



TRABALLO FIN DE GRAO
GRAO EN ENXEÑARÍA INFORMÁTICA
MENCIÓN EN ENXEÑARÍA DO SOFTWARE



Diseño y desarrollo de una plataforma de análisis de datos extraídos de cámaras tácticas en partidos de fútbol

Estudiante: Miguel Rodríguez Martínez

Dirección: Diego Seco Naveiras

Tirso Valera Rodeiro

Ignacio Lourido Fuertes

A Coruña, junio de 2024.

A mi familia y amigos.

Agradecimientos

Me gustaría agradecer a Diego, Tirso e Ignacio por ofrecerme la oportunidad de hacer este trabajo de fin de grado, por el interés que han puesto en guiarme y ayudarme durante todos estos meses. También me gustaría agradecer a mi familia y amigos por su apoyo durante esta etapa, ya que sin ellos tampoco hubiera sido posible.

Resumen

El trabajo de fin de grado consiste en desarrollar una plataforma modular y extensible para la gestión y el análisis de datos extraídos de cámaras tácticas en partidos de fútbol. Los datos que fueron utilizados para desarrollar el proyecto, archivos estructurados en diferentes formatos que contienen información detallada sobre el partido, fueron obtenidos a través del Real Club Deportivo de La Coruña [1]. Estos datos pertenecen a la empresa británica Opta [2], dedicada al análisis de datos deportivos.

El objetivo fundamental de este proyecto consiste en refactorizar, integrar y optimizar, en la medida de lo posible, dos trabajos de fin de grado del Grado en Ciencia e Ingeniería de Datos, muy distantes de ser una plataforma en sí mismos, al proporcionar únicamente análisis básicos empleando tecnologías y diseños poco escalables y mantenibles. La plataforma permite el análisis de diferentes partidos de fútbol, agrupando, a grandes rasgos, tres funcionalidades principales. Análisis posicional en un terreno de juego de dos dimensiones, extracción de clips de vídeos de diferentes tipos de jugadas a lo largo del partido y visualizar con la posibilidad de exportar un informe de estadísticas del partido.

Para el desarrollo del proyecto se procedió en primer lugar a realizar un análisis general de los requisitos funcionales y no funcionales que debería cumplir la plataforma y establecer la forma y metodología de trabajo con la que se llevaría a cabo. La metodología seguida es una metodología iterativa e incremental basada en Scrum, donde se definieron diferentes ciclos para el desarrollo de las funcionalidades (también llamados sprints). En cuanto a las tecnologías usadas diferenciamos de forma genérica Java y Spring para el backend, y TypeScript y Angular para el frontend. Además, se utilizó Postgres como Sistema Gestor de Bases de Datos, con su extensión PostGIS para datos espaciales.

Abstract

The final degree project consists of developing a modular and extensible platform for managing and analyzing data extracted from tactical cameras in football matches. The data used to develop the project, structured files in different formats containing detailed information about the match, were obtained through the Real Club Deportivo de La Coruña [1]. This data belongs to the British company Opta [2], dedicated to sports data analysis.

The main objective of this project is to refactor, integrate, and optimize, as much as possible, two final degree projects from the Degree in Data Science and Engineering, which are far from being a platform themselves, as they only provide basic analyses using technologies and designs that are not very scalable or maintainable. The platform allows for the analysis

of different football matches, broadly grouping three main functionalities: positional analysis on a two-dimensional playing field, extraction of video clips of different types of plays throughout the match, and visualization with the possibility of exporting a match statistics report.

For the development of the project, a general analysis of the functional and non-functional requirements that the platform should meet was first carried out, and the form and methodology of work to be followed were established. The methodology followed is an iterative and incremental methodology based on Scrum, where different cycles for the development of functionalities (also called sprints) were defined. Regarding the technologies used, we generically differentiate Java and Spring for the backend, and TypeScript and Angular for the frontend. Additionally, Postgres was used as the Database Management System, with its PostGIS extension for spatial data.

Palabras clave:

- Java
- Spring
- Angular
- TypeScript
- Postgres
- Posición
- Jugador
- Balón
- Fútbol
- Análisis
- Leaflet
- API

Keywords:

- Java
- Spring
- Angular
- TypeScript
- Postgres
- Position
- Player
- Ball
- Football
- Analysis
- Leaflet
- API

Índice general

1	Introducción	1
1.1	Motivación	1
1.2	Objetivos	2
2	Fundamentos tecnológicos	4
2.1	Estado del arte	4
2.2	Tecnologías utilizadas	6
3	Metodología y planificación	8
3.1	Metodología de desarrollo	8
3.1.1	Herramientas de apoyo a la metodología	9
3.2	Planificación y seguimiento	10
3.2.1	Planificación del proyecto	12
3.2.2	Seguimiento del proyecto	12
3.3	Coste económico del proyecto	16
4	Análisis	18
4.1	Requisitos	18
4.1.1	Actores	18
4.1.2	Requisitos funcionales	19
4.1.3	Requisitos no funcionales	22
4.2	Arquitectura del sistema	23
4.3	Interfaz de usuario	24
4.4	Modelo conceptual de datos	28
4.5	Principales “endpoints” de la aplicación web	34
5	Diseño	38
5.1	Arquitectura tecnológica del sistema	38

5.2	Diseño de la aplicación	39
6	Implementación y pruebas	44
6.1	Implementación	44
6.2	Pruebas	48
7	Solución desarrollada	49
7.1	Características de usuario.	49
7.2	Carga y selección de partidos.	51
7.3	Análisis con cámara táctica.	54
7.4	Análisis posicional.	55
7.5	Informe de partido.	58
8	Conclusiones y trabajo futuro	60
8.1	Conclusiones en relación a los objetivos planeados al inicio del proyecto . . .	60
8.2	Competencias adquiridas	61
8.3	Objetivos a futuro	62
A	Manual de instalación	65
A.1	Configuración general	65
A.2	Configuración de la Base de Datos	65
A.3	Ejecución del Backend	65
A.4	Ejecución del Frontend	66
B	Prototipos de la aplicación	67
Lista de acrónimos		78
Glosario		79
Bibliografía		80

Índice de figuras

2.1	Visualización 2D y 3D en Mediacoach.	5
2.2	Visualización 2D y 3D en Veo.	5
3.1	Sprint backlog de la primera iteración.	10
3.2	Tablero Scrum.	11
3.3	Planificación inicial del proyecto.	12
3.4	Seguimiento de la planificación inicial del proyecto.	13
3.5	Iconos de posición en el terreno de dos dimensiones.	15
3.6	Iconos numéricos en el terreno de dos dimensiones.	15
4.1	Arquitectura del sistema.	24
4.2	Pantalla de selección de partido.	25
4.3	Pantalla de carga de nuevos partidos.	25
4.4	Pantalla de selección de funcionalidades principales.	26
4.5	Pantalla de visualización de cámara táctica y selección de filtros.	27
4.6	Pantalla de visualización de clips de vídeo extraídos.	27
4.7	Pantalla de análisis en dos dimensiones.	28
4.8	Pantalla de estadísticas de un jugador.	29
4.9	Pantalla de visualización del informe.	29
4.10	Diagrama entidad-relación del modelo de datos.	30
5.1	Esquema de la arquitectura y tecnologías utilizadas.	40
5.2	Estructura del backend de la aplicación extraído de IntelliJ.	42
5.3	Estructura del frontend de la aplicación extraído de IntelliJ.	43
7.1	Pantalla inicial para usuarios no registrados.	50
7.2	Formulario de inicio de sesión.	50
7.3	Formulario de registro de nuevo usuario.	51

7.4	Selección de nuevo partido o navegación a añadir nuevo partido.	52
7.5	Carga de archivos de un nuevo partido.	53
7.6	Selección de partido.	53
7.7	Página inicial de la plataforma para usuarios con sesión iniciada.	54
7.8	Visualización completa de un partido.	55
7.9	Clips extraídos de un partido.	56
7.10	Visualización posicional de un partido.	57
7.11	Mapa de calor de jugador seleccionado.	57
7.12	Estadísticas de un jugador.	58
7.13	Primera parte del informe de un partido.	59
7.14	Segunda parte del informe de un partido.	59
B.1	Pantalla inicial al entrar a la plataforma.	67
B.2	Pantalla de inicio de sesión.	68
B.3	Pantalla de registro.	69
B.4	Pantalla de selección de partido.	70
B.5	Pantalla de carga de nuevos partidos.	71
B.6	Pantalla de selección de funcionalidades principales.	72
B.7	Pantalla de visualización de cámara táctica y selección de filtros.	73
B.8	Pantalla de visualización de clips de vídeo extraídos.	74
B.9	Pantalla de análisis en dos dimensiones.	75
B.10	Pantalla de estadísticas de un jugador.	76
B.11	Pantalla de visualización del informe.	77

Índice de cuadros

4.1	Endpoints de la API relacionados con los usuarios	35
4.2	Endpoints de la API relacionados con los partidos	35
4.3	Endpoints de la API relacionados con las jugadas de los partidos.	36
4.4	Endpoints de la API relacionados con las estadísticas.	36
4.5	Endpoints de la API relacionados con los datos de los jugadores.	36
4.6	Endpoints de la API relacionados con la visualización 2D.	37

Capítulo 1

Introducción

1.1 Motivación

En las últimas décadas y más concretamente en los últimos años el análisis que se realiza de los partidos de un equipo de fútbol se ha desarrollado y ha avanzado notablemente. En la actualidad la gran mayoría de clubes cuentan con equipos de analistas y tecnologías de vídeo que ayudan a extraer una cantidad innumerable de estadísticas. Por ejemplo, la liga española de fútbol profesional ya cuenta con una herramienta propia desarrollada por ellos mismos, llamada “Mediaccoach” [3], en la que se monitorizan a partir de vídeos cientos de estadísticas, ya sean individuales para un jugador en concreto, o colectivas de un equipo. El inconveniente es que aquí en el fútbol español esta aplicación solo está disponible para los clubes que pertenecen a la primera y segunda división, y tiene un altísimo coste para los equipos. A partir de aquí surge realmente la motivación y necesidad de crear una nueva plataforma de análisis que democratice el análisis táctico del fútbol.

Los clubes de divisiones más bajas no tienen acceso a la plataforma de La Liga y normalmente, tampoco disponen de los recursos económicos como para contratar a un equipo de desarrolladores y analistas que creen una plataforma de cero para el propio equipo. Por lo tanto, no disponen de ninguna plataforma de análisis que permita a sus analistas visualizar con un mayor detalle táctico y estadístico los partidos. La mayoría de clubes se limitan a la grabación de partidos y entrenamientos con sus equipos de grabación propios. Esta nueva plataforma permitiría a equipos más humildes, e incluso a equipos del fútbol base, analizar más aspectos estadísticos de sus equipos.

Para el análisis de un partido solamente es necesario cargar los datos de este mismo en la plataforma, pudiendo cargar un número de partidos sin límite, simplemente limitado por el tamaño de la base de datos. Esta carga masiva de datos se realiza a través de un archivo que contiene diferentes estadísticas de los jugadores y del balón en cada instante de un vídeo grabado a través de una cámara táctica. Una cámara táctica es una cámara que se encuentra en

uno de los laterales del campo, normalmente en un punto lo suficientemente alto y centrado como para enfocar a los 22 jugadores en un mismo plano. Permite realizar un exhaustivo análisis posicional de los jugadores y del balón a lo largo de todo el partido. A partir de dicho video-análisis es cuando se crearía el archivo de datos, que muchas veces las ligas ya ofrecen a sus equipos. En cualquier otro caso podría realizarse por los analistas e ingenieros de datos que cada equipo pudiera tener.

1.2 Objetivos

El objetivo de este proyecto es la creación de una plataforma modular, extensible y mantenible en el tiempo para el análisis de partidos de fútbol. La plataforma permite a los analistas de los equipos cargar los datos y vídeos de sus partidos, para posteriormente visualizar el posicionamiento de sus jugadores propios e incluso jugadores de otros equipos (posibles fichajes), extraer las jugadas destacadas e incluso visionar un informe con estadísticas y gráficos acerca del partido.

Se puede descomponer este objetivo general de la plataforma en objetivos más pequeños y concretos que fueron necesarios alcanzar uno a uno para llegar al resultado final:

- Gestión de usuarios y diferentes roles de usuario. Se diferencian dos roles, administrador y analista. El usuario administrador es el actor más importante dentro de la aplicación y es el que puede acceder y manipular los datos de todos los partidos. En el caso del resto de usuarios solamente podrán acceder a datos propios, es decir, a datos que hayan sido cargados por ese mismo usuario.
- Carga, almacenamiento, gestión y análisis de una gran cantidad de datos de partidos junto también a la carga y almacenamiento de sus respectivos vídeos, que en cualquier caso no suelen bajar de una hora y media de duración.
- Los vídeos cargados, comentados en el anterior punto, deben poder visualizarse en cualquier momento por el usuario.
- Permitir la visualización de una serie de clips de vídeos correspondientes a jugadas concretas durante el partido. Los clips que se muestran al usuario siguen unos filtros determinados según el tipo de jugadas que se quieran visionar (contraataque, ataque por banda, desmarque en ruptura, etc.), equipo y jugador.
- Visualizar en un terreno de dos dimensiones la posición de los jugadores y del balón a lo largo de todo el partido, pudiendo seleccionar concretamente momentos decisivos como pueden ser goles o saques de esquina.

- Generación de diferentes gráficos y mapas de calor de cada jugador y equipos para una mayor facilidad en el análisis y comparación de las estadísticas de un partido.
- Generación de un informe con las estadísticas y gráficos más destacados de un partido como pueden ser los jugadores más rápidos, los jugadores que más distancia han recorrido, posesión de cada equipo, etc.
- Gestión de los cambios en las plantillas de los equipos, ya sea durante una misma temporada o en temporadas diferentes.

Capítulo 2

Fundamentos tecnológicos

2.1 Estado del arte

En el mercado actual se han encontrado una serie de herramientas similares a esta plataforma. A continuación se van a describir algunas de estas mismas:

- **Mediaccoach** [3]: es una herramienta de vídeo análisis desarrollada por Mediapro y que se encuentra en colaboración con La Liga Española de Fútbol Profesional. Por esta razón, solamente los 42 equipos pertenecientes al fútbol profesional español tienen acceso a esta aplicación. La plataforma, al estar desarrollada por una empresa de ese calibre y llevar ya más de una década en desarrollo continuo, dispone de cientos de funcionalidades e incluso tiene paneles de customización a disposición del gusto y las necesidades de cada uno de los usuarios. Los usuarios pueden hacer un análisis de los partidos en tiempo real, vídeo análisis con múltiples ángulos de visión, segmentación de jugadas, obtener informes detallados, comparativas, acceso a datos históricos, análisis de rendimiento físico o visualización táctica en 2D y en 3D como se ve en la Figura 2.1, etc. Obviamente el número de posibilidades es altísimo, pero el coste de la plataforma es a su vez extremadamente alto.
- **Veo**: es el nombre de una empresa y a su vez de la herramienta desarrollada por esa empresa. Está especializada en la grabación y análisis de partidos mediante inteligencia artificial. Utilizan algoritmos que les permiten seguir el balón y las acciones del juego eliminando la necesidad de un operador de cámara. Además de eso, han desarrollado una plataforma de análisis que permite subir automáticamente el partido grabado. Posteriormente la plataforma pone a disposición del usuario una serie de funcionalidades que permiten visualizar el vídeo completo de los partidos y crear clips de vídeo de momentos destacados, por ejemplo cuando ocurren eventos como goles, faltas, saques de puerta, saques de esquina, etc. También dispone de generación de mapas de calor, in-

formes de estadísticas o visualización en 2D y 3D como se puede ver en la Figura 2.2. El coste de la cámara es de 999 euros y el coste de la suscripción a la plataforma va desde 39 a 165 euros mensuales dependiendo de las necesidades propias del cliente.



Figura 2.1: Visualización 2D y 3D en Mediacoach.



Figura 2.2: Visualización 2D y 3D en Veo.

Las dos plataformas mencionadas anteriormente disponen de aplicaciones muy desarrolladas y muy profesionalizadas. El principal inconveniente para los equipos es el alto coste que tienen y la necesidad que les surgiría al necesitar un gran equipo de analistas, ya que si no sacarle partido a estas plataformas tan grandes es muy complicado, sobre todo porque la curva de aprendizaje al final no es tan reducida. Este proyecto al final está centrado y desarrollado para equipos más humildes, muy accesible económicamente y con una curva de aprendizaje

que es muy reducida. Se enfoca en los aspectos fundamentales del análisis, que son el análisis 2D, la extracción de una serie de clips de jugadas concretas y determinantes en el transcurso de un partido, y por último la extracción de una serie de datos físicos de los jugadores. Podría ser utilizada y sacarle el máximo provecho simplemente con un solo analista, entrenador o mismo por los propios jugadores. Además, no requiere de apenas configuración ni mantenimiento. Obviamente todo esto también tiene sus puntos negativos. Las funcionalidades al ser solamente las esenciales, son a su vez muy limitadas. Además, al necesitar disponer del archivo de carga de datos es un rasgo de dependencia con respecto a otra entidad o a otro software. Veo y Mediacoach son dos empresas que esto ya lo traen incorporado, la primera de ellas porque trae un servicio de grabación y vídeo análisis automatizado, y la segunda porque la plataforma ya la recibes con todos los datos cargados y no necesitas realizar nada extra.

2.2 Tecnologías utilizadas

A continuación, se detallan algunas de las tecnologías utilizadas para el desarrollo del proyecto:

- **PostgreSQL** [4]. Sistema de gestión de bases de datos relacional de código abierto.
- **PostGIS** [5]. Extensión de PostgreSQL de código abierto facilitando el almacenamiento, manipulación y análisis de datos espaciales complejos.
- **Leaflet** [6]. Librería de JavaScript de código abierto para crear mapas dinámicos.
- **TypeScript** [7]. Lenguaje de programación tipado que es un superconjunto de JavaScript.
- **NPM** [8]. Gestor de paquetes y dependencias de JavaScript.
- **Angular** [9]. Framework utilizado para crear aplicaciones web de una sola página (SPA).
- **Angular Material** [10]. Biblioteca de componentes de Angular.
- **CryptoJS** [11]. Biblioteca que proporciona operaciones criptográficas para JavaScript.
- **ChartJS** [12]. Biblioteca que facilita la creación de gráficos en JavaScript.
- **Rxjs** [13]. Biblioteca de JavaScript que implementa programación reactiva a eventos.
- **Jspdf** [14]. Biblioteca de JavaScript que facilita la creación de documentos en formato PDF.

- **JUnit** [15]. Marco de trabajo que facilita la creación y ejecución de pruebas unitarias.
- **Mockito** [16]. Biblioteca de Java que facilita la creación de objetos que simulan el comportamiento de dependencias.
- **Spring** [17]. Framework de desarrollo de aplicaciones Java.
- **Maven** [18]. Herramienta que facilita el empaquetado y la gestión de dependencias.
- **Gson** [19]. Biblioteca de Java que facilita la conversión de objetos Java a JSON y viceversa.
- **OpenCSV** [20] . Biblioteca de Java que facilita la lectura y escritura de archivos de extensión CSV.
- **Lombok** [21]. Biblioteca de Java que permite reducir la cantidad de código agregando anotaciones.

Capítulo 3

Metodología y planificación

3.1 Metodología de desarrollo

El proyecto fue desarrollado siguiendo una metodología iterativa incremental [22]. Esta metodología sigue un enfoque de desarrollo que combina elementos del desarrollo secuencial y el desarrollo iterativo.

En este tipo de metodologías el proyecto se planifica en varios bloques. Se divide el trabajo a realizar en bloques diferenciados temporalmente en el que se suele repetir un modelo de trabajo. Estos bloques temporales son también llamados iteraciones. Cada iteración tiene una duración de entre dos y cuatro semanas. En cada una de ellas, el cliente alcanza a ver un resultado completo del proyecto. Tal y como indica el propio nombre, tras cada iteración se incrementa el número de funcionalidades que la plataforma soporta. De este modo el proyecto es un producto usable en el menor tiempo posible y el cliente puede empezar a trabajar con él lo antes posible. También es por esto que se establece un orden de prioridades. Las tareas y funcionalidades de mayor prioridad serán siempre desarrolladas antes que las tareas de menor prioridad.

Una de las mayores ventajas de esta metología es que permite una retroalimentación muy temprana y continuada en el tiempo por parte del cliente. Esto permite reducir el riesgo de cometer costosos fallos al final del proyecto y una gran flexibilidad en el desarrollo. No hace falta tener una rígida especificación de requisitos, sino que no hay ningún problema en que los requisitos varíen de iteración en iteración según vayamos recibiendo también la retroalimentación del cliente. Otra ventaja importante, ya comentada anteriormente, es la entrega continua de valor al cliente, generando un producto utilizable desde etapas tempranas. Todas estas ventajas llevan a su vez una serie de puntos débiles. La gestión del proyecto requiere de una comunicación y coordinación constante entre equipos y cliente. Además, por los cambios constantes, es muy difícil realizar una estimación precisa del alcance total del proyecto en recursos y tiempo. Por último, un problema muy común en los equipos de trabajo es la

sobrecarga de trabajo debido a la necesidad de entregar un producto final en cada iteración.

Para el caso concreto del desarrollo de este proyecto la metodología fue además basada en Scrum [23]. Al solo contar con una persona en el equipo de desarrollo no puede considerarse Scrum como tal, ya que sería necesario una reunión diaria con el equipo, que simplemente por no haber un equipo de varias personas, no se llevaron a cabo. Scrum es una metodología ágil utilizada especialmente en el desarrollo de software. Al ser una metodología iterativa incremental el trabajo se divide en iteraciones llamadas “sprints” que duran como ya se comentó anteriormente de dos a cuatro semanas. En cada “sprint” el equipo trabaja en un conjunto concreto de funcionalidades que tienen mayor prioridad. Tras cada iteración se presenta un producto final. Para llevar a cabo esta metodología se debe establecer varias técnicas y prácticas. Se debe definir un “Product Backlog” que podrá ser modificado entre iteraciones. Esto es básicamente una lista de historias de usuario a realizar en el que cada una de ellas tiene establecida una prioridad. A su vez en cada iteración se debe definir un “Sprint Backlog” como el de la Figura 3.1, que se trata de un subconjunto del backlog que se va a realizar en esa iteración en concreto. Un desarrollador cuando acaba una tarea debe escoger la siguiente a realizar entre las tareas de mayor prioridad. En el modelo original de Scrum se realizan varias reuniones por cada iteración, que en este caso fueron superpuestas en una reunión de mayor duración en la que se realizaban varias tareas a la vez. El modelo de Scrum define tres tipos de reuniones, además de las reuniones diarias que como ya fue comentado no se llevaron a cabo. Se define una reunión de planificación al inicio del “sprint” para definir lo que se realizará en esa iteración, una reunión de revisión para revisar lo que se ha realizado y por último una reunión para reflexionar sobre esa iteración e intentar mejorar procesos. En este caso, podemos considerar que estas tres reuniones, por cuestiones de tiempo, se han realizado en una misma reunión de mayor duración al final de cada iteración, que a su vez coincide con el inicio de la siguiente.

3.1.1 Herramientas de apoyo a la metodología

Para la gestión del desarrollo se han utilizado una serie de herramientas de gran ayuda o fundamentales:

- **GanttProject** [24]. Se trata de un software de gestión de proyectos a partir de diagramas de Gantt. Un diagrama de Gantt es una herramienta para programar y supervisar tareas a lo largo del tiempo. Se representan las tareas en un eje vertical junto a un eje temporal de forma horizontal para una mayor facilidad en la visualización y gestión del transcurso de las tareas realizadas y por realizar a lo largo del tiempo.
- **Jira** [25]. Herramienta de gestión de proyectos en la que principalmente se hizo la gestión del backlog, el backlog de cada sprint y se hizo uso también de tableros Scrum como

	SCRUM-23 Crear entidad home_away_players	TO DO	✗
SCRUM-24	Crear entidad player_metadata	TO DO	✗
SCRUM-25	Migrar servicio de desmarque en apoyo	TO DO	✗
SCRUM-26	Migrar servicio de desmarque en ruptura	TO DO	✗
SCRUM-27	Migrar servicio de contraataque	TO DO	✗
SCRUM-28	Migrar servicio de ataque por banda	TO DO	✗
SCRUM-31	Migrar BD	TO DO	✗
SCRUM-32	Migrar datos BD a nueva BD llamada football_data	TO DO	✗
SCRUM-29	Migrar servicio de guardar de la caché	TO DO	✗
SCRUM-30	Migrar servicio de obtener de la caché	TO DO	✗

+ Create issue

Figura 3.1: Sprint backlog de la primera iteración.

el de la Figura 3.2.

- **pgAdmin** [26]. Entorno gráfico para la gestión de bases de datos PostgreSQL.
- **IntelliJ** [27]. Entorno completo de programación y desarrollo especialmente dedicado a lenguajes como Kotlin o Java.
- **Overleaf** [28]. Herramienta que permite a los usuarios crear y gestionar documentos LaTeX.
- **GitLab** [29]. Herramienta para el control de versiones y gestión de código fuente de los proyectos.
- **Balsamiq** [30]. Herramienta de diseño de interfaces de usuario.
- **Postman** [31]. Postman es una plataforma para el diseño, desarrollo, prueba y documentación de APIs.
- **Draw.io** [32]. Herramienta para el diseño y modelado de un diagrama entidad relación tras el que posteriormente se desarrollaría la base de datos que da soporte al proyecto.

3.2 Planificación y seguimiento

En este apartado se explica la planificación inicial del proyecto junto con el seguimiento realizado al mismo. Por último, se hace un pequeño análisis de las desviaciones en tiempo y coste que este ha tenido.

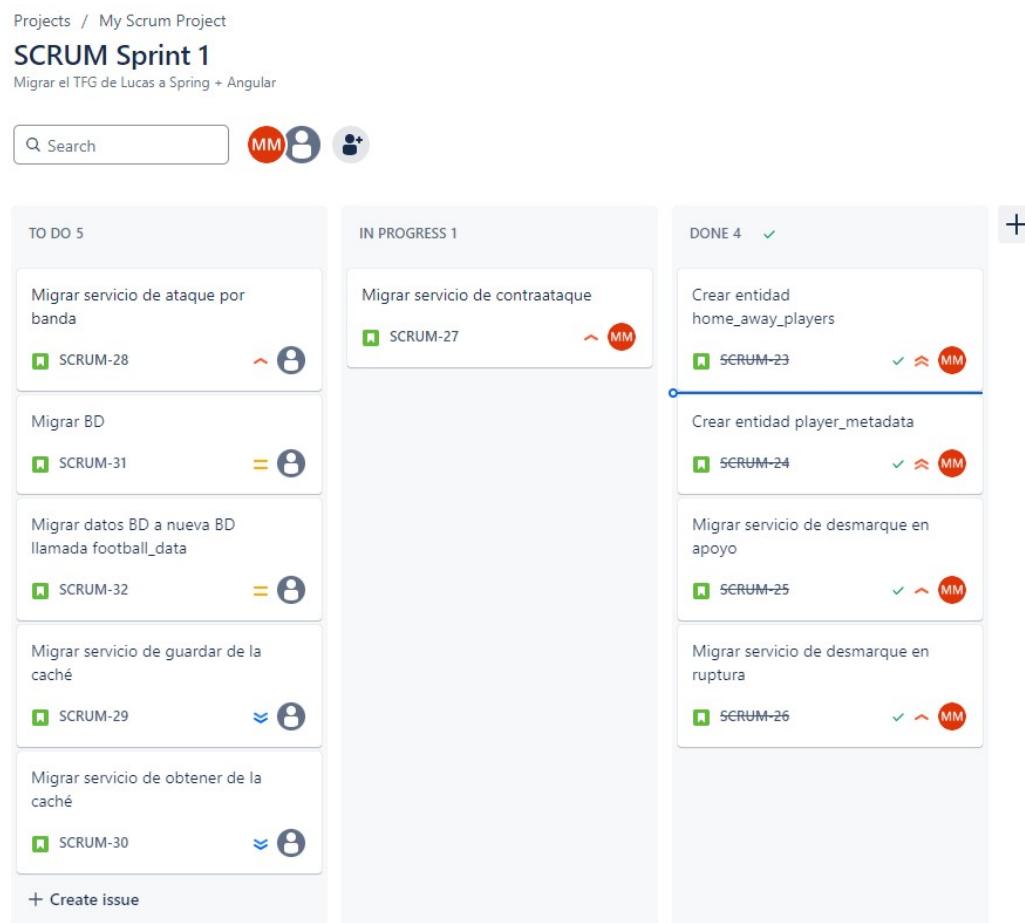


Figura 3.2: Tablero Scrum.

3.2.1 Planificación del proyecto

A grandes rasgos, podemos dividir el proyecto en tres etapas. Una primera etapa de planificación y análisis, una segunda etapa de desarrollo y para acabar la elaboración de la memoria.

Para el comienzo del trabajo se realizó una primera reunión inicial entre directores del proyecto y alumno en la que se definieron las bases y las directrices de lo que iba a ser el diseño y desarrollo de la plataforma y del trabajo. En esa reunión inicial se pusieron en conjunto las bases del proyecto, es decir, los dos trabajos de fin de grado de ingeniería de datos que ya fueron comentados con anterioridad en otros apartados, y se propusieron una serie de ideas generales a partir de las cuales habría que definir objetivos más concretos. Además de eso, se definió que la plataforma tendría tres grandes funcionalidades. Establecimos la metodología de trabajo y la duración de cada iteración, que sería de dos semanas, con un total de 9 sprints.

El primer sprint sería el correspondiente a la primera etapa de planificación y análisis, tras la comentada reunión inicial.

Tras esto, vendrían una serie de seis sprints, en los que se dividió el trabajo por funcionalidades. Cada dos sprints debería estar plenamente desarrollada cada una de las tres grandes funcionalidades que se diferencian en la plataforma.

Por último, quedarían dos sprints que fueron dedicados a la elaboración de la memoria y ajustes en las funcionalidades abordadas en las iteraciones anteriores.

Entre la finalización y el comienzo de cada sprint, se definió que se realizarían reuniones entre directores y alumno en el que se podrían ajustar o definir nuevos requisitos además de valorar el trabajo realizado hasta cada momento.

3.2.2 Seguimiento del proyecto

En este apartado se describen las tareas realizadas en cada sprint y las posibles desviaciones que hubo. En primer lugar, y para tener una mejor visualización de las tareas realizadas y de los comentarios que se harán posteriormente en este apartado, recomendaría ver con detalle la [Figura 3.3](#) y la [Figura 3.4](#) correspondientes a dos diagramas de Gantt de la planificación inicial y seguimiento del proyecto.

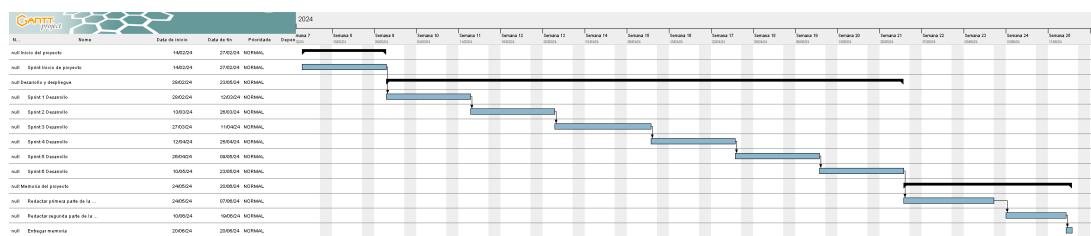


Figura 3.3: Planificación inicial del proyecto.

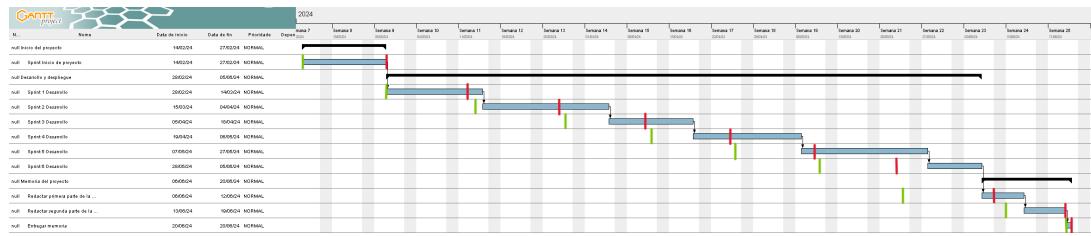


Figura 3.4: Seguimiento de la planificación inicial del proyecto.

En primer lugar, se realiza una descripción con mayor detalle de las tareas realizadas en cada sprint, partiendo desde el segundo sprint, el primero correspondiente a la etapa de desarrollo.

- **Segunda iteración.** En esta segunda iteración, y la primera de la etapa de desarrollo del proyecto, se comenzó por uno de los trabajos de fin de grado de ingeniería de datos dado al alumno. Para ello en primer lugar, se desplegó en el equipo propio del alumno ese proyecto. En este caso, consistía en un proyecto de una única gran funcionalidad que consistía en la visualización y extracción de clips a partir de una serie de filtros de un partido de fútbol. El título original de este mismo es “Sistema de detección de patrones en partidos de fútbol a través de un visor web”. El proyecto estaba realizado en Python y JavaScript. La primera de las tareas, tras haberlo lanzado ya en el equipo local, trataba de migrar y refactorizar el backend de esa plataforma a Java utilizando el framework Spring.
- **Tercera iteración.** La tercera iteración fue realmente una continuación de la segunda. Una vez ya estaba el backend terminado se comenzó a migrar y refactorizar el frontend, que en este caso estaba desarrollado en JavaScript, [HTML](#) y [CSS](#) muy simple. El aspecto de la interfaz web tampoco era muy vistoso de cara al usuario final. El frontend debía ser migrado por completo a una tecnología moderna y perdurable en el tiempo, por ello Angular y TypeScript fue una elección que creímos muy aceptable en el cumplimiento de estas condiciones. La realidad es que del frontend poco se pudo mantener y el desarrollo fue casi de cero.
- **Cuarta iteración.** Una vez migrado y refactorizado el primer proyecto se comenzó con el segundo. Aquí los pasos a seguir fueron muy similares. En este caso se trataba de otro proyecto que realmente también constaba de otra gran funcionalidad. En este caso el título del proyecto era “Aplicación web orientada al análisis de partidos de fútbol”. El proyecto se trataba de un análisis de un partido de fútbol completo a partir de un terreno de juego en dos dimensiones en donde se posicionaba a los jugadores y el balón en cada frame a partir de un archivo de datos. Estaba desarrollado igualmente en

Python y JavaScript. Esta iteración constó igualmente que para el anterior proyecto en la migración y refactorización del backend de la aplicación web a una tecnología actual y perdurable en el tiempo que igual que para el proyecto anterior se decidió que fuera Java y Spring, ya que uno de los objetivos de este proyecto es realizar una plataforma conjunta utilizando tecnologías comunes a ambas.

- **Quinta iteración.** Una vez desarrollado el backend de la aplicación, procedemos de nuevo a migrar y refactorizar el frontend. De nuevo, como para el otro proyecto, la interfaz de usuario era realmente muy pobre y fue desarrollada en JavaScript, HTML y CSS, provocando un código totalmente desestructurado y muy poco reutilizable y manejable. La reutilización de código en esta iteración fue prácticamente nula y de nuevo se realizó todo de cero, partiendo de ninguna base. Por lo ya comentado anteriormente, Angular y TypeScript fueron las tecnologías escogidas.
- **Sexta iteración.** Esta iteración según la planificación inicial correspondería al comienzo de una nueva funcionalidad. La realidad es que debido a consecutivos pequeños retrasos en las iteraciones y, sobre todo, la aparición de un problema de sincronización en la visualización de jugadores y balón en el terreno de juego en dos dimensiones esta iteración fue principalmente de resolución de problemas. Ya fue en estas dos semanas cuando se finalizó la migración y refactorización completa del segundo de los trabajos y se hizo un pequeño inicio de la última funcionalidad que corresponde a la creación y posible exportación de un informe.
- **Séptima iteración.** Esta iteración corresponde a la última en la etapa de desarrollo. Se definieron e implementaron ya de forma definitiva los requisitos de esta última funcionalidad de visualización de informes dentro de la plataforma y se finalizó ya casi de manera definitiva toda la plataforma completa.
- **Octava iteración.** Esta es la primera iteración de las dos que forman la fase de elaboración de la memoria. Entre otras cosas se completaron los primeros apartados de la memoria del trabajo, pero también surgió la posibilidad de realizar una reunión con el director del departamento de tecnología analítica y deportiva y también director de este trabajo de fin de grado, Ignacio Lourido Fuertes, en la que se realizó una explicación y demo del funcionamiento de la plataforma. Esta reunión se correspondería con la realización de pruebas de aceptación por parte del cliente para garantizar que el producto cumple con los requisitos y expectativas que el cliente pide en él. El cliente solicitó una serie de cambios, de no gran importancia, pero que podrían ser interesantes. Todos ellos no pudieron llevarse a cabos por motivos de plazos de entrega, pero sí uno de los que más interesantes le parecían. Se trató de un cambio de visualización en el terreno de

juego de dos dimensiones en que los iconos de posición pasaron de ser simples marcadore de posición, como se ve en la Figura 3.5, a fichas numéricas en las que se puede identifcar el dorsal y, por lo tanto, de qué jugador se trata cada icono como se puede ver en la Figura 3.6.



Figura 3.5: Iconos de posición en el terreno de dos dimensiones.



Figura 3.6: Iconos numéricicos en el terreno de dos dimensiones.

Por lo tanto, esta iteración a pesar de contemplarse en la planificación como de elabora-

ción de la memoria hubo que adaptarla al surgimiento de nuevos cambios y requisitos en la plataforma y tuvo parte también que constó de desarrollo de código fuente.

- **Novena iteración.** Ya en esta última iteración fue cuando se finalizó la elaboración de la memoria del proyecto y se realizó una revisión completa de este, finalizando con la entrega del mismo.

Tal y como ya se pudo ver en la [Figura 3.3](#) y la [Figura 3.4](#) la planificación inicial del proyecto se vio modificada desde el comienzo del proyecto. En la [Figura 3.4](#) para cada iteración se muestran dos líneas verticales verde y roja respectivamente. La línea verde se corresponde al día de inicio esperado de esa iteración y la roja al día de fin esperado. Esto se hizo para visualizar gráficamente con gran facilidad las desviaciones en tiempo que ha tenido el proyecto y cada iteración individualmente.

Desde la primera iteración de la etapa de desarrollo se produjeron retrasos que se fueron acumulando continuamente hasta la última iteración. A pesar de esto se pudo llegar en tiempo y forma al objetivo deseado, no sin acumular varias horas extra de trabajo durante las últimas semanas.

3.3 Coste económico del proyecto

En este apartado se realiza un análisis de los costes totales del proyecto y de las desviaciones que ha tenido con respecto a la estimación inicial.

En primer lugar, nos encontramos con las herramientas de software y recursos físicos que el alumno ha tenido que utilizar. Para el desarrollo del proyecto el alumno no ha utilizado ningún software de pago, más allá de los que le otorga la universidad, que no han sido por lo tanto un coste real. A pesar de esto, contaremos como coste una licencia de IntelliJ Ultimate durante un año que otorga la universidad al alumno, con un coste de aproximadamente 599 euros al año. En cuanto a recursos físicos, solamente ha sido utilizado el ordenador portátil del alumno que en su día costó aproximadamente 900 euros. A pesar de que a día de hoy el precio seguramente sea menor se tomará ese precio como coste del producto.

Por otro lado tenemos el coste de los recursos humanos. Se realizaron un total de nueve reuniones con dos de los directores de una duración media de una hora y otra con los tres directores de una hora. Además, dos de los directores han realizado un trabajo correspondiente a cinco horas cada uno para la revisión del proyecto. Por lo que obtenemos un total de 14 horas para dos de los directores pertenecientes a la UDC y una hora para el director perteneciente al Real Club Deportivo de La Coruña. Por lo tanto obtenemos que los dos directores pertenecientes a la UDC, con salarios aproximados de 31.000 euros al año o 14,90 euros la hora, han tenido un coste total de 208,6 euros cada uno o 417,2 euros en total. Para el tercer de los

directores del proyecto no se han podido obtener datos por lo que se ha establecido un salario de 25 euros la hora. Por lo que el total es también de 25 euros. Por último, también deberíamos sumar el coste del alumno. El alumno ha trabajado un total de 320 horas, correspondientes a las 19 semanas de trabajo correspondientes al proyecto, dando una media aproximada de casi 17 horas de trabajo a la semana. A esto habría que sumarle las 10 horas de reuniones en las que estuvo el alumno, dando un total de 330 horas. El salario de un desarrollador junior en A Coruña, según el portal de análisis Glassdoor [33], es de 22.540 euros, lo que resulta en un total de 10,84 euros la hora. Haciendo una multiplicación simple obtenemos un coste total del trabajo del alumno de 3.577,2 euros.

Tras la suma de todo obtenemos un coste total de 5.518,4 euros. Se ha producido una desviación económica, pero por suerte ha sido relativamente pequeña. La planificación inicial establecía una media de 15 horas de trabajo para el estudiante a la semana. Esto indica que se han realizado 38 horas extra, con un coste de 411,92 euros. Por otro lado se realizó una reunión extra a las planificadas, la correspondiente a las pruebas de aceptación. En ella formaron parte los 3 directores y tuvo una duración de una hora. Esto hace una suma de 65,64 euros extra, una hora perteneciente a cada director y una al alumno. Esto da un sobrecoste total de 477,56 euros sobre la planificación inicial del proyecto.

Capítulo 4

Análisis

Esta aplicación web se trata de una plataforma de análisis de partidos de fútbol a través de análisis de vídeo y análisis posicional en dos dimensiones. A continuación se van a describir los actores, es decir quién usa el sistema, y los requisitos funcionales y no funcionales que tiene la plataforma.

4.1 Requisitos

En este apartado se describen los actores, requisitos funcionales y requisitos no funcionales del sistema. Además, también se detallan aspectos sobre la arquitectura, la interfaz gráfica y el modelo de datos.

4.1.1 Actores

La plataforma está dirigida para equipos de fútbol. Dentro de ella se diferencian dos tipos de usuarios.

- **Usuario registrado.** Todo usuario de la plataforma debe estar registrado. Este es el tipo de usuario por defecto de la plataforma y al que tendrán acceso la gran mayoría de usuarios de la misma. Solo puede acceder a datos propios.
- **Administrador.** Un administrador es a su vez un usuario registrado, pero con un mayor número de permisos. Tiene acceso a todos los datos del sistema, tanto los propios como a los de los demás usuarios registrados. En un futuro, una de las próximas funcionalidades que serían desarrolladas sería un panel de administrador que permitiera también al administrador manipular los datos de los demás usuarios, ya sea añadir, modificar o eliminar usuarios y/o datos de los partidos guardados por cada usuario.

4.1.2 Requisitos funcionales

A continuación, se van a listar y a describir brevemente los requisitos funcionales del sistema, representados como historias de usuario.

- **HU-1.** Como usuario no registrado, quiero registrar un nuevo usuario. El usuario introduce nombre, apellidos, nombre de usuario, correo, contraseña y confirmación de contraseña para crear un nuevo usuario con el que posteriormente se podrá autenticar y autorizar en la plataforma. En caso de introducir algún dato incorrecto la plataforma muestra un mensaje de error.
- **HU-2.** Como usuario registrado y/o administrador, quiero iniciar sesión. El usuario introduce nombre de usuario y contraseña, y se autentica en la plataforma. En caso de introducir credenciales no válidas el sistema muestra un mensaje de error. En caso de ser satisfactorio el proceso el usuario habrá iniciado sesión con las credenciales introducidas.
- **HU-3.** Como usuario registrado y/o administrador, quiero cerrar sesión. El usuario puede cerrar sesión en cualquier momento y en cualquier página de la plataforma simplemente haciendo clic en su nombre de usuario y pulsando el botón de cerrar sesión.
- **HU-4.** Como administrador, quiero seleccionar un partido de entre todos los guardados por todos los usuarios. Un administrador dispone de un apartado de selección en donde puede escoger para analizar un partido de entre todos los que están subidos y guardados en la plataforma.
- **HU-5.** Como usuario registrado, quiero seleccionar un partido de entre todos los guardados por mi propio usuario. Un usuario dispone de un apartado de selección en donde puede escoger para analizar un partido de entre todos los que han sido subidos por ese propio usuario.
- **HU-6.** Como usuario registrado y/o administrador, quiero subir un nuevo archivo de metadatos de un partido. Un usuario de la plataforma puede subir un archivo de metadatos acerca de un partido para cargar los metadatos relacionados con un partido en la base de datos y guardarlos para futuros análisis del partido.
- **HU-7.** Como usuario registrado y/o administrador, quiero subir un nuevo archivo de datos de un partido. Un usuario de la plataforma puede subir un archivo de datos acerca de un partido para cargar los datos de cada frame relacionados con un partido en la base de datos y guardarlos para futuros análisis del partido.

- **HU-8.** Como usuario registrado y/o administrador, quiero subir un nuevo archivo de eventos registrados de un partido. Un usuario de la plataforma puede subir un archivo de registro de eventos acerca de un partido para cargar los eventos relacionados con un partido en la base de datos y guardarlos para futuros análisis del partido.
- **HU-9.** Como usuario registrado y/o administrador, quiero subir un nuevo archivo de estadísticas físicas de un partido. Un usuario de la plataforma puede subir un archivo de estadísticas físicas acerca de un partido para cargar las estadísticas físicas relacionados con un partido en la base de datos y guardarlas para futuros análisis del partido.
- **HU-10.** Como usuario registrado y/o administrador, quiero subir un nuevo archivo de vídeo mp4 de un partido. Un usuario de la plataforma puede subir un archivo en formato mp4 de un partido para cargar el archivo de vídeo relacionados con un partido en la pataforma y guardararlo para futuros análisis del partido.
- **HU-11.** Como usuario registrado y/o administrador, quiero visualizar el vídeo completo de un partido. Un usuario puede visualizar el archivo de vídeo relacionado con un partido y tener control sobre él. Puede parar el vídeo, iniciar el vídeo, seleccionar un minuto, subir el volumen, bajar el volumen, poner en pantalla completa, etc.
- **HU-12.** Como usuario registrado y/o administrador, quiero aplicar filtros para extraer clips de vídeo de un partido. Un usuario puede aplicar filtros para extraer clips de vídeo de un archivo de vídeo de un partido completo. Un usuario puede filtrar por contraataques, desmarques en apoyo, desmarques en ruptura y ataque por banda.
- **HU-13.** Como usuario registrado y/o administrador, quiero filtrar los clips de vídeo por jugador. Un usuario puede a su vez también filtrar los clips de vídeo extraídos del archivo de vídeo completo por jugador. Cada vídeo está identificado a través de un jugador principal en la jugada y los vídeos pueden ser filtrados por el nombre de esos jugadores.
- **HU-14.** Como usuario registrado y/o administrador, quiero visualizar un terreno de juego en dos dimensiones. Un usuario dispone de un panel en el que se muestra un terreno de juego de fútbol en dos dimensiones en el que posteriormente se hará la representación visual de jugadores y balón durante cada instante del partido.
- **HU-15.** Como usuario registrado y/o administrador, quiero visualizar la posición inicial de los jugadores y el balón en un partido. Un usuario podrá clicar un botón y los iconos de los 22 jugadores y del balón se mostrarán en el instante inicial del partido, cada uno en su posición correspondiente.

- **HU-16.** Como usuario registrado y/o administrador, quiero visualizar un periodo del partido entre dos minutos concretos. Un usuario puede seleccionar en dos elementos numéricos de entrada dos minutos. Un minuto será el instante inicial de inicio del periodo y otro será el momento de fin de la visualización. El minuto de fin siempre debe ser mayor que el minuto de inicio, si no se muestra un mensaje de error y no comienza la visualización.
- **HU-17.** Como usuario registrado y/o administrador, quiero parar la visualización del partido en el terreno de dos dimensiones si el tiempo se encuentra en movimiento. Un usuario tiene disponible un botón de pausa que para la visualización del partido si se encuentra activa en ese momento.
- **HU-18.** Como usuario registrado y/o administrador, quiero poder reiniciar la visualización del partido en el terreno de dos dimensiones y empezar por donde ha sido pausada anteriormente. Un usuario dispone de un botón de reinicio en el que la visualización en el terreno de dos dimensiones comenzará de nuevo en el último instante en el que fue parada.
- **HU-19.** Como usuario registrado y/o administrador, quiero eliminar todos los iconos del campo. Un usuario dispone de un botón de eliminación de iconos. En cualquier momento en el que el terreno de dos dimensiones disponga de iconos desplegados sobre él, el usuario puede clicar este botón de limpiado para dejar el terreno sin iconos tal y como comenzó en un inicio.
- **HU-20.** Como usuario registrado y/o administrador, quiero visualizar un mapa de calor de cada jugador. El usuario dispone de dos paneles de selección en el que se muestran los jugadores del equipo local y del equipo visitante respectivamente. Una vez seleccionado uno de estos jugadores se mostrará en el terreno de dos dimensiones, una vez limpiados todos los iconos, un mapa de calor de ese jugador a partir de su posición a lo largo del partido.
- **HU-21.** Como usuario registrado y/o administrador, quiero visualizar un panel de estadísticas de cada jugador. El usuario dispone de dos paneles de selección en el que se muestran los jugadores del equipo local y del equipo visitante respectivamente. Una vez seleccionado uno de estos jugadores se mostrará un pequeño panel de información acerca de ese jugador junto con un botón de acceso a más información. Una vez clicado ese jugador se mostrará el resto de información y estadísticas de ese jugador.
- **HU-22.** Como usuario registrado y/o administrador, quiero seleccionar y visualizar un gol concreto de un partido. Un usuario dispone de un panel de selección en el que se

muestran los minutos de los goles del partido preseleccionado. Una vez seleccione uno de esos minutos comenzará la visualización en el campo de dos dimensiones de un periodo que corresponde desde 10 segundos antes del instante del gol a 10 segundos después del instante del gol.

- **HU-23.** Como usuario registrado y/o administrador, quiero seleccionar y visualizar un saque de esquina concreto de un partido. Un usuario dispone de un panel de selección en el que se muestran los minutos de los saques de esquina del partido preseleccionado. Una vez seleccione uno de esos minutos comenzará la visualización en el campo de dos dimensiones de un periodo que corresponde desde 10 segundos antes del instante del gol a 10 segundos después del instante del gol.
- **HU-24.** Como usuario registrado y/o administrador, quiero ver un informe completo a modo de resumen de un partido y de las estadísticas del mismo. En la pantalla de inicio en el que el usuario puede acceder a las distintas funcionalidades que ofrece la plataforma, el usuario puede acceder a un informe a modo de resumen de datos y estadísticas principales del partido. Algunos de los datos mostrados son los equipos que se enfrentan, la fecha, la hora, los jugadores, la posesión, jugadores más rápidos, jugadores que más distancia han recorrido, etc.
- **HU-25.** Como usuario registrado y/o administrador, quiero exportar un informe completo a modo de resumen de un partido y las estadísticas del mismo a diferentes formatos. El informe comentado anteriormente debe tener un botón que lo permita a exportar a diferentes formatos (pdf, png, jpeg) a elección del usuario y se descargaran en su equipo local con un nombre descriptivo que lo relacione con el partido seleccionado.

4.1.3 Requisitos no funcionales

Los requisitos no funcionales de la aplicación, además de asegurar que esta misma cumpla con los requisitos funcionales, aseguran que el sistema sea eficiente, accesible, fácil de usar, fácil de mantener y que esté siempre disponible para los usuarios. A continuación se describen más detalladamente cada uno de los requisitos no funcionales del sistema.

- **Interfaz gráfica.** La interfaz debe ser todo lo simple que sea posible acorde a las funcionalidades requeridas y que por lo tanto la curva de aprendizaje sea lo más rápida posible.
- **Velocidad de respuesta.** La respuesta a las peticiones debe ser lo más veloz posible, ajustándose cada petición a la carga de trabajo que tenga.

- **Modularidad.** El sistema debe estar desarrollado en módulos diferenciados. De esta manera no habrá problema en añadir, modificar o eliminar módulos y la mantenibilidad y escalabilidad del sistema será mucho mayor.
- **Disponibilidad.** La aplicación debe tener una disponibilidad superior al 99% del año, garantizando que los usuarios puedan acceder al sistema siempre que lo necesiten.
- **Eficiencia.** La plataforma debe ser capaz de optimizar el uso de los recursos al servicio del servidor para maximizar la eficiencia y reducir los costes operativos.
- **Compatibilidad.** La aplicación debe ser compatible con navegadores web modernos (Chrome, Firefox, Safari, Edge, etc.) para asegurar la disponibilidad del sistema en las principales plataformas.

4.2 Arquitectura del sistema

El sistema está basado en una arquitectura cliente-servidor estructurada en una serie de capas lógicas. Cada capa tiene claramente definidas sus funciones, facilitando así el mantenimiento y escalabilidad del sistema.

Diferenciamos tres grandes capas, tal y como se puede ver en la Figura 4.1.

- **Capa de presentación (Cliente).** La capa de presentación o capa cliente es la que gestiona la interacción usuario-sistema. Permite a los usuarios enviar peticiones al sistema y recibir las respuestas del mismo. Es decir, se encarga de recoger las entradas del usuario, enviar solicitudes al servidor y mostrar los resultados de esas solicitudes al usuario a través de la interfaz gráfica.
- **Capa de lógica de negocio (Servidor web).** Tal y como indica su nombre, esta capa se encarga de la lógica de negocio del sistema. Procesa las peticiones que recibe el servidor por parte de la capa cliente y realiza la interacción con la base de datos para obtener, añadir, modificar o eliminar los datos necesarios. Por último, envía de nuevo las respuestas en un formato apropiado al cliente.
- **Capa de acceso a datos (Base de datos).** Por último, tenemos la capa de acceso a datos. Esta capa realiza la gestión de los datos. Almacena los datos relevantes y provee de mecanismos de consulta y actualización de los mismos. Asegura a su vez la integridad y consistencia de los datos del sistema.

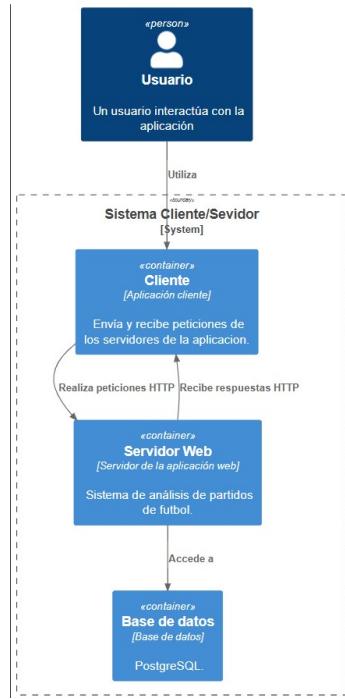


Figura 4.1: Arquitectura del sistema.

4.3 Interfaz de usuario

En la fase inicial del proyecto se realizó un prototipo de lo que sería la interfaz de usuario de la aplicación. A continuación, se muestran y se explican brevemente las ideas y pantallas principales que se diseñaron inicialmente para la plataforma. La versión final de la plataforma se basa en estos prototipos, pero aparecen algunos cambios decididos posteriormente.

Una vez el usuario ya ha iniciado sesión, ya sea como administrador o como usuario estándar, se accede a la pantalla de la Figura 4.2. En ella se puede ver un valor de entrada de selección en el que el usuario debe indicar el partido que desea analizar. Además, al seleccionar un partido nos aparecerán las opciones que se muestran en la barra de navegación para las diferentes funcionalidades. Esta barra de navegación nos acompañará por las diferentes pantallas siempre que esté un partido seleccionado. También se puede ver en la parte inferior un botón para añadir un nuevo partido. Este botón es simplemente un botón de navegación a la pantalla de la Figura 4.3.

En la Figura 4.3 se ve la pantalla en la que un usuario puede agregar datos para nuevos partidos. Aquí simplemente el usuario deberá seleccionar los archivos de datos que la plataforma se encargará de procesar y almacenar en la base de datos y almacenamiento local. Una vez se pulse el botón de confirmar se navega a la Figura 4.4.

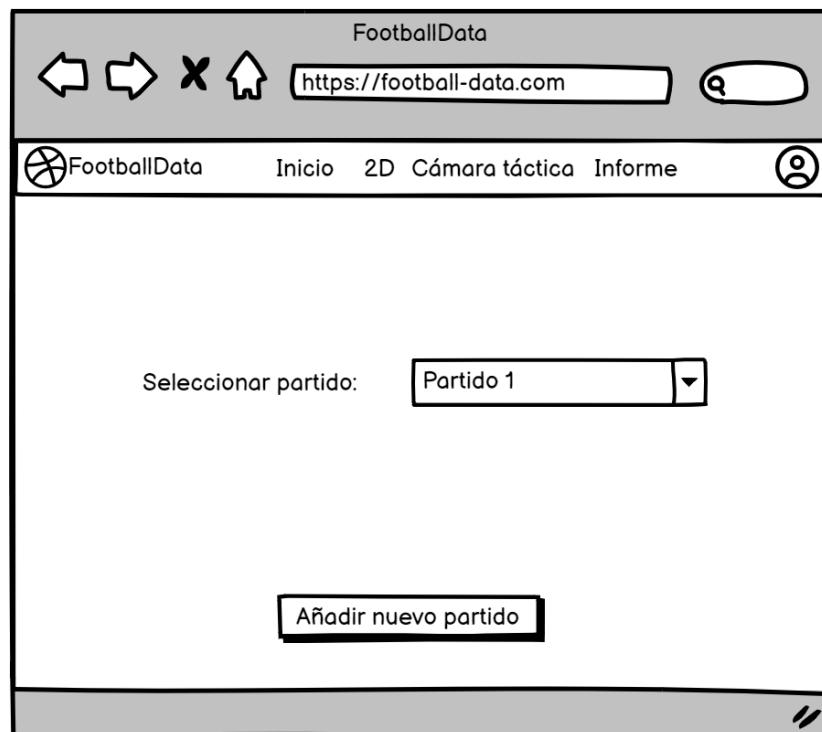


Figura 4.2: Pantalla de selección de partido.

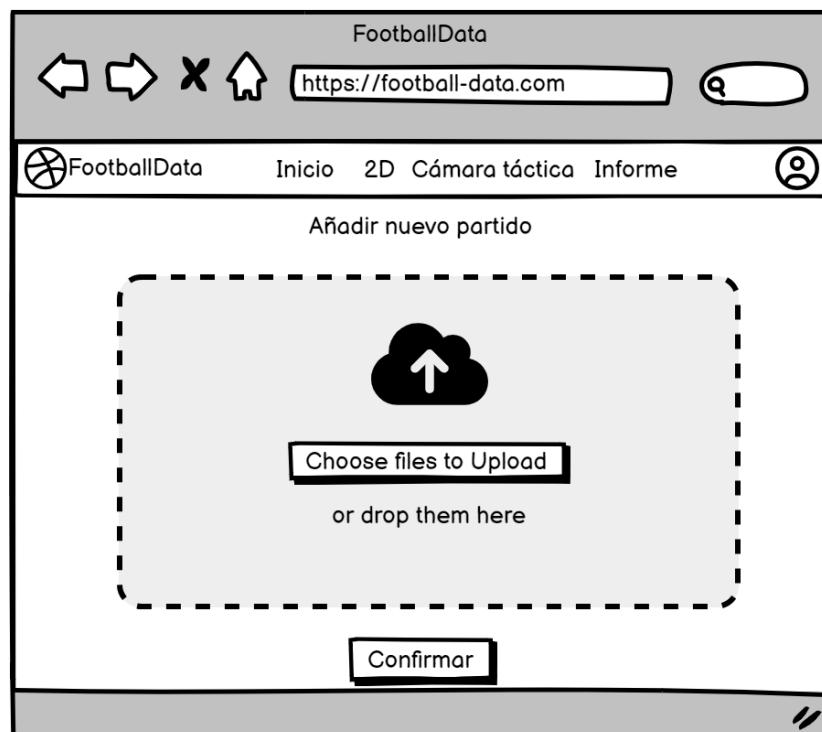


Figura 4.3: Pantalla de carga de nuevos partidos.

En la [Figura 4.4](#) se observa la pantalla correspondiente a la selección de la actividad deseada. En esta pantalla se puede seleccionar cada una de las tres grandes funcionalidades de la plataforma y el partido seleccionado anteriormente. Esta es la que se podría considerar la pantalla principal de la plataforma una vez hemos iniciado sesión. Clicando en uno de los tres paneles navegamos a las diferentes funcionalidades de la plataforma.

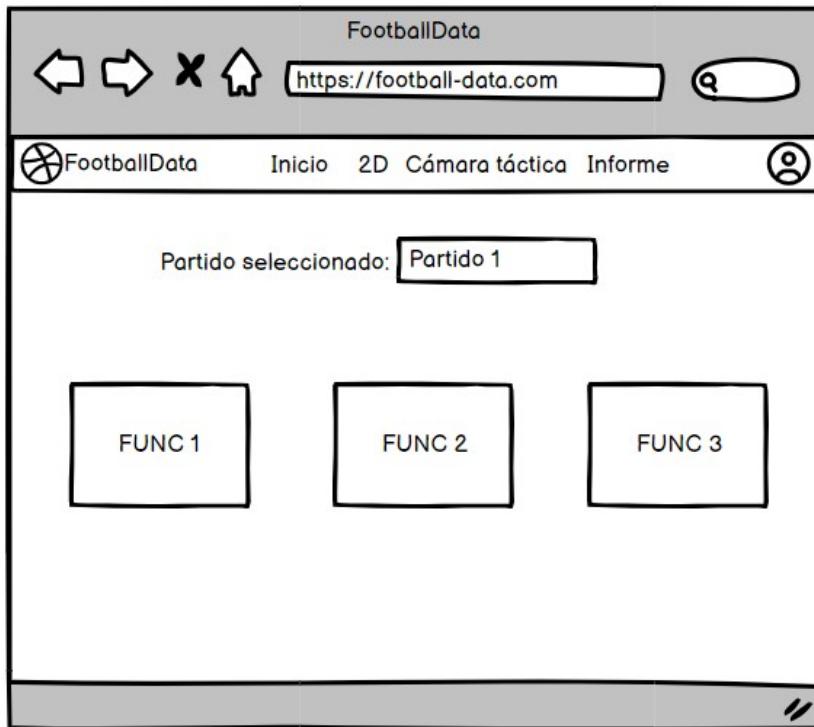


Figura 4.4: Pantalla de selección de funcionalidades principales.

En la [Figura 4.5](#) podemos ver que a la derecha es donde se visiona el partido grabado por una cámara táctica. En la izquierda se disponen los filtros aplicables junto con un botón que al clicarlo cargará los clips de vídeo de la [Figura 4.6](#).

En la [Figura 4.6](#) podemos ver los clips de vídeo ya cargados y su disposición en pequeños paneles en la pantalla. A su vez, también se mantienen los filtros aplicables para más búsquedas.

La [Figura 4.7](#) trata de la segunda de las funcionalidades. En el panel más grande es donde se dispone el terreno de juego en dos dimensiones y donde se mueven los iconos de los jugadores y el balón. En la parte superior del panel hay dos botones. El primero de ellos mostrará los iconos de jugadores y balón al inicio del encuentro. El segundo borrará todos los iconos en cualquier momento. Los dos valores de entrada de la parte inferior, minuto inicial y minuto final, se corresponden con los valores de los minutos entre los que se mostrará la visualización una vez pulsemos el botón de comienzo que podemos ver un poco más hacia la derecha, con

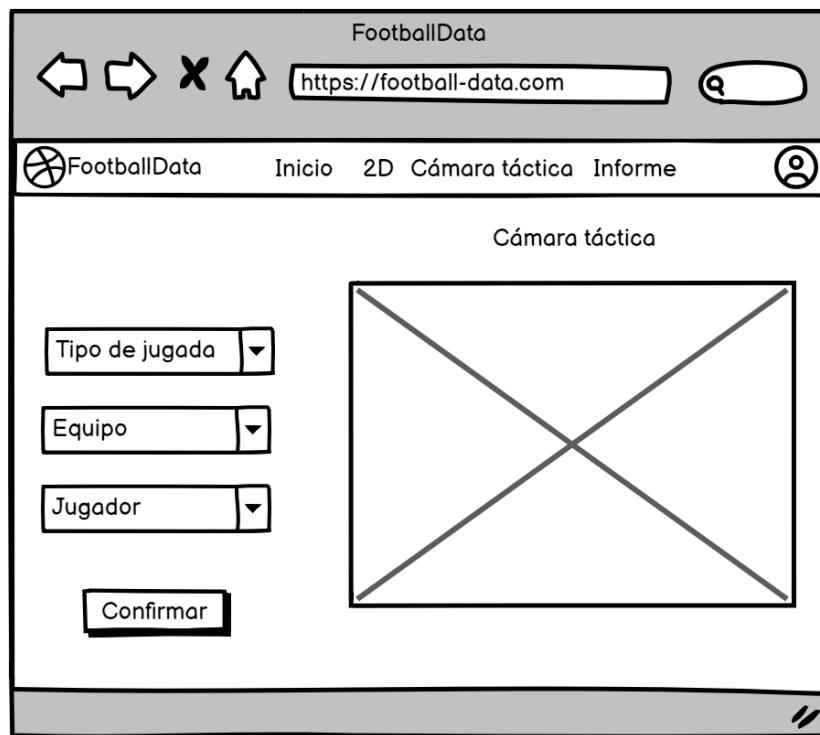


Figura 4.5: Pantalla de visualización de cámara táctica y selección de filtros.

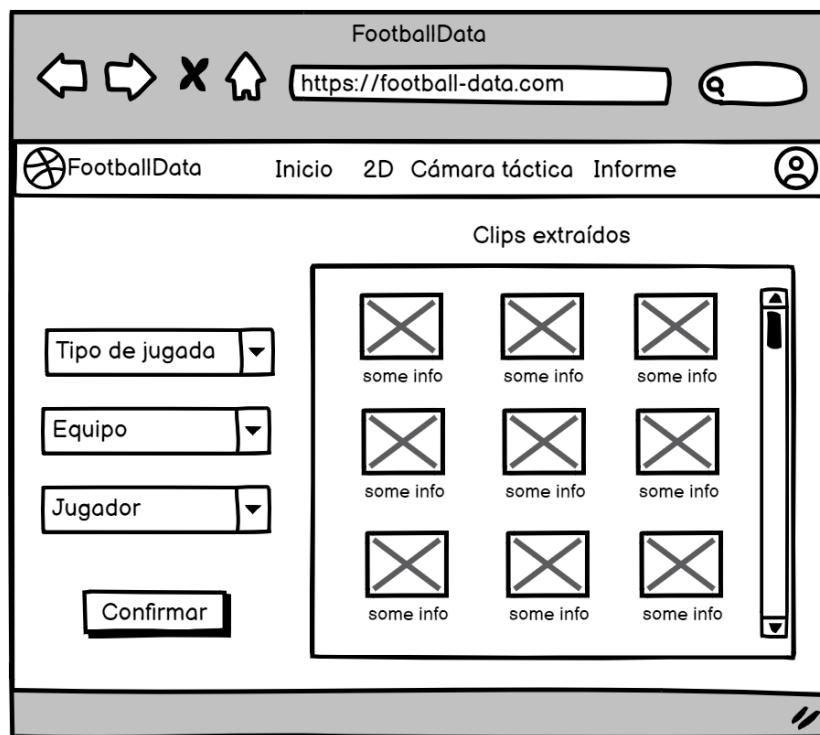


Figura 4.6: Pantalla de visualización de clips de vídeo extraídos.

símbolo de flecha. A su lado se encuentra también el botón de parar la visualización, que detendrá el tiempo de visualización en cualquier momento. Por último, vemos que hay a la derecha cuatro valores de selección. Los dos primeros se corresponden con los jugadores y nos dirigen a la [Figura 4.8](#) correspondiente a cada jugador. Por otro lado, la selección de un gol o corner nos lleva a la visualización en el terreno en dos dimensiones de esos eventos.

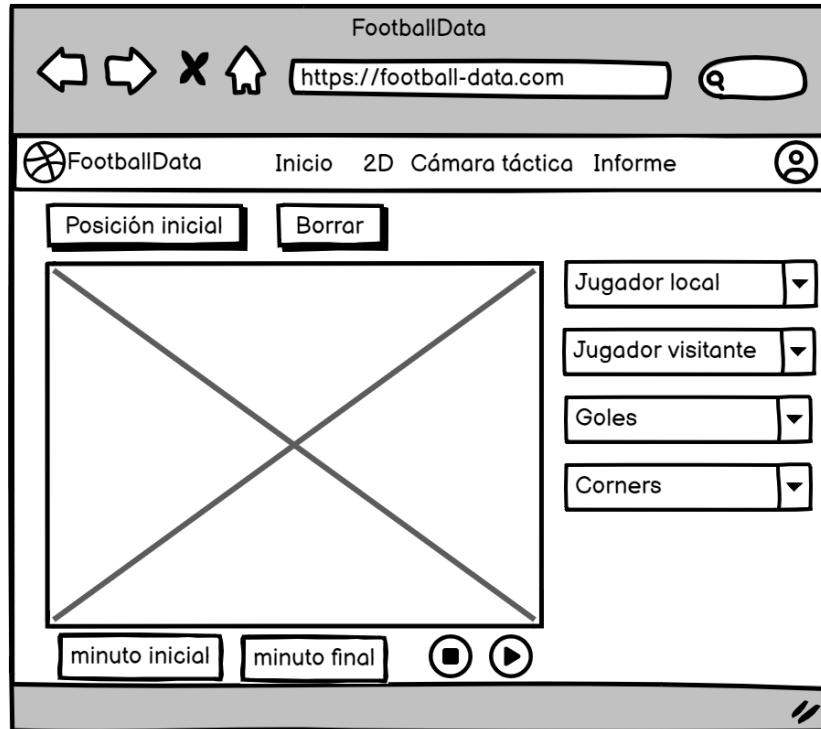


Figura 4.7: Pantalla de análisis en dos dimensiones.

En la [Figura 4.8](#) visualizamos las estadísticas de cada jugador. Aquí ha sido representados de manera muy simple por no tener todavía bien definidos los requisitos de este apartado. Solamente se muestran en este prototipo algunos valores del jugador y una imagen en representación de un posible gráfico. Lo mismo ocurre en la [Figura 4.9](#), funcionalidad que todavía no estaba bien definida a la hora de desarrollar el prototipo inicial de la aplicación y que por lo tanto no se muestra con gran detalle.

4.4 Modelo conceptual de datos

A continuación se describe el modelo de datos utilizado para el desarrollo del proyecto. Se van a describir las entidades creadas y los atributos de cada tabla. El modelo de datos se muestra de manera gráfica en la [Figura 4.10](#). A partir de ese diagrama, se han creado las siguientes tablas y atributos de cada tabla:



Figura 4.8: Pantalla de estadísticas de un jugador.

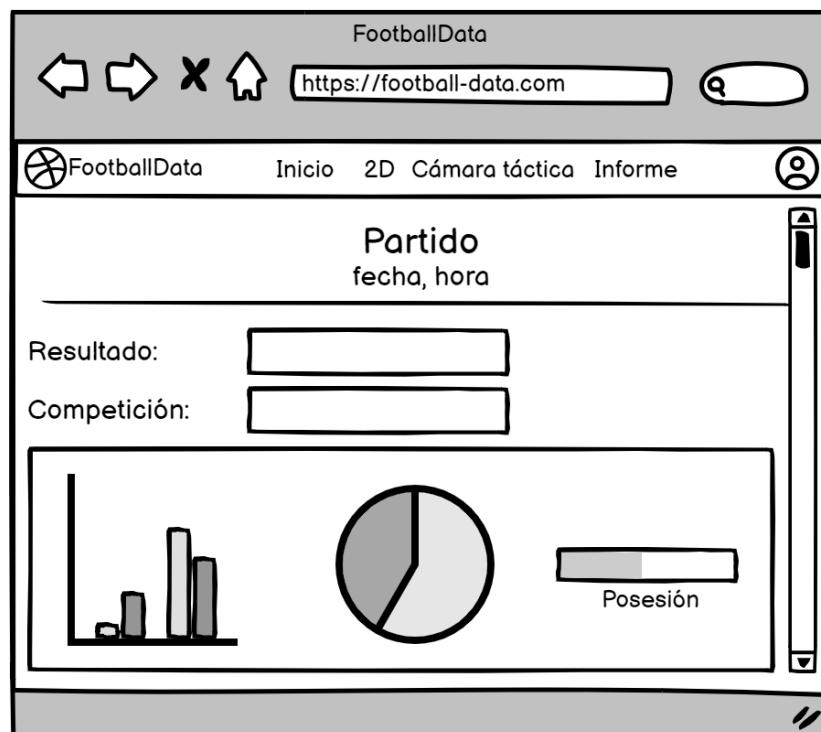


Figura 4.9: Pantalla de visualización del informe.

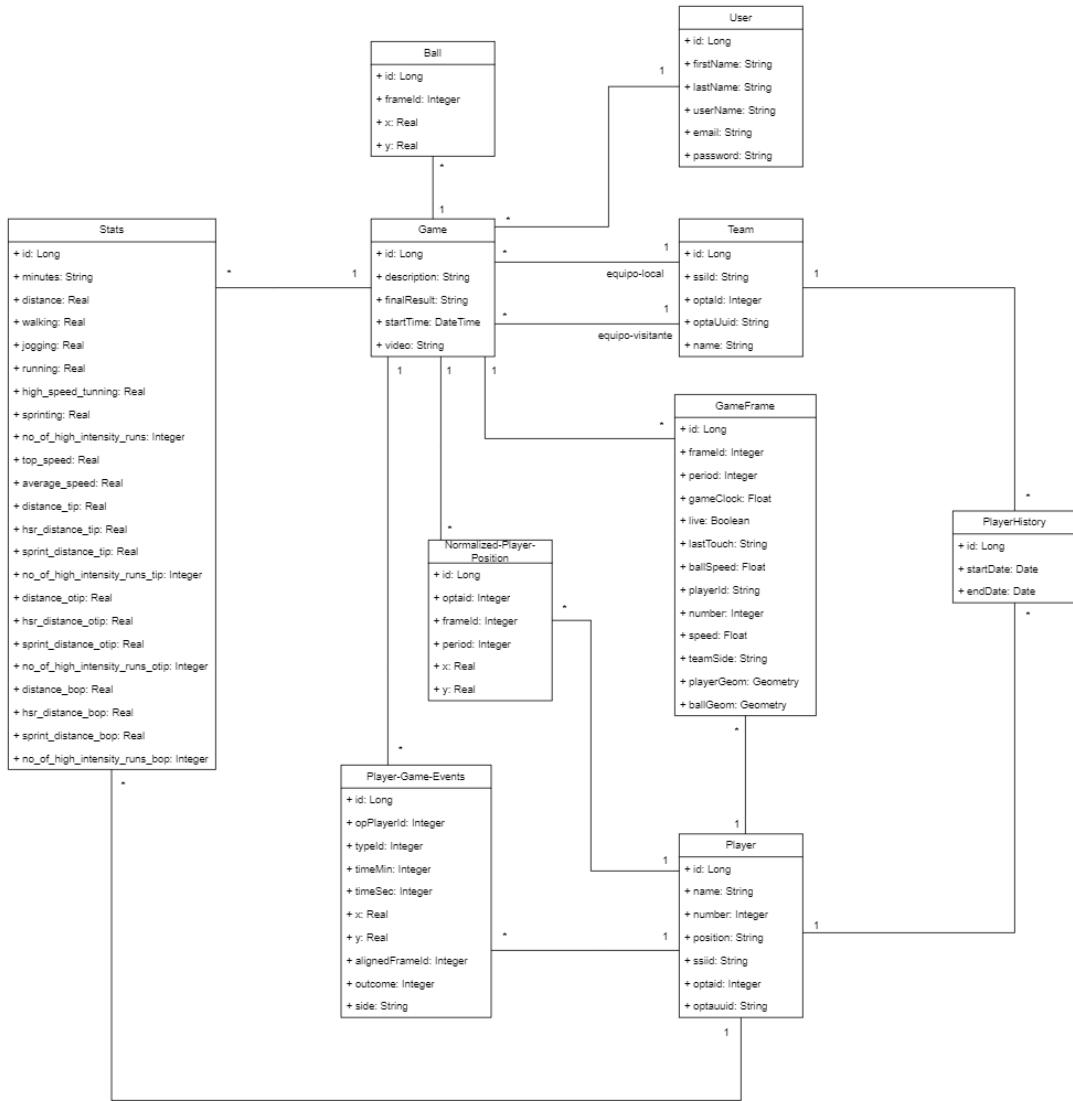


Figura 4.10: Diagrama entidad-relación del modelo de datos.

- **User (Usuario).**

- *Id (Identificador)*. Valor único que diferencia cada usuario de los demás. Funciona como identificador de cada usuario.
- *Email (Correo)*. Dirección de correo electrónico del usuario.
- *FirstName (Nombre)*. Nombre del usuario.
- *LastName (Apellidos)*. Apellidos del usuario.
- *Password (Contraseña)*. Contraseña del usuario.
- *Role (Rol)*. Rol asignado al usuario, indicando sus permisos y funciones dentro del sistema.
- *UserName (Nombre de usuario)*. Nombre de usuario utilizado para identificarse en el sistema.

- **Player (Jugador).**

- *Id (Identificador)*. Valor único que diferencia cada jugador de los demás. Funciona como identificador de cada jugador.
- *Name (Nombre)*. Nombre del jugador.
- *Number (Dorsal)*. Número que porta el jugador en su camiseta.
- *OptaId (Identificador opta de jugador)*. Identificador único asignado por Opta.
- *Position (Posición)*. Posición en la que juega el jugador.

- **Ball (Balón).**

- *Id (Identificador)*. Valor único que diferencia cada frame del balón de los demás. Funciona como identificador de cada frame del balón.
- *FrameId (Identificador de frame)*. Identificador del frame específico donde se encuentra el balón.
- *X (Eje x)*. Coordenada X del balón respecto al centro del campo.
- *Y (Eje y)*. Coordenada Y del balón respecto al centro del campo.
- *GameId (Identificador de partido)*. Identificador del partido al que pertenece el frame del balón.

- **Stats (Estadísticas).**

- *Id (Identificador)*. Valor único que diferencia cada estadística de un jugador y partido de las demás. Funciona como identificador de cada estadística para un partido y jugador concretos.

- *AverageSpeed (Velocidad media)*. Velocidad media del jugador durante el partido.
- *Distance (Distancia)*. Distancia total recorrida por el jugador.
- *HighSpeedRunning (Carreras a alta velocidad)*. Distancia recorrida a alta velocidad.
- *Jogging (Trote)*. Distancia recorrida trotando.
- *Minutes (Minutos)*. Minutos jugados por el jugador.
- *NumberOfHighIntensityRuns (Número de carreras de alta intensidad)*. Número de carreras realizadas a alta intensidad.
- *Running (Corriendo)*. Distancia recorrida corriendo.
- *Sprinting (Sprint)*. Distancia recorrida esprintando.
- *TopSpeed (Velocidad máxima)*. Velocidad máxima alcanzada por el jugador.
- *Walking (Caminando)*. Distancia recorrida caminando.
- *GameId (Identificador de partido)*. Identificador del partido al que pertenecen las estadísticas.
- *PlayerId (Identificador de jugador)*. Identificador del jugador al que pertenecen las estadísticas.

• **Team (Equipo).**

- *Id (Identificador)*. Valor único que diferencia cada equipo de los demás. Funciona como identificador de cada equipo.
- *Name (Nombre)*. Nombre del equipo.
- *OptaId (Identificador opta)*. Identificador único asignado por Opta.

• **GameFrame (Instante del partido).**

- *Id (Identificador)*. Valor único que diferencia cada instante (frame) del partido de los demás. Funciona como identificador de cada instante (frame).
- *BallGeom (Geometría del balón)*. Información sobre la posición del balón.
- *BallSpeed (Velocidad del balón)*. Velocidad del balón en ese instante.
- *FrameId (Identificador de frame)*. Identificador del frame específico en ese instante del partido.
- *GameClock (Reloj del partido)*. Tiempo transcurrido en el partido en ese instante.
- *LastTouch (Último toque)*. Información sobre el último equipo que tocó el balón.
- *Live (En vivo)*. Indica si está el balón en juego en ese frame.
- *Number (Número)*. Número de frame en la secuencia del partido.

- *OptaId (Identificador opta)*. Identificador único asignado por Opta para ese jugador.
- *Period (Periodo)*. Periodo del partido (primer tiempo o segundo tiempo).
- *PlayerGeom (Geometría del jugador)*. Información sobre la posición del jugador.
- *PlayerLargeId (Identificador de jugador de tipo cadena de caracteres.)*. Identificador único para un jugador.
- *Speed (Velocidad)*. Velocidad del jugador en ese instante.
- *TeamSide (Lado del equipo)*. Equipo al que pertenece el jugador (local o visitante).
- *GameId (Identificador de partido)*. Identificador del partido al que pertenece el frame.
- *PlayerId (Identificador de jugador)*. Identificador del jugador.

- **PlayerGameEvents (Eventos del partido y jugador).**

- *Id (Identificador)*. Valor único que diferencia cada evento de los demás. Funciona como identificador de cada evento.
- *AlignedFrameId (Identificador de frame alineado)*. Identificador del frame alineado con el evento.
- *Minute (Minuto)*. Minuto del partido en el que ocurre el evento.
- *OpPlayerId (Identificador de jugador oponente)*. Identificador del jugador oponente involucrado en el evento.
- *Second (Segundo)*. Segundo del minuto en el que ocurre el evento.
- *Side (Lado)*. Lado del campo donde se saca el saque de esquina (para el resto de eventos es nulo).
- *TypeId (Identificador de tipo)*. Valor numérico que identifica el tipo de evento.
- *X (Eje x)*. Coordenada X del evento respecto a la esquina inferior izquierda del campo.
- *Y (Eje y)*. Coordenada Y del evento respecto a la esquina inferior izquierda del campo.
- *GameId (Identificador de partido)*. Identificador del partido al que pertenece el evento.
- *PlayerId (Identificador de jugador)*. Identificador del jugador involucrado en el evento.

- **PlayerHistory (Historial de jugador).**

- *Id (Identificador)*. Valor único que diferencia cada historial de los demás. Funciona como identificador de cada historial.
- *EndDate (Fecha de finalización)*. Fecha en la que finaliza el historial del jugador con un equipo.
- *StartDate (Fecha de inicio)*. Fecha en la que inicia el historial del jugador con un equipo.
- *PlayerId (Identificador de jugador)*. Identificador del jugador al que pertenece el historial.
- *TeamId (Identificador de equipo)*. Identificador del equipo al que pertenece el historial del jugador.

- **PlayerPosition (Posición de jugador).**

- *Id (Identificador)*. Valor único que diferencia cada posición de un jugador de las demás. Funciona como identificador de cada posición.
- *FrameId (Identificador de frame)*. Identificador del frame específico donde se encuentra el jugador.
- *OptaId (Identificador opta)*. Identificador único de un jugador asignado por Opta.
- *Period (Periodo)*. Periodo del partido (primer tiempo o segundo tiempo).
- *X (Eje x)*. Coordenada X de la posición del jugador respecto al centro del campo.
- *Y (Eje y)*. Coordenada Y de la posición del jugador respecto al centro del campo.
- *GameId (Identificador de partido)*. Identificador del partido al que pertenece la posición.
- *PlayerId (Identificador de jugador)*. Identificador del jugador en esa posición.

4.5 Principales “endpoints” de la aplicación web

El código fuente del backend de la aplicación es el que define la API. Se van a definir algunos de los endpoints de la API, más bien a modo de ejemplo que a modo de documentación. Los endpoints de la aplicación están definidos a lo largo de seis controladores diferentes. Se van a definir los endpoints agrupados en los diferentes controladores.

- **Controlador de usuarios.** Se definen todos los endpoints correspondientes a las operaciones relacionadas con usuarios ([Cuadro 4.1](#)).
- **Controlador de partidos.** Se definen todos los endpoints correspondientes a las operaciones relacionadas con partidos ([Cuadro 4.2](#)).

- **Controlador de jugadas.** Se definen todos los endpoints correspondientes a las operaciones relacionadas con las jugadas de los partidos ([Cuadro 4.3](#)).
- **Controlador de estadísticas.** Se definen todos los endpoints correspondientes a las operaciones relacionadas con estadísticas de jugadores y equipos ([Cuadro 4.4](#)).
- **Controlador de datos de jugadores.** Se definen todos los endpoints correspondientes a las operaciones relacionadas con la obtención de datos de jugadores ([Cuadro 4.5](#)).
- **Controlador de mapa 2D.** Se definen todos los endpoints correspondientes a las operaciones relacionadas con la visualización 2D ([Cuadro 4.6](#)).

Método y ruta	Acción
POST /user/signUp	Registrar nuevo usuario
POST /user/login	Iniciar sesión de un usuario
PUT /user/{id}	Actualizar usuario

Cuadro 4.1: Endpoints de la API relacionados con los usuarios.

Método y ruta	Acción
GET /games/all	Obtener todos los partidos
GET /games/game	Obtener partido concreto
POST /games/upload/data	Cargar archivo de datos de frames
POST /games/upload/insight	Cargar archivo de eventos
POST /games/metadata	Cargar archivo de datos del partido
POST /games/summary	Cargar archivo de estadísticas
POST /games/video	Cargar archivo de vídeo

Cuadro 4.2: Endpoints de la API relacionados con los partidos.

Método y ruta	Acción
GET /jugadas/desmarca_ruptura	Obtener clips de desmarques en ruptura
GET /jugadas/desmarca_apoyo	Obtener clips de desmarques en apoyo
GET /jugadas/ataque_banda	Obtener clips de ataques por banda
GET /jugadas/contraataque	Obtener clips de contraataques

Cuadro 4.3: Endpoints de la API relacionados con las jugadas de los partidos.

Método y ruta	Acción
GET /stats/possession	Obtener datos de posesión de un partido
GET /stats/team-distance	Obtener distancia recorrida por un equipo
GET /stats/team-stats	Obtener estadísticas de un equipo
GET /stats/top-speed	Obtener los tres jugadores más rápidos
GET /stats/player-stats	Obtener estadísticas de un jugador

Cuadro 4.4: Endpoints de la API relacionados con las estadísticas.

Método y ruta	Acción
GET /jugadores/all	Obtener todos los jugadores
GET /jugadores/{id}	Obtener jugador por ID
GET /jugadores/optaId/{id}	Obtener jugador por Opta ID
GET /jugadores/local/{gameId}	Obtener jugadores locales
GET /jugadores/visitante/{gameId}	Obtener jugadores visitantes

Cuadro 4.5: Endpoints de la API relacionados con los datos de los jugadores.

Método y ruta	Acción
GET /2d/dataframe/players	Obtener datos jugadores
GET /2d/dataframe/ball	Obtener datos de balón
GET /2d/player_stats/{playerId}/{gameId}	Obtener estadísticas
GET /2d/heat_data/{playerId}/{gameId}	Obtener mapa de calor
GET /2d/top_three/{stat}/{gameId}	Obtener top 3 por estadística
GET /2d/getGoals/{gameId}	Obtener goles
GET /2d/getCorners/{gameId}	Obtener saques de esquina
GET /2d/getCards/{gameId}	Obtener tarjetas
GET /2d/getFouls/{gameId}	Obtener faltas
GET /2d/getSubOn/{gameId}	Obtener cambios
GET /2d/getSubOff/{gameId}	Obtener cambios
GET /2d/getEvents/{gameId}	Obtener eventos

Cuadro 4.6: Endpoints de la API relacionados con la visualización 2D.

Capítulo 5

Diseño

5.1 Arquitectura tecnológica del sistema

Profundizando en la arquitectura del sistema, y a partir del esquema mostrado en la Figura 4.1, va a ser explicado ahora un esquema de la arquitectura del sistema más completo. En él se pueden identificar los componentes mostrados anteriormente de una manera más detallada junto con las tecnologías utilizadas en cada uno de ellos.

El servidor web está desarrollado, casi en su totalidad, en lenguaje Java, apoyado a su vez por el framework Spring. Spring es un framework conocido por su facilidad a la hora de crear aplicaciones web y APIs. La estructura del servidor está organizada en varios componentes. Por un lado están los repositorios, que están encargados de realizar las operaciones en la base de datos. En segundo lugar tenemos los modelos, que son los que definen las estructuras de datos que dan forma a su vez a la base de datos. Los servicios son los que contienen toda la lógica de negocio del sistema. Por último, los controladores se encargan de manejar las peticiones HTTP que recibe la plataforma por parte del cliente y definen la API. Maven se utiliza como herramienta de gestión de proyectos para manejar las dependencias de la aplicación.

El otro de los componentes generales del sistema es el cliente web. Está desarrollado usando tecnologías modernas de desarrollo web. La interfaz de usuario fue desarrollada en HTML y CSS, dos lenguajes de programación que definen la estructura y los estilos visuales que tiene la plataforma. Para desarrollar las interacciones y dinámicas de la interfaz se utiliza Angular, un framework muy potente para la creación de aplicaciones web de una sola página, conocidas también por sus siglas como páginas web SPA. Angular a su vez utiliza TypeScript, un lenguaje de programación que se trata realmente de un superconjunto de JavaScript que añade tipos estáticos.

Para la base de datos se utilizó PostgreSQL, un sistema de gestión de bases de datos relacionales junto a su extensión PostGIS. Esta extensión permite y facilita el almacenamiento y manipulación de datos espaciales.

Todo lo citado anteriormente garantiza que la plataforma sea fácilmente mantenible en el tiempo, tanto por su estructura como las tecnologías utilizadas. Angular y Spring son tecnologías modernas altamente consolidadas en la industria del desarrollo de software, y cuentan con el apoyo de grandes comunidades activas que las sostienen. Además, ofrecen una arquitectura robusta y modular que las hace altamente escalables y eficientes. El alto respaldo del que disponen asegura que ambas tecnologías se mantengan actualizadas con las mejores prácticas y tendencias de este sector a lo largo de los años.

Se sigue claramente una arquitectura cliente-servidor. Un usuario interactúa a través de la interfaz de usuario con el cliente. El cliente web es el encargado de enviar solicitudes [HTTP](#) al servidor web, que debe recibir estas solicitudes y manejarlas. El manejo puede implicar extraer información de la base de datos, actualizar información en base de datos, ejecutar lógica de negocio del sistema, etc. Más concretamente, los controladores son los encargados de recibir y redirigir estas peticiones a los servicios adecuados, y son los servicios los que utilizan los modelos y repositorios para acceder a base de datos y realizar las operaciones oportunas. Por último, el servidor devuelve al cliente una respuesta [HTTP](#) que de alguna manera se verá reflejada en la interfaz de usuario llegando al usuario final.

5.2 Diseño de la aplicación

En esta sección se va a explicar la estructura tanto del [backend](#) como del [frontend](#). Se explica a través de la [Figura 5.2](#) y la [Figura 5.3](#), por lo tanto futuras posibles referencias de paquetes, directorios o archivos serán a partir de ambas figuras.

La carpeta `src` es la que contiene prácticamente todo el código fuente del backend de la aplicación. Dentro de esta se diferencian otros tres directorios: `main`, `sql` y `test`. Empezando por el directorio `test`, tal y como su nombre indica podemos identificar la localización de todas las pruebas implementadas en la aplicación. Por un lado están las pruebas unitarias, que son las que se encuentran en el directorio `entities` y por otro lado se encuentran las pruebas de integración, que son las que se encuentran en el directorio `services`. Siguiendo por el directorio de `sql`, se localiza un `script` de configuración de la base de datos. Este archivo contiene una serie de acciones que se deben de ejecutar en la base de datos `sql` anteriormente a ejecutar la aplicación.

En el directorio `main` se encuentra el grueso de la aplicación. Por un lado en el directorio `resources` se localiza el archivo de propiedades de la aplicación. Ahí están definidos aspectos fundamentales de configuración de la base de datos como el nombre de la misma, el nombre de usuario, la contraseña, el modo de actualización y otros valores como el puerto del servidor. En el directorio `java` se encuentran todas las clases de Java que definen la aplicación. Diferenciamos en primer lugar entre dos carpetas: `model` y `rest`. Dentro de `model` se diferen-

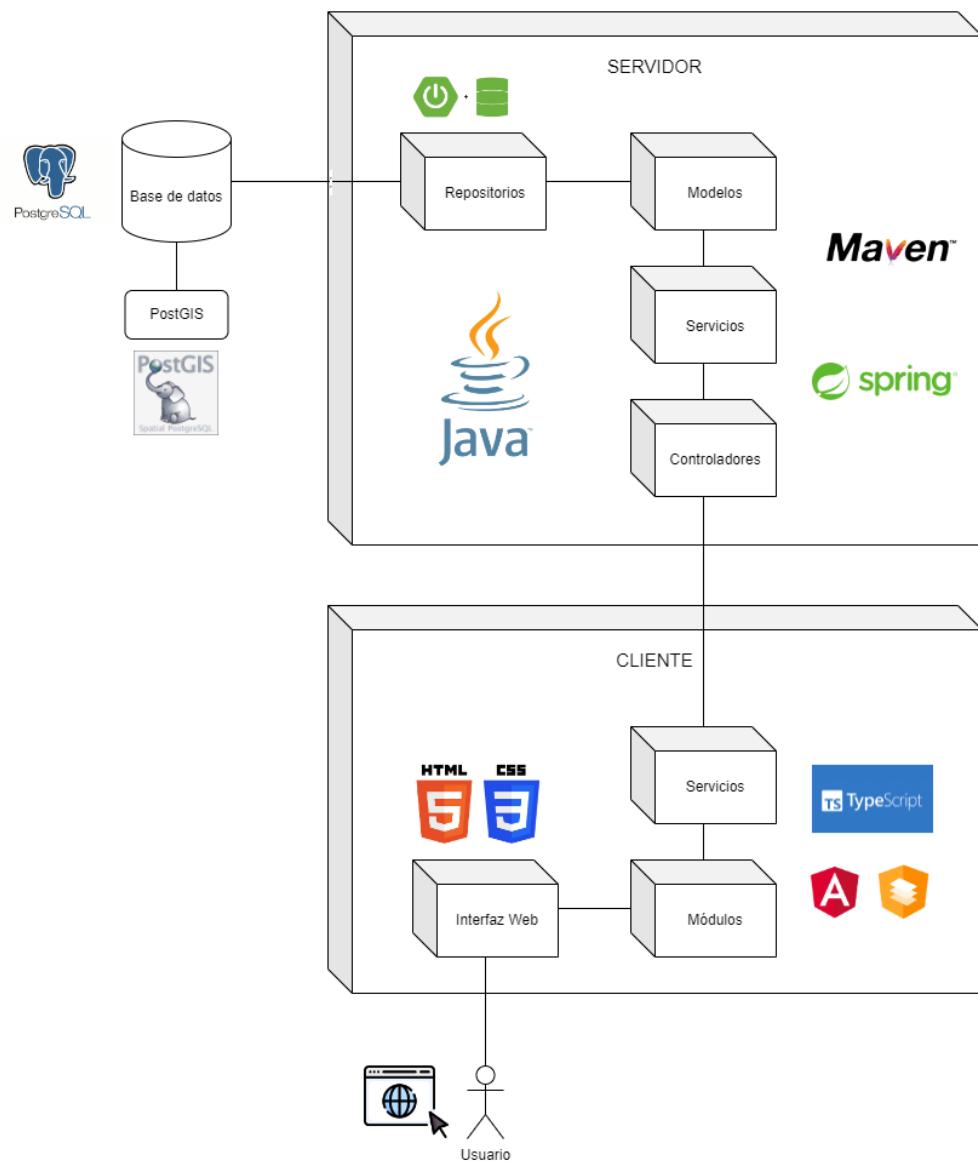


Figura 5.1: Esquema de la arquitectura y tecnologías utilizadas.

cian a su vez las carpetas *entities*, *exceptions* y *services*. Los nombres son bastante intuitivos. En el directorio *entities* se localizan todas las entidades de Spring que definen las tablas de la base de datos. En *exceptions* se encuentran algunos archivos de excepciones personalizadas que se han considerado oportunos crear. En *services* están todos los servicios que mantienen la lógica de negocio de la plataforma.

Dentro del directorio *rest* se diferencian también otros tres directorios: *common*, *controllers* y *dtos*. El primero de ellos almacena todos los archivos que son comunes a varios componentes y que no se localizan en los otros directorios. Ejemplos son el archivo de configuración de la seguridad y algunos modelos de datos que fueron necesarios en algunos servicios. En el segundo de los directorios se encuentran los controladores de la plataforma, que definen los endpoints que reciben y devuelven las respuestas [HTTP](#) al cliente. Ya por último tenemos los *DTOs* que definen los objetos que se utilizan para transferir datos entre el servidor y el cliente.

Por último, es importante también mencionar el archivo *pom.xml*, que contiene todas las dependencias gestionadas por Maven que necesita la plataforma para ejecutarse y funcionar adecuadamente.

Para el frontend, se diferencia en primer lugar el directorio *src* y varios archivos de configuración como el archivo *angular.json* que define la estructura del proyecto y las configuraciones de compilación.

La carpeta *src* es la que contiene todos los componentes y servicios Angular de la aplicación. El archivo *global-error-handler.ts* define un manejador de errores global. Se ven también otros dos directorios que son el *assets* y el *app*. El primero contiene todos los archivos multimedia. En *images* se almacenan todas los archivos de imágenes que son utilizados en la plataforma mientras que en *video* se encuentran los vídeos de los partidos que se suben a la plataforma. En *app* se diferencian a su vez otros tres directorios. En todos ellos hay definidos componentes y servicios de Angular. El directorio *shared* contiene los componentes, servicios e incluso modelos de datos que son utilizados por varios componentes. En *core* se localizan los componentes y servicios generales de la plataforma, como son la barra de navegación y el pie de página, y también los servicios que son utilizados por varios componentes. Por último, el directorio *modules* es el que contiene la mayor parte de los componentes de la aplicación. Es aquí donde se definen todos los componentes Angular restantes que son utilizados. Otros dos archivos que merecen mención son el *app.module.ts*, que define los componentes, servicios y otros módulos necesarios para la aplicación, y el archivo *app-routing.module.ts* que es donde se definen las rutas y subrutas de los componentes.

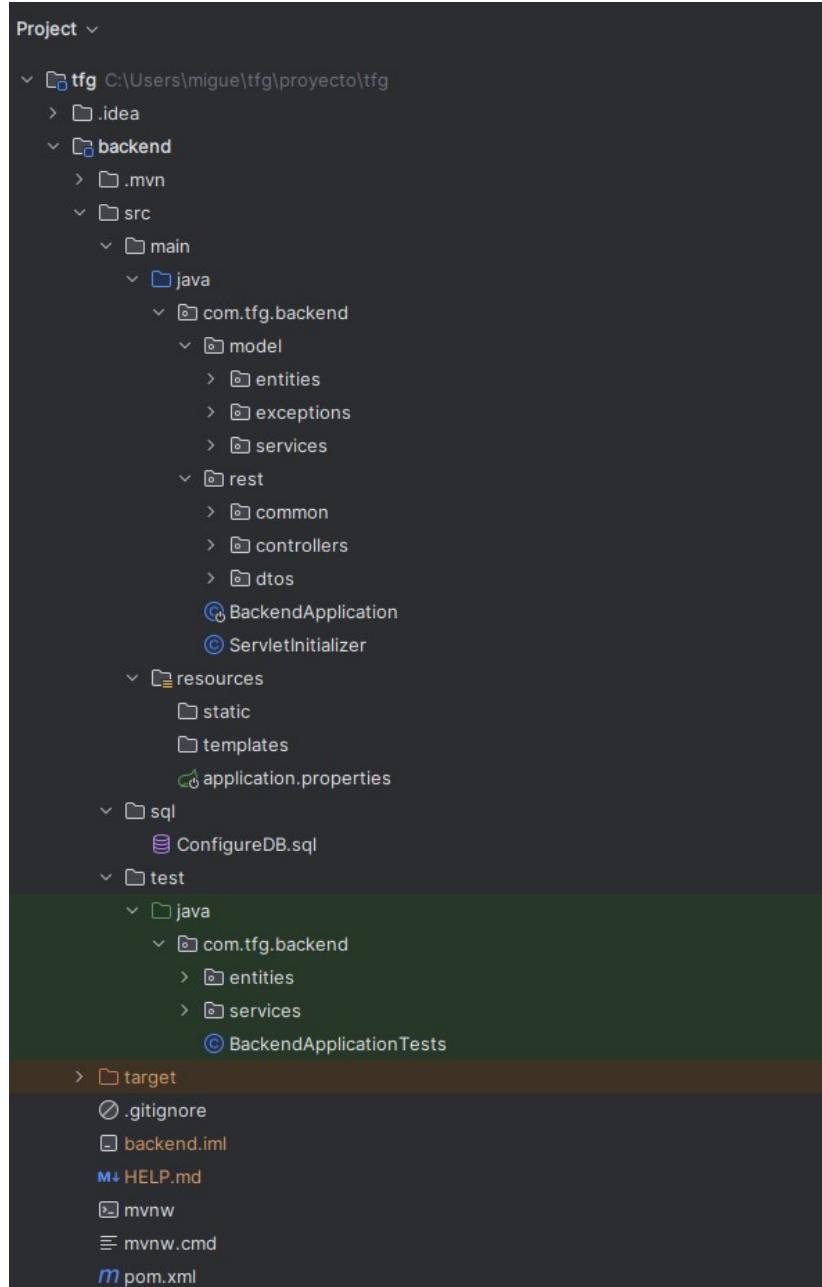


Figura 5.2: Estructura del backend de la aplicación extraído de IntelliJJ.

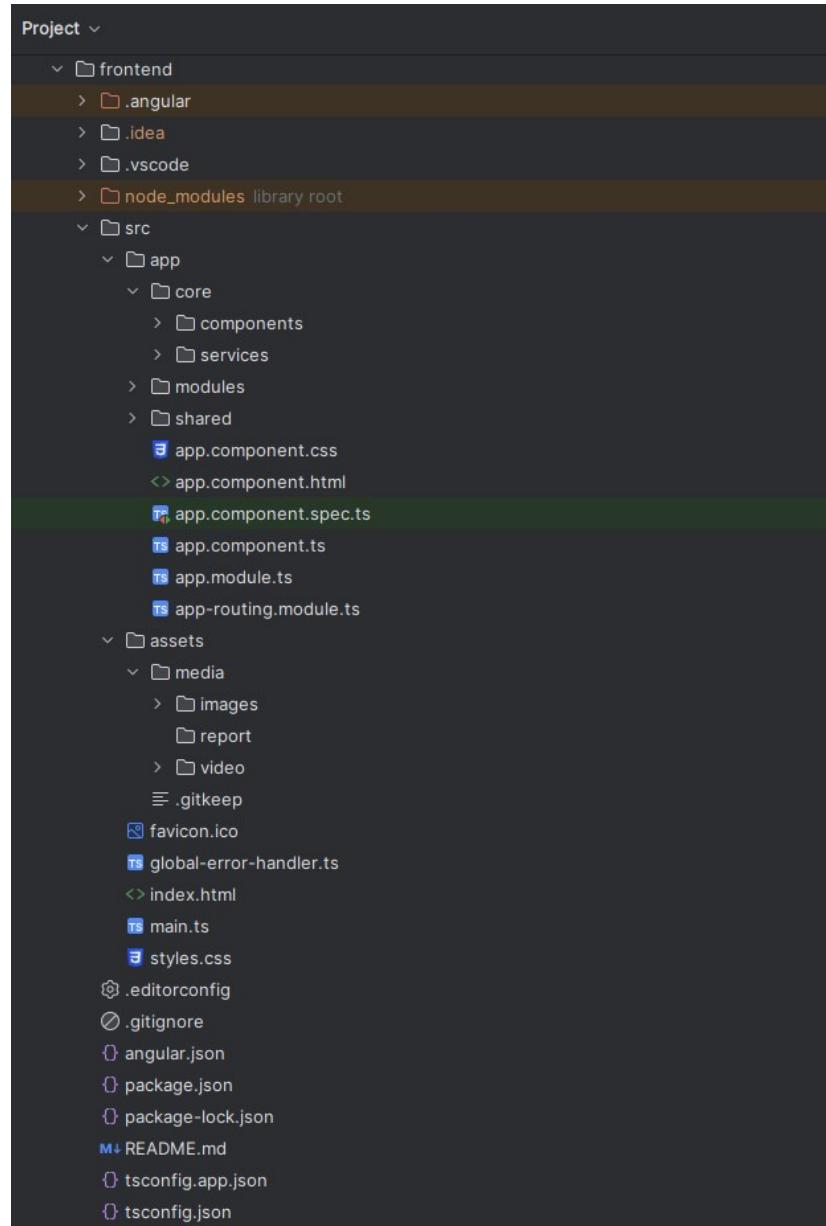


Figura 5.3: Estructura del frontend de la aplicación extraído de IntelliJ.

Capítulo 6

Implementación y pruebas

6.1 Implementación

A continuación se van a explicar algunos aspectos sobre el desarrollo de la aplicación a más bajo nivel, entrando incluso en algunos detalles del código fuente de la misma. Se van a destacar especialmente los tres siguientes puntos:

- **Sincronización de jugadores y balón en visualización 2D.** Para visualizar los iconos de los jugadores y del balón durante el partido es necesario la carga continua de los mismos en la aplicación por lotes. Estos datos se encuentran almacenados en la base de datos, y la carga de ellos simultáneamente y a tan rápida velocidad puede causar demoras. Hay que tener en cuenta que lo normal es tener datos de las posiciones de los jugadores y del balón de aproximadamente unos 25 **frames** por cada segundo de vídeo. Esto implica que se deben realizar peticiones al backend repetidamente y muy seguido, lo cual causa que si el equipo informático no es potente pueda causar demoras en la obtención de los datos, o que la obtención de los datos del balón y de los jugadores no estén sincronizadas al estar implementadas ambas solicitudes en llamadas diferentes a la [API](#). Había dos opciones para solucionar esto. La primera de ellas sería juntar la obtención de los datos en una misma consulta, una implementación más simple pero que realmente no sería una solución “ limpia ” y poco mantenible en futuros desarrollos. Por lo tanto se optó por buscar otra solución. La solución fue finalmente provocar una sincronización manual de ambas peticiones. Antes de continuar con el hilo de ejecución y por lo tanto continuar a los siguientes frames, cada vez que se realiza una petición a la API para la obtención de los datos el hilo de ejecución se detiene y espera a obtener una respuesta por parte de ambas peticiones. Esto fue conseguido a través de la utilización de promesas. Una promesa es un objeto que representa el resultado eventual de una operación asincrónica.

Como se puede ver en las siguientes líneas de código fuente se han convertido las funciones de obtención de datos en funciones asíncronas con la palabra clave de “`async`”. Por otro lado con la creación de la promesa y el uso de otra palabra clave como “`await`” conseguimos que se resuelva la promesa antes de continuar con el hilo de ejecución. De esta manera conseguimos cargar los datos y que el hilo de ejecución espere a la terminación de ambas promesas para continuar con la ejecución de las siguientes líneas de código.

```

1  async fetchBallData(index: number) {
2      try {
3          this.dataFramesBall = await new Promise((resolve, reject) => {
4              this.getBallDataFrames(index, this.tamPagBall).subscribe({
5                  next: data => resolve(data),
6                  error: err => reject(err)
7              });
8          });
9      } catch (error) {
10         console.error('Error al obtener datos:', error);
11     }
12   }
13
14  async fetchPlayerData(index: number) {
15      try {
16          this.dataFramePlayers = await new Promise((resolve, reject) => {
17              this.getPlayerDataFrames(index,
18                  this.tamPagPlayers).subscribe({
19                      next: data => resolve(data),
20                      error: err => reject(err)
21                  });
22          });
23      } catch (error) {
24         console.error('Error al obtener datos:', error);
25     }
26 }
```

Las siguientes líneas de código corresponden a una pequeña parte de la función en donde se solicita la carga de los datos. Como se puede ver se utiliza de nuevo la palabra clave “`await`” que es la que provoca finalmente que el hilo de ejecución no siga de forma libre y todo funcione de manera sincronizada, a pesar de ser las funciones asíncronas.

```

1 if (this.currentFrame != 1 && this.currentFrame %
2     this.numFramesPerPage == 1) {
3     index++;
4     await this.fetchBallData(index);
```

```

4     await this.fetchPlayerData(index);
5 }
```

- **Integrar modelos de datos.** Otro de los aspectos más destacadas del desarrollo fue el diseño e implementación de un nuevo modelo de datos. Como ya se ha comentado, este trabajo corresponde en parte a la integración y refactorización de otros dos trabajos de fin de grado. Esos dos trabajos disponían cada uno de sus propias bases de datos. El diseño de estas bases de datos realmente no era óptimo. Una de ellas basaba casi en su totalidad el almacenamiento de los datos en una única tabla lo que provocaba una enorme redundancia de datos, y si simplemente se unieran ambas bases de datos, esto generaba aún más redundancia. Fue por lo tanto necesario hacer un trabajo de refactorización, diseño y normalización prácticamente desde cero de todo un modelo de datos que finalmente resultó en el de la Figura 4.10. Además, por la implementación diferente de ambos trabajos, fue necesario hacer un trabajo de normalización en cuanto a las posiciones de los jugadores y balón, ya que para el análisis posicional en dos dimensiones se utilizan posiciones normalizadas según el tamaño del terreno de juego y no era para nada eficiente almacenar las posiciones normalizadas y sin normalizar en la base de datos. Simplemente se podrían normalizar posteriormente una vez obtenidos de la base de datos. Además, para el manejo de los datos geoespaciales es necesario serializar y deserializar los datos de tipo “Geometry”. Se solucionó este problema implementando la interfaz “Serializable” de Java para indicar que los objetos de esa clase pueden ser convertidos en una secuencia de bytes y luego reconstruidos a partir de esa misma secuencia. Esto resultó de nuevo en una base de datos que garantiza una mayor escalabilidad y rendimiento de la plataforma, permitiendo una muy fácil adición de nuevas filas y/o columnas a las tablas.
- **Carga de datos a partir de los archivos.** La carga de datos a partir de archivos de diferentes formatos es otro de los aspectos más destacados del desarrollo. A pesar de la existencia de diferentes formatos para los diferentes archivos que el usuario debe subir, la problemática surge de que la cantidad de datos a cargar es enorme. Los archivos de datos pueden tener miles y miles de filas, dependientes del número de frames, eventos, número de jugadores, etc. Esto provocó que inicialmente la carga de videos se pudiera demorar a largos períodos de tiempo, pudiendo alcanzar fácilmente más de una hora simplemente para la carga de datos para un partido.

Por lo tanto, como se puede ver en el siguiente fragmento de código se desarrolló un procesamiento del archivo en diferentes hilos de ejecución de forma concurrente. Se hicieron diferentes pruebas y finalmente se optó por diez hilos de ejecución por ser el número de hilos más rápido y eficiente en procesar todas las líneas. De esta manera

se mejoró drásticamente el rendimiento de las operaciones de lectura, procesamiento y almacenamiento en la base de datos.

```

1   try {
2       ObjectMapper objectMapper = new ObjectMapper();
3       ExecutorService executor =
4           Executors.newFixedThreadPool(10);
5
6       try (BufferedReader br = new BufferedReader(new
7           InputStreamReader(file.getInputStream()))) {
8           String line;
9           while ((line = br.readLine()) != null) {
10              final String jsonLine = line;
11              executor.submit(() -> {
12                  try {
13                      JsonNode jsonNode =
14                          objectMapper.readTree(jsonLine);
15                      gamesService.saveData(jsonNode, userId,
16                      gameId);
17                  } catch (Exception e) {
18                      e.printStackTrace();
19                  }
20              });
21          }
22      }
23      executor.shutdown();
24      return "Archivo subido correctamente";
25  } catch (Exception e) {
26      e.printStackTrace();
27      return "Error al subir el archivo";
28  }

```

También es de gran valor destacar que durante la implementación se mantuvo una estructura en módulos que provoca que añadir nuevas funcionalidades a la plataforma sea muy sencillo. El equipo de desarrollo sólamente debe añadir al código fuente de la plataforma los archivos que mantengan la nueva funcionalidad, sin tener que modificar ninguno de los ya existentes. Para el backend, sólamente debe añadir las nuevas entidades a la carpeta *entities*, crear los servicios en la carpeta *services* y los controladores que hagan uso de estos servicios en la carpeta *controllers*. Para el frontend, se pueden reutilizar algunos de los componentes ya creados y sólamente se deben añadir los nuevos componentes que sean necesarios a la carpeta *modules*. El resto de funcionalidades deberían seguir funcionando correctamente sin importar las nuevas adiciones. Esto de nuevo provoca que la plataforma sea altamente mantenible y escalable.

6.2 Pruebas

Una parte fundamental del desarrollo es también la implementación de pruebas. Para el proyecto se han implementado tres tipos de pruebas.

- **Pruebas unitarias.** Se implementaron pruebas unitarias, en donde se verificó la correcta implementación de partes aisladas de la aplicación. Se centró la atención principalmente en la prueba de clases individuales en representación de las entidades utilizadas en el backend.
- **Pruebas de integración.** Se implementaron también pruebas de integración. Estas pruebas verifican la interoperabilidad e integración entre diferentes componentes. En estas pruebas se dedicaron especialmente a la validación de la integración entre los métodos de los servicios, las entidades y las operaciones de los repositorios de cada entidad.
- **Pruebas de aceptación.** Este tipo de pruebas se realizan en colaboración con el cliente o usuario final para determinar, según los criterios del cliente o usuario final, si la aplicación cumple con los requisitos y expectativas. En este caso se realizó una reunión con el cliente. En la reunión, el producto cumplió las expectativas, pero se requirieron una serie de pequeños cambios y propuestas de mejora. Algunas de ellas, como el ya comentado cambio de iconos de los jugadores para ver los dorsales en la visualización de dos dimensiones se han podido realizar, y otros que requieren de más desarrollo todavía quedan pendientes para futuros desarrollos, como la visualización conjunta de el terreno en dos dimensiones y el vídeo del partido (esto es comentado en el capítulo 8 de esta memoria).
- **Otras pruebas.** También es razonable mencionar las pruebas realizadas con Postman. Esta herramienta se ha utilizado para testear las solicitudes realizadas a la [API](#) de la plataforma. Se han podido verificar cómo enviar los parámetros de la [URL](#), el cuerpo de las peticiones y sobre todo que el formato de la respuesta y los datos de la misma fueran los deseados.

Capítulo 7

Solución desarrollada

A continuación se detalla a nivel de usuario el funcionamiento de la aplicación. Dividiremos en cuatro bloques principales, cada una de las tres funcionalidades y la carga de datos de partidos.

7.1 Características de usuario.

En esta sección se comentan las características de la aplicación relacionadas con el registro e inicio de sesión de los usuarios.

En la [Figura 7.1](#) se ve lo que corresponde a la página inicial de la plataforma y la que los usuarios se encuentran al entrar en la plataforma por primera vez. Se puede ver un mensaje de bienvenida y con dos botones, uno para navegar a inicio de sesión y otro para navegar a registro de un nuevo usuario. También se dispone de una barra de navegación, que se mostrará siempre durante la navegación por todas las funcionalidades del sistema, pero cambiando alguna de sus opciones. Lo primero que hay que decir de esta barra de navegación es que el símbolo de un campo de fútbol que se encuentra a la izquierda de la barra es realmente un botón de navegación a esta misma página o a la página inicial para usuarios con sesión iniciada, dependiendo de si los usuarios han iniciado sesión o no. Por último, tenemos también otras dos opciones en la barra de navegación, Iniciar sesión y registrarse. Estos botones simplemente disponen las mismas funcionalidades que los dos botones principales de esta pantalla, pero para poder tenerlos también fácilmente accesibles desde otras pantallas de la plataforma o desde este si se prefieren.

En la [Figura 7.2](#) se puede ver un formulario de inicio de sesión. Al no estar todavía con la sesión iniciada, podemos ver que la barra de navegación de la parte superior es idéntica a la anterior. Un usuario debe introducir el nombre de usuario y contraseña con los que previamente se ha registrado. En caso de no ser válidas las credenciales se muestra un error. En el caso de ser la información correcta se dirige al usuario a la página de la [Figura 7.4](#) una vez

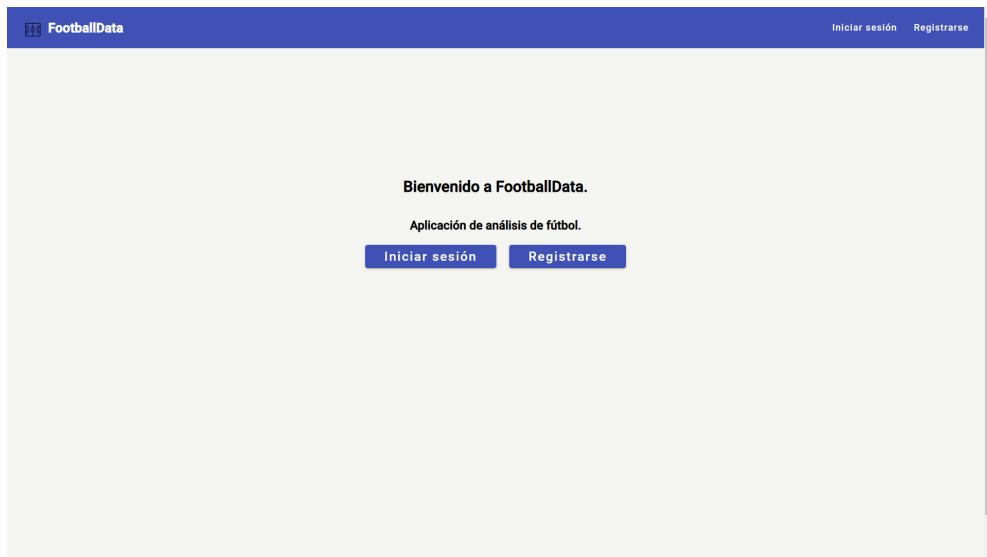


Figura 7.1: Pantalla inicial para usuarios no registrados.

haya sido clicado el botón de iniciar sesión que se encuentra justo debajo del formulario.

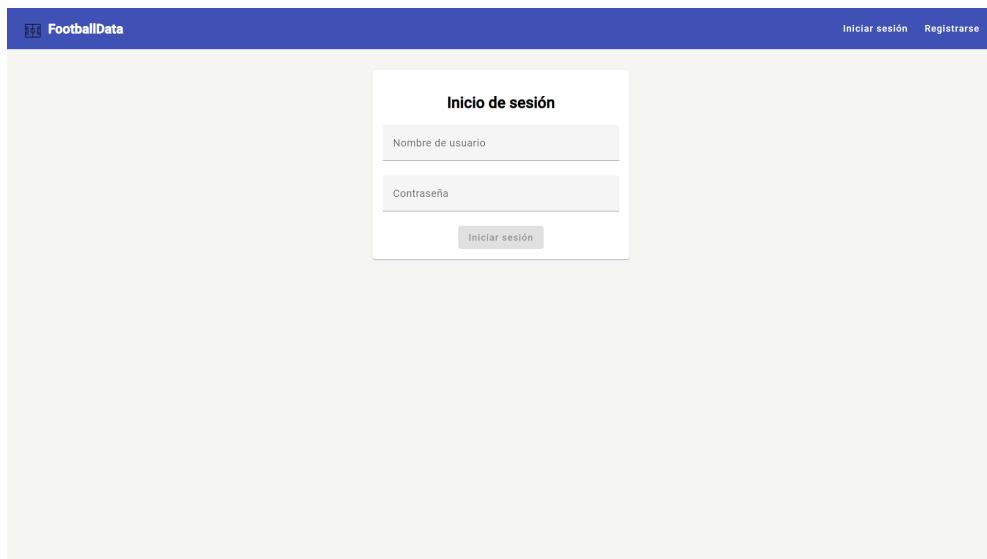


Figura 7.2: Formulario de inicio de sesión.

En la Figura 7.3 se puede observar el formulario de registro de nuevo usuario. Aquí todavía no se ha iniciado sesión tampoco y, por lo tanto, la barra de navegación sigue intacta a como se encontraba al inicio. Para registrar un nuevo usuario es necesario completar el formulario introduciendo nombre, apellidos, nombre de usuario, email, contraseña y una confirmación de la contraseña que debe ser idéntica a la contraseña. Al introducir la contraseña no se muestran los símbolos, si no que simplemente se mostrarán un número de puntos que corresponden

con el número de caracteres de la contraseña y aportan seguridad a la plataforma. Al pulsar el botón de registrarse, si hay algún dato que no es correcto se muestra un error. En cualquier otro caso se navega a la página de inicio de sesión, pero la sesión todavía no está iniciada. Es necesario posteriormente completar el formulario de inicio de sesión correctamente.

The screenshot shows a registration form titled "Registrar nuevo usuario". The form consists of six input fields: "Nombre" (Name), "Apellidos" (Last Name), "Nombre de usuario" (User Name), "Email", "Contraseña" (Password), and "Confirmar contraseña" (Confirm Password). Below the fields is a "Registrarse" (Register) button. At the top of the page, there is a header bar with the "FootballData" logo, a search bar, and navigation links for "Iniciar sesión" (Log in) and "Registrarse" (Register).

Figura 7.3: Formulario de registro de nuevo usuario.

7.2 Carga y selección de partidos.

En esta sección se comentan características relacionadas con el almacenamiento y carga de los datos para la inclusión de nuevos partidos para analizar, además de cómo se realiza la selección de un partido también para analizar.

La Figura 7.4 corresponde a la primera pantalla una vez un usuario ha iniciado sesión. Aquí se puede ver como la barra de navegación ya es diferente. Mientras que el logo y botón del icono de un terreno de juego y el nombre de la plataforma se mantienen podemos ver que las opciones de navegación a inicio de sesión y registro han desaparecido. Ahora tenemos una nueva opción en la que se muestra el nombre de usuario. Al clicar en este botón tendremos la opción de cerrar sesión que se comenta posteriormente en la descripción de la Figura 7.7. Por otro lado también tenemos un botón de selección en la que un usuario administrador puede seleccionar un partido de todos los subidos a la plataforma y un usuario con privilegios por defecto podrá seleccionar uno de los partidos que el mismo ha subido. El botón de un color cercano a rojo, que indica “Añadir nuevo partido” simplemente se trata de un botón de navegación a la página para añadir un nuevo partido.

Para la carga de datos de nuevos partidos un usuario debe cargar todos los archivos nece-



Figura 7.4: Selección de nuevo partido o navegación a añadir nuevo partido.

sarios junto al vídeo en formato mp4 grabado con una cámara táctica de ese partido. Además debe añadir un identificador único a cada partido. Como se puede ver en la Figura 7.5 el usuario se encuentra con un formulario de 6 acciones. La primera de ellas se trata de elegir el identificador del partido. Las cuatro siguientes tratan de cargar los archivos de datos. El usuario debe pulsar en seleccionar archivo y elegir los archivos correspondientes a cada opción. Una vez hecho esto debe pulsar en “save file” y la carga de datos comienza. La última opción tiene una interfaz similar a las anteriores, pero en este caso se trata del archivo de vídeo del partido. Estas tareas pueden tardar varios minutos. Para ir a la opción siguiente o anterior el usuario puede clicar encima de ellas o pulsar los botones de anterior y siguiente.

La Figura 7.6 muestra una pantalla muy similar a las anteriores con dos cambios fundamentales. La barra de navegación dispone ahora de una serie de opciones adicionales. Se tratan de nuevo de simples botones de navegación a las diferentes funcionalidades. Ahora ya se muestran porque en este momento hay un partido ya seleccionado. Si en cualquier momento se deselecciona el partido estas opciones volverán a desaparecer. La primera de ellas indica “Inicio” y navega a la pantalla de la Figura 7.7. La opción de “Seleccionar partido” navega a esta misma pantalla de la que se está hablando. Las siguientes tres opciones navegan a las tres funcionalidades principales de la plataforma, las correspondientes a la Figura 7.8, la Figura 7.10 y la Figura 7.14, respectivamente. Ya para acabar se puede visualizar el menú de selección desplegado en el que simplemente se seleccionan los partidos. Si se quiere deseleccionar un partido simplemente se hace pulsando en la opción vacía. La ya comentada barra de navegación nos acompañará con este aspecto durante el resto de las pantallas y por lo tanto no se harán más comentarios respecto a esta a lo largo de lo que resta de sección.

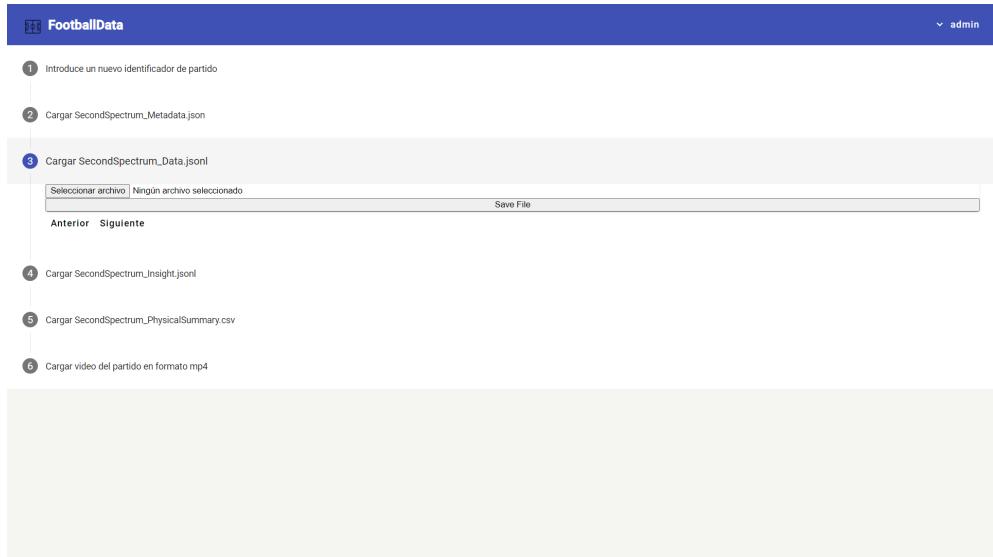


Figura 7.5: Carga de archivos de un nuevo partido.

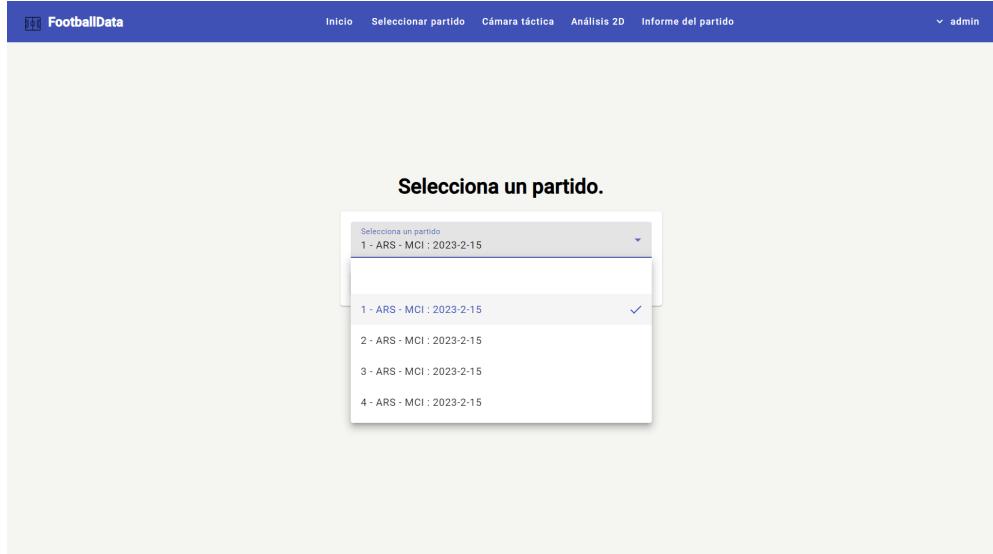


Figura 7.6: Selección de partido.

Una vez seleccionado un partido accedemos a la página principal de la plataforma para usuarios con sesión iniciada, como se puede ver en la Figura 7.7. En primer lugar se ve el nombre del partido seleccionado en la esquina superior izquierda, debajo de la barra de navegación. En la esquina superior derecha, otra de las novedades que se ven es que el botón indicado a través del nombre de usuario ha sido clicado y que por lo tanto se muestra la única opción disponible hasta ahora, “Cerrar sesión”. Al pulsar sobre este botón se terminará la sesión del usuario y por lo tanto habrá que iniciar sesión de nuevo antes de acceder a las funcionalidades de la plataforma. Los tres paneles que se encuentran superpuestos a la imagen de un campo de fútbol muestran las tres principales funcionalidades de la aplicación. El primero de ellos nos dirige a la Figura 7.8, el del medio a la Figura 7.10 y el que está más a la derecha a la Figura 7.14.



Figura 7.7: Página inicial de la plataforma para usuarios con sesión iniciada.

7.3 Análisis con cámara táctica.

En esta sección se comentan las características de una de las funcionalidades principales de la aplicación, el análisis y extracción de clips de una cámara táctica a partir de diversos filtros.

En la Figura 7.8 se puede observar un pequeño formulario y un panel de vídeo. El panel de vídeo muestra la grabación del partido a través de una cámara táctica. Además el panel dispone de una serie de controles. Se puede poner el vídeo en pantalla completa, subir y bajar el volumen, y también parar y reanudar la visualización. En el formulario se pueden observar varias opciones y dos botones. Las dos opciones superiores corresponden a la selección de equipo y tipo de jugada para las cuales se quieren extraer clips. Al clicar el botón que se sitúa

en la zona inferior a estos dos datos de entrada se obtendrán los clips según estos dos filtros y se navega a la Figura 7.9. El tercero de los datos de entrada se corresponde al filtro por jugador. Una vez seleccionado un jugador se habilita el botón que se encuentra justo debajo del panel de selección. Al pulsar este botón se filtrarán los clips también por jugador y de igual manera se navegará a una pantalla como la de la Figura 7.9.

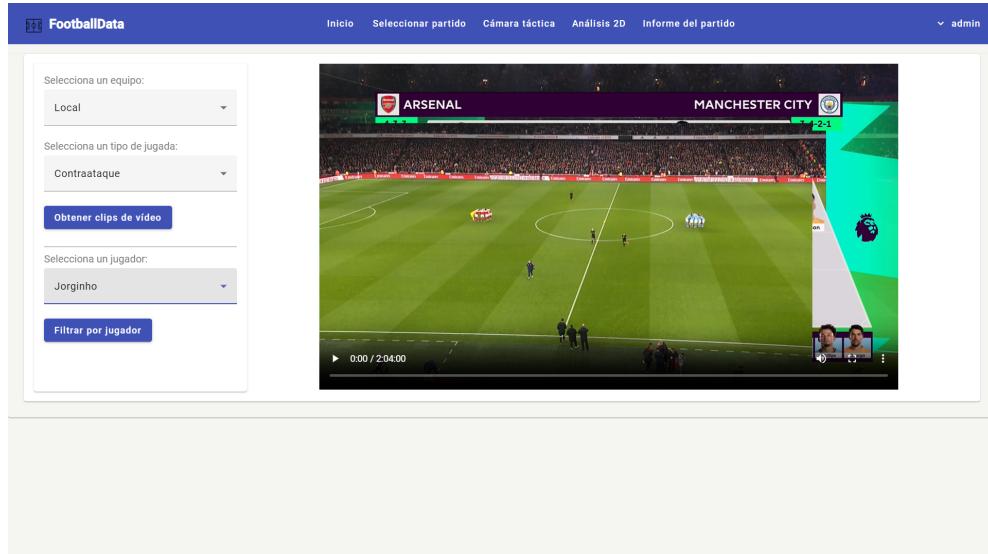


Figura 7.8: Visualización completa de un partido.

Una vez nos encontramos en la Figura 7.9 el formulario no desaparece. Se podrán seguir realizando peticiones y modificando los filtros y se cargarán en la misma página en la que se encuentra el usuario.

7.4 Análisis posicional.

Esta sección corresponde a las características de la segunda de las funcionalidades de la plataforma. En este caso se trata del análisis posicional en un terreno de dos dimensiones.

En la Figura 7.10 se puede ver la página principal de esta funcionalidad y donde se realizan la gran mayoría de acciones. Empezando por el panel de mayor tamaño se puede ver un dibujado un terreno que representa un campo de fútbol en dos dimensiones. Con los dos botones que se encuentran en la parte superior izquierda del panel se puede aumentar el zoom o disminuir el zoom sobre el campo. La posición inicial del zoom es tal cual se muestra en la imagen. En este terreno se muestran tanto los iconos de los jugadores, como del balón, como de los eventos ocurridos durante el partido.

Ya fijándonos en el resto de elementos de la pantalla, en la parte izquierda vemos una serie de elementos. Los números de la parte superior representan el tiempo de juego actual.

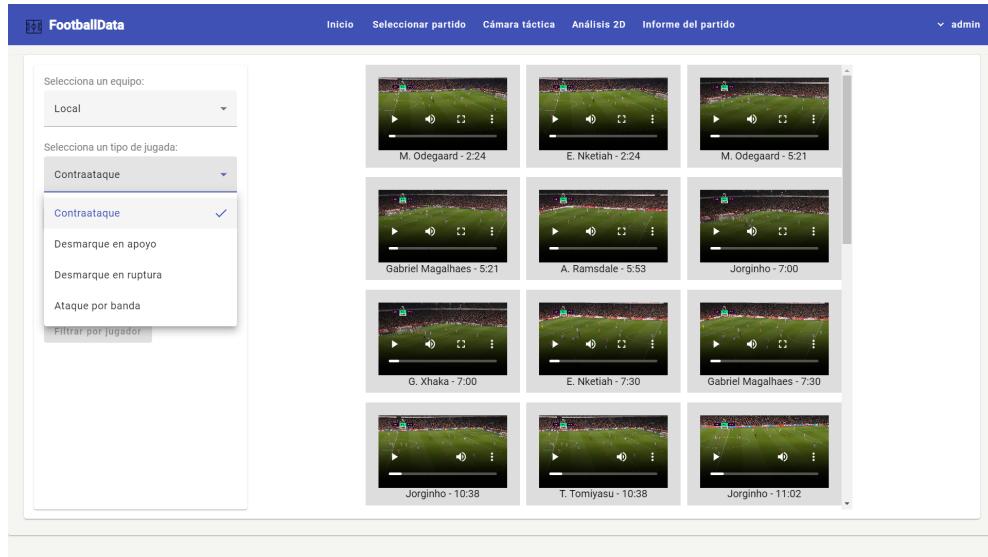


Figura 7.9: Clips extraídos de un partido.

El primero de los botones azules muestra la posición de los jugadores en el minuto cero y segundo cero. El segundo de los botones azules inicia la visualización del partido en el periodo que corresponde a los minutos marcados por el usuario. Esos minutos se marcan en los dos valores de entrada que tienen como nombre minuto inicial y minuto final. Si el minuto final no es superior al minuto inicial la plataforma muestra un error al iniciar la visualización. También se dispone un botón para parar la visualización, las dos rayas verticales, y un botón para reanudar la visualización, la flecha azul apuntando hacia la derecha. Para borrar todos los iconos del terreno de juego se debe pulsar el botón rojo que indica “Limpiar datos”.

En la parte de la pantalla más a la derecha vemos un pequeño formulario con cuatro valores de selección. Empezando por abajo tenemos uno de ellos llamado corner. Aquí se muestran una opción por cada corner del partido indicando el minuto en el que ocurre. Al seleccionar una de las opciones empieza la visualización del partido desde diez segundos antes de que ocurra este evento. Lo mismo de manera idéntica ocurre para el valor de selección de goles. Ya por último, se puede seleccionar tanto un jugador del equipo local como del equipo visitante. La selección de un jugador de uno de los equipos deseleccionará el del otro equipo, si es que había alguno seleccionado. Al elegir una de las opciones la plataforma mostrará algo parecido a la Figura 7.11.

En la Figura 7.11 se puede ver que ha sido seleccionado un jugador del equipo local. Tras la selección, se borran todos los iconos si es que los hay y se dibuja un mapa de calor de ese jugador en el terreno de juego. Además, se abre un nuevo panel en la esquina inferior izquierda. En él se pueden ver algunos datos del jugador y un botón de navegación a la página de la Figura 7.12.



Figura 7.10: Visualización posicional de un partido.



Figura 7.11: Mapa de calor de jugador seleccionado.

La Figura 7.12 corresponde a las estadísticas del jugador seleccionado. Se muestra el nombre, el dorsal, la posición, los minutos totales jugados, la distancia total recorrida, la distancia recorrida a alta velocidad, el número de sprints a alta velocidad, la velocidad máxima alcanzada y la velocidad media durante un partido. Además, en la parte derecha se observa un gráfico que divide la distancia recorrida en cuatro tipos de carrera.

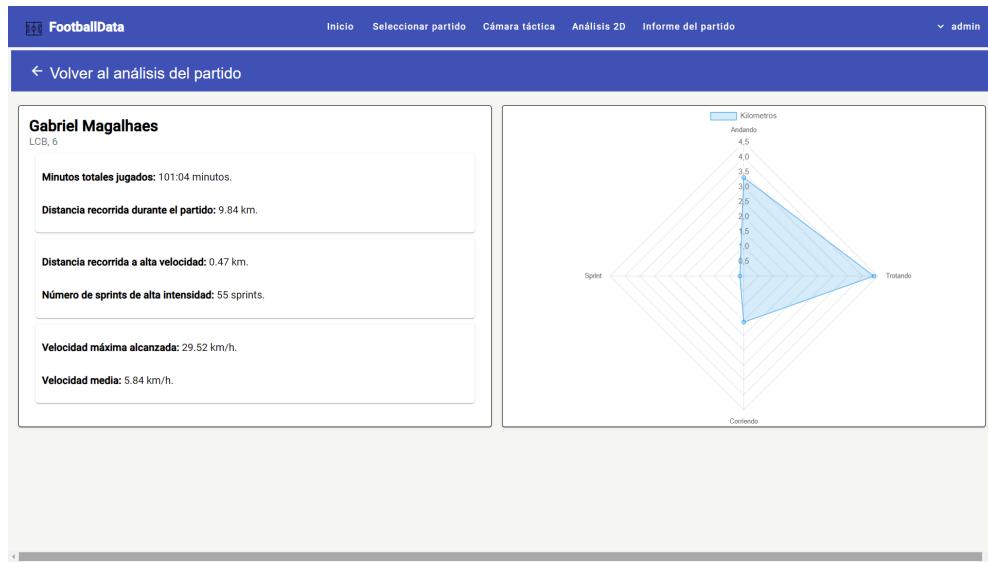


Figura 7.12: Estadísticas de un jugador.

7.5 Informe de partido.

Por último, la Figura 7.13 y la Figura 7.14 muestran un informe de un partido. En la esquina superior derecha se dispone de un panel de selección. Ahí podemos seleccionar entre tres formatos diferentes y una vez seleccionado uno se habilitará el botón de exportar que descargará el informe en el formato seleccionado en nuestro equipo local.

Por otro lado, se pueden ver diferentes datos y estadísticas que ofrece el informe sobre el partido, que serán simplemente de visualización. A partir de aquí ya no hay ningún aspecto modificable ni con el que el usuario pueda interaccionar. Se puede ver la fecha, la hora, los equipos, los jugadores ordenados por dorsal, la posesión, una serie de gráficas, etc.

Resumen del Partido

Datos del Partido:

Partido: ARS - MCI
Fecha del partido: 2023-2-15
Hora de inicio: 20:30
Resultado final: 1-3

Jugadores Equipo Local			Jugadores Equipo Visitante		
Dorsal	Nombre	Posición	Dorsal	Nombre	Posición
1	A. Ramsdale	GK	2	K. Walker	RCB
4	B. White	SUB	3	Ruben Dias	CB
6	Gabriel Magalhaes	LCB	4	K. Phillips	SUB
7	B. Saka	RW	6	N. Ake	LCB
8	M. Odegaard	RCM	8	I. Gundogan	LAM
11	Gabriel Martinelli	LW	9	E. Haaland	ST
12	W. Saliba	RCB	10	J. Grealish	LW
14	E. Nketiah	CF	16	Rodri	RDM
18	T. Tomiyasu	RB	17	K. De Bruyne	RAM
19	L. Trossard	SUB	20	Bernardo Silva	LDM
20	Jorginho	CDM	25	M. Akanji	SUB
21	Fabio Vieira	SUB	26	R. Mahrez	RW
34	G. Xhaka	LCM	31	Ederson	GK
35	O. Zinchenko	LB	47	P. Foden	SUB

Figura 7.13: Primera parte del informe de un partido.

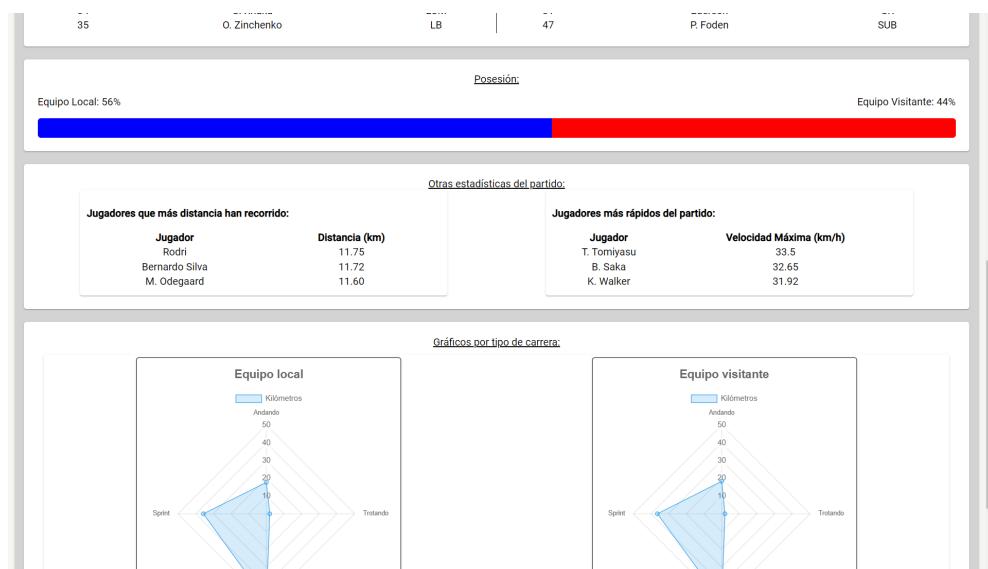


Figura 7.14: Segunda parte del informe de un partido.

Capítulo 8

Conclusiones y trabajo futuro

En esta sección se describe el grado de cumplimiento de los objetivos planificados al inicio del proyecto, algunas de las competencias que han sido necesarias para el proyecto y adquiridas por el alumno y las posibles líneas de ampliación que tiene el proyecto.

8.1 Conclusiones en relación a los objetivos planeados al inicio del proyecto

Los objetivos que fueron planificados, de forma genérica, en la reunión inicial, y, de forma más detallada, durante la etapa de análisis han sido plenamente cumplidos en tiempo y forma. Se han solventado los problemas que han ido apareciendo a lo largo del desarrollo con plenas garantías, y a pesar de que sí ha habido retrasos durante las etapas de desarrollo del código fuente de la plataforma, finalmente se ha llegado al resultado objetivo en la fecha límite acordada.

Se ha construido una plataforma de análisis de partidos de fútbol dirigida a equipos de fútbol base o equipos con un no muy alto presupuesto, que por sus recursos tanto económicos como humanos, solo requieren de las características más fundamentales del análisis. La plataforma cuenta con tres funcionalidades principales, que a su vez coinciden con los aspectos que los equipos más necesitan. La primera de ellas es la realización del análisis posicional de sus jugadores y de los jugadores de equipos rivales, ya sea para preparar partidos, corregir errores o incluso realizar seguimientos personalizados a determinados jugadores del propio equipo o posibles fichajes. Por otro lado la plataforma dispone almacenadas las grabaciones de los partidos, previamente subidas por los usuarios, con cámaras tácticas. Los analistas pueden seleccionar una serie de filtros por jugada, equipo y jugador para extraer una serie de clips del partido que haya seleccionado. Por último, destaca la visualización de un informe a modo de resumen del partido con los datos y estadísticas más importante del partido que haya sido seleccionado, que también podrá ser exportado a diferentes formatos.

Además, ya es una realidad desde hace años que el análisis táctico y estadístico es crucial en el fútbol moderno. Los equipos que lo realizan pueden identificar de manera más rápida fortalezas y debilidades, ajustar estrategias antes y durante el partido y mejora el rendimiento de los equipos. Todos los equipos que han tenido éxito durante los últimos años realizan un exhaustivo análisis táctico y estadístico de todos sus partidos y de los de los rivales, tienen equipos de analistas muy grandes y cada año que pasa esto simplemente parece ir a más.

8.2 Competencias adquiridas

Para la realización del proyecto se han tenido que utilizar y aplicar un gran número de herramientas, tecnologías, lenguajes de programación, librerías, conceptos y metodologías. En gran parte, muchas de las tareas han sido resueltas con ayuda de conocimientos adquiridos durante las asignaturas impartidas en el grado, pero muchas otras veces ha sido necesario aprender conceptos y tecnologías nuevas.

Dos aspectos fundamentales aprendidos durante la carrera y que se han utilizado en el proyecto fueron el modelado de bases de datos y el desarrollo de aplicaciones web, con especial atención al desarrollo de una [API REST](#). También la metodología que se ha seguido, la implementación de pruebas, etc.

Por otro lado, también se van a destacar una serie de habilidades que han sido adquiridas a medida que se iba desarrollando el proyecto y, por lo tanto, también utilizadas en el mismo.

- **TypeScript.** Lenguaje de programación que se utiliza en el desarrollo con Angular y que se ha utilizado para la implementación del frontend.
- **Angular.** Angular es un framework de desarrollo web que se ha utilizado para desarrollar también el frontend de la plataforma.
- **Angular Material.** Es una biblioteca de componentes para la interfaz de usuario de Angular. Ha sido utilizado en conjunto con Angular.
- **Leaflet.** Es una biblioteca utilizada para crear mapas interactivos en aplicaciones web. Ha sido utilizada para la creación del mapa para el terreno de juego en dos dimensiones donde se dibujan los iconos de los jugadores y del balón.
- **PostGIS.** Extensión de PostgreSQL para soportar la utilización de datos geoespaciales. Se ha tenido que aprender a manejar el formato de los datos geoespaciales de tipo geométrico. Para el caso específico de la aplicación simplemente se han usado datos de tipo “Point”. Ha sido necesario también serializar y deserializar estos datos en el backend a la hora de guardarlos y obtenerlos de la base de datos.

- **ChartJS.** Biblioteca utilizada en el frontend para la creación de gráficos para las estadísticas de los jugadores y equipos.
- **Mockito.** Biblioteca de Java que se utiliza para crear objetos simulados (mocks) en pruebas unitarias que simulan el comportamiento de objetos reales, permitiendo probar componentes de software de forma aislada.

8.3 Objetivos a futuro

Los objetivos planificados inicialmente para el proyecto han sido plenamente completados. Aún con esto, el proyecto tiene varias líneas sobre las que se podrían orientar las futuras ampliaciones. A continuación, se especifican algunas de ellas.

- **Panel de control para los administradores.** La única función extra actualmente que tienen los administradores es que pueden seleccionar partidos que hayan sido subidos por cualquier usuario. Se podría crear un panel de administración en donde los administrados pudieran gestionar tanto usuarios como partidos. Un administrador debería poder añadir, modificar y eliminar usuarios, e incluso visualizar una lista de todos los usuarios con botones para realizar las acciones citadas con rapidez. Lo mismo sería también de gran ayuda para los partidos. Así un administrador podría eliminar tanto los videos y datos de los partidos que ya no sean necesarios, y la plataforma se ahorraría mucho espacio en base de datos que ya no es necesario tener ocupado.
- **Sincronización de vídeo y análisis posicional.** Hasta ahora la plataforma solamente permite o visualizar el partido con la cámara táctica o en el terreno de juego en dos dimensiones. El próximo objetivo prioritario debería de ser permitir a los usuarios visualizar, de forma sincronizada, ambos paneles a la vez. A pesar de que esto se pueda llegar a ver muy sencillo al estar ya ambas funcionalidades implementadas por separado, afrontar la sincronización del vídeo y de los iconos en el terreno de dos dimensiones no es una tarea fácil y normalmente aparecerán dificultades.
- **Internacionalización de la plataforma.** Actualmente la plataforma soporta un único lenguaje, el castellano. Lo ideal, para que la plataforma pudiera ser usada globalmente por gente que hable otros idiomas, o al menos los más populares en el mundo, sería que los usuarios pudieran elegir el idioma de la aplicación. Inicialmente el idioma por defecto de la aplicación sería el del `locale` asociado a ese usuario.
- **Análisis de más tipos de jugadas.** La plataforma hasta ahora solamente permite filtrar por cuatro tipos de jugadas. Estas jugadas son contraataque, ataque por banda, desmarque en apoyo y desmarque en ruptura. En especial destacaría la importancia de

analizar jugadas a balón parado como saques de esquina, faltas cerca del área, saques de banda e incluso penaltis.

Apéndices

Apéndice A

Manual de instalación

A.1 Configuración general

1. Asegúrate de tener instalados los siguientes software:

- Java 19
- Node
- Angular 17
- PostgreSQL junto a su extensión PostGIS

2. Clona el repositorio del proyecto:

-> *git clone https://gitlab.lbd.org.es/tfgs/miguel.rodriguez6.tfg/tfg.git*

A.2 Configuración de la Base de Datos

1. Crea una base de datos con el nombre tfg_test.

-> *CREATE DATABASE tfg_test;*

2. Ejecuta el script ConfigureDB.sql en postgres.

-> *\i backend/src/sql/ConfigureDB.sql*

A.3 Ejecución del Backend

1. Una vez clonado el repositorio, cambia el directorio actual a .../backend:

-> *cd tfg/backend*

2. Una vez completados los anteriores pasos, inicia el backend de la aplicación:

-> *mvn spring-boot:run*

A.4 Ejecución del Frontend

1. Una vez clonado el repositorio, cambia de directorio actual a .../frontend e instala las dependencias necesarias:

-> *cd tfg/frontend*

-> *npm install*

2. Una vez completados los anteriores pasos, inicia el frontend de la aplicación:

-> *ng serve*

Apéndice B

Prototipos de la aplicación

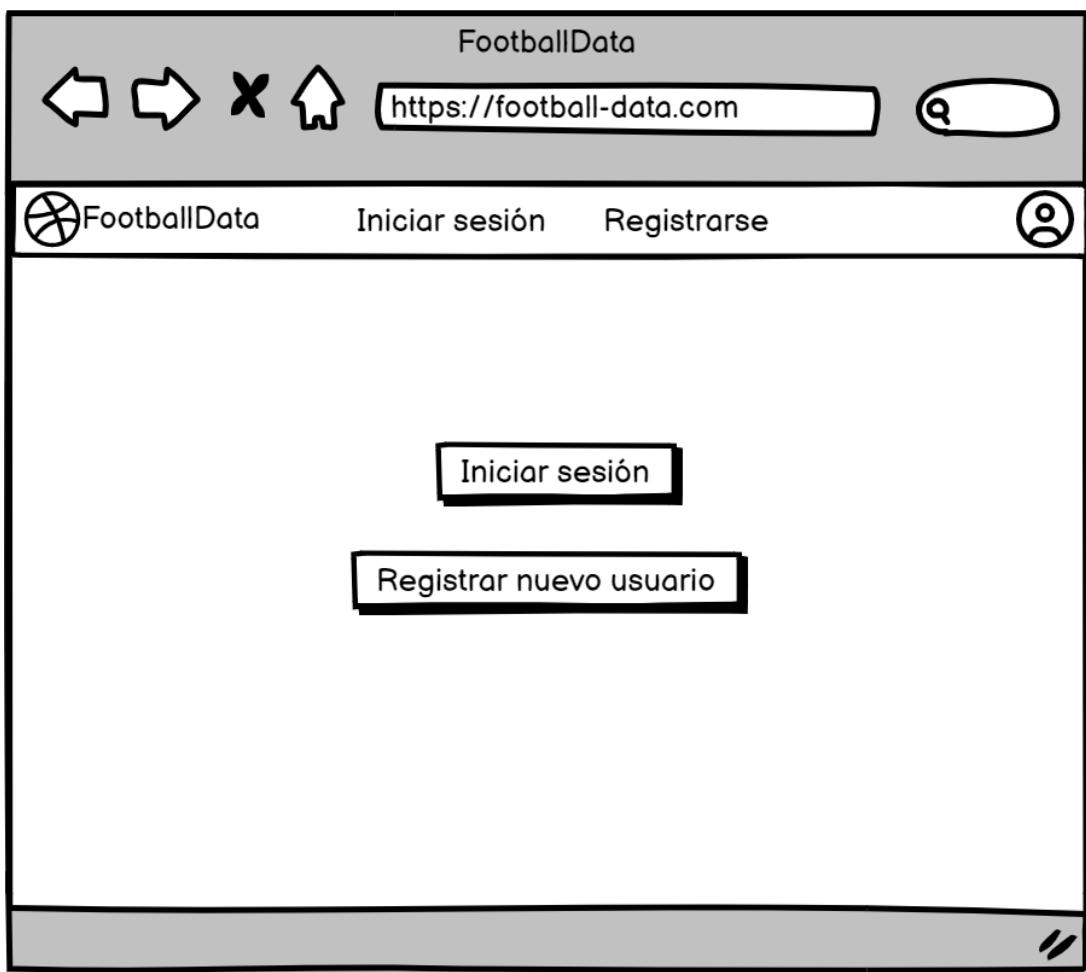


Figura B.1: Pantalla inicial al entrar a la plataforma.

APÉNDICE B. PROTOTIPOS DE LA APLICACIÓN

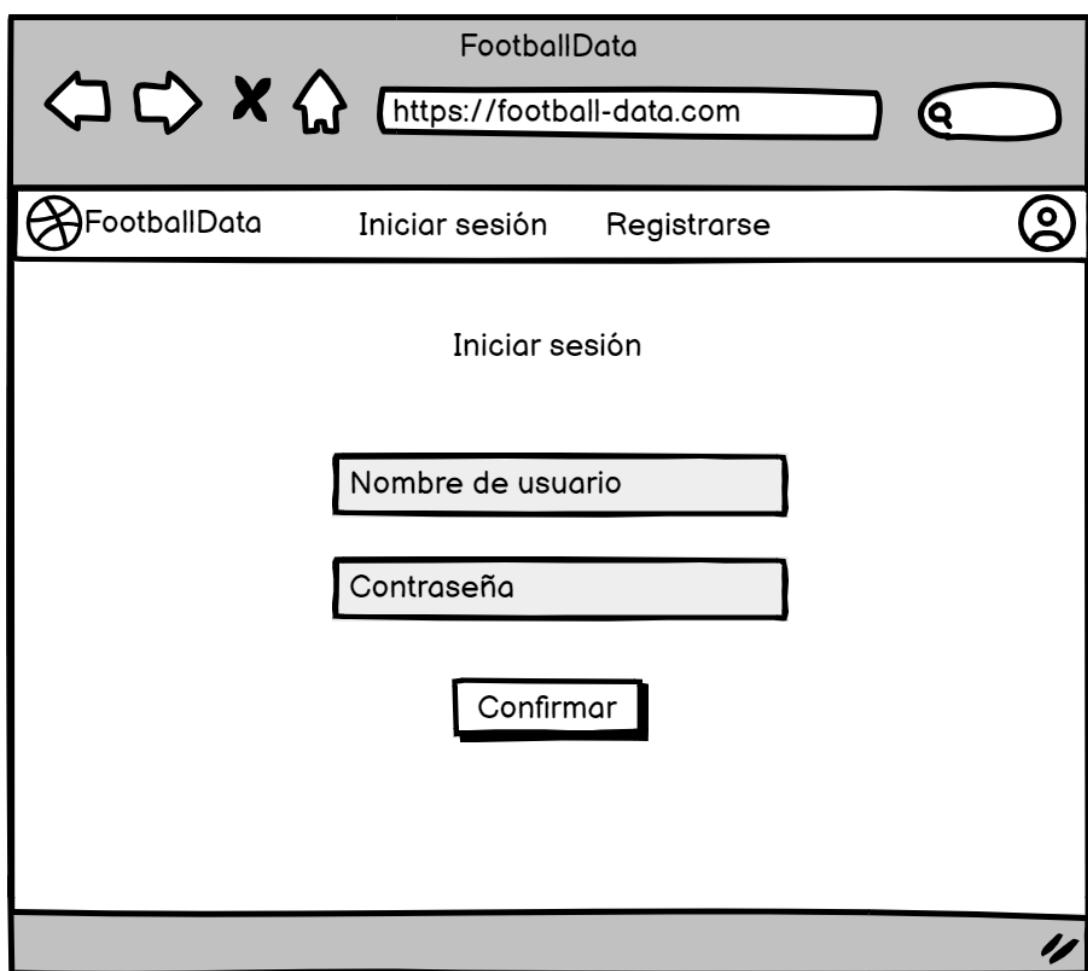


Figura B.2: Pantalla de inicio de sesión.

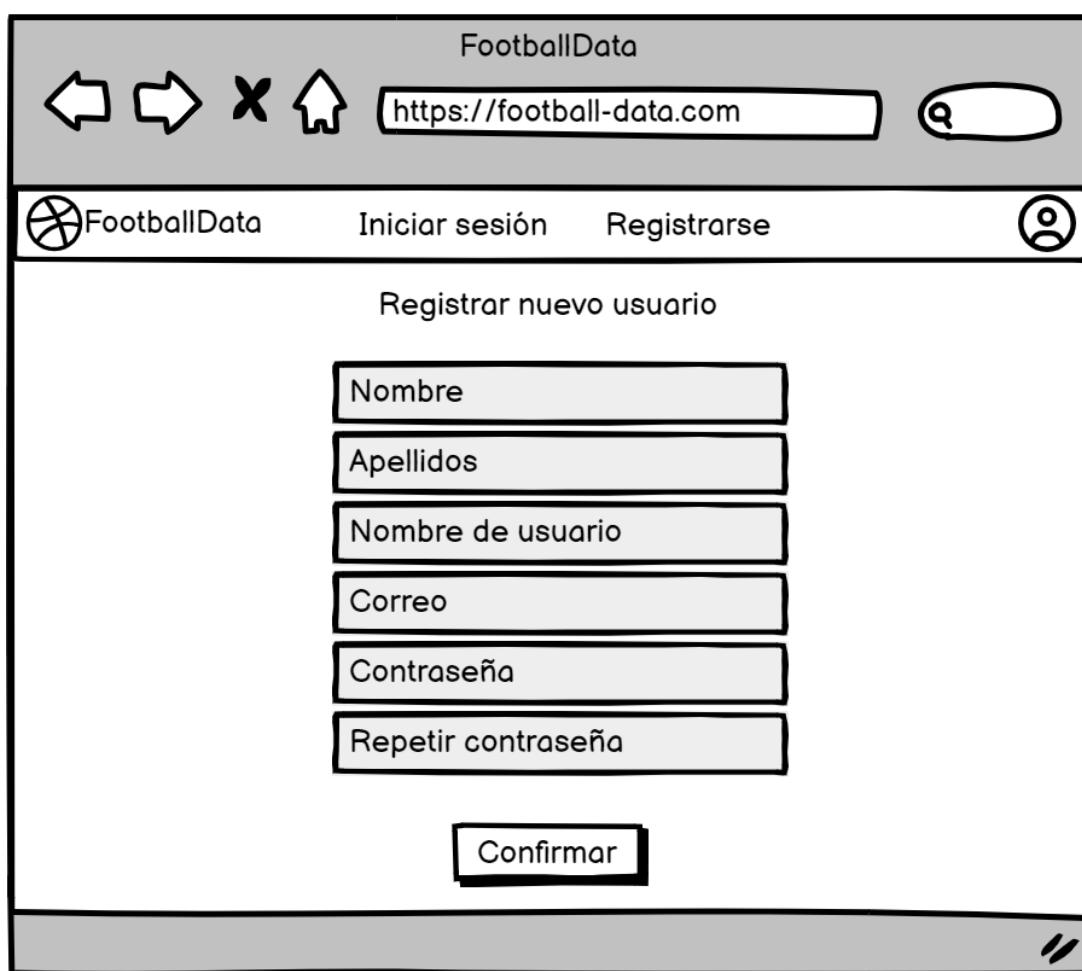


Figura B.3: Pantalla de registro.

APÉNDICE B. PROTOTIPOS DE LA APLICACIÓN

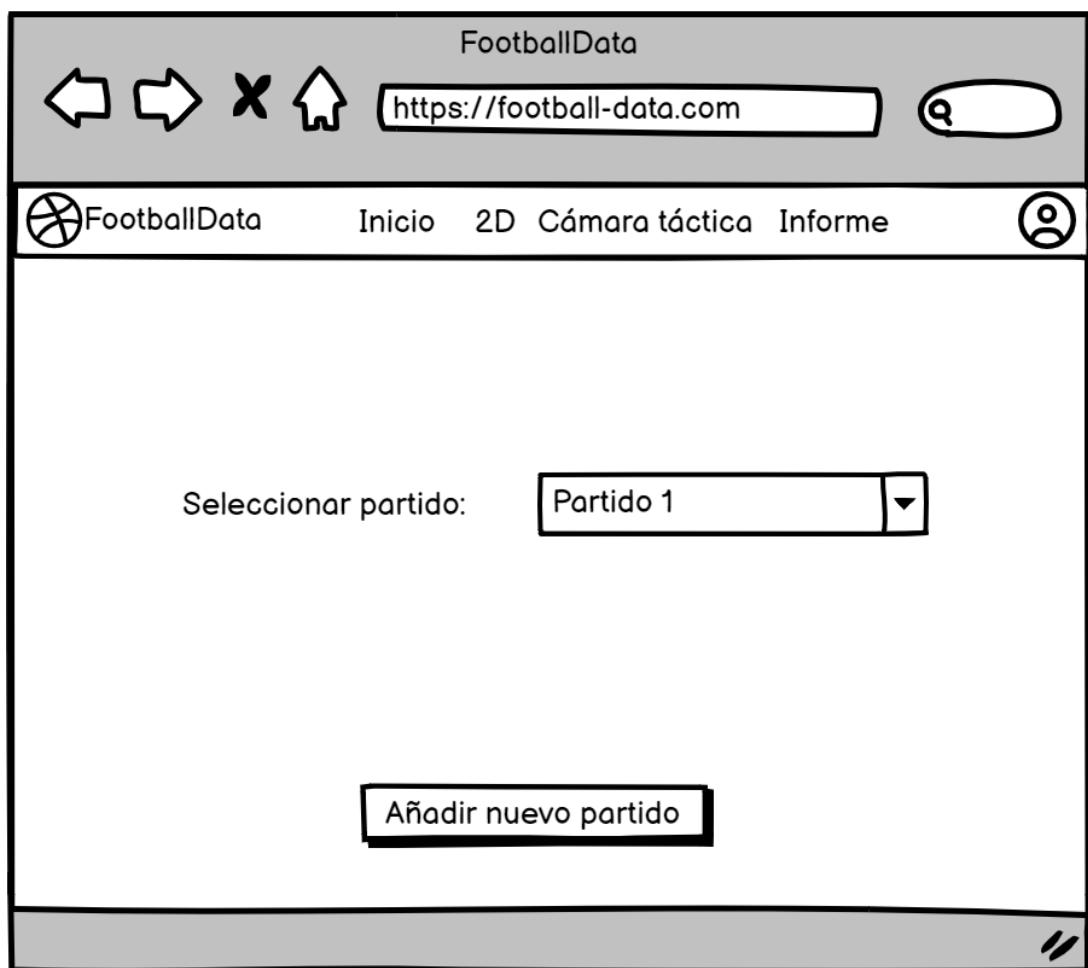


Figura B.4: Pantalla de selección de partido.

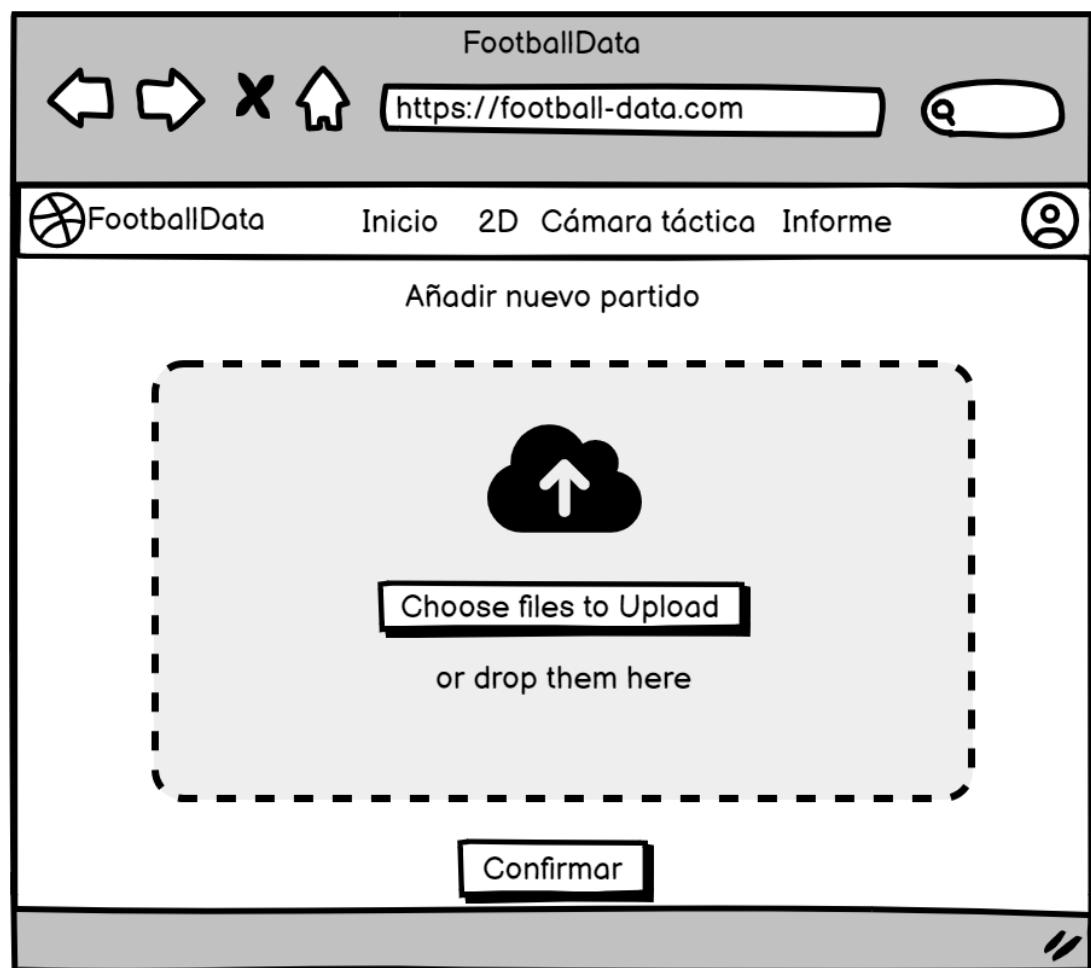


Figura B.5: Pantalla de carga de nuevos partidos.

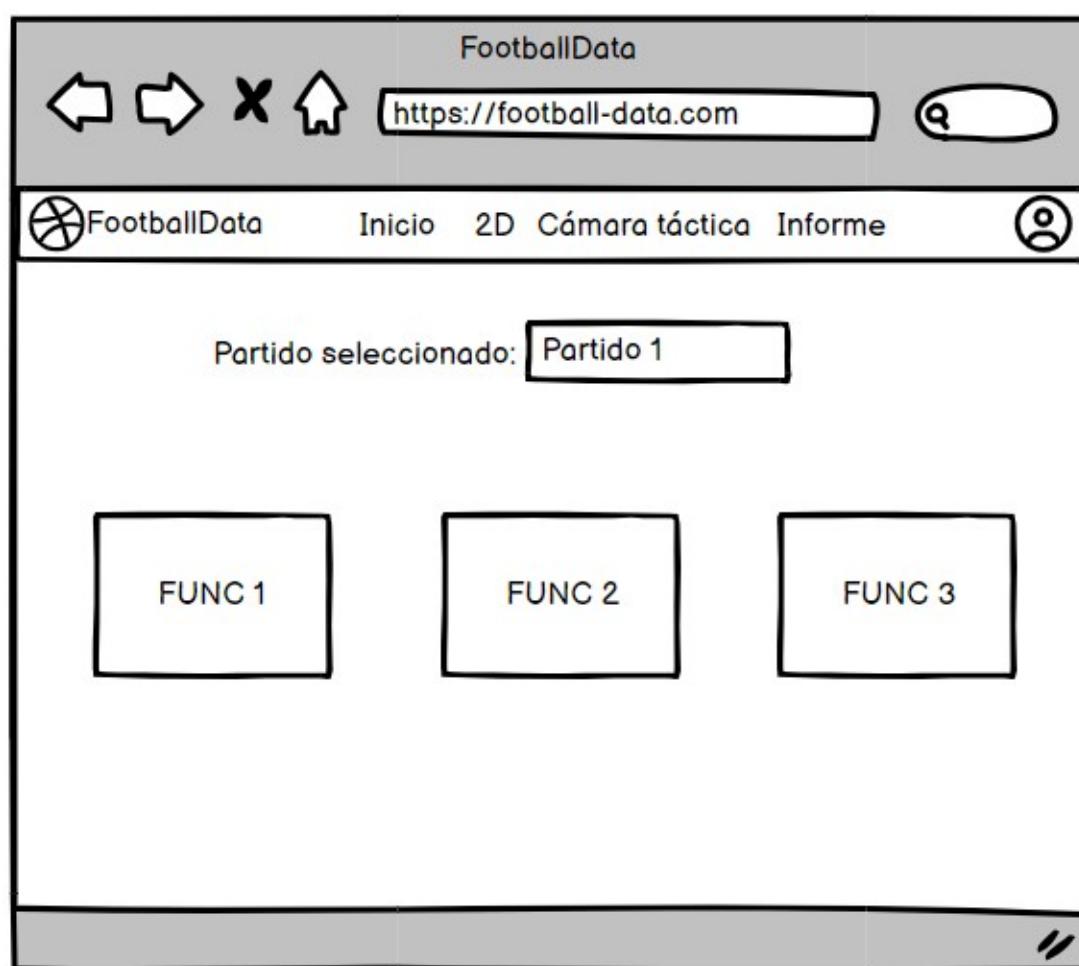


Figura B.6: Pantalla de selección de funcionalidades principales.

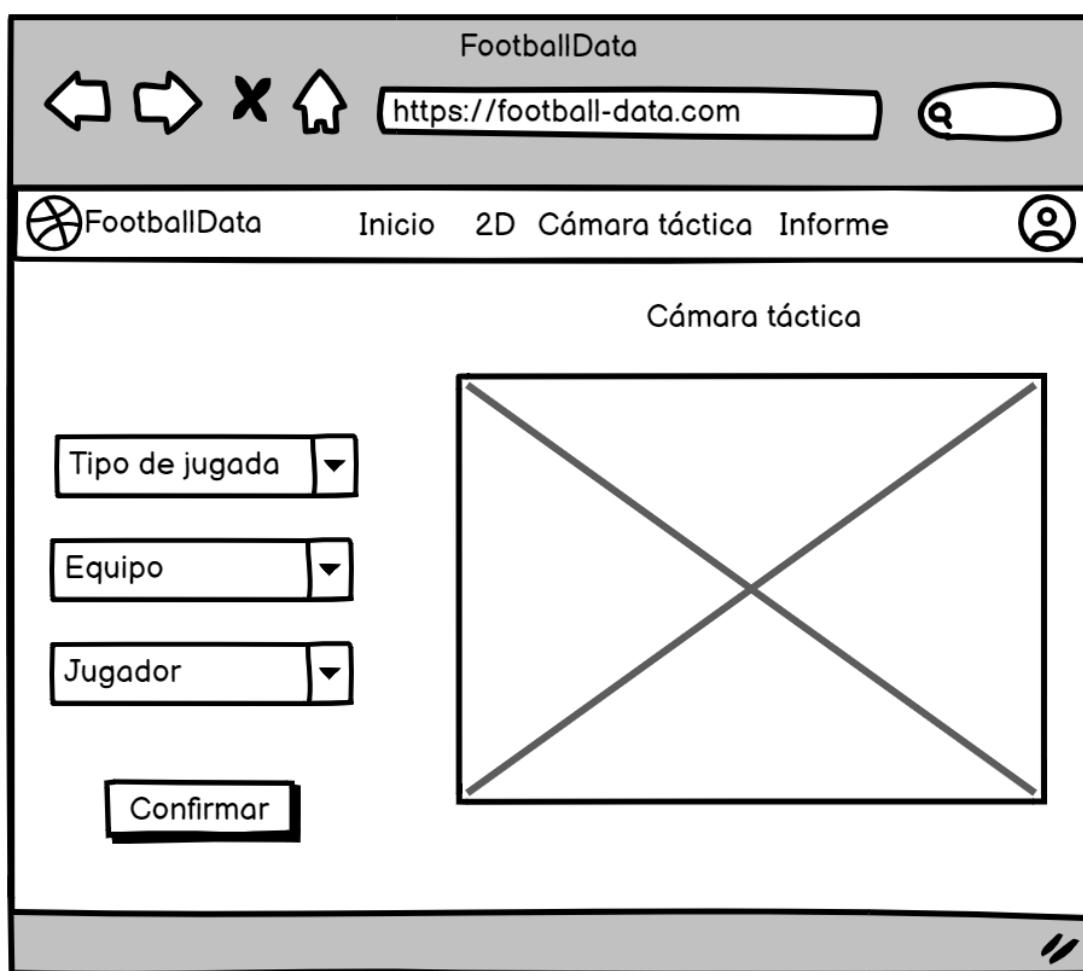


Figura B.7: Pantalla de visualización de cámara táctica y selección de filtros.

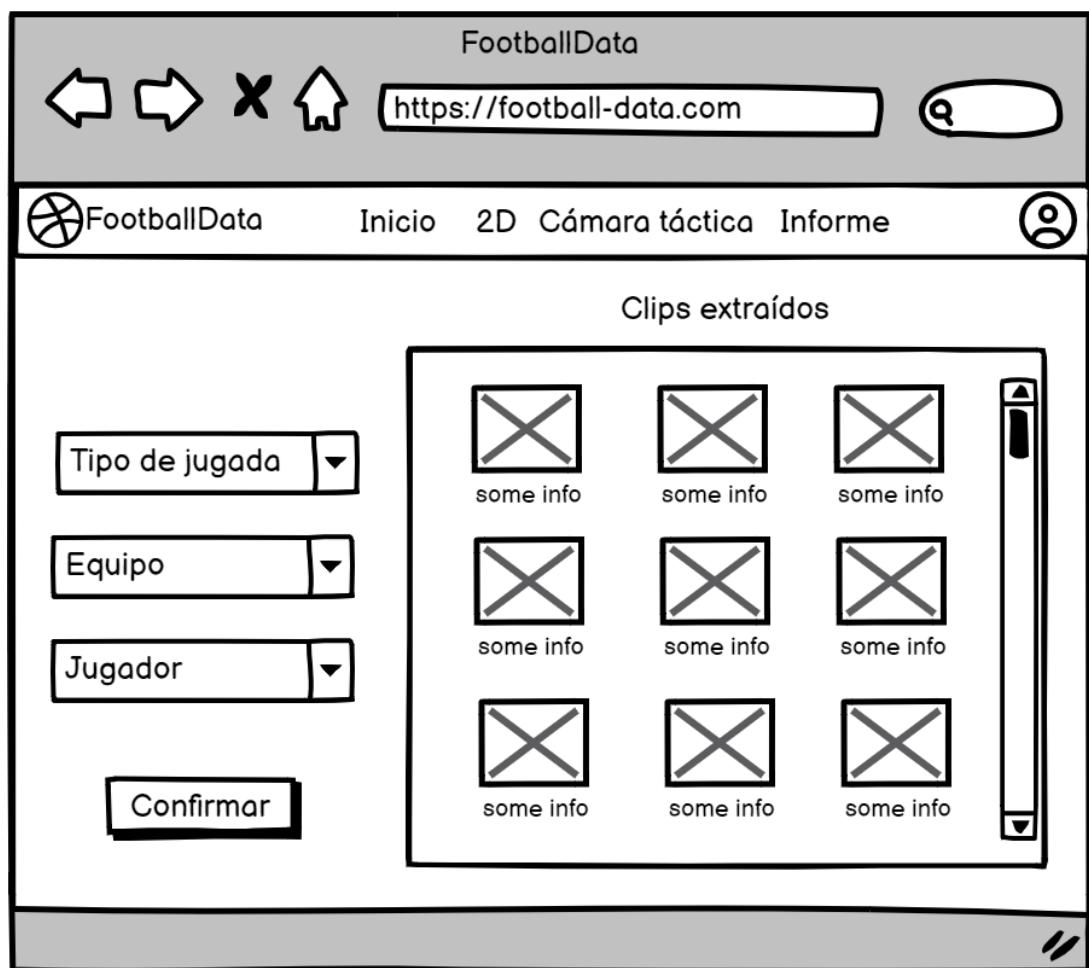


Figura B.8: Pantalla de visualización de clips de vídeo extraídos.

APÉNDICE B. PROTOTIPOS DE LA APLICACIÓN

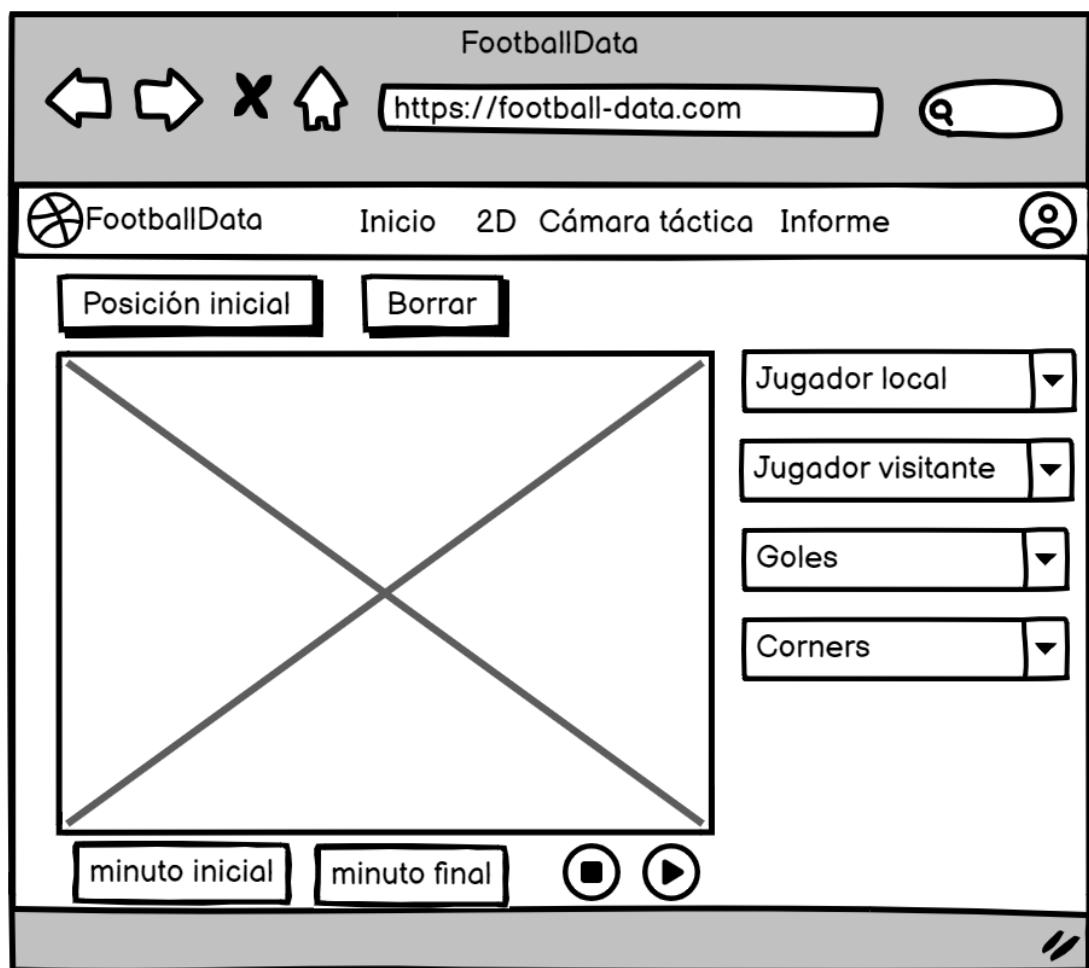


Figura B.9: Pantalla de análisis en dos dimensiones.

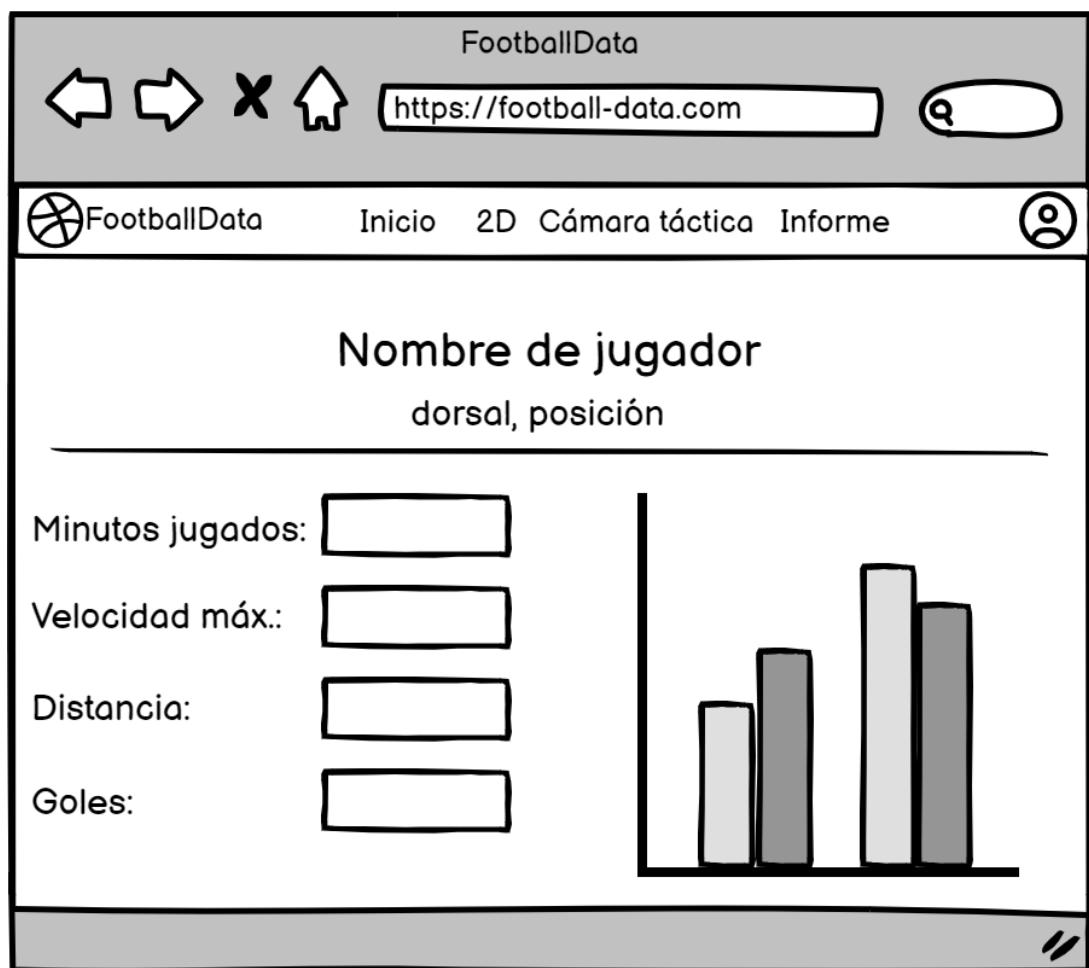


Figura B.10: Pantalla de estadísticas de un jugador.

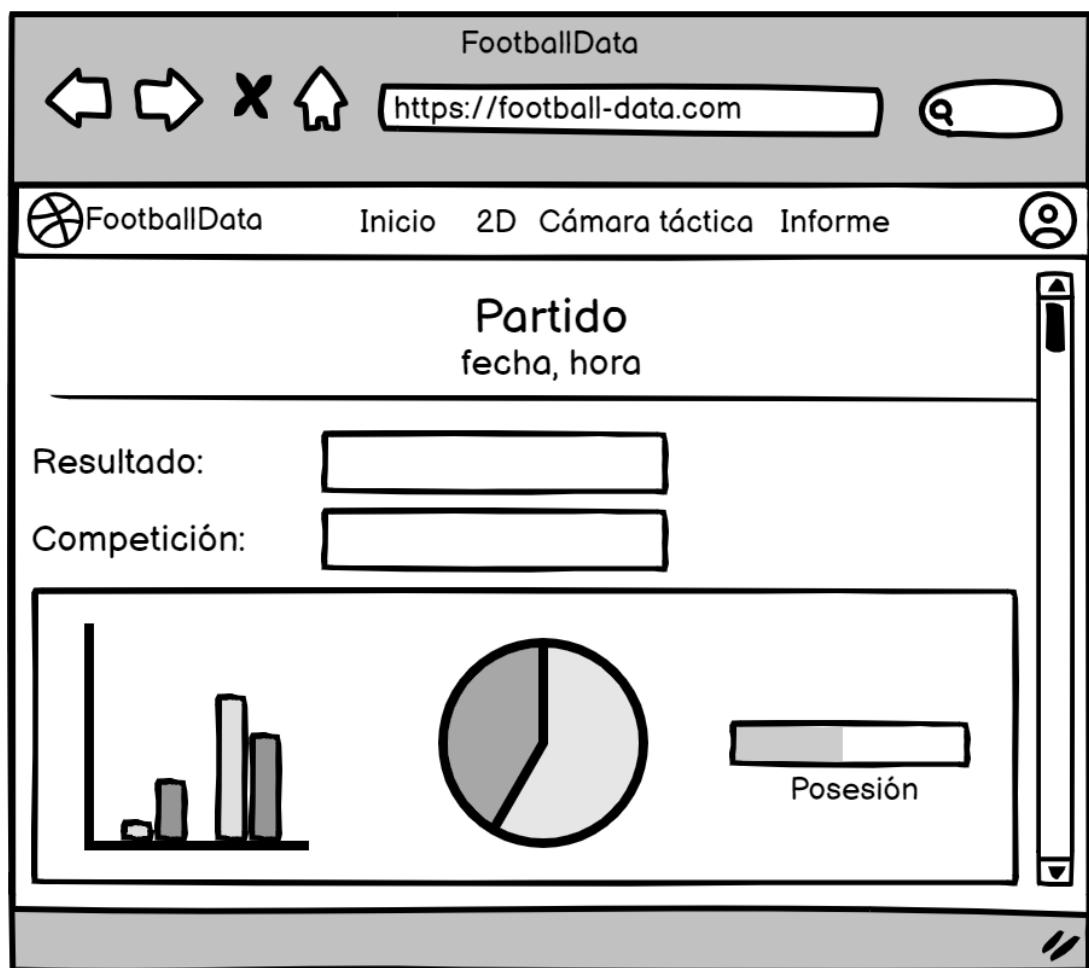


Figura B.11: Pantalla de visualización del informe.

Lista de acrónimos

API Application Programming Interface. [34](#), [38](#), [44](#), [48](#), [61](#)

CSS Cascading Style Sheets. [13](#)

CSV Comma-Separated Values. [7](#)

DTO Data Transfer Objects. [41](#)

HTML HyperText Markup Language. [13](#)

HTTP Hypertext Transfer Protocol. [39](#), [41](#)

REST Representational State Transfer. [61](#)

SPA Single Page Application. [38](#)

URL Uniform Resource Locators. [48](#)

Glosario

backend Se trata de la parte de una aplicación que se encarga de la lógica del servidor, la base de datos y la gestión de las solicitudes del cliente.. [39](#)

endpoint URL específica en una API que recibe y responde a solicitudes HTTP.. [34](#)

frame Imagen estática que forma parte de una secuencia de vídeo.. [44](#)

frontend Se trata de la parte visible de una aplicación web, con la que interactúan los usuarios.. [39](#)

locale Configuración que define las preferencias del usuario en cuanto a idioma, formato de fecha, hora, moneda, y más.. [62](#)

script Conjunto de instrucciones o código que se ejecuta automáticamente para realizar tareas específicas dentro de un programa o sistema.. [39](#)

Bibliografía

- [1] “Página web del Real Club Deportivo de La Coruña.” [Online]. Available: <https://www.rcdeportivo.es/>
- [2] “Página web de Opta.” [Online]. Available: https://optaplayerstats.statsperform.com/en_GB/soccer
- [3] “Página web de Mediacoach.” [En línea]. Disponible en: <https://www.mediacoach.es/>
- [4] “Página web de PostgreSQL.” [En línea]. Disponible en: <https://www.postgresql.org/>
- [5] R. Obe, *PostGIS in action*. Stamford, CT: Manning Publications Co., 2011.
- [6] Vladimir Agafonkin, “Página web de Leaflet,” 2019. [En línea]. Disponible en: <https://leafletjs.com/>
- [7] “Página web de TypeScript.” [En línea]. Disponible en: <https://www.typescriptlang.org/>
- [8] “Página web de NPM.” [En línea]. Disponible en: <https://www.npmjs.com/>
- [9] “Página web de Angular.” [En línea]. Disponible en: <https://angular.dev/>
- [10] “Página web de Angular Material.” [En línea]. Disponible en: <https://material.angular.io/>
- [11] “introducción a CryptoJS.” [En línea]. Disponible en: <https://dev.to/beresiartejuan/cifrado-y-descifrado-con-cryptojs-3h5g>
- [12] “Página web de ChartJS.” [En línea]. Disponible en: <https://www.chartjs.org/>
- [13] “Página web de Rxjs.” [En línea]. Disponible en: <https://rxjs.dev/>
- [14] “Documentación de Jspdf.” [En línea]. Disponible en: <https://parallax.github.io/jspdf/docs/jspdf.html>
- [15] “Página web de JUnit.” [En línea]. Disponible en: <https://junit.org/junit5/>

- [16] “Página web de Mockito.” [En línea]. Disponible en: <https://site.mockito.org/>
- [17] “Página web de Spring.” [En línea]. Disponible en: <https://spring.io/>
- [18] “Página web de Maven.” [En línea]. Disponible en: <https://mvnrepository.com/>
- [19] “Introducción a Gson.” [En línea]. Disponible en: <https://howtodoinjava.com/gson/gson/>
- [20] “Introducción a OpenCSV.” [En línea]. Disponible en: <https://www.baeldung.com/opencsv>
- [21] “Introducción a Lombok.” [En línea]. Disponible en: <https://programandoenjava.com/lombok-en-java/>
- [22] “Metodología iterativa incremental.” [En línea]. Disponible en: <https://proyectosagiles.org/desarrollo-iterativo-incremental/>
- [23] “Introducción a Scrum.” [En línea]. Disponible en: <https://www.atlassian.com/es/agile/scrum>
- [24] “Página web de GanttProject.” [En línea]. Disponible en: <https://www.ganttproject.biz/>
- [25] “Página web de Jira.” [En línea]. Disponible en: <https://www.atlassian.com/software/jira>
- [26] “Página web de pgAdmin.” [En línea]. Disponible en: <https://www.pgadmin.org/>
- [27] “Página web de IntelliJ.” [En línea]. Disponible en: <https://www.jetbrains.com/es-es/idea/>
- [28] “Página web de Overleaf.” [En línea]. Disponible en: <https://es.overleaf.com/project>
- [29] “Página web de GitLab.” [En línea]. Disponible en: <https://about.gitlab.com/>
- [30] “Página web de Balsamiq.” [En línea]. Disponible en: <https://balsamiq.com/>
- [31] “Página web de Postman.” [En línea]. Disponible en: <https://www.postman.com/>
- [32] “Página web de Draw.io.” [En línea]. Disponible en: <https://app.diagrams.net/>
- [33] “Página web de Glassdoor.” [En línea]. Disponible en: https://www.glassdoor.es/Sueldos/a-coru%C3%B1a-programador-j%C3%A1bano-suelo-SRCH_IL.0,8_IC2626610_KO9,27.htm