 8](#_Toc103977092)

[***3.*** ***TECNOLOGÍAS*** 8](#_Toc103977093)

[*3.1.* *Diseño* 10](#_Toc103977094)

[3.1.1. Diagrama de aplicación 10](#_Toc103977095)

[3.1.2. Diagrama E/R y esquema de BBDD 13](#_Toc103977096)

[3.1.3. Diseño de interfaces 14](#_Toc103977097)

[*3.2.* *Pruebas* 17](#_Toc103977098)

[***4.*** ***CONCLUSIONES FINALES*** 27](#_Toc103977099)

[*4.1.* *Grado de cumplimiento de los requisitos fijados* 27](#_Toc103977100)

[*4.2.* *Propuestas de mejora o ampliaciones futuras* 28](#_Toc103977101)

[***5.*** ***GUÍAS*** 28](#_Toc103977102)

[***6.*** ***REFERENCIAS*** 28](#_Toc103977103)

# ***INTRODUCCIÓN***

## *Descripción*

En el siguiente documento se describe el proceso de creación de LiveFútbol, una aplicación enmarcada en el Ciclo de desarrollo de aplicaciones web. La App está destinada al sector del deporte, concretamente dirigida al futbol.

Para un hincha o aficionado, es tan importante poder ir al campo a ver a su equipo como tener cerca un medio con el que conocer de una manera sencilla y sobre todo accesible las estadísticas tanto de su equipo como de otros diferentes, esta aplicación pretende dar esa información.

Falta destacar que esta aplicación esta subida en el siguiente dominio: devadrian.es

## *Justificación*

La idea surge por mi experiencia con este tipo de aplicaciones, ya que las uso constantemente debido a mi pasión por el futbol. La idea principal era conocer cómo podría hacerla yo mismo con el conocimiento que tengo hasta ahora y a partir de ahí utilizarla como mi app de referencia que vaya a utilizar en un futuro, poco a poco mejorándola cada vez más.

# ***PLANIFICACIÓN***

## *Requisitos*

### Requisitos funcionales

### Requisitos no funcionales

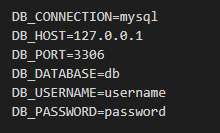
Despliegue de la aplicación en entorno local:

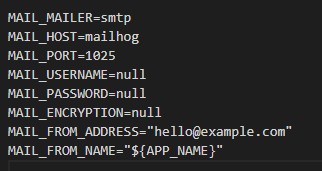
Descargar el proyecto desde mi repositorio publico <https://github.com/devadrian16/proyecto_laravel>

Gracias a Artisan (una interfaz de comandos que incluye Laravel) ejecutarlos siguientes comandos desde la ruta del proyecto para acabar de ajustar la configuración.

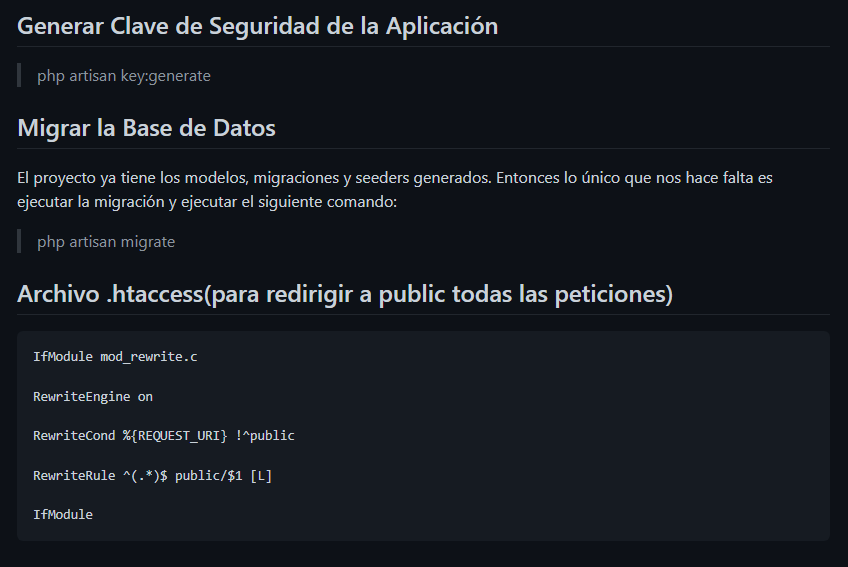


Como dice en el último párrafo será necesario cambiar los siguientes parámetros en el archivo “.env” respecto a la configuración de la base de datos, servidor de correo electrónico para poder recuperar la contraseña de una cuenta y la key de la API:









Ya tendríamos la aplicación lista para ejecutarla de manera local con todas las configuraciones hechas.

Es destacable mencionar que se puede hacer correr una aplicación con el servidor que viene incluido con Laravel (utiliza el puerto 8000) de manera fácil y rápida con el siguiente comando:





## *Recursos*

### Recursos software

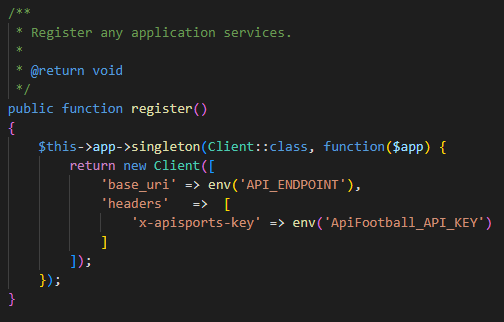
Conexión API:

Para esta aplicación que he elegido hacer, un recurso imprescindible y casi el más importante diría yo es la utilización de una API, tenía ganas de poner en práctica el uso de algo así ya que me llama la atención por la magnitud que puede llegar a tener.

Así que me puse a buscar un API por internet sobre resultados de futbol, que tuviera un poco de todo sobre este deporte, hay un montón de ellas en el mercado, más profesionales, menos profesionales, con planes gratis y solo pagando (y no poco dinero).

Después de investigar durante un tiempo encontré la siguiente, <https://www.api-football.com/,> y utilizando el plan gratis que tiene (claramente me limita en algunos aspectos) decidí montar mi aplicación en base a esta API.

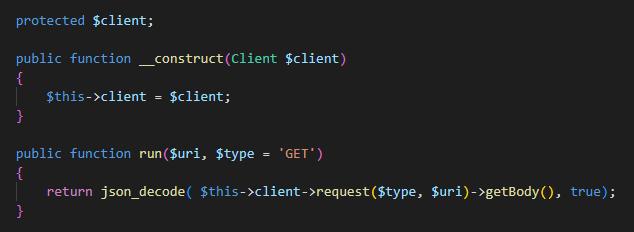
Mediante GuzzleHttp y el siguiente método realizo la conexión con la API

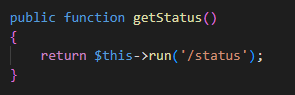




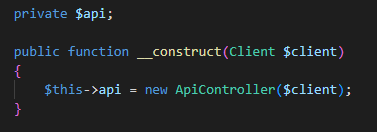
Este es un ejemplo de la clase donde están todos los métodos que necesito para hacer peticiones a la API

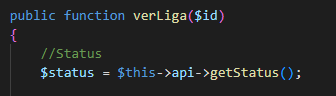






Este es un ejemplo de cómo desde la clase LigaController llamo a la clase de la API para recoger el status que me devuelve información sobre el estado de mi suscripción.





## *Planificación temporal*

En este punto, se indicarán las etapas para el desarrollo y la planificación temporal que he llevado a cabo en la realización del proyecto:

Sobre investigación y aprendizaje:

**En primer lugar**, fue elegir que plataformas quería utilizar, después de un par de semanas trabajando en las practicas del ciclo con Laravel PHP me di cuenta de que me facilitaba las cosas mucho para algunas cosas que me interesaban para mi aplicación, es por ello que decidí enfocarlo de esta manera.

Por lo demás seguí utilizando Visual Studio Code ya que me era igualmente compatible y MySql con PhpMyAdmin por lo mismo.

**Después** tuve que ponerme al día con el framework mencionado anteriormente, no me llevo mucho tiempo aprender lo básico debido a su corta curva de aprendizaje, aunque hay partes que todavía intento mejorarlas, es por eso que al final he reescrito bastante código, porque a la hora de ir aprendiendo más me daba cuenta de cómo podía hacer las cosas mejor (con menos código y más legible).

Desarrollo de la propia aplicación:

La aplicación **empezó** con pruebas sobre la conexión con la API, al terminar encontrando una que se acercaba a mis necesidades y después de varias pruebas, acabe dando con cómo realizar esa conexión y así ya tendría la manera de recoger aquellos datos que me interesaban.

**Seguí** con la ventana de Ligas donde lo utilicé para trastear o jugar con los datos que me llegaban y así saber exactamente lo que podía llegar a hacer con estos junto a las funcionalidades de laravel.

**Después de esto** pase a la realización del Login, pensé en el tiempo que me llevaría y buscando un poco encontré JetStream la cual me ayudo instalando unos paquetes en mi proyecto a hacerlo más sencillo y sobre todo darme unas funcionalidades como autenticación de dos factores y verificación entre otros.

**Al terminar** con lo anterior me centre en la parte de LiveScore, la más importante de la aplicación diría yo, es de las partes que más tiempo me ha llevado debido a todos los parámetros que hay que tener en cuenta, como si ha empezado el partido, cuando no haya empezado cuanto tiempo queda, información acerca de ellos, etc. Pero con todo lo que había trasteado anteriormente fue un reto aceptable.

**Después** seguí con la parte de Equipos y Favoritos, después de elegir como iba a ser el funcionamiento de estas y realizarlo pensé que sería mejor que al elegir un equipo como favorito no se recargara la página, y así finalmente lo hice con Ajax.

**Con todas las funcionalidades** que tenía pensado hechas, durante un tiempo me centre en buscar unos colores para darle algo de vida a la aplicación (escogí unos verdes por la relación con el césped de un campo de futbol) y en maquetar todas las pantallas para que siguiera un diseño Responsive junto a Bootstrap además de CSS puro.

**Por último**, a la vez que voy terminando esta memoria darle a la aplicación pequeños retoques que mejoran su uso, así como el funcionamiento, por ejemplo, en las pantallas que muestran partidos, sean en directo o no, poder ver más estadísticas sobre estos mediante un botón en su lado derecho y una ventana emergente utilizando solo en este caso un widget que me ofrece la API.

Tabla 2‑1: Planificación temporal del proyecto

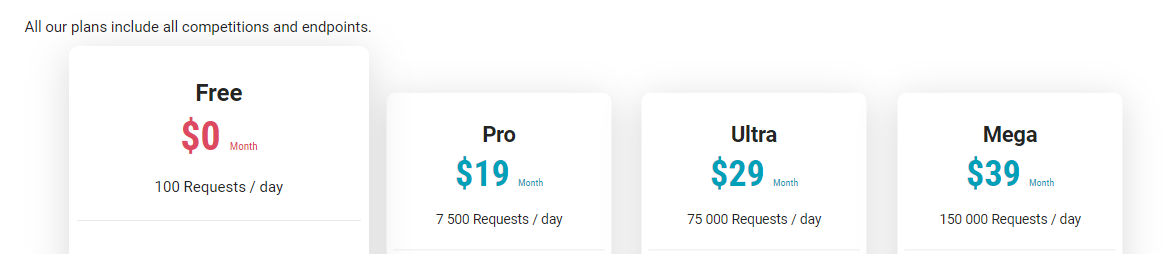
|  |  |
| --- | --- |
| **Descripción de la tarea** | **N.º semanas** |
| Investigación y aprendizaje | 3 |
| Pruebas API | 1 |
| Pantalla Liga | 2 |
| Login JetStream | 0,5 |
| Pantalla LiveScore | 2 |
| Pantalla Equipos y Favoritos | 1,5 |
| Colores y maquetación | 0,5 |
| Retoques | 0,5 |
| **SEMANAS TOTALES** | **10,5** |

## *Planificación económica*

En este apartado se mostrarán la posible planificación económica necesario para el proyecto, donde estarían dos principales componentes, la API y el hosting.

**API (api-football.com)**

En la siguiente imagen se muestran los planes que ofrecen, en este caso yo he elegido la opción gratis ya que me da un servicio bueno para presentar la aplicación como proyecto de fin de ciclo.

Aunque en el mínimo caso de que la aplicación se usara por mas de una persona y creciera sería muy recomendable al menos elegir el plan Pro, ya que si no sería muy poco eficiente debido a las restricciones que te da el gratis.

**HOSTING (devadrian.es)**

En el apartado del hosting, algo esencial en una pagina web desde mi punto de vista, al principio estuve usando uno gratuito y era mi idea seguir con el hasta que me encontré con que la aplicación había crecido tanto que por lo que probe ningún hosting gratuito tenía la capacidad necesaria.

Debido a ello elegí hostalia.com básicamente porque tenía muy buenas ofertas y a la vez prestaciones como almacenamiento ilimitado y 2 dominios incluidos entre muchas otras.

*Tabla 2‑2- Presupuesto del proyecto*

|  |  |
| --- | --- |
| **Descripción de la tarea** | **Precio** |
| API-Football | 0€ - 18€ /mes |
| Hosting | 1,25€ /mes |
| **TOTAL** | **1,25€ - 19,25 /mes** |

# ***TECNOLOGÍAS***

**PHP (Laravel)**

Laravel es un framework de PHP que nos simplifica la construcción de aplicaciones web robustas de forma rápida y minimiza la cantidad de codificación necesaria.

Características:

Su arquitectura es conocida como MVC (Modelo-Vista-Controlador) que da muchas facilidades para relacionar de manera sencilla todas las partes de una aplicación.

Utiliza un motor de plantillas, llamado Blade, capaz de utilizar sus propias variables y reutilizarlas con el poder de hacer unas páginas visualmente muy potentes.

Funciona con Artisan, que es el nombre que le dan a la interfaz de comandos para ejecutar muchas funcionalidades como arrancar la aplicación o pararla o ver todas las rutas disponibles, además de muchas otras.

Eloquent ORM que es muy intuitivo para escribir consultas PHP sobre objetos y bases de datos.

Migraciones, permite actualizar y migrar la base de datos una vez que el desarrollo ya este comenzado, gracias a esto el riesgo de perder datos sean del valor que sean es mínimo.

Ventajas y desventajas:

Entre las ventajas de Laravel quizás la más destacada es que es fácil de usar y aprender además de tener una amplia documentación e información por internet.

Por otro lado, la principal desventaja es que algunas librerías y demás dependen de Symfony que es otro framework del mismo estilo.

**JavaScript**

Se utiliza principalmente en su forma del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas.

La función que ha tenido este lenguaje en mi aplicación no ha sido tan densa como la de otros lenguajes, pero me ha ayudado en la parte del front-end a darle en algunos ámbitos un toque de realismo y accesibilidad.

También en una pequeña parte he utilizado jQuery, una biblioteca de JavaScript para abrir unas ventanas modales con la información de las estadísticas de los partidos ya que me facilitaba el código.

JQuery permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.

**Bootstrap**

Bootstrap es un framework CSS utilizado en la parte front-end, es decir, en la pantalla de interfaz con el usuario para desarrollar aplicaciones que se adapten a cualquier dispositivo (Diseño Responsive), está constituido por una serie de archivos CSS y JavaScript responsables de asignar caracteristicas especificas a los elementos de la página.

Razones para utilizarlo:

Bootstrap sigue el concepto de mobilefirst, que significa que la preocupación del framework es, primero, desarrollar una página que funcione perfectamente en dispositivos móvil y luego en escritorio.

Para desarrollar muchas de las caracteristicas existentes en Bootstrap, sería necesario escribir una gran cantidad de líneas de código lo que provoca un aumento importante en el tamaño de los archivos y en la cantidad de datos transferidos a la carga de la página.

**MySQL**

El sistema de gestión de bases de datos elegido ha sido MySql junto a phpMyAdmin que es la aplicación web que sirve para administrar bases de datos MySql de forma sencilla e intuitiva desarrollada en php.

**JetStream**

Laravel JetStream es un kit de inicio de apps diseñadas específicamente para el framework Laravel y proporciona un punto de partida para su aplicación. Proporciona la implementación para el inicio de sesión, registro, verificación y autenticación de dos factores, además de otras funciones.

**GuzzleHttp**

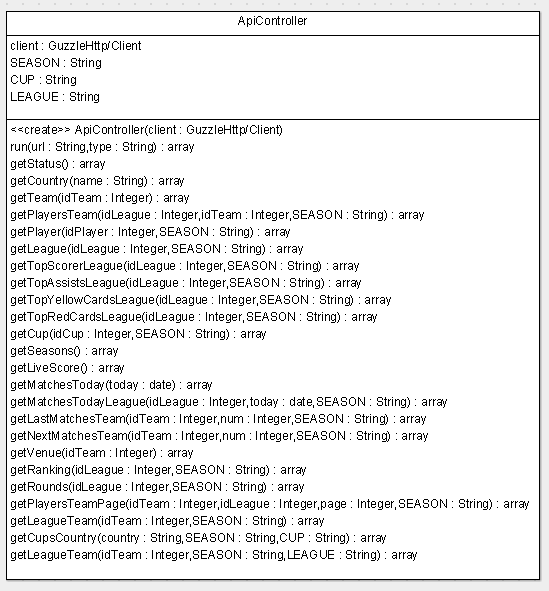
Un Guzzle en laravel es un cliente PHP HTTP que facilita él envió de solicitudes HTTP con datos, encabezados y triviales para integrarse con servicios web, esta herramienta es la que me facilito conectarme a la API que utilizo en mi aplicación.

## *Diseño*

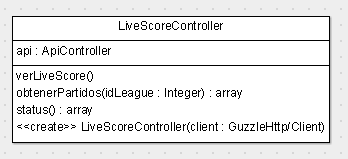
### Diagrama de aplicación

En este apartado se mostrará el diagrama de clases que se lleva a cabo a través de un conjunto de capturas y una breve descripción de las diferentes clases.

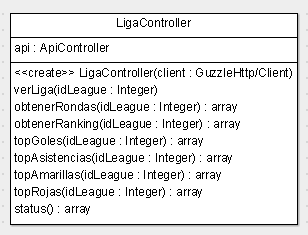
Clase ApiController: es la clase encargada de hacer la conexión con la API además de recoger mediante sus métodos los diferentes datos en forma de arrays que se utilizan para mostrarse en todas las vistas.



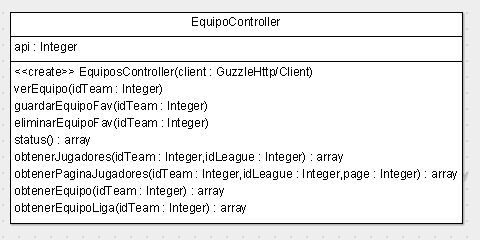
Clase LiveScoreController: es la clase encargada de mostrar la vista LiveScore.



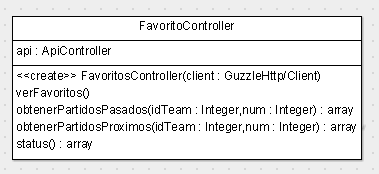
Clase LigaController: es la encargada de mostrar la vista Ligas.



Clase EquipoController: se encarga de mostrar la vista Equipos, además es el encargado de crear o eliminar registros en la base de datos (tabla favoritos) para poder registrar el equipo como favorito o no.



Clase FavoritoController: es la encargada de mostrar la vista Favoritos, hace una conexión con la base de datos para saber si el usuario registrado tiene equipos favoritos y si los tiene cuales son.



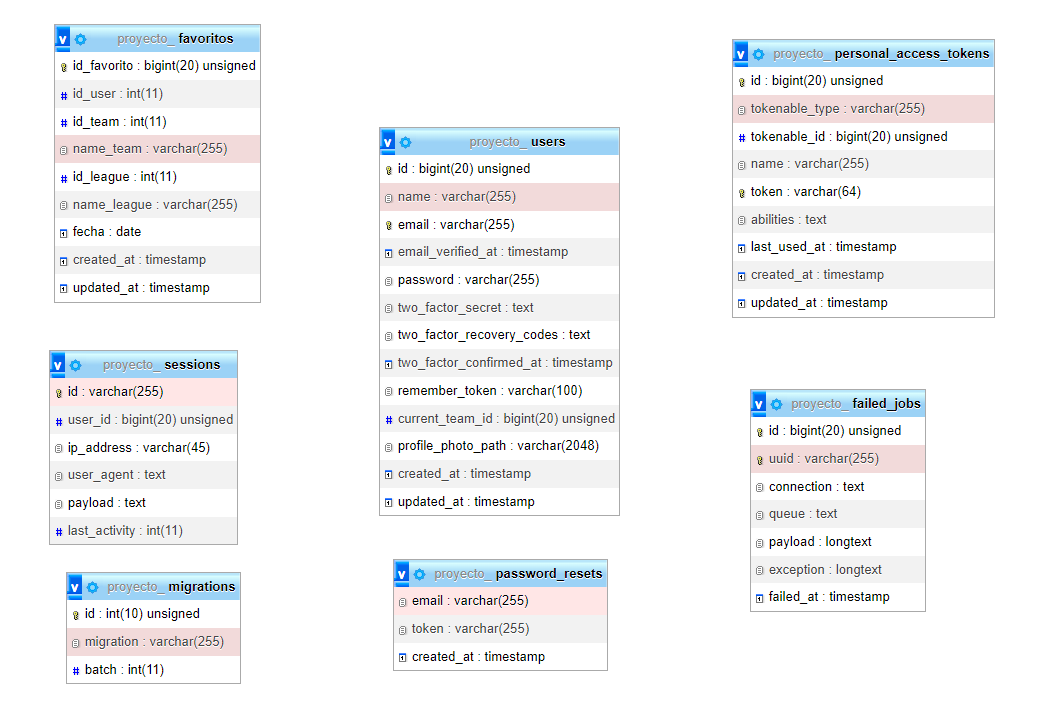
En todas las clases anteriores como se puede apreciar los constructores se conectan con la clase ApiController que es la que les ofrece mediante la API los arrays con toda la información que necesitan.

Así mismo, todas tienen un método llamado status que consulta cual es el porcentaje de uso que tiene en ese momento la API, así se puede controlar cuando el porcentaje este completo se pueda notificar en vez de mostrar un error.

### Diagrama E/R y esquema de BBDD

El modelo relacional de una base de datos define la implementación lógica de los datos mediante tablas, campos y sus relaciones.

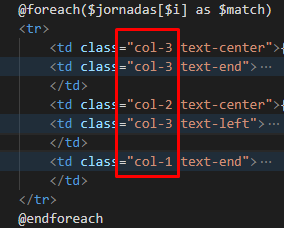
En este punto se mostrará cuál es el modelo relacional de la base de datos que se utiliza en la aplicación:



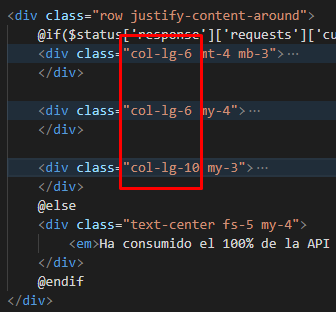
### Diseño de interfaces

En este caso he contado con la ayuda de Bootstrap para poder seguir un diseño responsive, que busca la correcta visualización de una misma página en distintos dispositivos según el ancho de la pantalla.

Ejemplo de maquetación de las jornadas:

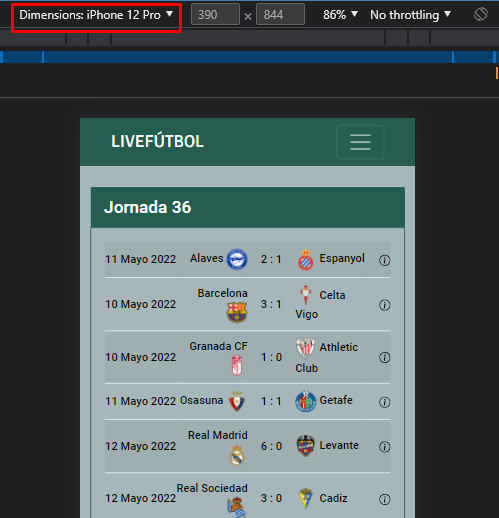


Ejemplo de maquetación de la vista Ligas:

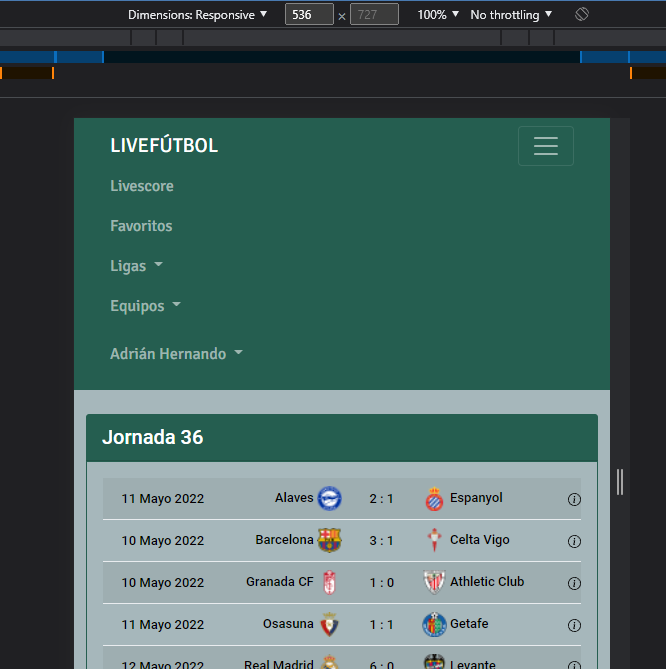


Para ello, he creado los siguientes “breakpoints”:

Dimensiones en móvil:



Dimensiones en tablets o pantallas pequeñas con un ancho menor a 768px:



Dimensiones en pantallas medianas entre 768px y 992px:



Dimensiones en pantallas grandes entre 992px y 1400px:



Dimensiones en pantallas grandes con un ancho mayor a 1400px:



## *Pruebas*

***DESCRIPCION***

Comprobar funcionamiento hipervínculos de los equipos o ligas en las diferentes pantallas de la aplicación.







***RESULTADO ESPERADO***

Según el equipo o liga que selecciones te llevara a su página.

***RESULTADO***

Equipo seleccionado:

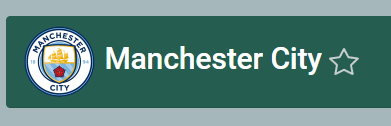


Liga seleccionada:

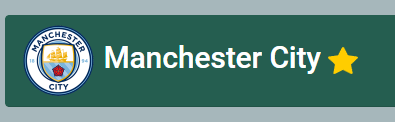


***DESCRIPCION***

Marcar como favorito un equipo y que aparezca en la página de favoritos de una cuenta concreta y a su vez desmarcar el equipo de favoritos y que desaparezca de la página favoritos.



Pulsar en la estrella para marcar como favorito.



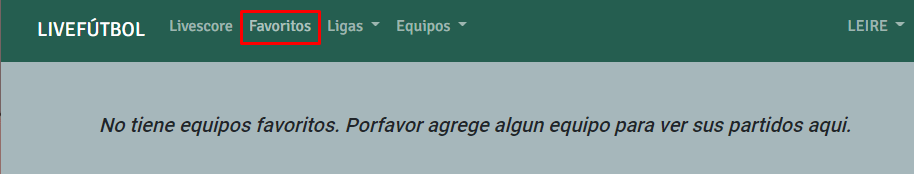
***RESULTADO ESPERADO***

Si no este marcado que no aparezca en la pantalla favoritos.

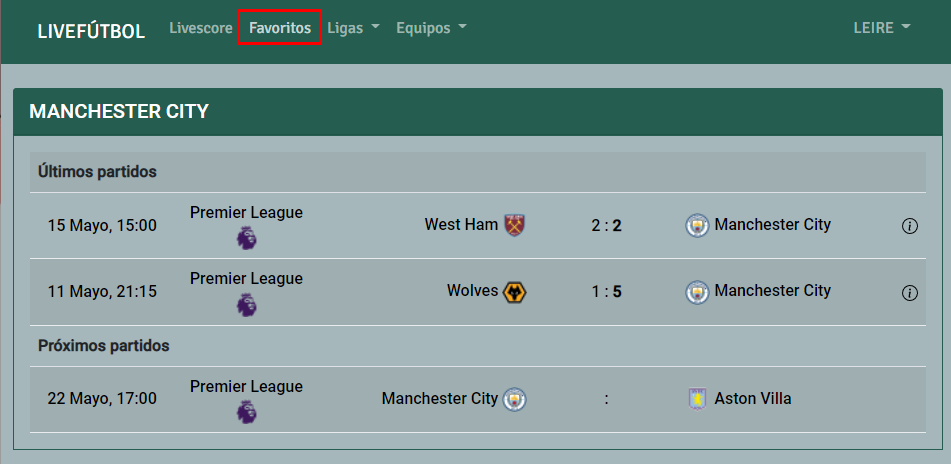
Si está marcado que se añada a la página favoritos.

***RESULTADO***

No marcado:

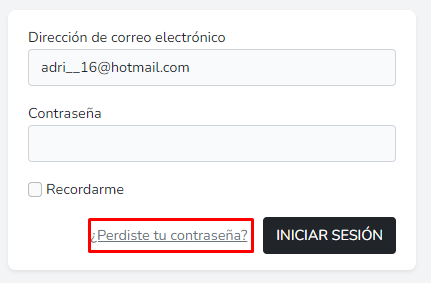


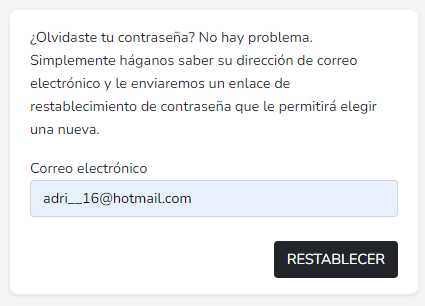
Marcado:



***DESCRIPCION***

Comprobar funcionamiento al recuperar contraseña de una cuenta.



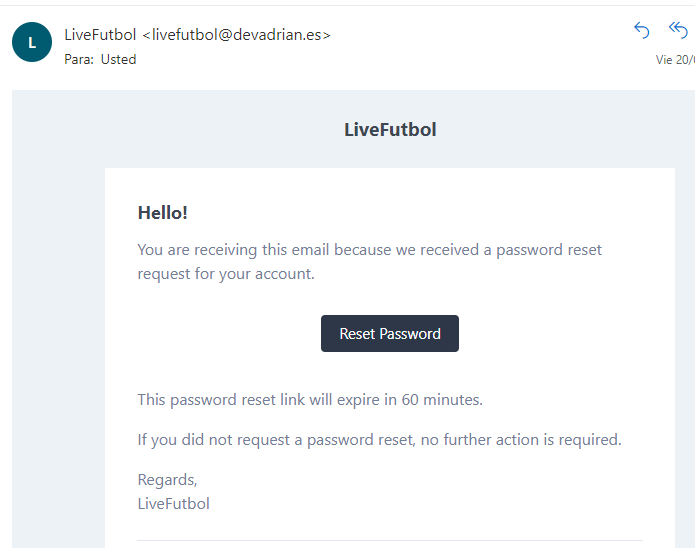


***RESULTADO ESPERADO***

Recibir un correo con un link para restablecer la contraseña.

***RESULTADO***

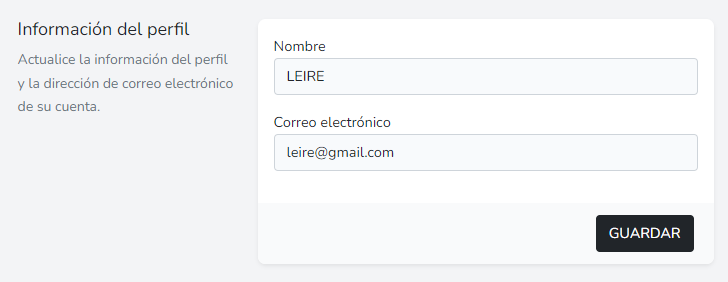
Correcto





***DESCRIPCION***

Comprobar funcionamiento de cambio de nombre y correo electrónico de un perfil.

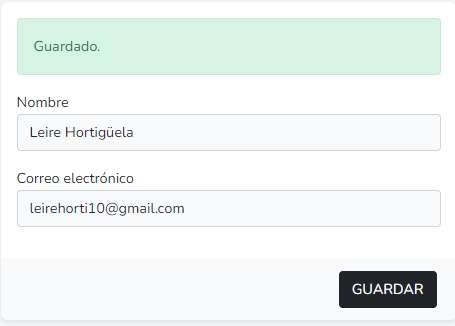


***RESULTADO ESPERADO***

Al volver a iniciar sesión solo lo conseguirá con el correo electrónico nuevo y con el correo electrónico anterior dará error.

Cambiara el nombre de la cuenta.

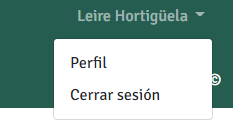
***RESULTADO***



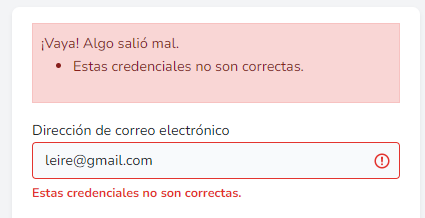
Con el correo electrónico nuevo:



Como vemos entra y además se actualiza el nombre de la cuenta.

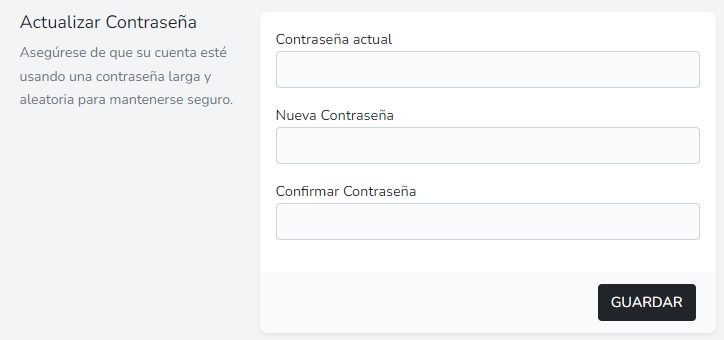


Con el correo electrónico anterior:



***DESCRIPCION***

Comprobar funcionamiento al actualizar la contraseña de un perfil.



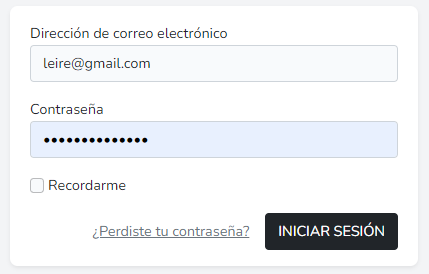
***RESULTADO ESPERADO***

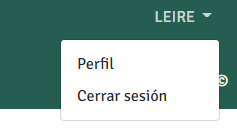
Al volver a iniciar sesión solo lo conseguirá con la contraseña nueva y con la contraseña anterior dará error.

***RESULTADO***

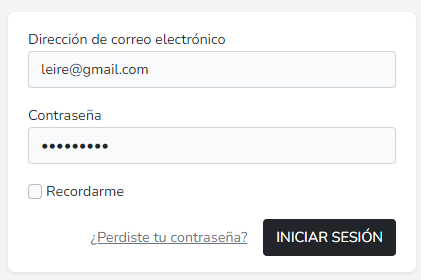


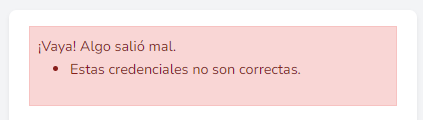
Con la contraseña nueva entra en la aplicación:





Con la contraseña anterior:





***DESCRIPCION***

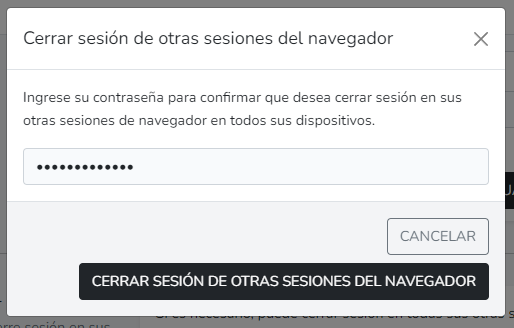
Comprobar funcionamiento al cerrar sesiones en otros navegadores o dispositivos con varias sesiones abiertas de un perfil.

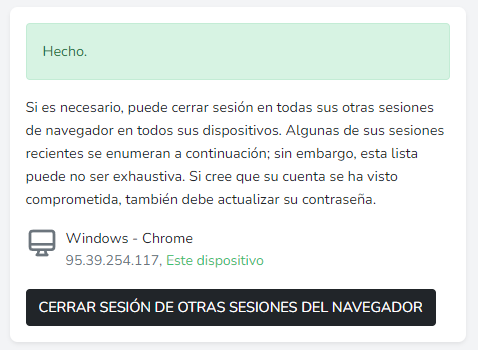


***RESULTADO ESPERADO***

Desaparecen las otras sesiones del perfil y si actualizas las sesiones en los otros navegadores o dispositivos hacen logout solas.

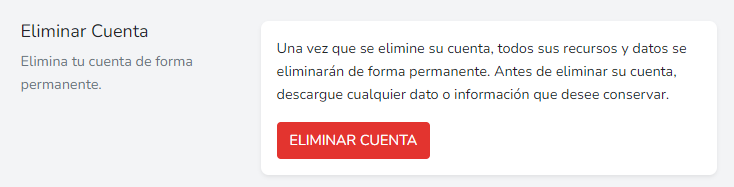
***RESULTADO***





***DESCRIPCION***

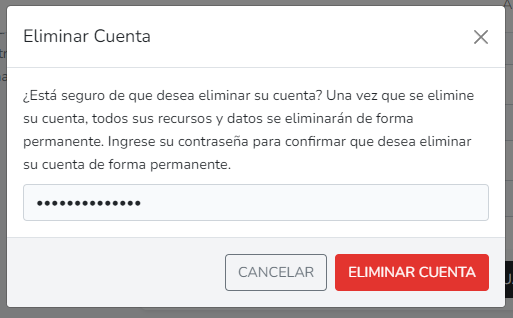
Comprobar funcionamiento al eliminar una cuenta.

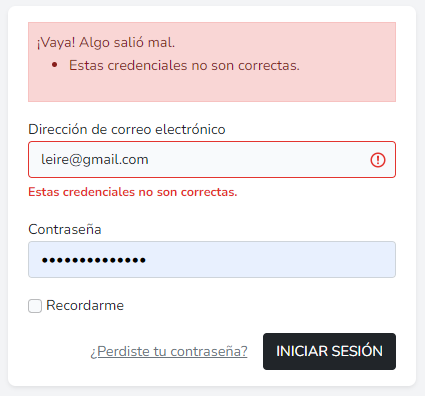


***RESULTADO ESPERADO***

Al intentar volver a iniciar sesión con las credenciales de la cuenta borrada da error porque no son correctas.

***RESULTADO***





# ***CONCLUSIONES FINALES***

## *Grado de cumplimiento de los requisitos fijados*

He seguido un proceso constante desde el principio pues era consciente de la dedicación y tiempo que requiere debido a mi experiencia en el proyecto del Ciclo desarrollo de aplicaciones multiplataforma. Y, lo que, es más, por el hecho de hacerlo al mismo tiempo que las prácticas.

Es por eso que cada parte del proyecto la he dedicado el tiempo necesario para poder estudiarla, entenderla y por último implementarla con satisfacción.

El mayor obstáculo para el desarrollo ha sido tener que utilizar la versión gratuita de la API ya que al principio me era más que suficiente. Pero a medida que progresaba la aplicación no podía continuar pues al día solo podía hacer un total de 100 llamadas (requests) y cuando estas terminaban tenía que esperar al día siguiente, por lo que en algunos ámbitos me retrasaba.

Mi grado de satisfacción con la aplicación presentada es muy positiva porque cumple con los objetivos que me propuse al principio y por todo lo aprendido en esta etapa. Asimismo, me gustaría seguir implementando mejoras de cara al futuro porque tengo la certeza de que me va a ser útil y se adapta a mis intereses.

## *Propuestas de mejora o ampliaciones futuras*

De cara a ampliaciones futuras tengo varias propuestas como, por ejemplo, implementar un buscador que pueda dirigirse por el nombre a cualquier equipo o liga que exista, ya que ahora mismo solo tengo unos links de las más importantes.

Para esta primera etapa de la aplicación me he centrado en las ligas y sus equipos por lo que en un futuro me gustaría hacer lo mismo pero enfocado a torneos como mundiales, eurocopas, champions, copas del rey... con los cambios que eso conlleva ya que no son las mismas reglas.

También quiero centrarme en la versión de móvil ya que no es el fuerte de la aplicación y me gustaría mejorarla para que sea más atractiva y fácil de usar.

Por último, una mejora bastante importante seria crear un modo sin conexión, que se refiere a que al cargar una página se guarden los datos en la base de datos y así en vez de tener que recurrir a la API, se cargaran la próxima vez que se consulten desde la base de datos. Esto daría la opción de cuando se acaben las peticiones del plan gratis de la API por día, poder consultar sin problemas las anteriormente visitadas gracias al modo sin conexión.

# ***GUÍAS***

# ***REFERENCIAS***