CLUB RIVALS: APP ANDROID DE GAMIFICACIÓN PARA EQUIPOS DE FÚTBOL AMATEUR

CLUB RIVALS: ANDROID APP FOR GAMIFICATION OF AMATEUR FOOTBALL TEAMS

TRABAJO FIN DE GRADO
CURSO 2024-2025

AUTOR
GONZALO MARTÍNEZ CARRERA

DIRECTOR
PABLO RABANAL BASALO

GRADO EN INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

NOTA FINAL: 9.5

CLUB RIVALS: APP ANDROID DE GAMIFICACIÓN PARA EQUIPOS DE FÚTBOL AMATEUR

CLUB RIVALS: ANDROID APP FOR GAMIFICATION OF AMATEUR FOOTBALL TEAMS

TRABAJO DE FIN DE GRADO EN INGENIERÍA INFORMÁTICA

AUTOR
GONZALO MARTÍNEZ CARRERA

DIRECTOR
PABLO RABANAL BASALO

CONVOCATORIA: SEPTIEMBRE 2025

GRADO EN INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

9 DE SEPTIEMBRE

NOTA FINAL: 9.5

DEDICATORIA

A los profesores de FAL.

AGRADECIMIENTOS

A todos los profesores que me han ayudado durante el grado y a Pablo, que me ha guiado en el proceso para elaborar este trabajo.

RESUMEN

La aplicación que se presenta consiste en una plataforma móvil de modalidad Fantasy Football orientada a ligas menores y equipos federados no profesionales. Su principal objetivo es ofrecer a los jugadores una herramienta sencilla para visualizar estadísticas individuales y colectivas, gestionada por los administradores.

Esta aplicación permite a estos administradores crear temporadas y equipos, registrar partidos con todos los eventos relevantes y mantener actualizada la información en la base de datos. Los usuarios pueden consultar sus estadísticas personales mediante un perfil propio, además de acceder a clasificaciones generales, rankings de goles o asistencias y otras comparativas de rendimiento.

El sistema se apoya en el sistema de base de datos Google Firebase para la autenticación y el almacenamiento de datos, garantizando la sincronización entre dispositivos.

La interfaz se desarrolla en Android Studio, utiliza Java, XML y componentes modernos de esta herramienta para conseguir una navegación simple y fluida entre secciones. En conjunto, se busca ofrecer una experiencia atractiva y sobre todo funcional que combine la gestión deportiva y entretenimiento, adaptadas a las necesidades específicas de equipos locales y ligas regionales.

Palabras clave

Fantasy football, estadísticas, Firebase, Android Studio, temporadas, equipos, jugadores, perfil.

ABSTRACT

The application to be presented consists of a Fantasy Football mobile platform aimed at minor leagues and non-professional federated teams. Its main objective is to offer players a simple tool that allows them to visualize individual and collective statistics, managed by administrators.

This application allows these administrators to create seasons and teams, save record of matches with all their relevant events, and keep the information in the database up to date. Users can visualize their personal statistics through their own profile, as well as access general rankings, goal or assists standings, and other performance comparisons.

The system relies on Google Firebase database system for authentication and data storage, ensuring synchronization between devices.

The interface is developed in Android Studio, using Java, XML and modern components of this tool to achieve simple and smooth navigation between sections. Overall, the aim is to offer an attractive and, above all, functional experience that combines both sports management and entertainment, tailored to the specific needs of local teams and regional leagues.

Key Words

Fantasy football, statistics, Firebase, Android Studio, seasons, teams, players, profile.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Capítulo 1 - Introducción

En este primer apartado de la memoria se expone la motivación, los objetivos, la metodología y el plan de trabajo, además de la estructura de la propia memoria.

1.1 Motivación

Desde que de bastante pequeño descubrí el mundo de la tecnología y a una edad más avanzada el del desarrollo de aplicaciones, sabía que en un futuro querría dedicarme profesionalmente a ello. Esta motivación inicial fue la razón principal que me impulsó a iniciar los estudios de Ingeniería Informática, ya que de siempre me ha atraído la idea de poder crear herramientas tecnológicas para resolver problemas de la vida cotidiana.

Al mismo tiempo, el deporte ha estado muy presente en mi vida, y lo sigue estando, tanto como forma de ocio como medio para poder mantenerme activo y en forma.

La unión de ambos intereses, tecnología y deporte, que a día de hoy diría que son los dos aficiones que más tiempo ocupan en mi vida y con los que más disfruto, ha resultado ser compatible para poder realizar este trabajo, donde el objetivo es desarrollar una aplicación de gamificación del deporte, combinando perfectamente ambos temas.

La sociedad actual está en constante transformación digital. Cada vez más son los aspectos del día a día que se trasladan al ámbito tecnológico, buscando eficiencia y comodidad. Un ejemplo claro y cotidiano es la sustitución de las agendas físicas por calendarios digitales que, aparte de permitir anotar citas o eventos, también se pueden utilizar para programar recordatorios o integrarse con otros servicios. Otro caso más complejo y elaborado es el uso de aplicaciones de salud que monitorizan el sueño, la alimentación o la actividad física en tiempo real, generando informes automáticos del estado actual del usuario.

En este contexto de digitalización, durante los últimos años se han ido creando aplicaciones para todo tipo de situaciones cotidianas. Precisamente por esto, y

obviamente motivado por el propio tema que se eligió para el trabajo de fin de grado, surge la idea de crear una aplicación que permita a los usuarios llevar un seguimiento de las estadísticas de las competiciones de fútbol en las que es partícipe, similar al concepto de las ligas Fantasy[1] (importante ver referencia) de fútbol profesional. A diferencia de estas, donde los usuarios seleccionan jugadores de equipos internacionales en el ámbito profesional, esta aplicación está orientada a los propios equipos amateurs, permitiendo registrar estadísticas reales de sus partidos y generar clasificaciones internas basadas en goles, asistencias, minutos jugados, etc.

Esta propuesta busca fomentar la participación activa de los jugadores en su propio rendimiento, ya que visualizar una mala racha goleadora puede impulsar al usuario a trabajar para mejorar en ese aspecto del juego. Además, aumenta el compromiso con el equipo y ofrece una experiencia lúdica más allá del terreno de juego.

Este trabajo representa no solo el cierre de una etapa académica, sino también el inicio de un camino en el que la tecnología y el deporte seguirán estando presentes en mi desarrollo profesional y personal. Por tanto, es también una pequeña muestra de lo que me gustaría seguir haciendo en un futuro: crear soluciones tecnológicas que tengan un impacto real combinando mis intereses personales con lo aprendido hasta el día de hoy.

1.2 Objetivos

El objetivo de este trabajo es el desarrollo de una aplicación para el entorno Android, que permita a los usuarios recoger y visualizar estadísticas sobre el desempeño de un equipo de fútbol en una competición.

A lo largo del desarrollo de este Trabajo de Fin de Grado, se han establecido una serie de objetivos claros que han guiado tanto la planificación como la implementación del proyecto.

Para poder comprender los objetivos marcados para el desarrollo, es recomendable conocer cuál va a ser la propuesta exactamente y qué es lo que va a permitir la aplicación.

Se quiere construir una aplicación que recoja estadísticas reales. Deberá habilitarse un usuario administrador (el entrenador, por ejemplo) que registre los datos de los partidos en un apartado de la aplicación limitado únicamente a estos administradores, y los usuarios reales (jugadores) podrán visualizar las estadísticas que se han ido introduciendo. Esta información deberá ser persistente, por lo que será necesario incluir una base de datos que aloje todos estos valores. Además, se pretende presentar una interfaz clara, simple y accesible.

Para ello, se han definido los siguientes objetivos:

En primer lugar, uno de los propósitos fundamentales ha sido construir el sistema principal de la aplicación, que debía presentar diferentes pantallas que mostrarían datos acerca de las competiciones, sentando así las bases sobre las que se construyen todas las funcionalidades posteriores. Este sistema debía permitir una navegación fluida, intuitiva y organizada entre las distintas pantallas, garantizando que el usuario pudiera acceder fácilmente a cada una de las secciones.

En segundo lugar, ha sido imprescindible construir una base de datos robusta, flexible y bien estructurada, que permitiera almacenar toda la información relevante del sistema, como competiciones, equipos, usuarios, partidos y sus estadísticas detalladas.

El tercer bloque consiste en desarrollar los diferentes componentes que conforman la aplicación, como la creación y gestión de equipos y jugadores, la introducción de los datos de los partidos y la visualización de estadísticas, siempre pensando en el uso práctico que los jugadores aficionados, harán de la herramienta.

Con todos estos objetivos bien definidos, el proyecto podrá ir tomando forma hasta llegar a la versión que queremos presentar, preparada para utilizarse y, además, para poder seguir creciendo.

1.3 Metodología

Para poder abordar correctamente el desarrollo de la aplicación, se ha optado por una metodología incremental y modular a la vez, que permite ir construyendo el proyecto poco a poco, probando cada parte a medida que se implementa, y asegurando en todo momento que el sistema funciona de forma estable. Esta forma de

trabajar resulta especialmente útil porque, al tratarse de una aplicación con varias funcionalidades y pantallas, el dividir las tareas por módulos ayuda a centrar el foco en lo importante en cada fase y evitar bloqueos por no tener todas las piezas listas desde el principio.

De hecho, lo que más ha ayudado y lo que luego además menos problemas se espera que genere, es que desde un principio existe una estructura clara de cómo va a construirse la aplicación, con una base de navegación definida desde el inicio a la que poco a poco se van a ir añadiendo las distintas actividades y pantallas en cada una de las fases.

Por tanto, es necesario detallar cómo se estructura el trabajo, en cuatro fases claramente diferenciadas:

1.3.1 Prototipado

En primer lugar, se dedicará tiempo al diseño de las pantallas utilizando Figma[2]. Esto permitirá visualizar cómo va a estar estructurada la aplicación antes de empezar a programarla. Se diseñarán las pantallas principales, así como las secundarias de estas principales, con una barra inferior de navegación fija y un diseño limpio, que tenga sentido tanto para el administrador como para el jugador. Esta fase es clave para tener una idea clara de los requisitos funcionales de la navegación, los elementos visibles en cada sección y la jerarquía visual.

1.3.2 Diseño de base de datos

Esta fase es una de las partes más importantes, ya que, para este tipo de aplicación, donde la mayoría del valor recae en la recopilación de datos e información, crear una buena estructura de base de datos es indispensable desde un principio para tener claro cómo van a ir integrándose los distintos objetos. Se definen bien las entidades necesarias y sus relaciones, teniendo en cuenta cómo va a utilizarlas tanto el administrador (que introduce los datos) como los jugadores (que sólo consultan). Esta parte es fundamental porque permite también organizar de manera lógica cómo se va a acceder a la información en cada pantalla más adelante, ya que las clases encapsuladas deben tener información de sus superiores, etc.

1.3.3 Construcción de la aplicación base

Con toda la base de datos bien planificada, se quiere construir inicialmente una aplicación base en Android Studio[3], es decir, una estructura mínima que ya incluya la navegación básica entre pantallas, el patrón de diseño general y una primera integración con los datos en memoria. Es importante, e incluso imprescindible, tener esta base sobre la que ir añadiendo las diferentes funcionalidades que se van a ir integrando poco a poco. A partir de ahí, se desarrollará cada módulo de forma independiente, lo que permite trabajar de manera ordenada y sin depender del resto de componentes.

1.3.4 Desarrollo de módulos

Una vez se tenga la base, se incluirán módulos de manera incremental. Esto permite centrar el foco cada vez en una parte, y no pasar a la siguiente hasta que la anterior no esté cerrada. Además, al tener la aplicación bien estructurada, el desarrollo de unas no influye en el funcionamiento de otras, como ya se ha comentado anteriormente.

El primer módulo a desarrollar será el de las estadísticas, ya que es el núcleo de la aplicación. Centraremos el desarrollo en que los datos de los partidos, los goles, asistencias, minutos, etc., se puedan visualizar correctamente y se actualicen de forma coherente. Es verdad que, en un principio, al no tener el módulo administrador construido, no se podrán ir modificando los valores, pero hacerlo al revés sería todavía más lioso porque al introducir los datos en el administrador no se podrán comprobar los datos que se incluyen. El siguiente módulo que se construirá será el del modo administrador, que permite crear temporadas, equipos, jugadores y partidos, y registrar todos los eventos de estos. Como se ha mencionado justo antes, al tener la parte de visualización de estadísticas ya implementada, se podrá ir probando lo que se construya en este módulo de administrador.

Posteriormente, se desarrollarán las pantallas “General” o “Home” y la del “Perfil”, ya que para entonces se tendrán todos los datos necesarios a mostrar y sólo será necesario crear las vistas.

1.4 Plan de trabajo

Una parte fundamental del proyecto es organizar correctamente el tiempo disponible para abordar todas las fases sin dejar tareas importantes para el final. Desde el principio es importante tener en cuenta que, aunque la aplicación no es demasiado compleja, sí tiene muchas partes diferentes y dependientes entre sí, por lo que es imprescindible establecer una planificación por bloques de trabajo, siempre dejando cierto margen para reajustes o por si no se ha calculado correctamente la duración de cada uno.

La planificación se divide en cuatro grandes fases: prototipado, diseño de base de datos, construcción de la aplicación base y desarrollo por módulos. Finalmente se realizará también una fase adicional de pruebas, revisión y documentación. Estas fases se han pensado para poder avanzar de forma progresiva, primero asentando las bases (estructura, diseño y modelo de datos) y después construyendo cada funcionalidad concreta sobre ese esquema.

Las primeras semanas se analizarán otras aplicaciones de ese tipo, tanto de ligas Fantasy como de estadísticas deportivas, y así poder definir los requisitos del proyecto. Cuando se tenga esta parte bien definida, entonces se podrá diseñar el prototipo de Figma, algo que se hará en paralelo con el diseño de la base de datos, ya que muchas decisiones visuales dependerán de cómo se va a estructurar la información internamente, y viceversa. Con esta parte clara, se empezará a construir la aplicación base en Android Studio, añadiendo progresivamente los distintos módulos funcionales. Se empezará por la lógica de estadísticas (que es la parte central de la aplicación) y luego se integrarán el resto de pantallas.

Cada módulo tiene su propia planificación: primero se implementará la interfaz, luego los datos, después la lógica y finalmente se harán pruebas para asegurar que queda todo bien integrado. Esta es la fase para la que, previsiblemente, se dedicará más tiempo.

Aunque a medida que se añadan funcionalidades se probará todo lo nuevo, sí que será necesario, en las últimas semanas, dejar un periodo de tiempo para realizar las

pruebas finales, corrección de errores y preparación de la memoria escrita, además de algunos ajustes visuales y mejoras en la navegación para pulir la experiencia de usuario.

Aunque no habrá entregas parciales obligatorias como tal, sí que se mantendrá contacto con el tutor a lo largo del desarrollo, aparte de tener varias sesiones en las que se irán revisando el trabajo hecho hasta el momento.

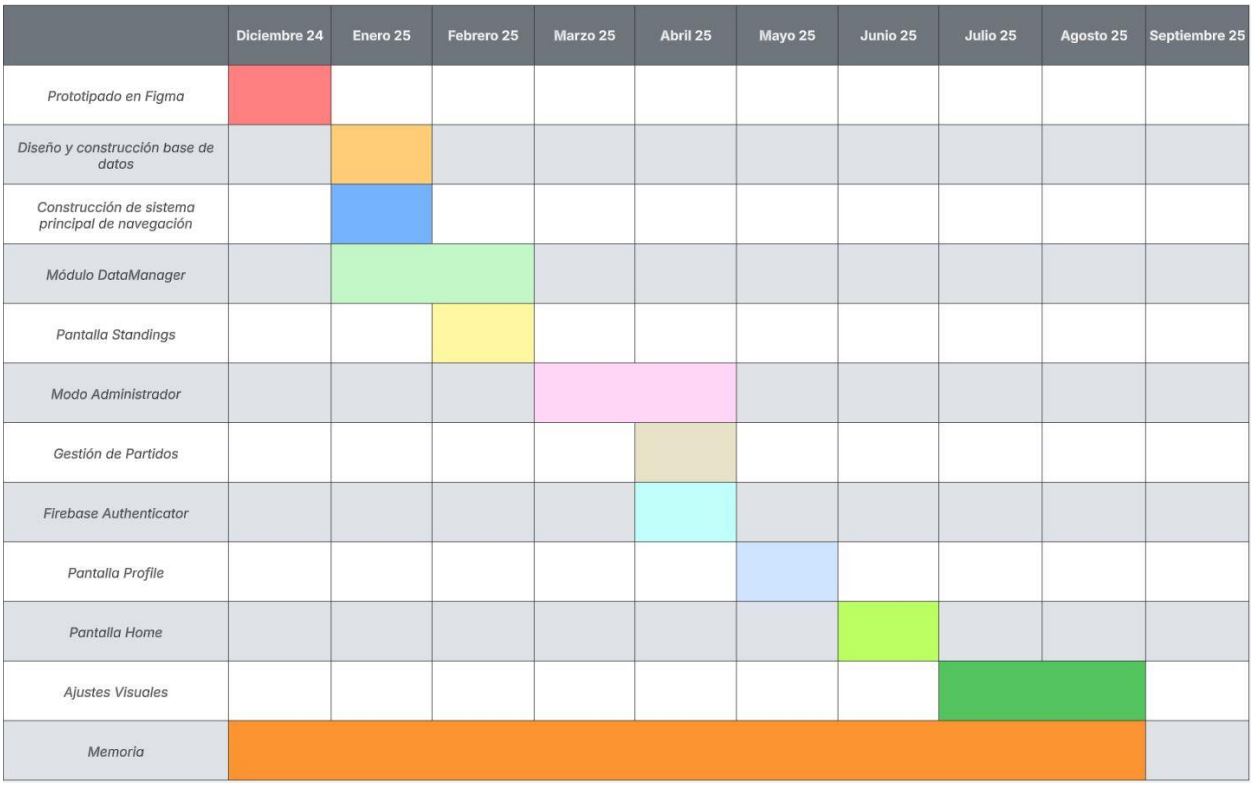


Tabla 1 : Diagrama de Gantt

1.5 Estructura de la memoria

Este documento se ha organizado en varios capítulos que siguen el mismo orden que se ha llevado a cabo durante el desarrollo del proyecto. La idea es que quien lo lea pueda entender el proceso completo, desde la motivación inicial hasta el resultado final, pasando por todas las fases intermedias como el diseño, la implementación y las pruebas.

En el capítulo 1, se presenta una introducción general del proyecto. Aquí se expone la motivación personal, el contexto, los objetivos principales y la metodología de trabajo, además de una planificación temporal del desarrollo.

El capítulo 2 recoge un análisis de las soluciones existentes en el mercado. Se comparan otras aplicaciones relacionadas con el seguimiento de estadísticas deportivas y se identifican sus limitaciones, lo cual sirve para justificar la necesidad y el enfoque de la aplicación desarrollada.

En el capítulo 3, se describe todo el diseño conceptual del sistema. Se definen los tipos de usuarios que interactúan con la aplicación, los requisitos funcionales y no funcionales, los casos de uso y la arquitectura general. También se explica cómo se estructura la navegación entre pantallas.

El capítulo 4 está centrado en el prototipado. Se describe el uso de Figma como herramienta de diseño visual y se presentan las pantallas principales de la aplicación, explicando su función y diseño.

El capítulo 5 recoge todo el proceso de desarrollo. Se explican las herramientas utilizadas, el diseño de la base de datos, y cómo se ha implementado cada módulo de forma progresiva, incluyendo tanto la perspectiva del administrador como la del jugador.

El capítulo 6 muestra el funcionamiento real de la aplicación. Se incluyen capturas de flujo y ejemplos reales.

En el capítulo 7 se abordan las pruebas realizadas. Se analiza cómo se han validado las funcionalidades y se describen los problemas encontrados durante el desarrollo junto con las soluciones aplicadas.

El capítulo 8 incluye las conclusiones finales, un repaso a los objetivos cumplidos y una propuesta de mejoras y funcionalidades futuras que podrían implementarse más adelante.

Finalmente, el documento se completa con un glosario de términos técnicos y la bibliografía utilizada, además de algunos apéndices.

Capítulo 1 - Introduction

In this first section of the report, the motivation, objectives, methodology and work plan are presented, along with the structure of the document itself.

1.1 Motivation

Since I was quite young, I discovered the world of technology, and later, at a more advanced age, the world of application development, so I always knew that in the future I would want to steer my professional life to it. This initial motivation was the main reason that led me to start a degree in Computer Engineering, as I have always been attracted to the idea of creating technological tools to solve everyday life problems.

At the same time, sports have always played a major role in my life, and it still is, both as a hobby and as a way to stay active and in shape.

The combination of both interests, technology and sports, which to this day I would say are the hobbies that occupy most of my free time and the ones I enjoy the most, has turned out to be the perfect match for carrying out this project, and it just made a lot of sense. The goal is to develop a sports gamification application that seamlessly combines both areas.

Today´s society is undergoing constant digital transformation. More and more aspects of daily life are quickly shifting into the technological sphere, seeking efficiency and convenience. A clear and common example is the replacement of physical agendas with digital calendars which, apart from allowing the scheduling of appointments or events, can also be used to program reminders or integrate with other services. A more complex case is the use of health applications that monitor sleep, diet or physical activity in real time, automatically generating reports on the user´s current condition.

Within this context of digitalization, in recent years applications have been created for all types of everyday situations. For this reason, and obviously motivated by the theme chosen for this Final Degree Project, the idea arose to create an application that allows users to track statistics from the football competitions in which they

participate, similar to the concept of professional Fantasy[1] (see reference). Unlike those, where users select players from international professional team, this application is aimed at amateur clubs, allowing them to record real statistics from their matches and generate internal rankings based on goals, assists, minutes played, and so on.

This project represents not only the conclusion of an academic stage but also the beginning of a path where technology and sports will continue to be present in my professional and personal growth. Therefore, it also serves as a small example of what I would like to continue doing in the future: creating technological solutions that have a real impact by combining my personal interests with what I have learned so far.

1.2 Objectives

The main goal of this project is the development of an Android application that allows users to collect and view statistics on the performance of a football team in a competition.

Throughout the development of this Final Degree Project, a series of objectives are established that guide both the planning and the implementation of the project. To understand these objectives, it is necessary to clarify the exact proposal and the scope of the application.

The application is designed to record real statistics. An administrator user (such as a coach) is entitled to register match data within a section of the app restricted exclusively to administrators, while regular users (players) are able to view the statistics that have been entered. This information must remain persistent, which requires the inclusion of a database to store all values. Furthermore, the interface is designed to be clear, simple, and accessible.

The specific objectives are as follows:

First, to build the main system of the application, which must include different screens displaying competition data, laying the foundations for the rest of the functionalities. This system must allow smooth, intuitive, and organized navigation between the different screens, ensuring that users can easily access each section.

Second, to design a robust, flexible, and well-structured database capable of storing all relevant information, including competitions, teams, users, matches, and detailed statistics.

Third, to develop the different components of the application, such as the creation and management of teams and players, the recording of match data, and the visualization of statistics, always considering the practical use that amateur players make of the tool.

With these objectives defined, the project gradually takes shape until reaching a version ready for use and capable of future expansion.

1.3 Methodology

The development of the application follows an incremental and modular methodology. This approach allows the project to be built step by step, testing each part as it is implemented, and ensuring system stability at all times. This proves particularly useful because, as the application includes multiple functionalities and screens, dividing the work into modules helps to maintain focus on each phase and avoid bottlenecks caused by incomplete components.

A clear structure of the application is defined from the beginning, with a navigation base to which the different activities and screens are gradually added.

The work is structured into four clearly differentiated phases:

1.3.1 Prototyping

The first step consists of designing screens using Figma. This tool makes it possible to visualize the structure of the application before programming begins. The main screens are designed along with their secondary screens, with a fixed bottom navigation bar and a clean design suitable for both administrators and players. This phase is key to defining navigation requirements, visible elements in each section, and visual hierarchy.

1.3.2 Database design

This phase is one of the most important, since in this type of application most of the value lies in data collection. The necessary entities and their relationships are defined, taking into account both the role of administrators (who input data) and players (who only consult it). This part is also essential for organizing how information will later be accessed from each screen.

1.3.3 Building the base application

Once the database is planned, a base application is created in Android Studio. This minimal structure already includes basic navigation between screens, the overall design pattern, and an initial integration with in-memory data. This base proves crucial as the foundation on which new functionalities are gradually added.

1.3.4 Module development

Once the base is ready, modules are incrementally included, focusing on one part at a time and not moving forward until the previous one is complete. The first module developed is the statistics module, as it forms the core of the application. The priority is to ensure that match data (goals, assists, minutes, etc.) are displayed correctly and updated consistently. Next, the administrator module is developed, enabling the creation of seasons, teams, players, and matches, and the recording of events. Finally, the "Home" and "Profile" screens are implemented, as by that stage all necessary data are available.

1.4 Work plan

Scheduling is essential to address all phases without leaving important tasks unfinished. Although the application is not overly complex, it includes many interdependent components, making block-based planning indispensable, while leaving room for adjustments.

The plan is divided into four main phases: prototyping, database design, base application construction, and incremental development by modules. Finally, an additional phase of testing, revision, and documentation is included.

The first weeks are dedicated to analysing other similar applications, both Fantasy leagues and sports statistics tools, in order to define requirements. Prototyping with Figma is carried out in parallel with database design. Once these are ready, the base application is created in Android Studio, progressively integrating each module. Development begins with the statistics logic, followed by the other screens.

Each module follows its own workflow: interface implementation, data integration, logic, and testing. This phase is expected to require the most time. At the end, a period is reserved for final testing, bug fixing, and report writing.

Although all new functionalities are tested as they are added, it is necessary, in the final weeks, to reserve a period of time for final testing, error correction, and preparation of the written report, as well as for some visual adjustments and navigation improvements to refine the user experience.

Although no mandatory partial submissions are established, regular contact with the supervisor is maintained throughout the development, in addition to several sessions in which the work completed up to that point is reviewed.

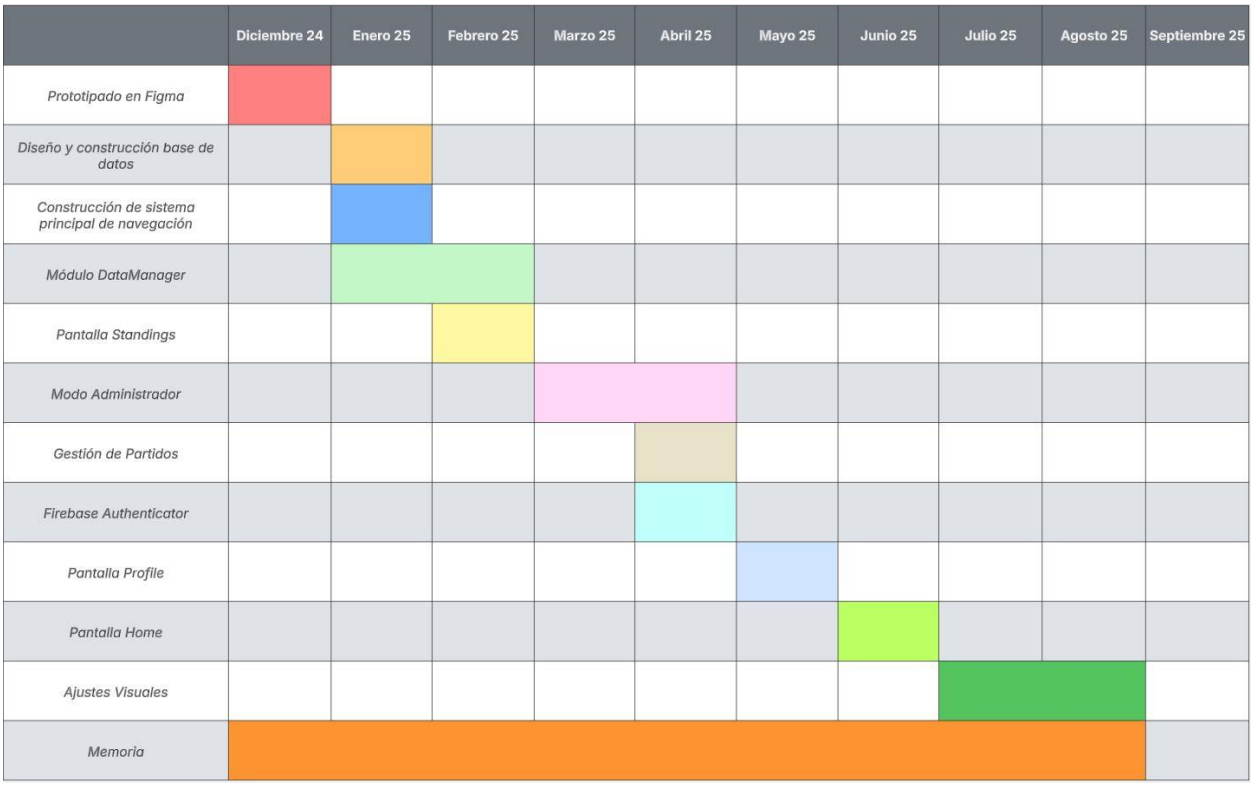


Table 1: Gantt Chart

1.5 Structure of the report

This document is organized into several chapters that follow the same order as the development process. The objective is to enable the reader to understand the entire process, from the initial motivation to the final result, including all intermediate phases such as design, implementation, and testing.

In Chapter 1, a general introduction of the project is presented. The personal motivation, the context, the main objectives, and the working methodology are outlined, together with a temporal planning of the development.

Chapter 2 contains an analysis of the existing solutions in the market. Other applications related to the tracking of sports statistics are compared, and their limitations are identified, which serves to justify the need and the approach of the developed application.

In Chapter 3, the entire conceptual design of the system is described. The types of users who interact with the application, the functional and non-functional requirements, the use cases, and the general architecture are defined. The structure of the navigation between screens is also explained.

Chapter 4 focuses on prototyping. The use of Figma as a visual design tool is described, and the main screens of the application are presented, explaining their function and design.

Chapter 5 covers the entire development process. The tools used are explained, the database design is presented, and the progressive implementation of each module is detailed, including both the administrator's and the player's perspective.

Chapter 6 shows the real functioning of the application. Flow diagrams and real examples are included.

In Chapter 7, the tests carried out are addressed. The validation of the functionalities is analyzed, and the problems encountered during development are described together with the solutions applied.

Chapter 8 includes the final conclusions, a review of the objectives achieved, and a proposal of improvements and future functionalities that could be implemented later.

Finally, the document is completed with a glossary of technical terms and the bibliography used, as well as several appendices.

Capítulo 2 - Análisis y Estudio Previo

Antes de iniciar el desarrollo de la aplicación, se consideró importante realizar un análisis de las soluciones ya existentes en el mercado que ofrecieran funcionalidades similares a las planteadas. El objetivo no era replicar otras aplicaciones, sino comprender qué características ofrecían, a qué tipo de usuarios se dirigían y, especialmente, qué carencias presentaban en relación con las necesidades del fútbol amateur desde una perspectiva cercana. Este análisis permitió identificar tanto aspectos bien resueltos en otras apps como oportunidades de mejora, con el fin de desarrollar una solución que aportase valor añadido a jugadores y entrenadores de equipos no profesionales.

2.1 Aplicaciones de seguimiento de estadísticas deportivas

Existen muchas aplicaciones orientadas a seguir estadísticas de fútbol a nivel profesional. Algunas de las más conocidas son LaLiga Fantasy[4] (ver Figura 1) o FotMob[5] (ver Figura 2). Estas aplicaciones están muy enfocadas al seguimiento de partidos de ligas oficiales o a la explotación de datos de los mismos para la creación de ligas virtuales, pero ambas ofrecen un nivel de detalle muy alto, tanto en estadísticas de jugadores como en tiempo real de encuentros.

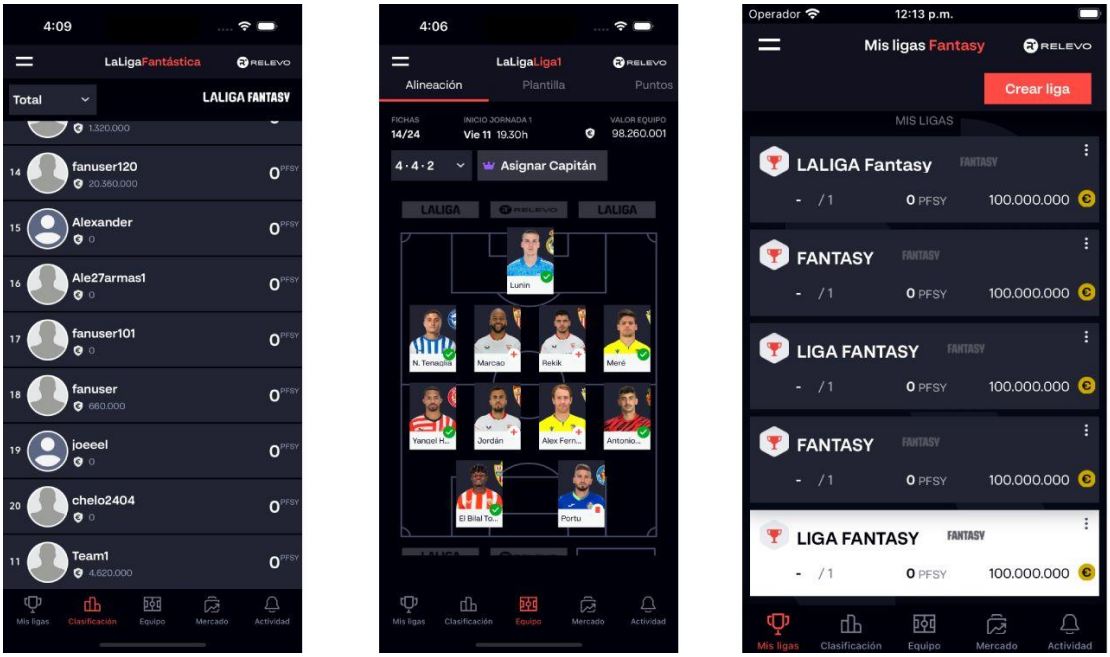


Figura 1: Capturas de pantalla de aplicación de ligas virtuales LaLiga Fantasy

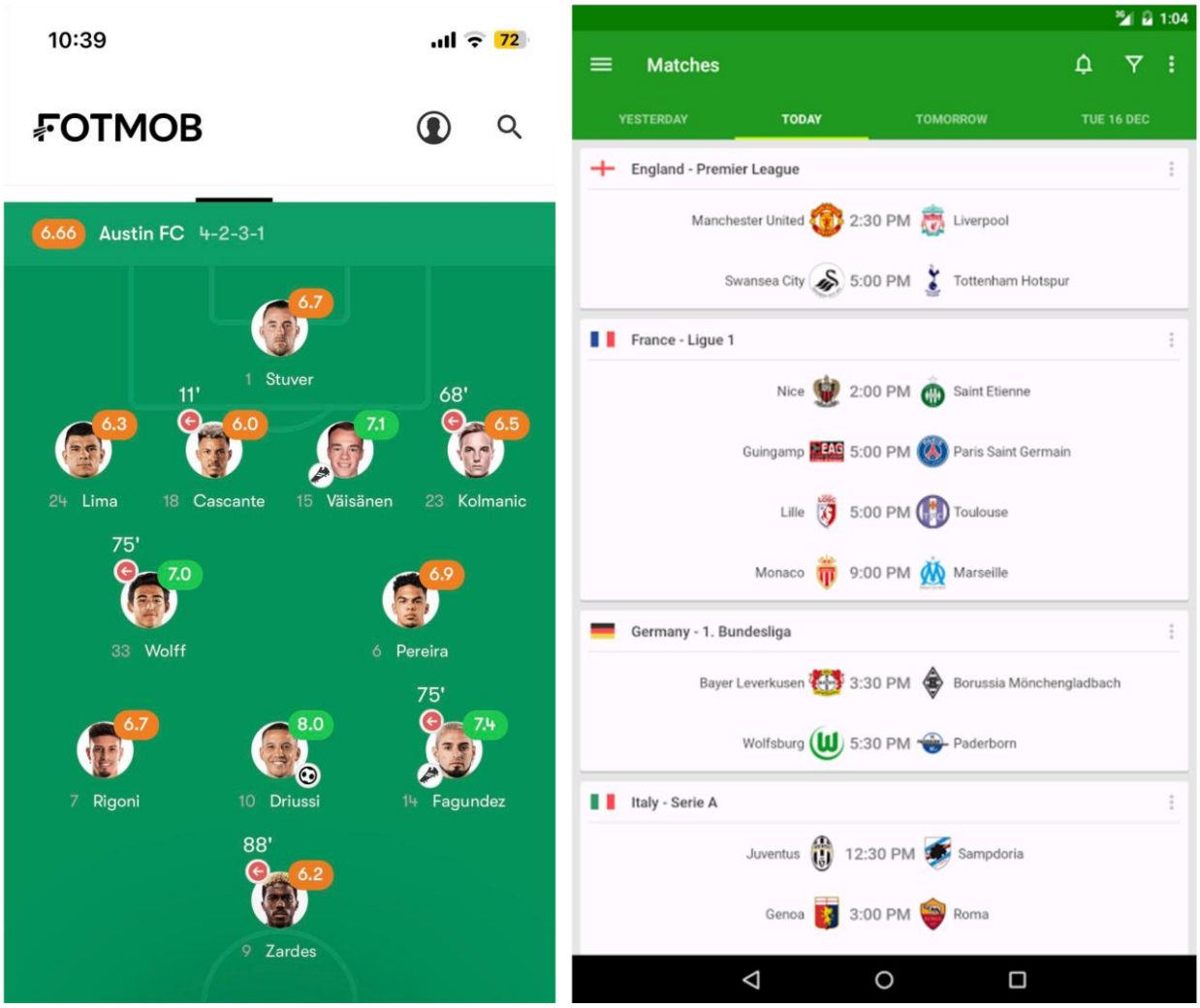


Figura 2: Capturas de pantalla de aplicación de estadísticas en directo FotMob

Por ejemplo, en Flashscore[6] (ver Figura 3) se pueden consultar los goles, tarjetas, asistencias, alineaciones, posesión, remates, etc., de prácticamente cualquier partido profesional, en directo o ya finalizado, lo cual es bastante irreal en ligas amateur ya que hay estadísticas que evidentemente no se pueden obtener. En el caso de LaLiga Fantasy, además de seguir partidos reales y estadísticas verídicas, los usuarios pueden gestionar un equipo propio de jugadores profesionales y competir contra amigos en función del rendimiento real de esos futbolistas.

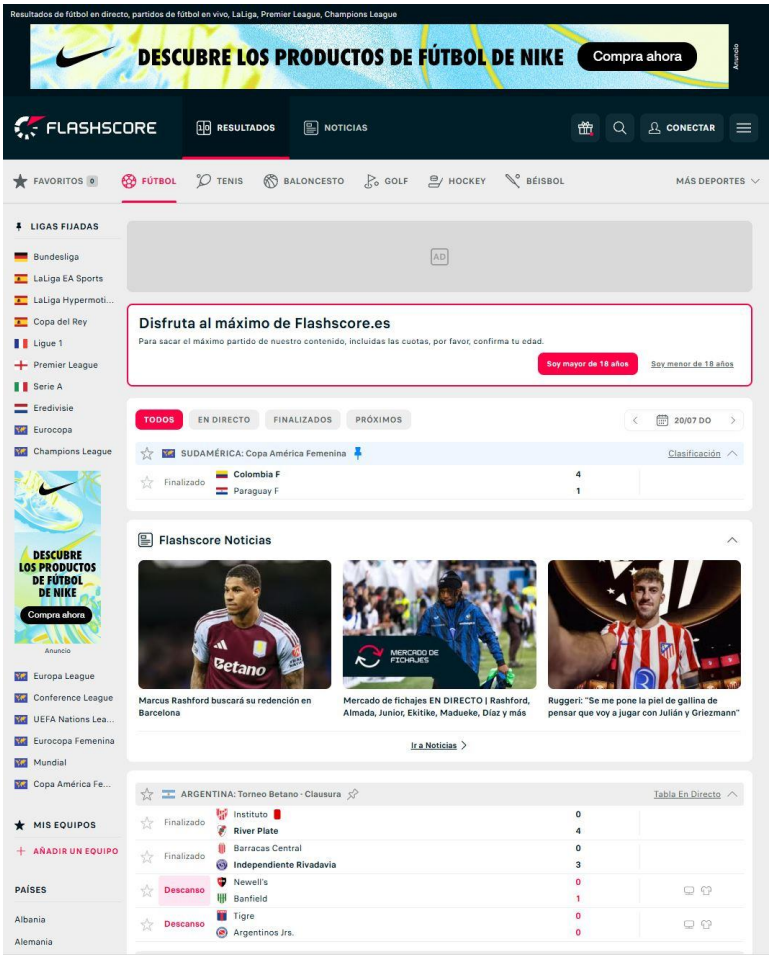


Figura 3: Captura de pantalla de la aplicación de estadísticas en directo Flashscore

Estas aplicaciones están muy bien diseñadas, tienen interfaces cuidadas y actualizan datos de forma casi instantánea. Sin embargo, todas ellas se basan en información pública y profesional, y no permiten introducir datos personalizados, algo que es clave en un entorno amateur donde no hay estadísticas automáticas o ningún otro tipo de cobertura en directo.

2.2 Aplicaciones orientadas a fútbol amateur o ligas fantasy

Dentro del fútbol amateur también existen algunas apps que se centran más en la gestión de equipos o en crear ligas de estilo fantasy entre amigos. Algunas de las que más destacan son Matchapp[7], SFL Football[8], Prematch[9] o Mister Fantasy[10] (ver Figuras 4 y 5).

Matchapp, por ejemplo, está muy enfocada a la gestión de partidos, consulta de resultados y algunos datos más como goles o tarjetas, pero con muy poco detalle. De hecho, lo que empezó siendo una herramienta privada y ajena a la organización, a día de hoy está reconocida oficialmente por la Federación de Fútbol de Madrid para la publicación de resultados y actas, así como para proveer información acerca de los equipos, aparte de la página oficial de la que disponen.

SFL Football, por el contrario, sí que es una app que organiza ligas amateur y permite ver clasificaciones y resultados, pero todo lo gestiona una empresa externa y no hay una entrada libre de datos por parte del usuario o entrenador. Está pensada para torneos organizados por ellos, aunque también dan la posibilidad de asignar algún delegado por equipo que pueda introducir datos limitados.

En cuanto a Mister Fantasy, aunque también es una app tipo Fantasy como la de LaLiga, permite crear ligas privadas y tiene algo más de libertad, pero sigue basándose en jugadores profesionales y no contempla datos reales de partidos amateur.

Existe la app Prematch, una aplicación que tiene una idea similar a la de esta aplicación que vamos a desarrollar, pero sólo permite seleccionar equipos de Inglaterra y Alemania y además no de todas las ligas.

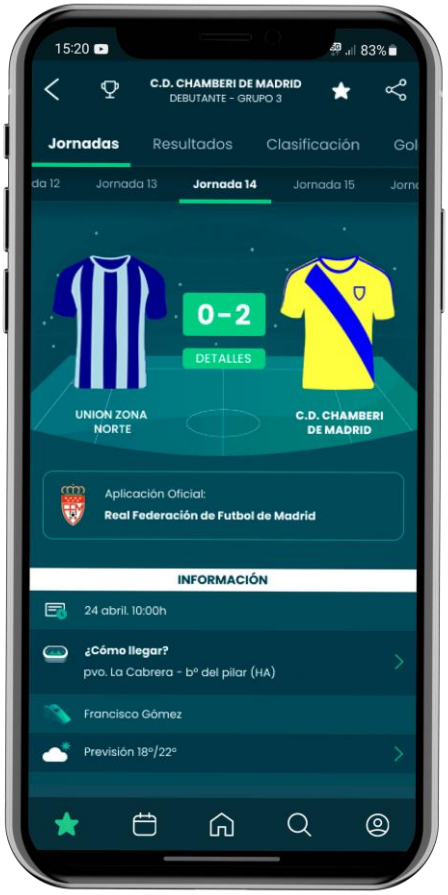


Figura 4: Captura de pantalla de aplicación Matchapp

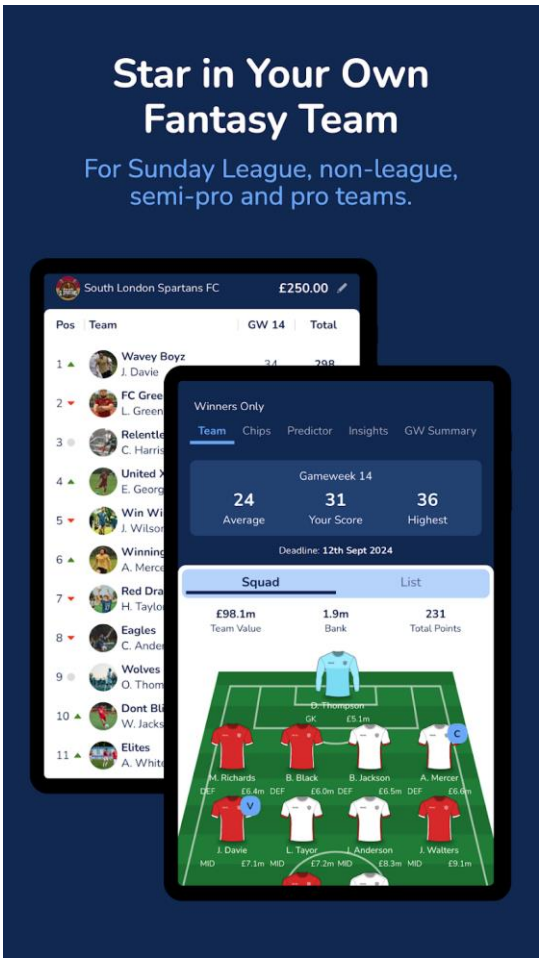


Figura 5: Captura de pantalla de aplicación SFL Football

2.3 Funcionalidades comunes y limitaciones

Después de revisar todas estas aplicaciones, lo que queda claro es que la mayoría están pensadas para contextos muy distintos al que se quería cubrir. Las apps profesionales tienen muchísima información, pero son completamente externas: los usuarios sólo consumen datos, no pueden añadir los suyos. Por otro lado, las apps de equipos amateurs se centran mucho en la parte de la organización, o en el caso de que se centren en la parte del seguimiento deportivo, están bastante controladas por entidades.

La mayor diferencia entre estas aplicaciones y esta propuesta es que en estas no hay forma de que un entrenador pueda introducir los datos de sus propios partidos, o que los jugadores puedan consultar su evolución a lo largo del tiempo con estadísticas reales del campo.

En resumen, se echa en falta una herramienta, que sea simple de usar, pensada desde dentro de un vestuario amateur y adaptada a esta realidad donde muchas veces no hay crónicas, ni estadísticas, ni formas de seguir la evolución de un jugador más allá de algunos datos muy limitados o lo que recuerde el propio equipo.

2.4 Valor diferencial de la propuesta

Teniendo en cuenta todo lo anterior, la aplicación desarrollada en este proyecto pretende cubrir precisamente ese hueco: ofrecer a equipos amateur una herramienta con la que puedan registrar y consultar sus propias estadísticas de manera fácil, útil y adaptada a su realidad.

A diferencia del resto, en esta aplicación es el entrenador (o administrador, que también podría ser un delegado) quien introduce los datos reales del partido. Tiene control sobre los equipos, los jugadores, las competiciones y los eventos registrados, como goles, asistencias, penaltis o sustituciones. Esto permite que los datos estén actualizados jornada a jornada y que reflejen realmente lo que pasa en el campo.

Por otro lado, los jugadores pueden consultar sus estadísticas personales, seguir su evolución y compararse con el resto del equipo, fomentando así la implicación, el compromiso y el seguimiento de su propio rendimiento para una posible inducción a la mejora.

La aplicación, en general, está pensada para ser ligera, visualmente clara y fácil de usar, sin complicaciones.

Todo esto convierte la aplicación en una propuesta diferente que no pretende sustituir a las apps profesionales ya existentes, ni competir con las Fantasy que hay en el mercado, sino dar a los equipos que entrenan y compiten cada semana una forma de llevar su propio seguimiento real, desde dentro.

Capítulo 3 - Diseño del Sistema

El diseño del sistema constituye una fase clave dentro del desarrollo de la aplicación, ya que define la estructura funcional que dará soporte a todas las interacciones posibles entre los distintos usuarios y la herramienta. Este apartado recoge las decisiones adoptadas en relación a la arquitectura general, la navegación entre pantallas, la organización de los datos y los distintos perfiles de usuario contemplados y sus responsabilidades.

Para llevar a cabo este diseño inicial, se han tenido en cuenta tanto aspectos funcionales como no funcionales para garantizar una experiencia de usuario fluida, segura y escalable, adecuada al contexto amateur al que queremos dirigir la aplicación.

En las siguientes secciones se detallan los perfiles de usuario, casos de uso, requisitos y arquitectura técnica empleada.

3.1 Perfil de usuarios y contexto de uso

En el desarrollo de esta aplicación se ha optado por una diferenciación clara entre dos tipos de usuario, atendiendo a las distintas responsabilidades y formas de interacción que cada uno tiene con la herramienta. Esta distinción resulta esencial tanto a nivel técnico, dependiendo de las funcionalidades de las que cada tipo de usuario podrá hacer uso, como desde el punto de vista de la experiencia de usuario, ya que permite ofrecer una interfaz diferenciada adaptada a las necesidades y tareas específicas de cada perfil.

3.1.1 Administrador

El rol del administrador está pensado principalmente para el entrenador del equipo, aunque podría extenderse a otros responsables técnicos o delegados de cada club. Este perfil tiene acceso completo a las funcionalidades relacionadas con la gestión y registro de información deportiva, ya que es el encargado de introducir los datos estadísticos tras cada jornada o partido disputado. Además, en una primera instancia

de la aplicación, el administrador va a ser también el encargado de crear usuarios, que luego serán asignados a cada jugador para que pueda acceder a la aplicación.

Entre las funciones asignadas al administrador, por tanto, se encuentran:

- Creación y gestión de temporadas: el administrador puede iniciar nuevas temporadas cuando comienza una competición, permitiendo así organizar las estadísticas por ciclos anuales (o en el tipo de ciclo que se considere).
- Creación de equipos: el administrador puede generar los distintos equipos que vaya a dirigir esa temporada, es decir, dentro de un mismo ciclo, un entrenador podrá gestionar la información de más de un equipo a la vez. Tendrá la posibilidad de crear estos equipos además de poder editar toda su información.
- Gestión de usuarios: una vez creados los equipos, el administrador deberá crear los usuarios (jugadores) de cada equipo que vaya a manejar dentro de una temporada. Además, como hemos comentado antes, en una primera versión de la aplicación, no existirá registro por parte de los usuarios, sino que será el administrador el que cree los usuarios y les asigne un correo, nombre de usuario y contraseña y, posteriormente, el usuario podrá iniciar sesión para ver las estadísticas, además de poder editar parte de su información (nombre, dorsal, contraseña, etc). El administrador podrá modificar posteriormente estos datos también, o eliminar los usuarios existentes.
- Registro de partidos: tras cada encuentro, el administrador tiene la capacidad de introducir los datos del mismo, como nombre del rival, fecha y algún otro dato relevante.
- Registro de datos del partido: al crear el propio partido, el usuario administrador introducirá los eventos del mismo, como goles a favor, en contra, cambios, asistencias, etc.

Este usuario tiene pleno control sobre la base de datos interna del equipo, ya que actúa como fuente oficial de los datos que posteriormente se visualizan en la interfaz pública. Por tanto, su perfil requiere acceso completo y sin restricciones a todas las funcionalidades administrativas de los equipos que gestiona.

Es importante destacar que todas las estadísticas introducidas por el administrador tienen un impacto directo en la experiencia de los jugadores, ya que son

la base sobre la que se calculan las clasificaciones, los rankings o las estadísticas individuales.

3.1.2 Jugador

El jugador representa al usuario más habitual de la aplicación, que será quien vaya a disfrutar del valor real de las posibilidades que ofrece la aplicación. Su interacción con la app se limita a la consulta de información, sin posibilidad de editar o modificar sus datos, salvo los suyos propios (perfil). Este enfoque garantiza la integridad de las estadísticas, lo que evitará manipulaciones de los datos por parte de los propios jugadores.

Entre las funciones disponibles para el jugador / usuario general, se incluyen:

- Visualización de estadísticas personales: el jugador puede acceder a su ficha individual, donde consulta los datos acumulados de cada temporada (goles, asistencias, minutos jugados, etc.).
- Consulta de clasificaciones: el jugador podrá ver rankings de su equipo y por tanto de sus compañeros, basados en diferentes métricas (ordenados por goles, asistencias, minutos, etc).
- Acceso a los datos de los partidos: el jugador podrá repasar los detalles de cada partido, como los eventos destacados o los resultados.
- Configuración del perfil: accede a su información personal básica, con posibilidad de visualizar datos esenciales y, además, poder modificar datos no relevantes para las estadísticas (como el nombre, el dorsal o el correo).

Este tipo de usuario tiene permisos únicamente de lectura sobre los datos almacenados (salvo los que vaya a poder editar en su perfil). No puede crear ni modificar partidos, registrar estadísticas ni editar datos del equipo o de otros jugadores. Su experiencia de uso está enfocada en la explotación de los datos para la motivación personal y el seguimiento del rendimiento deportivo, dentro de un entorno competitivo organizado.

3.1.3 Comparativa de permisos

En la Tabla 2 se presenta una comparativa de permisos entre los perfiles de administrador y jugador dentro de la aplicación. En ella se especifican las funcionalidades disponibles para cada rol, diferenciando claramente las acciones de gestión y configuración reservadas al administrador (como la creación de temporadas, equipos o usuarios, así como el registro de partidos y la edición de datos) frente a las funciones orientadas al jugador, centradas en la consulta de estadísticas, clasificaciones y partidos.

Funcionalidad	Administrador	Jugador
Crear temporadas	Sí	No
Crear equipos	Sí	No
Crear usuarios en equipos	Sí	No
Registrar partidos	Sí	No
Editar datos de usuarios	Sí	No
Consultar estadísticas de jugadores	No (no tiene)	Sí
Consultar clasificaciones del equipo	Sí	Sí
Visualizar partidos y estadísticas de los mismos	Sí	Sí
Acceso a la configuración del perfil	Sí	Sí (y podrá modificar datos no relevantes para las estadísticas)

Tabla 2: Comparativa de permisos entre usuarios

3.2 Casos de uso: registro de estadísticas, consulta de datos, navegación

En esta sección se describen los principales casos de uso que estructuran el funcionamiento de la aplicación. Estos casos de uso permiten comprender cómo interactúan los distintos actores con el sistema y qué funcionalidades son esenciales para alcanzar los objetivos del proyecto.

Se aborda el registro de estadísticas, que constituye una de las funcionalidades clave de la aplicación, ya que permite almacenar y gestionar los datos derivados de los partidos, como goles, asistencias o minutos jugados entre otros.

Se presenta también la consulta de datos, que proporciona a los usuarios acceso a la información almacenada, permitiendo visualizar clasificaciones, estadísticas individuales y colectivas.

Se detalla además la navegación dentro de la aplicación, que garantiza una interacción fluida e intuitiva, facilitando el acceso a las diferentes secciones mediante menús, fragmentos y pantallas adaptadas a cada tipo de usuario.

De esta manera, los casos de uso expuestos reflejan los escenarios fundamentales en los que se sustenta el desarrollo del sistema, mostrando tanto la parte administrativa como la de consulta y experiencia de usuario.

3.2.1 Casos de uso

Caso de uso 1	**Iniciar Sesión**		
Descripción	El actor inicia sesión en la aplicación.		
Actores	Jugador y Administrador		
Precondiciones	El actor debe tener una cuenta autenticada en el sistema de almacenamiento creada previamente.		
Post condiciones	Se muestra la pantalla principal de la aplicación.		
Secuencia Normal	**#**	**Acción (actor)**	**Reacción (sistema)**
	1	Introduce datos de inicio de sesión.	Espera a que se presione el botón de inicio de sesión.
	2	Presiona el botón de inicio de sesión.	Comprueba que existe cuenta y que credenciales son correctas y si lo son, accede a la aplicación, cargando los datos relativos a ese usuario.
Excepciones	#	**Acción (actor)**	**Reacción (sistema)**
	1	En el caso de que las credenciales introducidas por el usuario no sean	En el caso de que las credenciales introducidas por el usuario no sean

		correctas, el sistema deberá advertir de que los datos son erróneos.	correctas, el sistema deberá advertir de que los datos son erróneos.

Caso de uso 2	**Consultar estadísticas personales**		
Descripción	El actor consulta las estadísticas relativas a su usuario en los equipos a los que pertenece. Inicialmente en una vista previa y más en detalle en la pantalla Profile.		
Actores	Jugador		
Precondiciones	El actor debe haber iniciado sesión correctamente y por tanto se habrán cargado los datos de sus equipos.		
Post condiciones	Se muestran las estadísticas personales del jugador.		
Secuencia Normal	**#**	**Acción (actor)**	**Reacción (sistema)**
	1	Navega por las diferentes pestañas de estadísticas en la pantalla Profile.	Se actualizan los datos dependiendo de la pestaña seleccionada.
	2	Intercambia entre diferentes temporadas y/o equipos en los que haya jugado.	Se actualizan los datos dependiendo del valor de temporada y/o equipo seleccionados.

Excepciones	#	**Acción (actor)**	**Reacción (sistema)**
	1	En el caso de que no existan datos para alguna temporada o equipo seleccionado, no debería aparecer en los desplegables.	En el caso de que no existan datos para alguna temporada o equipo seleccionado, no debería aparecer en los desplegables.

Caso de uso 3	**Consultar estadísticas del equipo**		
Descripción	El actor consulta las estadísticas relativas a todos los usuarios de los equipos a los que pertenece o los que gestiona si es administrador.		
Actores	Jugador y administrador		
Precondiciones	El actor debe haber iniciado sesión correctamente y por tanto se habrán cargado los datos de sus equipos.		
Post condiciones	Se muestran las estadísticas del equipo seleccionado, del que el usuario forma parte.		
Secuencia Normal	**#**	**Acción (actor)**	**Reacción (sistema)**
	1	Navega por las diferentes pestañas de estadísticas en la pantalla Standings.	Se actualizan los datos dependiendo de la pestaña seleccionada.

	2	Intercambia entre diferentes temporadas y/o equipos en los que haya jugado.	Se actualizan los datos dependiendo del valor de temporada y/o equipo seleccionados.
Excepciones	#	**Acción (actor)**	**Reacción (sistema)**
	1	En el caso de que no existan datos para alguna temporada o equipo seleccionado, no debería aparecer en los desplegables.	En el caso de que no existan datos para alguna temporada o equipo seleccionado, no debería aparecer en los desplegables.

Caso de uso 4	**Consultar datos de partidos**		
Descripción	El actor consulta los datos de partidos creados previamente.		
Actores	Jugador y Administrador		
Precondiciones	El actor debe pertenecer a un equipo o ser administrador.		
Post condiciones	Se muestra la lista de partidos del equipo seleccionado y se facilita una versión detallada de los mismos.		
	#	**Acción (actor)**	**Reacción (sistema)**

Secuencia Normal	1	Navega hasta la sección MyTeam y selecciona temporada y equipo.	Se muestra la pantalla MyTeam con la información del equipo y los partidos que hayan sido creados para el mismo.
	2	Hace click sobre alguno de los partidos que ya han sido creados.	Se muestra una vista detallada y no editable del partido seleccionado.
Excepciones	#	**Acción (actor)**	**Reacción (sistema)**
	1	En el caso de que no existan partidos para ese equipo.	Se mostrará la lista vacía y por tanto no habrá datos.

Caso de uso 5	**Registro de nuevo Administrador**
Descripción	El actor se registra en la aplicación como un nuevo administrador.
Actores	Administrador

Precondiciones	Es necesario conocer la clave para poder registrarse como administrador. Además, el correo que se va a introducir no debe existir en el sistema de autenticación de la base de datos.		
Post condiciones	Se registra el nuevo administrador en la aplicación y se presenta la pantalla de Login para que pueda iniciar sesión.		
Secuencia Normal	**#**	**Acción (actor)**	**Reacción (sistema)**
	1	Introducir datos de registro.	Se comprueba la existencia de ese usuario y la clave a introducir. Si todo es correcto, se crea una nueva cuenta de administrador.
	2	Iniciar sesión con la nueva cuenta.	Comprueba que existe cuenta y que credenciales son correctas y si lo son, accede a la aplicación, cargando los datos relativos a ese usuario administrador.
Excepciones	#	**Acción (actor)**	**Reacción (sistema)**
	1	En el caso de que no coincida la clave de creación de administrador.	No se creará la cuenta de administrador y se avisará del error.

		En el caso de que ya exista una cuenta con el correo que se quiere registrar.	No se creará la cuenta de administrador y se avisará del error.

Caso de uso 6	**Crear temporada**		
Descripción	El actor crea una nueva temporada.		
Actores	Administrador		
Precondiciones	Es necesario tener permisos de administrador.		
Post condiciones	Temporada creada.		
Secuencia Normal	**#**	**Acción (actor)**	**Reacción (sistema)**
	1	Introducir datos de la nueva temporada.	Se comprueban datos correctos y se crea nueva temporada en ejecución y en base de datos.
Excepciones	#	**Acción (actor)**	**Reacción (sistema)**

Caso de uso 7	**Crear equipo para una temporada**		
Descripción	El actor crea un nuevo equipo.		
Actores	Administrador		
Precondiciones	Es necesario tener permisos de administrador y haber seleccionado una temporada.		
Post condiciones	Equipo creado.		
Secuencia Normal	**#**	**Acción (actor)**	**Reacción (sistema)**
	1	Introducir datos del nuevo equipo.	Se comprueban datos correctos y se crea nuevo equipo para esa temporada, en ejecución y en base de datos.
Excepciones	#	**Acción (actor)**	**Reacción (sistema)**

Caso de uso 8	**Crear User en un equipo**		
Descripción	El actor crea un nuevo User en un equipo.		
Actores	Administrador		
Precondiciones	Es necesario tener permisos de administrador y haber seleccionado una temporada y un equipo.		
Post condiciones	User creado en el equipo.		
Secuencia Normal	**#**	**Acción (actor)**	**Reacción (sistema)**
	1	Introducir datos del nuevo User. El correo introducido puede o no estar ya registrado. Si está registrado ya, debe pertenecer al mismo Administrador, pero puede estar en otra Season.	Se comprueban datos correctos y se crea nuevo User para ese equipo dentro de la temporada seleccionada, en ejecución y en base de datos. Se debe comprobar previamente que, si ya existe una cuenta con ese mismo correo, ese jugador pertenezca al mismo administrador. Se crea la nueva cuenta en el sistema de autenticación de la base de datos para ese usuario, así como la

			entrada en la tabla User y la clave foránea en la tabla Team.
Excepciones	#	**Acción (actor)**	**Reacción (sistema)**
		Se introduce un correo que ya está dado alta para otro administrador.	El sistema avisa y no crea ese usuario.

Caso de uso 9	**Crear Match para un equipo**		
Descripción	El actor crea un nuevo partido en un equipo.		
Actores	Administrador		
Pre condiciones	Es necesario tener permisos de administrador y haber seleccionado una temporada y un equipo. Es necesario que el equipo tenga once jugadores creados.		
Post condiciones	Partido creado para el equipo.		
Secuencia Normal	**#**	**Acción (actor)**	**Reacción (sistema)**
	1	Introducir datos informativos del partido.	Se registran los datos del partido.

	2	Introducir estadísticas del partido.	Se registran las estadísticas en el partido creado, así como en los usuarios implicados en los eventos del partido.
Excepciones	#	**Acción (actor)**	**Reacción (sistema)**
		No se introducen los datos mínimos del partido como once inicial o el nombre o fecha.	El sistema avisa y no crea ese partido.

Caso de uso 10	**Editar datos de un Match**		
Descripción	El actor edita los datos y las estadísticas de un partido.		
Actores	Administrador		
Pre condiciones	Es necesario tener permisos de administrador y haber seleccionado una temporada y un equipo, y seleccionar un partido creado previamente.		
Post condiciones	Los cambios en los datos del partido son guardados.		
	#	**Acción (actor)**	**Reacción (sistema)**

Secuencia Normal	1	Editar los datos estadísticos del partido. Para ello eliminar y crear nuevas estadísticas. Guardar la edición.	Se registran los nuevos datos del partido si estos son correctos. Se actualizan las estadísticas en los jugadores implicados en el partido editado.
Excepciones	#	**Acción (actor)**	**Reacción (sistema)**
		No se introducen los datos mínimos del partido como once inicial o el nombre o fecha.	El sistema avisa y no edita ese partido.

Caso de uso 11	**Borrar un partido**
Descripción	El actor borra un partido.
Actores	Administrador
Pre condiciones	Es necesario tener permisos de administrador y haber seleccionado una temporada y un equipo, y seleccionar un partido creado previamente.

Post condiciones	Se elimina el partido y se actualizan las estadísticas de los jugadores implicados.		
Secuencia Normal	**#**	**Acción (actor)**	**Reacción (sistema)**
	1	Eliminar el partido.	Se elimina la entrada de la tabla de partidos. Se actualizan las estadísticas en los jugadores implicados en el partido borrado.
Excepciones	#	**Acción (actor)**	**Reacción (sistema)**

Caso de uso 12	**Editar nombre y/o contraseña**
Descripción	El actor edita algunos de sus datos.
Actores	Administrador y Jugador
Pre condiciones	Es necesario tener una cuenta y haber iniciado sesión.
Post condiciones	Se cambia el nombre y/o la contraseña del usuario seleccionado para todos los equipos de los que forma parte.

Secuencia Normal	**#**	**Acción (actor)**	**Reacción (sistema)**
	1	Editar nombre	Se actualiza el nombre en base de datos para ese usuario, para todos los equipos de los que forma parte.
	2	Editar contraseña (escribiéndola dos veces)	Si coinciden, se actualiza la contraseña para el usuario en el sistema.
Excepciones	#	**Acción (actor)**	**Reacción (sistema)**
	1	No coinciden las dos contraseñas introducidas.	No se actualiza la contraseña.
	2	El nombre tiene más de 16 caracteres.	No se actualiza el nombre.

3.3 Requisitos funcionales

Los requisitos funcionales describen las acciones y comportamientos específicos que la aplicación debe ser capaz de ejecutar para cumplir con sus objetivos. Están centrados en las funcionalidades que los diferentes tipos de usuarios (administrador y jugador) pueden realizar dentro del sistema y constituyen la base para el diseño de los casos de uso, pantallas y flujos de navegación.

A continuación, se detalla la lista de requisitos funcionales identificados durante el análisis y diseño de la aplicación. Se han clasificado según el rol del usuario que los ejecuta, para facilitar su comprensión y vinculación con las funciones reales.

3.3.1 Requisitos funcionales del administrador (entrenador)

- RF1: puede iniciar sesión en la aplicación con sus credenciales.
- RF2: puede crear nuevas temporadas y asignarles un nombre y año identificativo.
- RF3: puede crear equipos dentro de una temporada determinada.
- RF4: puede crear jugadores dentro de un equipo.
- RF5: puede consultar la información de un equipo.
- RF6: puede consultar la información de un jugador.
- RF7: puede registrar nuevos partidos dentro de un equipo y temporada.
- RF8: puede introducir eventos dentro del partido:
    - Goles a favor (jugador y minuto)
    - Goles en contra (minuto)
    - Asistencias (jugador)
    - Penaltis parados (jugador)
    - Penaltis fallados (jugador)
    - Penaltis provocados (jugador)
    - Jugadores del Once inicial (jugadores)
    - Sustituciones (jugador que entra, jugador que sale, minuto)
- RF9: puede consultar partidos anteriores y sus eventos.
- RF10: puede modificar los datos de los partidos ya creados dentro de un equipo y temporada.
- RF11: puede visualizar estadísticas globales del equipo.
- RF12: puede borrar partidos y sus estadísticas y, a su vez, reflejar esos cambios en los usuarios.

3.3.2 Requisitos funcionales del jugador

- RF13: puede iniciar sesión en la aplicación con sus credenciales.

- RF14: puede acceder a su pantalla principal (Home) y ver un resumen del rendimiento y las clasificaciones.
- RF15: puede consultar su perfil personal y sus estadísticas individuales.
- RF16: puede consultar la información del equipo al que pertenece.
- RF17: puede visualizar el detalle de cada partido.
- RF18: puede acceder a las clasificaciones por estadísticas, visualizando rankings del equipo según la métrica (goles, asistencias, minutos jugados, etc).
- RF19: puede acceder a su perfil y ver su información básica, así como editar algunos datos no relevantes o contraseñas.

3.4 Requisitos no funcionales

Los requisitos no funcionales definen características de calidad que debe tener la aplicación, independientemente de las funcionalidades que implemente. Afectan al rendimiento, usabilidad, mantenibilidad, compatibilidad o aspectos técnicos del sistema que influyen directamente en la experiencia de uso y en la fiabilidad del producto.

A continuación, se detallan los requisitos no funcionales establecidos para esta aplicación:

3.4.1 Rendimiento

- RNF1: La aplicación debe ser capaz de cargar y mostrar los datos de cada pantalla en un tiempo inferior a 1 segundo, garantizando la fluidez en la navegación
- RNF2: Debe soportar equipos de hasta 25 jugadores y temporadas con al menos 36 partidos, sin que se degrade el rendimiento perceptible por el usuario

3.4.2 Escalabilidad

- RNF3: El diseño del sistema y la estructura interna, así como la base de datos, deberán permitir la incorporación de nuevas funcionalidades futuras, como el modo Fantasy, sin necesidad de reestructurar completamente el sistema.

3.4.3 Usabilidad

- RNF4: La interfaz debe ser clara, coherente y adaptada a un público no técnico, con iconografía sencilla, textos descriptivos y una clara jerarquía visual.
- RNF5: la aplicación debe mostrar una barra de navegación inferior que permita acceder fácilmente a las cinco secciones principales (Home, Social, Standings, MyTeam, Profile).
- RNF6: La navegación debe ser intuitiva, permitiendo acceder a cualquier sección en no más de tres pulsaciones desde cualquier punto de la App.
- RNF7: la aplicación debe mostrar los datos de manera ordenada y actualizada.

3.4.4 Accesibilidad

- RNF8: La aplicación debe ser usable en dispositivos Android con versiones desde Android 7 (Nougat) en adelante.
- RNF9: El diseño debe adaptarse a diferentes tamaños de pantalla, garantizando una correcta visualización tanto en móviles pequeños como en tablets.

3.4.5 Seguridad y privacidad

- RNF10: Cada usuario debe acceder únicamente a los datos que le correspondan según su rol y equipo.
- RNF11: La base de datos debe garantizar que un jugador no pueda modificar sus estadísticas ni las de otros compañeros.
- RNF12: Autenticación. La aplicación debe proporcionar un sistema de identificación y autenticación de los usuarios de forma única.

3.4.6 Mantenibilidad

- RNF13: El código debe estar estructurado de forma modular, permitiendo su actualización y corrección futura con facilidad.
- RNF14: Las clases deben estar documentadas mediante comentarios que expliquen su función.

3.4.7 Persistencia de datos

- RNF15: Todos los datos introducidos por los administradores deben almacenarse en la base de datos de forma inmediata y mantenerse aunque se cierre la app o el dispositivo se reinicie.

3.4.8 Consistencia de datos

- RNF16: Los datos mostrados deben estar sincronizados entre pantallas y mantenerse actualizados, evitando errores de visualización o incoherencias.
- RF17: la aplicación debe cargar la información desde la base de datos al iniciar la aplicación, mostrando la información correspondiente al usuario conectado. Además, deberá actualizar en base de datos la información nueva que se introduzca o se edite.

3.5 Arquitectura general del sistema

La arquitectura de la aplicación está basada en una estructura modular y clara, utilizando el patrón MVVM[11] (Model - View - ViewModel) como punto de referencia principal, aunque adaptado a las necesidades del proyecto. Este enfoque ha permitido mantener un cierto orden entre la interfaz de usuario, la lógica de negocio y los datos, facilitando tanto el desarrollo como la lectura y mantenimiento del código.

En cada una de las vistas principales de la aplicación se ha creado un ViewModel específico, que actúa como intermediario entre la interfaz y los datos. Esto ha permitido separar la lógica de la parte visual, y centralizar el acceso a los datos desde una única fuente. Además, los ViewModel han servido para conservar los datos durante los cambios de configuración o cambios en la información, ya que en ellos se guardan los registros para poder ir actualizándolos en tiempo real, aparte de en memoria y en la base de datos.

3.5.1 Estructura general del sistema

La aplicación está organizada en tres capas principales:

- Vista (UI): formada por las Activity[12] y los Fragment[13], además de los Adapter[14] para los RecyclerView[15]. Aquí es donde se muestran los datos al usuario y se recogen sus acciones. Las vistas, por lo general, no acceden directamente a los modelos, sino que se comunican con su ViewModel correspondiente para obtener la información o realizar cambios. Es necesario recalcar que no en todos los casos es así porque en algunas vistas se han implementado diálogos con su correspondiente funcionalidad, que requerían estar en la propia Activity o Fragment.
- ViewModel: cada pantalla tiene su propio ViewModel. Este gestiona los datos que necesita la vista, encapsula la lógica de presentación y se comunica con las clases del modelo para recuperar o actualizar información.
- Model: hacen referencia a los objetos con los que se va a trabajar y serán utilizados por las vistas para mostrar información sobre Temporadas, Equipos, Usuarios y Partidos.

Capítulo 4 - Prototipado de la aplicación

Antes de comenzar con el desarrollo de la aplicación en Android Studio, se dedica una fase al diseño visual de la interfaz y la estructura general de navegación. Esta etapa es fundamental para tener una visión clara de cómo se organizarán los distintos elementos en cada pantalla y cómo se guiará al usuario a lo largo de la experiencia de uso. En esta sección se aborda el proceso de prototipado, cómo se plantea la navegación entre pantallas y de qué forma se estructura visualmente la aplicación. También se incluirán capturas de los diseños realizados, que servirán como referencia directa para la fase posterior de implementación.

El prototipo cumple un papel clave como guía conceptual y visual del proyecto.

Para la etapa de diseño visual de la aplicación, se decidió utilizar Figma como herramienta principal de prototipado. Desde el principio fue importante saber que, antes de comenzar con la parte de desarrollo en Android Studio, era fundamental tener una idea clara de cómo iba a ser la interfaz gráfica de la app y cómo se iban a distribuir los elementos en cada pantalla.

Ya se conocía la plataforma Figma anteriormente y, por tanto, se determinó como la opción más cómoda y accesible para esta tarea, ya que además permite trabajar en la nube sin necesidad de instalaciones y ofrece una gran versatilidad a la hora de crear diseños interactivos.

Otra de las razones por las que se optó por Figma fue la posibilidad que ofrece de poder tener una versión global de la aplicación desde el principio. Esto fue especialmente útil en las primeras fases del trabajo, cuando todavía se estaba decidiendo cómo iban a ser los menús, qué pantallas se querían incluir, qué botones se iban a implementar y cómo se iba a guiar al usuario entre secciones. Figma ofrece bastante personalización en este aspecto y por eso ayudó a visualizar cómo se quería que la aplicación funcionase y así poder trasladarlo posteriormente al código.

En el diseño en Figma, se definieron las cinco pantallas principales que inicialmente componen la estructura de navegación de la aplicación: Home, Social, Standings, MyTeam, Profile.

Cada una de ellas responde a una necesidad concreta y está pensada para un uso diferente. El diseño está basado en una barra de navegación inferior, que permite al usuario moverse rápidamente entre secciones desde cualquier punto de la App. Esta decisión de diseño se tomó para seguir un patrón común en muchas aplicaciones móviles actuales, facilitando así la experiencia de uso.

El proceso de prototipado en Figma también sirvió para pensar la jerarquía visual de la información. Por ejemplo, en la pantalla "Home" (ver Figura 6), se da prioridad a poder ver muchos tipos de información, pero más generalizada, acerca del rendimiento del jugador y de las estadísticas de su equipo, mientras que en "Standings" (ver Figura 8), el foco está en los rankings detallados por estadísticas, que ofrecen una mayor profundidad en los datos. En todas las pantallas se intenta mantener una estética coherente, con colores llamativos pero simples, tipografía legible y espaciado entre elementos, priorizando siempre la claridad y simplicidad, por encima de elementos decorativos innecesarios, ya que, por experiencia propia, siempre se ha preferido la interfaz minimalista.

Además del diseño estático, Figma permite crear una navegación simulada entre pantallas, enlazando botones y elementos para poder visualizar el flujo de uso completo como si se tratase de una aplicación ya funcional. Esta funcionalidad es especialmente útil a la hora de revisar el diseño con el tutor, ya que facilita mucho el entender cómo va a funcionar la aplicación en la práctica, sin necesidad de haber empezado a programar todavía.

Una vez completado el diseño, se utilizará este primer mapa gráfico como guía durante el desarrollo. Aunque puede haber algunos ajustes más adelante, la estructura básica del prototipo se mantendrá durante todo el proyecto, sirviendo como referencia constante.

Seguidamente, se incluyen capturas de las distintas pantallas diseñadas en Figma, organizadas por orden de navegación, y que reflejan tanto la estética general de la aplicación (aunque posteriormente se podrá cambiar el diseño) como las funciones básicas pensadas para cada sección.

Es importante destacar que no necesariamente todas las funcionalidades que se proponen en un principio en la versión inicial de Figma, serán implementadas para la primera versión de la aplicación.

A continuación, se muestran las capturas de los diseños iniciales elaborados en Figma.

Pantalla Home: se muestra una vista previa de la información que se va a visualizar posteriormente en detalle en las siguientes pantallas (ver Figura 6).

Pantalla Social: se muestran noticias y eventos sociales que ocurren durante la temporada (ver Figura 7).

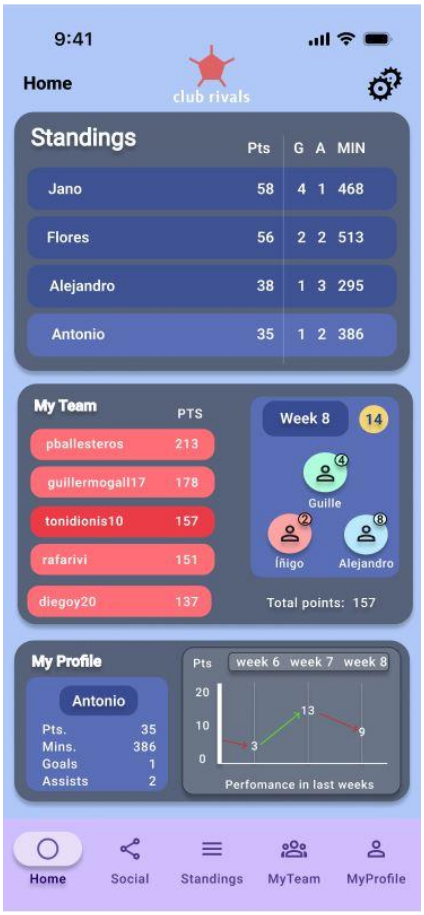


Figura 6: Captura de Figma de pantalla Home

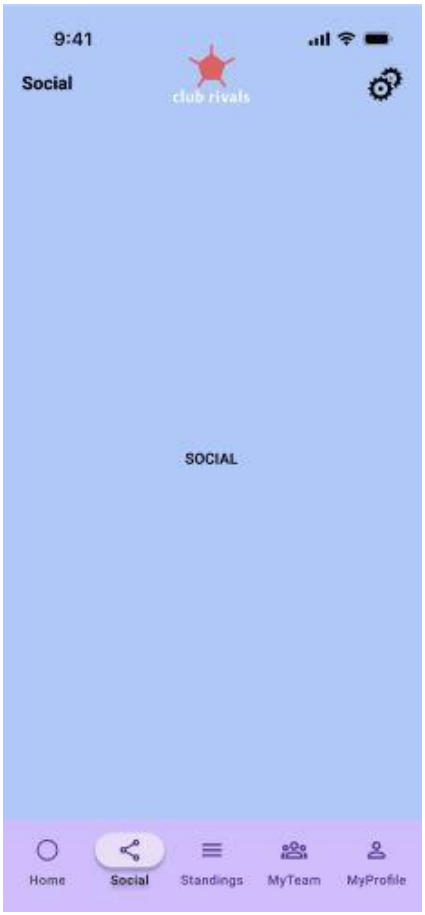


Figura 7: Captura de Figma de pantalla Social

Pantalla Standings: se muestran las clasificaciones detalladas según cada tipo de métrica (General por puntos, por goles, por asistencias, etc.) (ver Figuras 8, 9, 10, 11, 12 y 13).

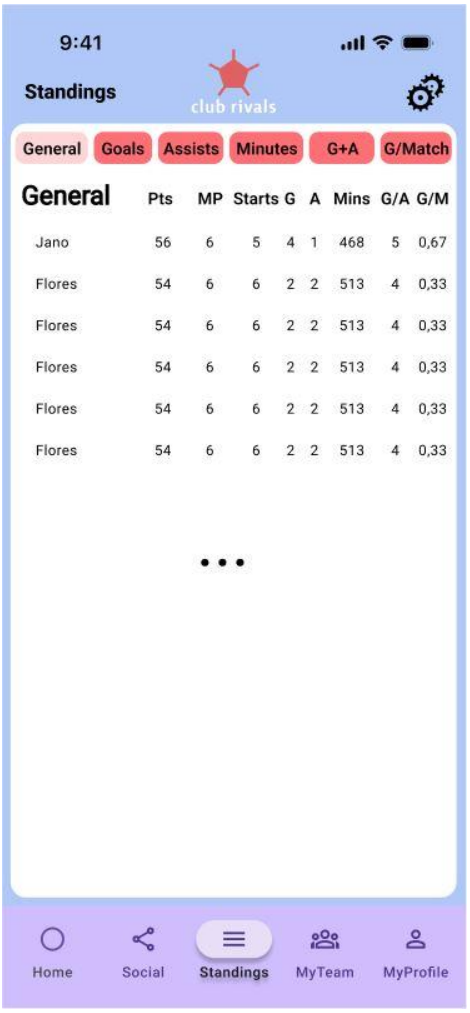


Figura 8: Captura de Figma de pantalla Standings General

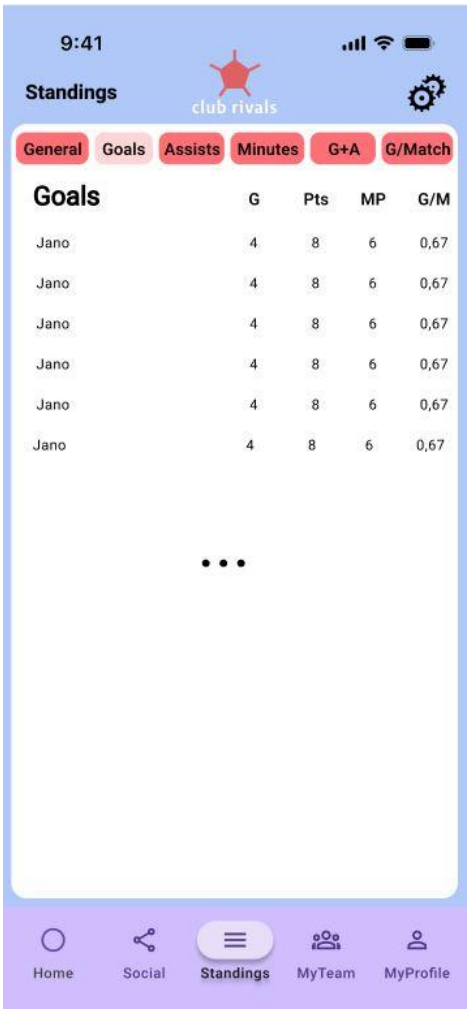


Figura 9: Captura de Figma de pantalla Standings Goals

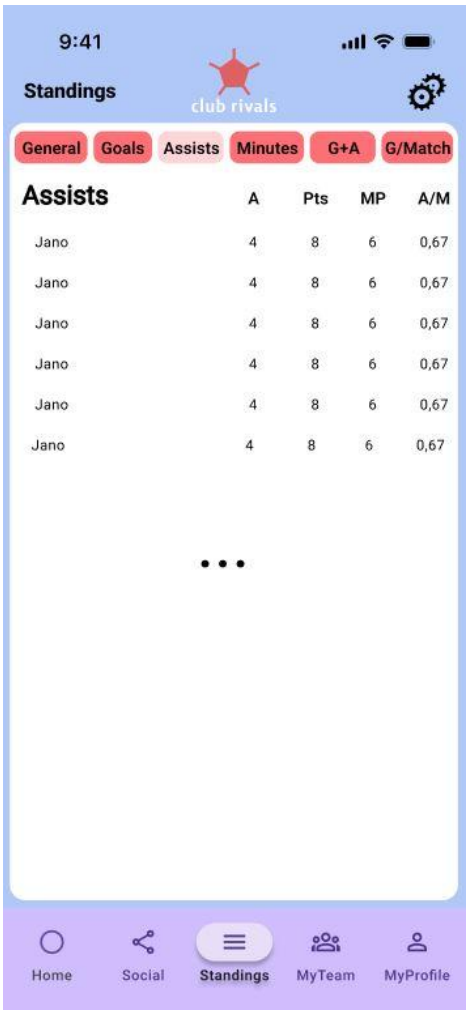


Figura 10: Captura de Figma de pantalla Standings Assists

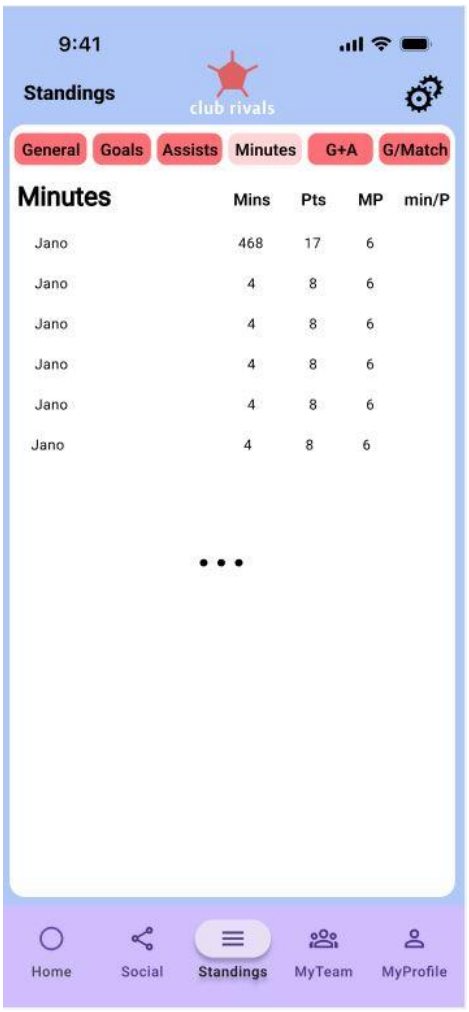


Figura 11: Captura de Figma de pantalla Standings Minutes

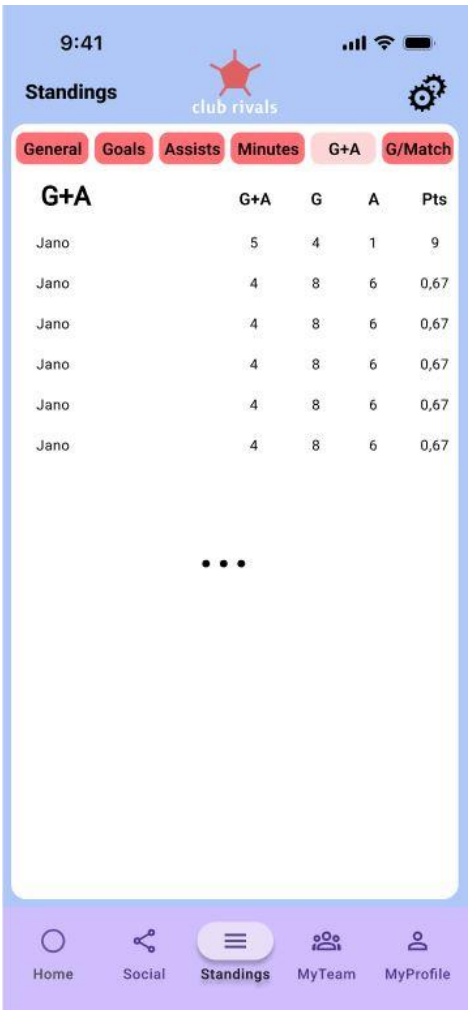


Figura 12: Captura de Figma de pantalla Standings Goals + Assists

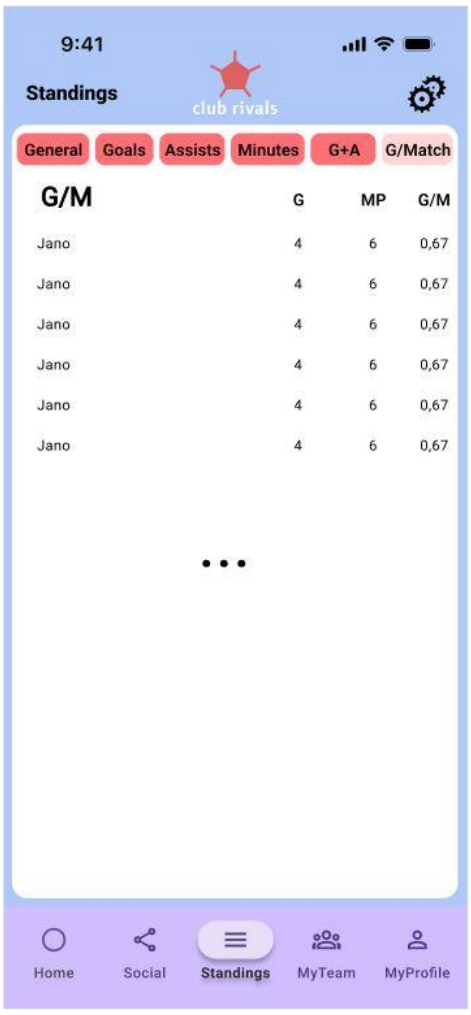


Figura 13: Captura de Figma de pantalla Standings Goals per Match

Pantalla My Team: se muestra la clasificación del juego secundario My Team así como el equipo del usuario y el que ha resultado ganador esa jornada. Se podrá crear un equipo para cada jornada y usuario (ver Figuras 14, 15, 16 y 17).

Figura 14: Captura de Figma de pantalla My Team (General)

Figura 15: Captura de Figma de pantalla My Team (Crear)

Figura 16: Captura de Figma de pantalla MyTeam (Añadir jugador)

Figura 17: Captura de Figma de pantalla My Team (Equipo creado)

Capítulo 5 - Desarrollo de la aplicación

Este capítulo recoge el proceso de desarrollo completo de la aplicación, detallando desde las herramientas utilizadas hasta los principales retos técnicos afrontados. Se explicarán las tecnologías empleadas y se justifica la elección de las mismas para cada funcionalidad del proyecto.

A continuación, se describe el diseño de la base de datos, la arquitectura del sistema principal y la lógica de navegación entre pantallas. También se profundiza en la implementación de los módulos clave, tanto desde el punto de vista del administrador como del jugador, abordando la gestión de partidos y la visualización de datos.

Finalmente, se analiza cómo se gestiona la persistencia y sincronización de la información, así como los desafíos técnicos que surgen durante el desarrollo y cómo se van resolviendo progresivamente.

5.1 Herramientas y tecnologías empleadas

Las herramientas utilizadas vienen definidas en parte por los requisitos del propio TFG. Se sugería que el trabajo se realizase en Android Studio, por lo que esta decisión realmente fue dirigida.

Para el desarrollo se utiliza Java como lenguaje principal, combinado con XML para la interfaz. Este es el lenguaje base de Android Studio que, aunque permite otro lenguaje, ya se había trabajado antes con ello y no hubo razón para cambiarlo (la otra opción es Kotlin[16]).

En cuanto a la base de datos, ya que se contaba con algo de experiencia de proyectos anteriores, se ha elegido Firebase[17], una solución gratuita de Google que ofrece las posibilidades justas que se necesitan, que en este caso van a ser dos. Por una parte, la del apartado de autenticación, para la que esta herramienta ofrece Firebase Authenticator[18], y por otra, la parte de almacenamiento de datos de Firebase Realtime Database[19].

Por último, se utilizará Github[20] para el control de versiones durante todo el desarrollo, lo que permite guardar copias del proyecto en diferentes estados, retroceder en caso de errores y organizar los avances por fases.

Gracias a este enfoque progresivo, se podrán ir añadiendo funcionalidades de forma controlada, asegurando en cada momento que todo funciona antes de pasar al siguiente paso. Esto ayuda a evitar errores acumulados y a tener siempre una versión funcional sobre la que seguir trabajando.

5.1.1 Github

Como se ha comentado, se ha utilizado Github como repositorio para almacenar todas las versiones de la aplicación. Debajo de estas líneas se encuentra una referencia al repositorio donde se encuentra el código.

Github: https://github.com/gmartinezc7/TFG_GonzaloMartinezCarrera.git

5.1.2 Android Studio, Java, XML

La elección de Android Studio como entorno de desarrollo viene determinada por los requisitos del propio proyecto. No obstante, su uso resulta natural y amigable gracias a la experiencia previa adquirida en asignaturas anteriores, lo que facilita el trabajo desde el principio.

Para el desarrollo de la lógica de la aplicación se utiliza Java, el lenguaje base de Android Studio, combinado con XML para el diseño de interfaces. Aunque la plataforma permite el uso de otros lenguajes como Kotlin, se opta por Java al haber trabajado con él anteriormente y ser cómoda su sintaxis y funcionamiento. Esta combinación permite una construcción clara y estructurada tanto del comportamiento interno de la aplicación como de su presentación visual.

5.1.3 Firebase

La base de datos utilizada es Firebase, una solución gratuita ofrecida por Google que se ajusta perfectamente a las necesidades del proyecto. Su elección se apoya en

la experiencia previa con esta herramienta en otros trabajos, lo que permite aprovechar su integración sencilla con Android Studio y sus servicios.

Sin embargo, durante las primeras fases del desarrollo, se simula el acceso a base de datos desde memoria con unas cargas manuales. De esa manera, se facilita la organización de las clases de forma que el cambio posterior a una conexión con Firebase resulte sencillo. Esta separación se consigue gracias al módulo personalizado DataManager, encargado de cargar la información de base de datos. Inicialmente, este módulo trabaja con datos de prueba y no es hasta que la aplicación alcanza un estado funcional cuando se conecta Firebase como fuente real de datos. Este enfoque facilita el desarrollo incremental y evita complicaciones prematuras durante las primeras etapas del proyecto.

5.2 Diseño e implementación de la base de datos

La base de datos utilizada en la aplicación está diseñada para reflejar con claridad la jerarquía lógica del sistema: administradores, temporadas, equipos, usuarios y partidos. Esta estructura se implementa en Firebase Realtime Database, un sistema no relacional que permite organizar la información en ramas anidadas y no anidadas y acceder a los datos de forma rápida y eficiente.

El diseño se base en una separación clara por entidades. En la raíz del árbol se encuentran las ramas principales: "admins", "seasons", "teams", "users", "matches". Cada una de estas ramas contiene objetos identificados por claves únicas compuestas, formadas cada una por los IDs de sus padres y su propio ID único, según corresponda.

Como particularidad, Firebase mantiene una estructura opaca en cuanto a las claves únicas, pero como esto no corrompe la seguridad de nuestra base de datos, cada objeto tendrá un campo con ese propio ID de Firebase para poder acceder a dichos datos desde sus hijos.

5.2.1 Admins

En la rama "admins", cada administrador tiene su propio nodo, donde se almacena información básica, además de una lista de temporadas asociadas

mediante la subrama "seasons", lo que permite relacionar directamente cada temporada con su creador. La estructura es la siguiente:

- Clave de Firebase: este identificador se crea aleatoriamente y debe ser único. Ejemplo: "admin_def456".
    - adminEmail: campo que guarda el email del usuario administrador, para que pueda identificarse.
    - adminName: campo informativo con el nombre del administrador.
    - adminID: campo en el que se guarda el identificador de Firebase de cada admin.
    - seasons: lista de identificadores de seasons de ese administrador.

5.2.2 Seasons

La rama "seasons" agrupa las distintas temporadas del sistema. Cada temporada tiene como clave un identificador compuesto con el ID del administrador y posteriormente se almacena la información de dicha temporada.

- Clave de Firebase: esta clave está formada por el identificador de su administrador más un identificador único para cada season. Ejemplo: "admin_def456_season146".
    - adminID: identificador del administrador para poder acceder a su información desde un hijo.
    - seasonID: identificador de Firebase de cada season.
    - year: nombre/año de la temporada.
    - teams: lista de identificadores de teams de esa season.

5.2.3 Teams

Cada equipo está definido en la rama "teams", también usando una clave compuesta. Cada objeto "team" contiene información sobre el equipo y una lista de usuarios.

- Clave de Firebase: clave formada por el identificador de su temporada (que a su vez incluye el identificador de su administrador) más un identificador único para cada team. Ejemplo: "admin_ghi789_season146_team376".
    - seasonID: identificador de la temporada para poder acceder a su información desde un hijo.
    - teamID: identificador de Firebase de cada team.
    - teamCategory: categoría (por edad) del equipo, como por ejemplo: Aficionado, Cadete, Benjamín, etc.
    - teamLeagueRanking: ranking dentro de la categoría, por ejemplo: Primera, Segunda, Autonómica, etc.
    - users: lista de identificadores de users de ese team

5.2.4 Users

Los jugadores se guardan en la rama users. Aquí se combinan dos tipos de entradas. Por un lado, los usuarios del sistema con privilegios (administradores), y por otro, los jugadores de cada equipo. Cada jugador se identifica por una clave formada por sus claves padre más una clave única del usuario (correo de autenticación único). Esta estructura permite que un mismo jugador pueda existir en distintos contextos (por ejemplo, en varias temporadas o equipos) sin duplicar información. Esto es necesario porque un mismo usuario de la aplicación puede tener entradas en más de un equipo. Dentro de cada entrada se guarda la siguiente información:

- Clave de Firebase: clave única del jugador (por equipo) formada por el identificador de su equipo (que a su vez contiene el identificador de su

temporada y su admin), y el correo de autenticación del jugador, para identificarlo unívocamente.

- email: email con el que se ha autenticado el usuario.
- firebaseKey: identificador de Firebase de este usuario.
- id: identificador de autenticación para búsquedas internas.
- name: nombre del jugador.
- lastname: apellido del jugador.
- position: posición en la que juega el usuario (dependiendo de esta función, la puntuación varía).
- userAdmin: identificador del usuario admin que ha creado este usuario, para búsquedas internas.
- username: nombre de usuario que se utilizará en funciones futuras de la aplicación.

A partir de aquí, guardamos las estadísticas de ese jugador para ese equipo en los campos correspondientes:

- goals
- assists
- matches
- minutes
- myteampoints
- points
- pointsassists
- pointsminutes
- pointsstarts
- starts

5.2.5 Matches

Por último, los partidos se gestionan en la rama "matches", donde cada encuentro tiene un identificador único que incluye el administrador, temporada, equipo

y clave única del partido. Cada partido almacena información del mismo, coherente con la que se guarda en los usuarios. Este formato permite consultar rápidamente todos los eventos de un partido y actualizar las estadísticas correspondientes.

Dentro de cada entrada se guarda:

- Clave de Firebase: identificador único formado por el identificador del equipo, formado a su vez por el identificador de la temporada y de su administrador, más una clave única de partido.
    - matchDate: fecha del partido.
    - matchName: nombre que queramos ponerle (Jornada 1, Match 1, etc).
    - matchID: identificador de Firebase del partido utilizado para búsquedas internas.

  A continuación, se detallan los campos estadísticos del partido:
    - matchAssists: lista de identificadores de usuarios que han realizado una asistencia.
    - matchGoalsFor: lista de pares "minuto-identificador de jugador" de los goles a favor del partido.
    - matchGoalsAgainst: lista de minutos en los que se han recibido goles en contra.
    - matchPenaltiesMissed: lista de identificadores de usuarios que han fallado un penalti.
    - matchPenaltiesSaved: lista de identificadores de usuarios que han parado un penalti (porteros).
    - matchPenaltiesProv: lista de identificadores de usuarios que han provocado un penalti.
    - matchStartingEleven: lista de identificadores de usuarios que han formado parte del once inicial.
    - matchSubs: lista de objeto "Sub", que contiene a su vez:
        - subIDin: identificador del jugador que entra al campo
        - subIDout: identificador del jugador que sale del campo
        - min: minuto en el que se efectúa la sustitución

Durante la primera fase del desarrollo, los datos se cargan desde memoria para facilitar las pruebas. Esta simulación permite trabajar con estructuras similares a las reales sin necesidad de conectarse a Firebase desde el inicio. Gracias a este diseño modular del sistema y, en particular, al uso del módulo DataManager, se puede hacer la transición a Firebase de forma sencilla y sin necesidad de modificar el resto de la lógica de la aplicación, más allá de la introducción de datos posterior. El DataManager es responsable de centralizar toda la carga de datos, desde Firebase, y facilita tanto la persistencia como la sincronización con la interfaz, ya que cada vez que se realiza un cambio, se hace un "data reload" para obtener todos los nuevos datos. Este enfoque permite mantener una versión funcional y estable desde el principio del desarrollo.

En la Figura 18 podemos ver cómo se han estructurado los diferentes objetos de la aplicación.

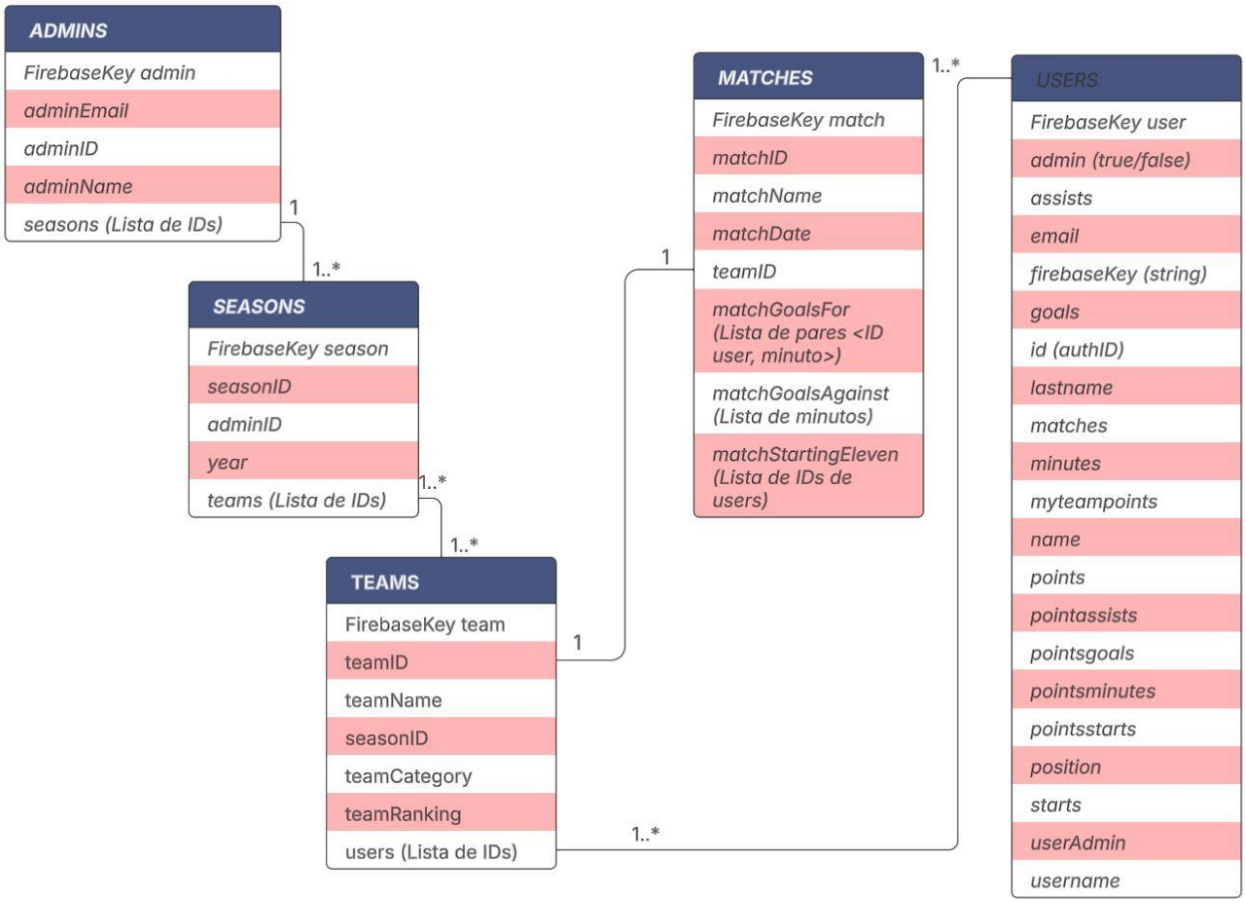


Figura 18 : Modelo de Datos

5.2.6 Firebase Authentication

Además del almacenamiento de datos, Firebase también ofrece un sistema de autenticación que se utiliza en la aplicación para gestionar el acceso de los usuarios. Firebase Authentication permite registrar e iniciar sesión con correo y contraseña, de forma segura y sencilla, sin necesidad de implementar un sistema propio desde cero.

Cada vez que un usuario inicia sesión, Firebase devuelve un identificador único (UID) que se conserva durante toda la sesión. Este identificador sirve para vincular al usuario con sus datos dentro de la base de datos y controlar su acceso a determinadas funciones, como por ejemplo, mostrar el botón de administración solo si el usuario autenticado tiene permisos, es decir, si es administrador.

5.3 Desarrollo del sistema principal y navegación

El sistema de navegación principal de la aplicación se estructura en torno a una actividad principal, "LobbyActivity.java", que actúa como contenedor de múltiples fragmentos, cada uno correspondiente a una sección funcional distinta. La navegación entre estas secciones se gestiona a través de una barra de navegación inferior (BottomNavigationView[21]), que está conectada al sistema de navegación de Android mediante el archivo "mobile_navigation.xml".

5.3.1 Pantalla de acceso (Login)

Este fragmento supone el punto de entrada al sistema. Utiliza Firebase Authentication para gestionar el acceso de los usuarios registrados. Al completar el inicio de sesión, se lanza "LobbyActivity.java".

5.3.2 Pantalla principal (LobbyActivity)

Es el contenedor general del sistema tras el Login. Utiliza un View Binding[22] para enlazar el layout principal. Contiene un Toolbar[23] personalizado con título dinámico y botón de acceso a la zona de administrador, cuya visibilidad se controla con el DataManager y la obtención de los permisos para ese usuario.

Además, integra una barra de navegación inferior (BottomNavigationView) y la gestiona con un NavController[24]. Esto hace que todos los fragmentos estén encapsulados visualmente en este Lobby y mantengan la estructura de la barra de navegación y el fragmento interior.

El sistema de navegación XML define los destinos posibles dentro de la aplicación. Cada destino representa un fragmento (ver Figura 19), y está vinculado a un layout y una clase.

```
<fragment
    android:id="@+id/navigation_standings"
    android:name="es.ucm.fdi.clubrivals.ui.Standings.StandingsFragment"
    android:label="Standings"
    tools:layout="@layout/fragment_standings" />
```

Figura 19: Inicialización de cada fragmento en el sistema de navegación

La estructura de navegación incluye los siguientes fragmentos:

- Home: muestra resumen general y clasificación por defecto, aparte de información resumida del usuario que ha iniciado sesión.
- Standings: consulta de estadísticas por pestañas
- MyTeam: reservada para la funcionalidad MyTeam
- Profile: información relativa al usuario

El NavController implementado en LobbyActivity permite el cambio de fragmentos al pulsar los botones del menú inferior. Cada fragmento mantiene su estado al cambiar de sección. Se usa un listener para actualizar dinámicamente el título del Toolbar en función del fragmento activo.

Algunas ventajas de este diseño modular, por ejemplo, son las relacionadas con la mantenibilidad y escalabilidad del sistema. LobbyActivity actúa como espacio principal sin acoplarse a la lógica de cada sección.

Los datos se comparten entre fragmentos mediante ViewModels o argumentos de navegación.

5.4 Implementación de componentes clave

Podemos encontrar los detalles de implementación de los fragmentos principales en el Apéndice 1.

5.5 Retos técnicos durante el desarrollo

Durante el desarrollo de la aplicación, los retos más importantes no aparecen tanto en el uso de la base de datos como tal, sino cómo se gestiona y actualiza la información una vez que está cargada en memoria.

Uno de los principales problemas surge a la hora de mantener los datos sincronizados en tiempo real dentro de los fragmentos que observan los LiveData[25], sobre todo cuando hay cambios frecuentes o múltiples vistas dependen de la misma información. Actualizar correctamente estas vistas al modificar un partido, añadir un usuario o simplemente cambiar el valor de un Spinner[26], implica asegurar que todos los LiveData afectados se actualizan, se notifican y reciben los cambios. Esto requiere controlar bien el ciclo de vida de cada ViewModel, forzar recargar o lanzar manualmente eventos de actualización, ya que si no se hace de forma ordenada, es fácil que la vista muestre información desactualizada aunque los datos en Firebase o incluso ya en la memoria de ejecución, sean correctos.

Otro reto destacado está en la parte de edición de partidos. Aunque crear un partido desde cero resulta relativamente sencillo, elaborar una vista que permita editar un partido existente de forma visual, organizada y sin errores, y que luego actualice la base de datos, es bastante más complejo. Es necesario cargar todos los eventos anteriores (goles, asistencias, cambios, once inicial…), representarlos correctamente, permitir que se modifiquen o eliminen por separado, y luego guardar los nuevos datos sin que se acumulen o mezclen con los antiguos. Además, se deben revertir correctamente las estadísticas aplicadas al usuario en el partido original antes de guardar los nuevos eventos, lo que añade una capa extra de lógica y validaciones.

Estos retos hacen que la parte de actualización y edición en memoria sea más compleja que el resto.

Capítulo 6 - Presentación de la aplicación

En este capítulo se describe el funcionamiento general de la aplicación desde el punto de vista del usuario, ya sea jugador o administrador. Se detalla el propósito y la lógica de cada una de las principales pantallas que componen la interfaz: la pantalla de inicio (Home), la sección de clasificación (Standings), el perfil de usuario (Profile) y la vista del administrador. Finalmente, no se ha implementado la funcionalidad MyTeam en esta primera instancia por lo que no se enseñará.

A lo largo de cada sección se explica cómo se estructura la información, qué acciones puede realizar el usuario y cómo interactúan entre sí los distintos componentes de la aplicación. De esta manera, podemos entender cómo un usuario real puede hacer uso de la aplicación.

6.1 Pantalla Login / Registro

La pantalla Login es la primera vista que aparece al iniciar la aplicación. Su función principal es permitir el acceso de los usuarios registrados mediante el inicio de sesión con correo electrónico y contraseña. Para ello, se muestran dos campos de texto donde se deben introducir ambos datos obligatoriamente (ver Figura 20).

Una vez rellenados, se pulsa el botón de acceso y, si las credenciales son correctas, se accede a la pantalla principal de la aplicación. En caso de error, se muestra un mensaje informativo indicando que el correo o la contraseña no son válidos.

Esta pantalla está conectada con el sistema de autenticación de Firebase, por lo que el inicio de sesión se valida de forma segura y automática. Sólo pueden acceder los usuarios que ya han sido registrados previamente por un administrador dentro de la aplicación.

También existe la opción de registrarse como un administrador (ver Figura 21). Para ello, habrá que pulsar sobre el botón de registro y se abrirá una ventana para registrarse como administrador. Se deberá introducir el nombre del administrador, un correo que todavía no esté inscrito en la aplicación (ni como administrador ni como jugador), y un código que será proporcionado por el desarrollador. Si se cumplen todas

las condiciones, se creará una cuenta de administrador que podrá gestionar nuevos equipos y temporadas.

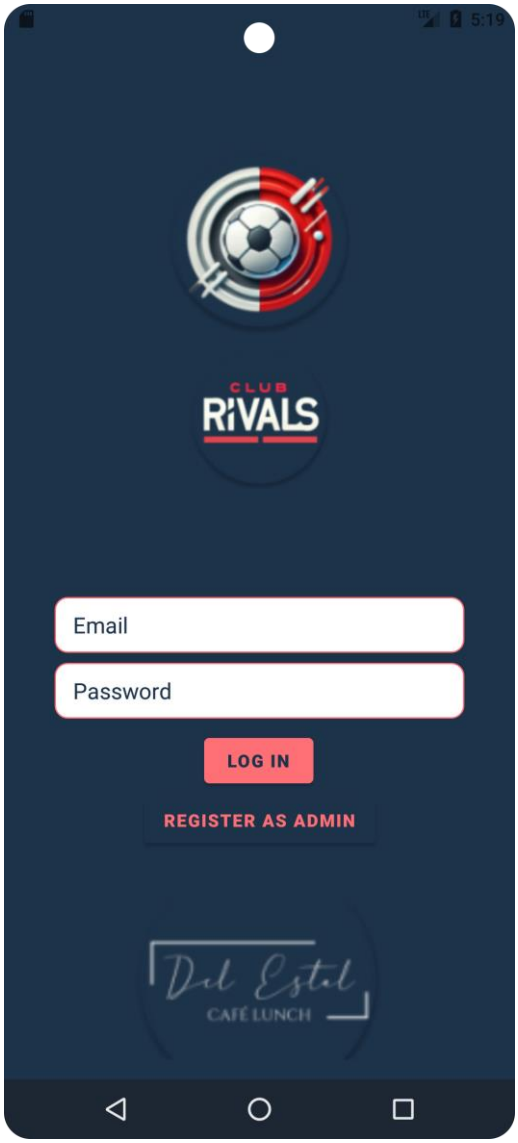


Figura 20: Captura de pantalla Login

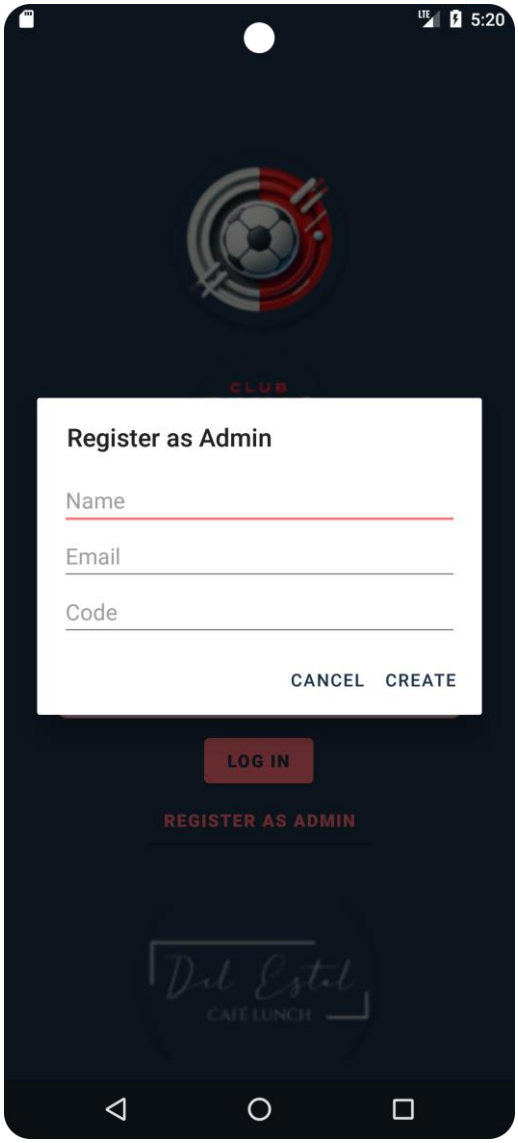


Figura 21: Captura de pantalla de registro de nuevo administrador

Para la creación de usuarios jugadores, será el administrador el que, al crear cada entrada de jugador en el equipo, introduzca su correo y una contraseña temporal. Esto creará directamente la cuenta en Firebase Authenticator y a su vez, el nuevo User en la base de datos y su correspondiente entrada en el equipo para el que está siendo

creado. Posteriormente será el propio usuario el encargado de modificar algunos de sus datos (no todos serán editables), así como la contraseña.

6.2 Pantalla Home (Inicio)

La pantalla de inicio presenta al usuario una vista previa de los datos de sus equipos y su perfil (ver Figura 22).

En primer lugar, se muestra una tabla con la clasificación general (Standings) del equipo seleccionado, ordenada por puntos de mayor a menor. Esta tabla corresponde únicamente a la clasificación por puntos y ofrece una visión rápida del rendimiento de los jugadores dentro del equipo. Para que se muestre la información deseada, aunque habrá preseleccionados unos datos por defecto, se debe seleccionar la temporada y el equipo en los spinners situados en la parte superior.

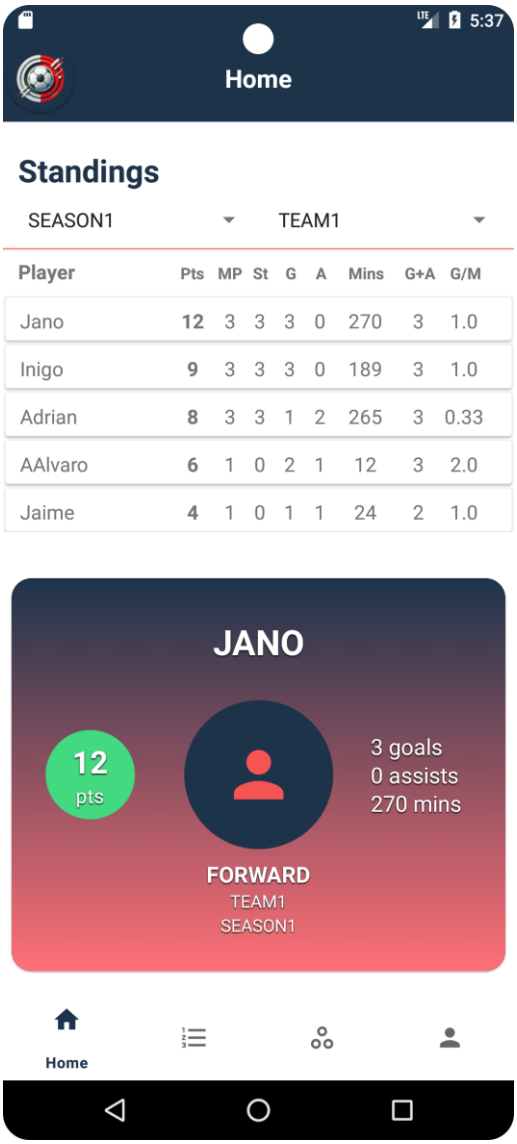


Figura 22 : Captura de pantalla inicial de la aplicación

Los spinners permiten elegir entre todas las temporadas disponibles del administrador asociado al usuario que ha iniciado sesión. Además, si el usuario no participa en alguna de estas temporadas, no aparecerá como opción seleccionable para simplificar los datos. Al seleccionar una temporada, el segundo spinner se actualiza automáticamente para mostrar sólo los equipos pertenecientes a dicha temporada. Una vez seleccionados ambos campos, se actualiza la tabla de clasificación y los datos del jugador.

Debajo de la tabla, se muestra una vista previa del rendimiento del propio jugador autenticado. Esta sección incluye el nombre, la posición y las estadísticas más destacadas para la temporada y equipo seleccionados. Esta información permite al usuario consultar de forma rápida sus datos más relevantes en la competición y compararse con el resto de jugadores de su equipo, aunque también aparezca en la tabla de clasificación.

6.3 Pantalla Standings

La pantalla Standings permite consultar distintas clasificaciones de los jugadores del equipo, ordenadas por estadísticas diferentes. Al igual que en la pantalla Home, en la parte superior se encuentran dos spinners que permiten seleccionar la temporada y el equipo deseado. Se cargan automáticamente todas las temporadas disponibles del administrador del usuario autenticado y, al elegir una temporada, el segundo spinner se actualiza para mostrar los equipos asociados a esa temporada. Una vez seleccionados ambos campos, se cargan las estadísticas correspondientes a ese equipo y temporada.

Cada clasificación se muestra en una pestaña diferente, y se puede navegar entre ellas deslizando horizontalmente o pulsando en el nombre de la pestaña.

A continuación, se explica el contenido de cada una.

6.3.1 General

En esta pestaña se muestra la clasificación general de los jugadores del equipo, ordenada por puntos de mayor a menor. Los puntos se calculan a partir del rendimiento del jugador en los partidos: goles, asistencias, titularidades, minutos jugados, etc. En cada fila aparece el nombre del jugador y el número total de puntos acumulados, que es el dato a través del cual se ordena en primera instancia. Además, se muestran otros datos de carácter general como las asistencias o los minutos jugados (ver Figura 23).

Figura 23: Captura de pantalla Standings General

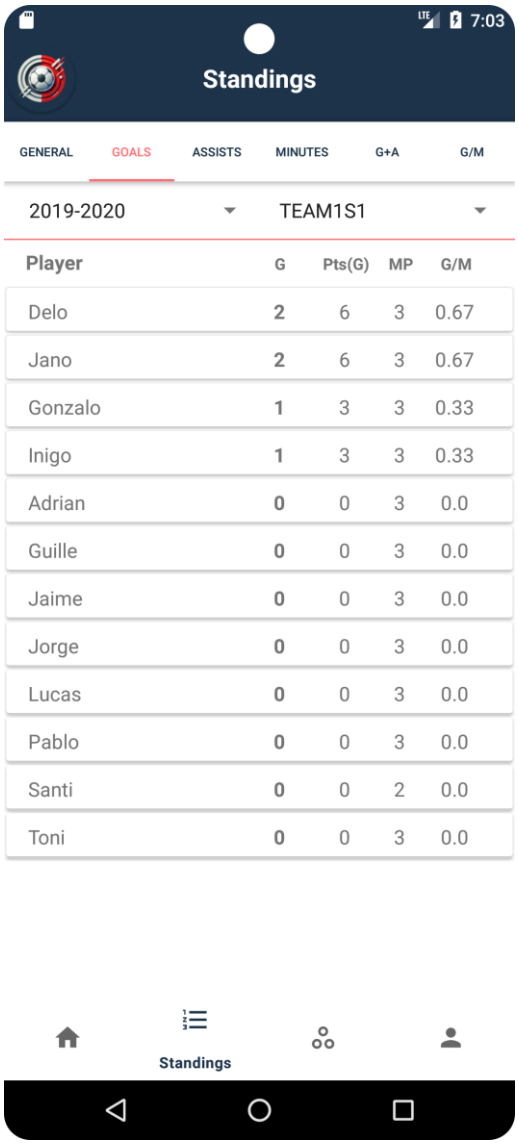


Figura 24: Captura de pantalla Standings Goals

6.3.2 Goals

En esta sección se muestra una clasificación con información relativa a los goles. La tabla se ordena de mayor a menor según la cantidad de goles anotados. Junto al nombre del jugador, se indica el número total de goles, los puntos generados por esos goles, los partidos jugados y la métrica goles por partido. Es útil para ver quiénes son los jugadores más goleadores del equipo (ver Figura 24).

6.3.3 Assists

Aquí se presenta la clasificación de los jugadores según el número de asistencias realizadas. Al igual que en la pestaña de goles, los datos se ordenan de mayor a menor. Se muestra el nombre, el número de asistencias, los puntos generados por asistencias, los partidos jugados y la métrica de asistencias por partido. Permite identificar a los jugadores más participativos en creación de jugadas (ver Figura 25).

Figura 25: Captura de pantalla Standings Assists

Figura 26: Captura de pantalla Standings Minutes

6.3.4 Minutes

Esta pestaña refleja el total de minutos jugados por cada jugador a lo largo de la temporada. La clasificación se ordena también de mayor a menor, mostrando primero a los jugadores con más participación en el campo. Junto al nombre, se muestran los minutos jugados, los puntos generados por minutos jugados, el número de partidos jugados y el promedio de minutos por partido. Es útil para saber cuáles son los jugadores más utilizados por el entrenador (ver Figura 26).

6.3.5 Goals and Assists

Esta pestaña muestra la suma total de goles y asistencias de cada jugador. Se ordena de mayor a menor según la suma de ambas estadísticas. Permite ver qué jugadores han contribuido directamente en más acciones ofensivas. Se muestra la suma total, los goles, asistencias, y los puntos obtenidos por ambas (ver Figura 27).

Figura 27: Captura de pantalla Standings Goals and Assists

Figura 28: Captura de pantalla Standings Goals per Match

6.3.6 Goals per Match

En esta sección se calcula el promedio de goles por partido de cada jugador. Se obtiene dividiendo el número total de goles entre la cantidad de partidos jugados. La tabla se ordena de mayor a menor en función de ese promedio, mostrando a los jugadores con mayor eficacia goleadora. También se muestra el número total de goles y los partidos jugados para dar contexto al dato (ver Figura 28).

Esta pantalla Standings ofrece una vista completa y comparativa de las estadísticas individuales de los jugadores, ayudando a analizar el rendimiento en detalle desde distintas perspectivas.

6.4 Pantalla MyTeam

La pantalla MyTeam permite al usuario consultar los partidos de los equipos en los que participa. En la parte superior se incluyen dos desplegables (season y team), que funcionan de forma similar a los de otras secciones: el primero carga todas las temporadas disponibles para el administrador asociado al usuario autenticado, en las que el jugador ha participado porque forma parte de un equipo, y el segundo muestra los equipos correspondientes a la temporada seleccionada. Una vez definidos estos valores, se carga la lista de partido asociados al equipo elegido.

Por un lado, se muestra la información relativa al equipo seleccionado. Justo debajo, se pueden visualizar los partidos en formato lista, donde cada elemento corresponde a un encuentro disputado (ver Figura 29). Al hacer click sobre cualquiera de ellos, se abre una segunda vista dedicada al detalle del partido. En esta vista de detalle se muestra la información completa del encuentro, incluyendo sus datos principales y las estadísticas registradas (ver Figura 30).

Esta pantalla está diseñada para ofrecer al usuario una visión detallada de los partidos de sus equipos, permitiendo tanto una visión general de todos los encuentros como una exploración detallada de cada uno en particular.

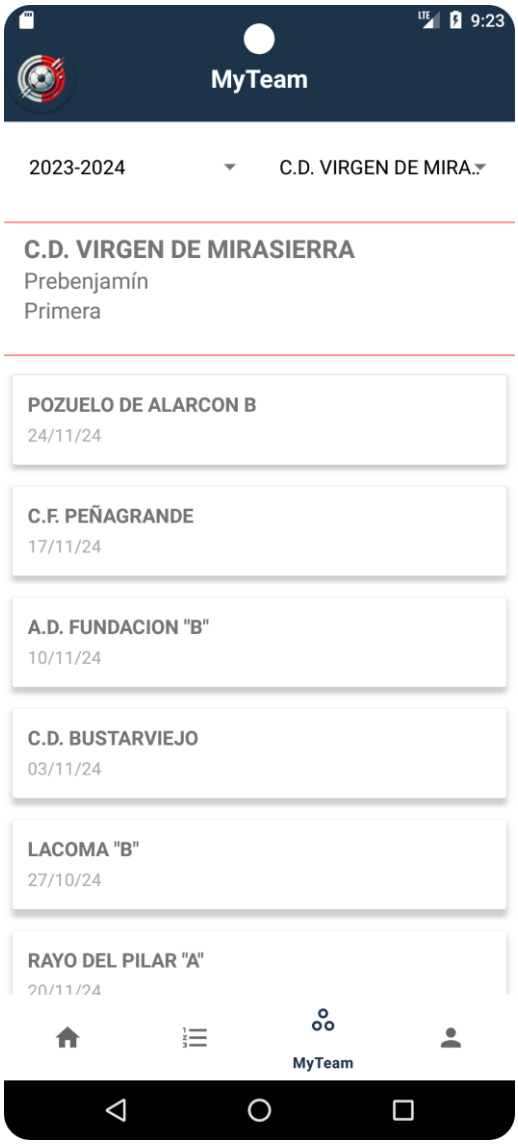


Figura 29 : Captura de pantalla MyTeam – Vista general

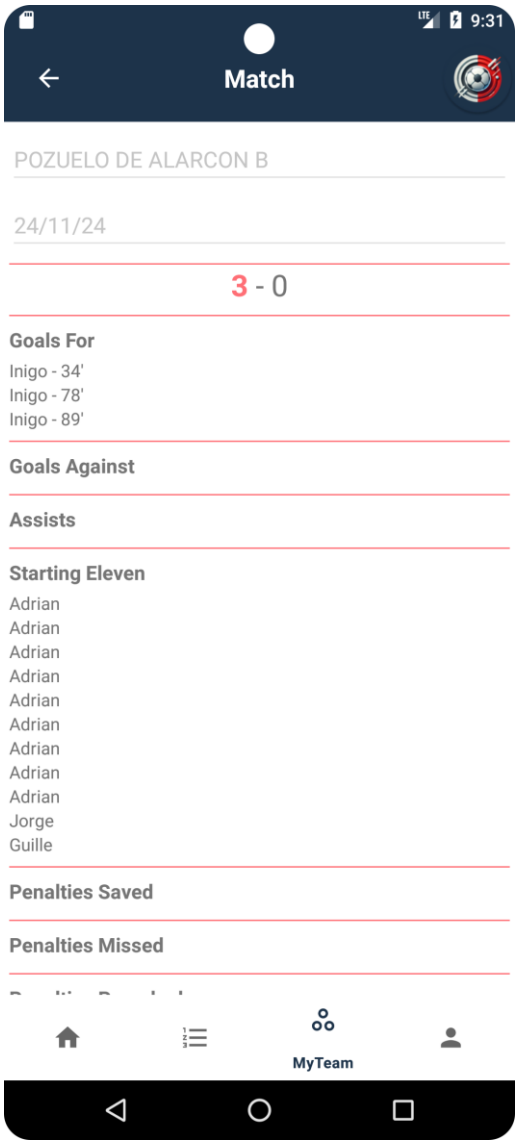


Figura 30 : Captura de pantalla MyTeam - Vista de partido

6.5 Pantalla Profile

La pantalla Profile (ver Figura 31) permite al usuario consultar sus propios datos y estadísticas personales dentro de un equipo. En la parte superior se muestra el nombre del jugador. Justo debajo, igual que en otras secciones, se encuentran los spinners para seleccionar la temporada y el equipo. Se cargan todas las temporadas del administrador del usuario autenticado, y al seleccionar una de ellas, el segundo desplegable muestra los equipos asociados a esa temporada. Se cargan las estadísticas correspondientes según los valores seleccionados en los spinners.

La parte inferior de la pantalla muestra una tabla similar a la de la sección de clasificación, pero en este caso filtrada únicamente para mostrar los datos del jugador autenticado. Se pueden consultar distintas pestañas con estadísticas personales. Para ser más exactos, se pueden consultar las mismas tablas que en la vista Standings.

Además de los datos de la tabla, se incluye también un gráfico visual donde se representan los goles, asistencias y minutos del jugador, que son las estadísticas más importantes. Esto permite tener una referencia rápida del rendimiento del jugador durante la temporada.

Esta pantalla está pensada para que cada usuario pueda hacer un seguimiento individual de su propio progreso entre las diferentes temporadas, de forma clara y visual.

Además, el usuario puede cambiar algunos de sus datos, como son el nombre para todos los equipos de los que forma parte, y la contraseña con la que se autentica en el sistema (ver Figura 32).

Figura 31 : Captura de pantalla Profile

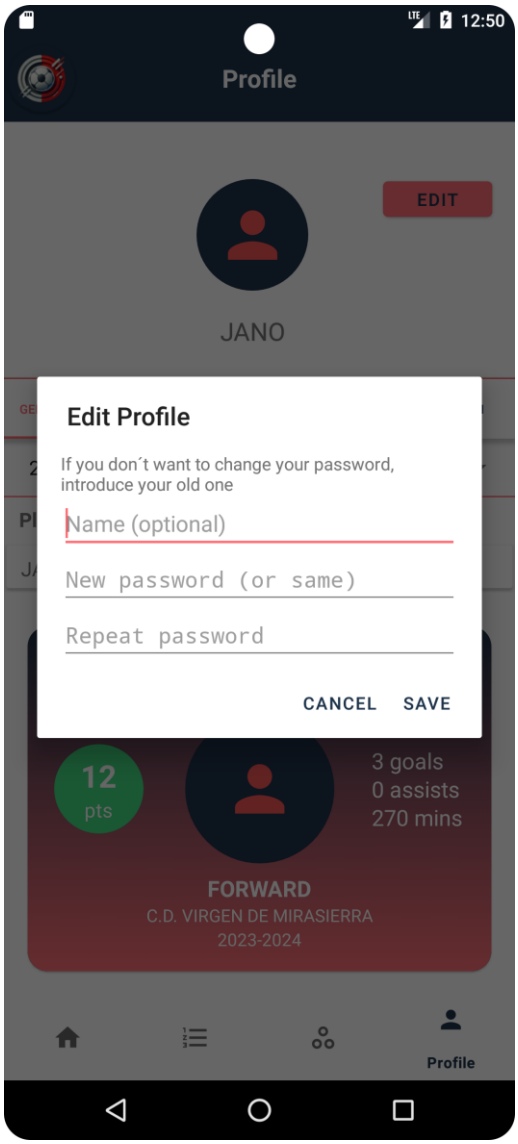


Figura 32 : Captura de pantalla Profile (Editar datos)

6.6 Pantalla Administrador

El acceso a la pantalla del administrador está habilitado únicamente para los usuarios con permisos de administración. Se accede desde un botón ubicado en la parte superior de la pantalla principal que sólo se mostrará a este tipo de usuarios (ver Figura 33). Al pulsarlo, se abre una vista en la que se muestra la lista de temporadas gestionadas por ese administrador, junto con un botón para añadir una nueva

temporada. Si todavía no se ha creado ninguna, la lista aparecerá vacía y sólo se mostrará el botón de creación.

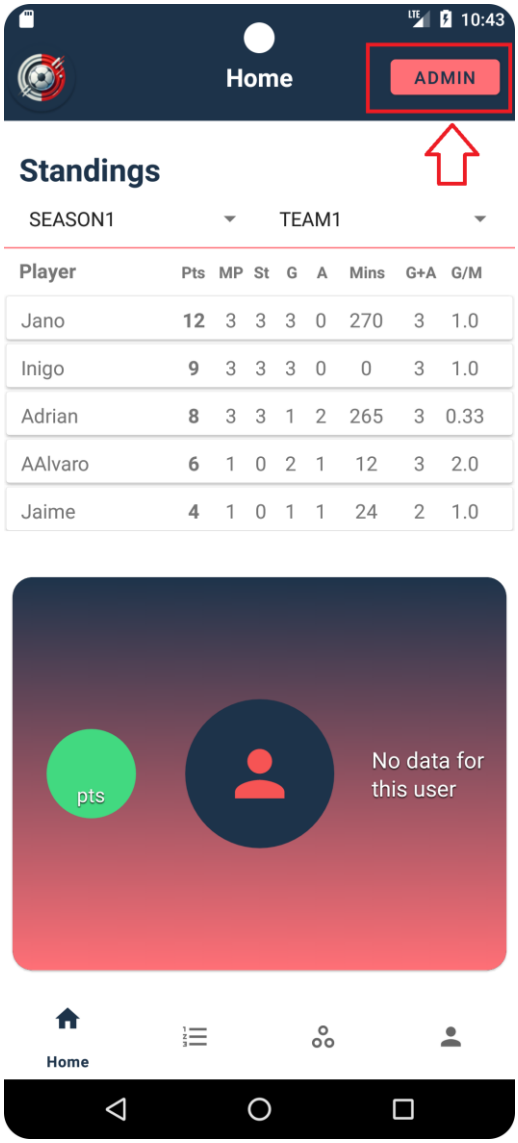


Figura 33 : Captura de pantalla de selector de modo Administrador

Figura 34: Captura de pantalla inicial del modo Administrador

Si queremos crear una temporada, podemos hacer click en el botón "+" y nos aparecerá un cuadro de diálogo para crearla. En caso contrario, podemos pinchar en una de ellas y acceder a la información de la misma (ver Figuras 34, 35 y 36).

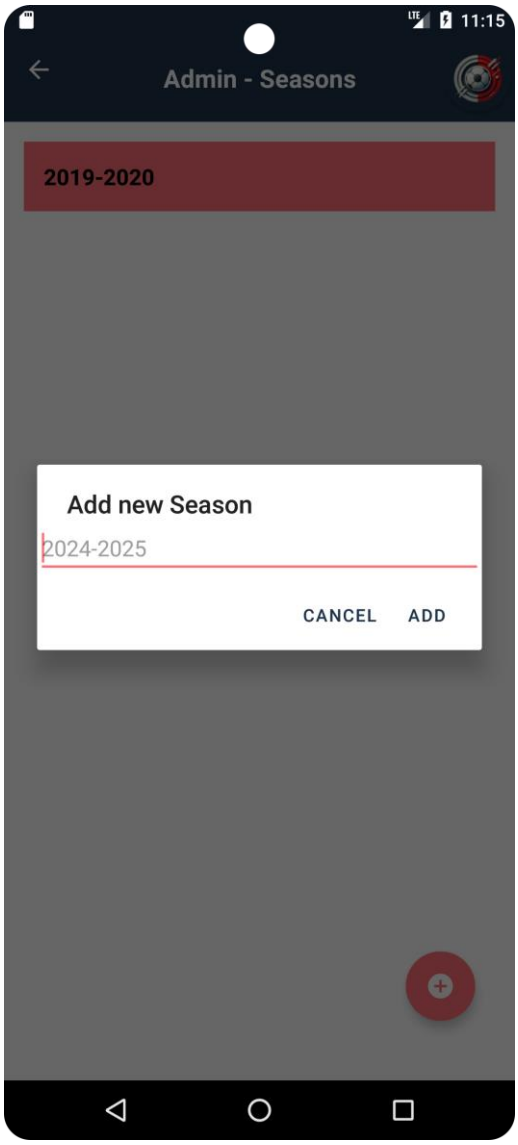


Figura 35: Captura de pantalla de creación de nueva Season

Figura 36: Captura de pantalla de lista de Teams

Como vemos en la Figura 36, al seleccionar una temporada de la lista, se accede a una nueva pantalla en la que se muestran los equipos asociados a dicha temporada. En la parte inferior aparece el botón para añadir un nuevo equipo, y justo encima se

muestra la lista de equipos ya existentes. Si no hay ningún equipo creado, simplemente se mostrará el botón de añadir.

Si creamos un nuevo equipo, debemos introducir algunos datos informativos, y ya podremos pasar a introducir los usuarios y partidos de ese equipo (ver Figura 37).

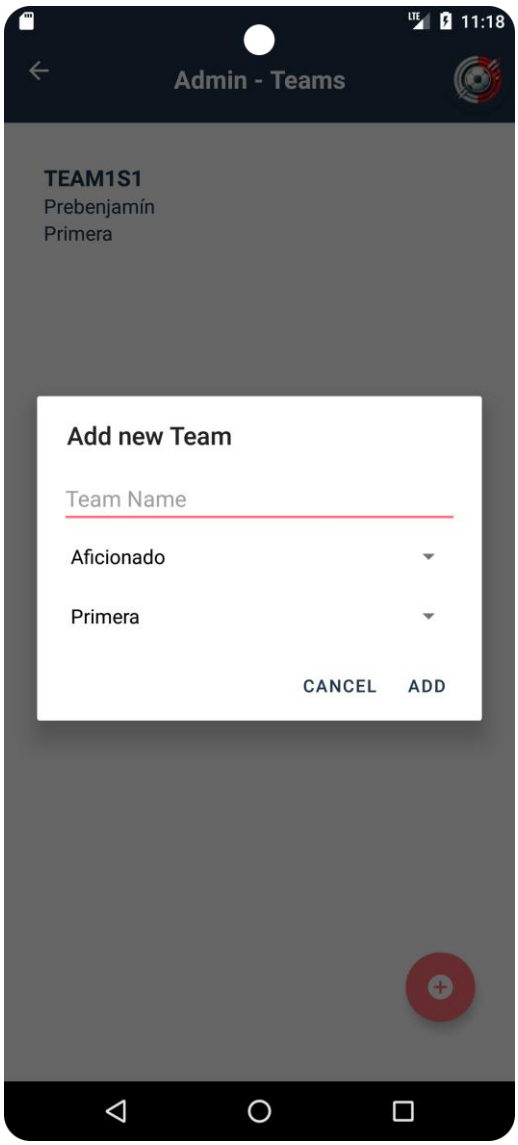


Figura 37: Captura de pantalla de creación de nuevo Team

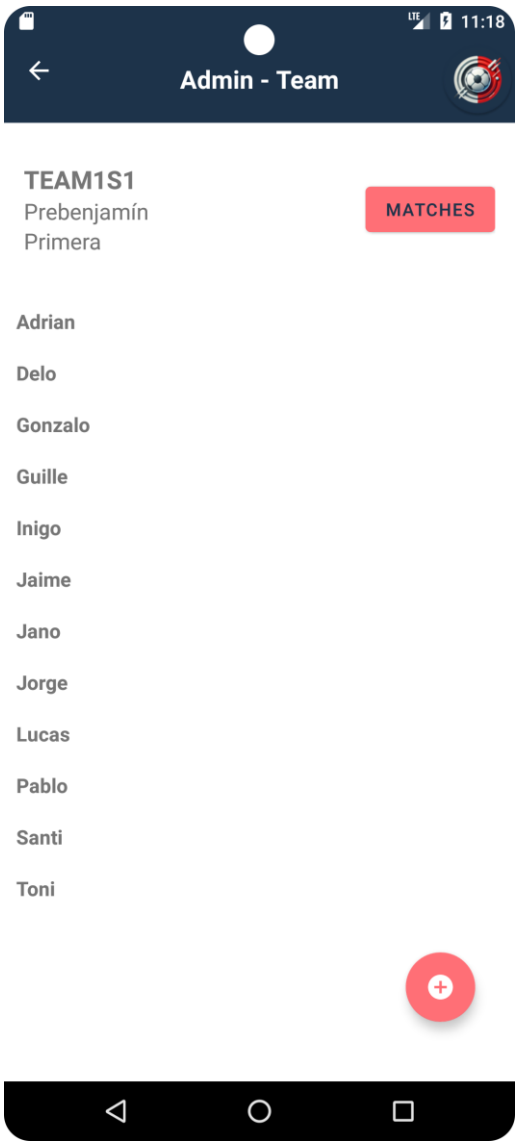


Figura 38: Captura de pantalla de información de equipo

Al acceder al equipo, se abre una vista con información básica del equipo, un botón para ir a la sección de partidos (Matches) y una lista de los jugadores que lo

forman. Esta lista puede estar vacía si no se ha añadido ningún jugador todavía. Desde esta misma pantalla se puede pulsar un botón para añadir nuevos jugadores al equipo (ver Figura 38).

Para añadir un nuevo jugador habrá que pulsar en el botón de añadir y se abrirá un cuadro de diálogo como el de la Figura 39, donde habrá que rellenar los datos identificativos y los deportivos. Si el administrador que va a introducir un nuevo usuario intenta crear un usuario con un correo que ya existe para un usuario en otro administrador, dará un error. Sólo se pueden crear usuarios con el mismo correo (y por tanto mismo usuario autenticado) para seasons y teams del mismo administrador. El usuario podrá cambiar su contraseña posteriormente.

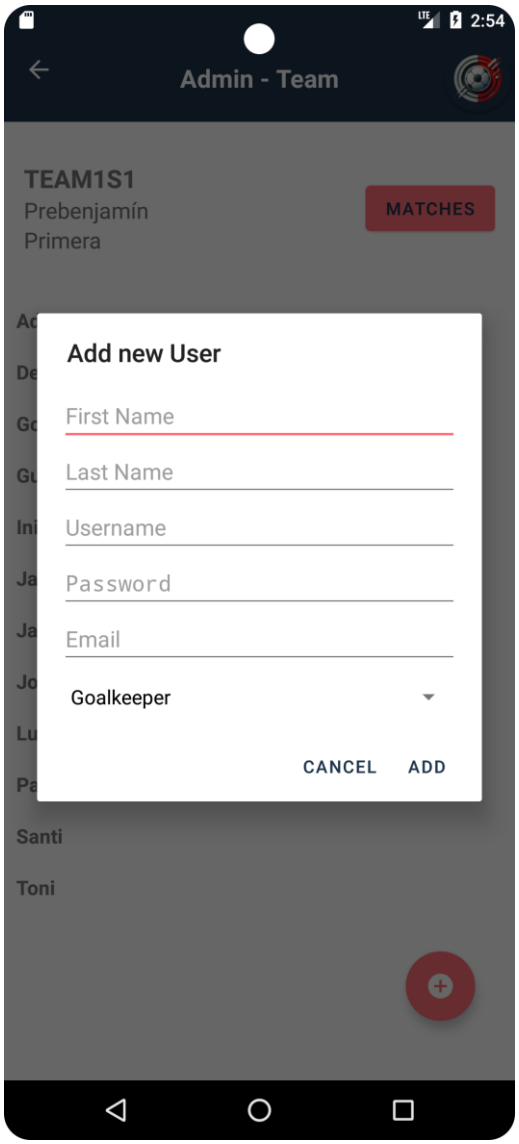


Figura 39: Captura de pantalla de creación de nuevo User

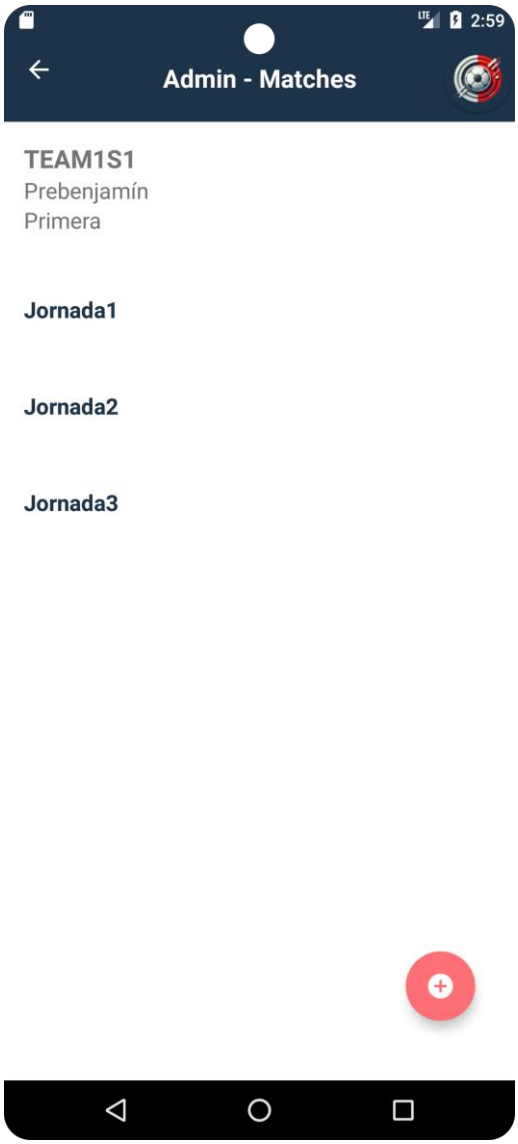


Figura 40: Captura de pantalla de los partidos de un equipo

Una vez se hayan creado los usuarios, podemos pasar a crear o editar los partidos. Para ello, pulsamos en el botón de partidos (botón Matches en Figura 38), y se abre una nueva pantalla en la que se muestra de nuevo la información general del equipo y la lista de partidos asociados, en caso de que haya. Si no, aparecerá vacía. En cualquier caso, se dispone un botón para poder crear nuevos partidos (ver Figura 40).

Al crear un partido, se accede a una vista donde es necesario introducir los datos identificativos del encuentro y registrar todos los eventos relevantes que puedan hacer puntuar a los jugadores, que se pueden añadir mediante botones específicos que permiten construir cada evento (ver Figura 41). Algunos necesitan varios datos y otros sólo necesitan uno. Una vez completado, se pulsa el botón "Create Match" y el partido se guarda en ejecución y en base de datos. Por ejemplo, para añadir un gol (ver Figura 42), se necesita introducir el jugador que lo ha metido, y el minuto en el que ha sido.

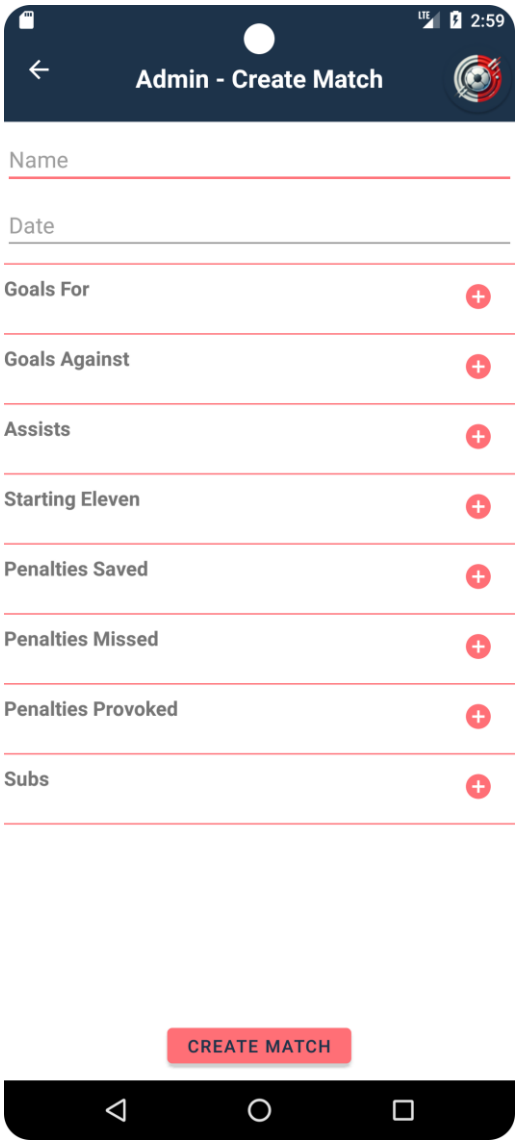


Figura 41: Captura de pantalla de creación de nuevo Match

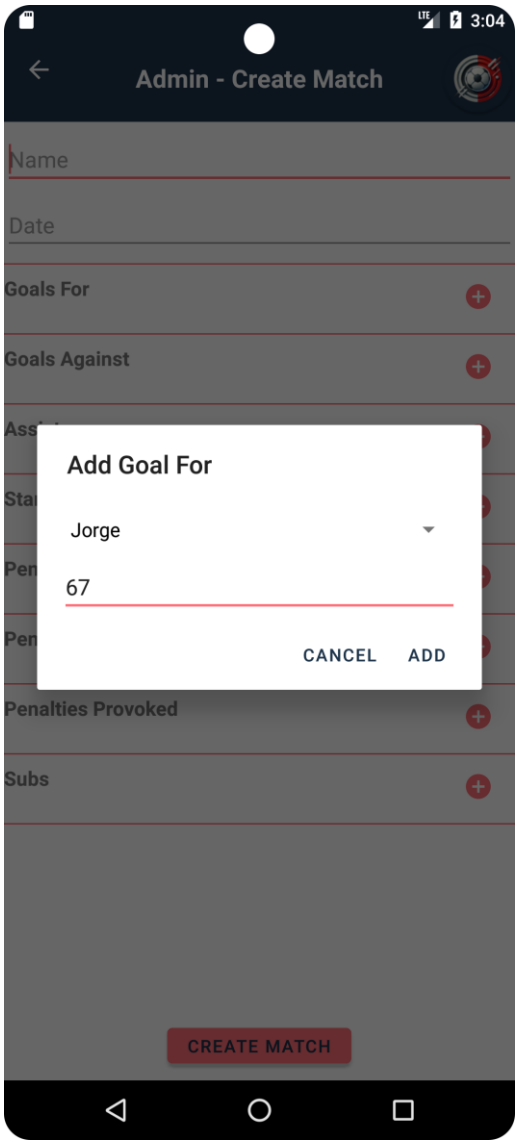


Figura 42: Captura de pantalla de registro de datos de un partido

Una vez añadido cada evento, irá apareciendo en la vista. Algunos datos como los minutos o los jugadores están restringidos. Por ejemplo, no se puede seleccionar un minuto menor que el 0 o mayor que el 100 (90 más descuento, ya que en partidos de liga nunca hay prórrogas). Tampoco se pueden seleccionar jugadores de otro equipo, o crear un once inicial donde haya jugadores repetidos, o crear una sustitución donde

el jugador que entra sea el mismo que el que sale. En las siguientes capturas se muestra un partido a punto de ser creado (Figuras 43 y 44).

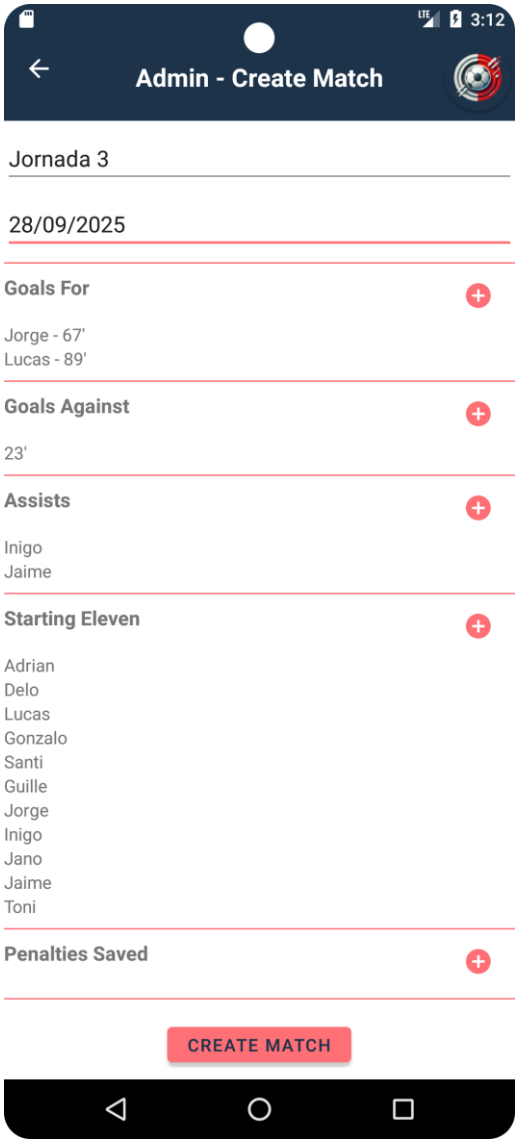


Figura 43: Captura de pantalla de creación de nuevo Match

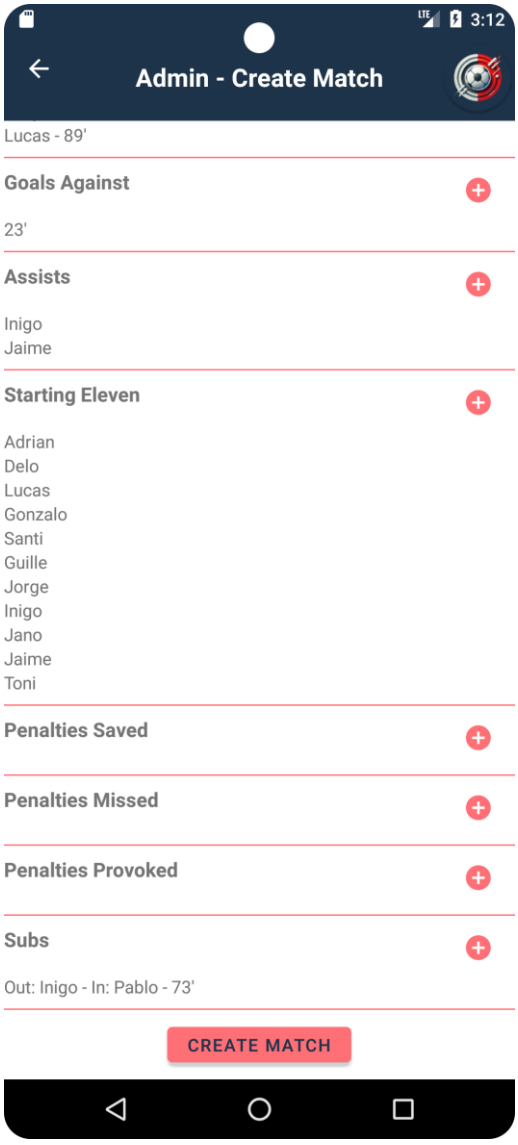


Figura 44: Captura de pantalla de registro de datos de un partido

Desde la lista de partidos, al pulsar sobre cualquiera de ellos, se abre una vista con la información completa del encuentro. En esta pantalla se ofrecen dos opciones, eliminar o editar el partido.

- Si se pulsa eliminar, se abre un cuadro de confirmación y, si se acepta, el partido se elimina por completo, incluyendo todos los eventos y estadísticas asociadas, también en los usuarios en los que fueron añadidas (ver Figura 45).

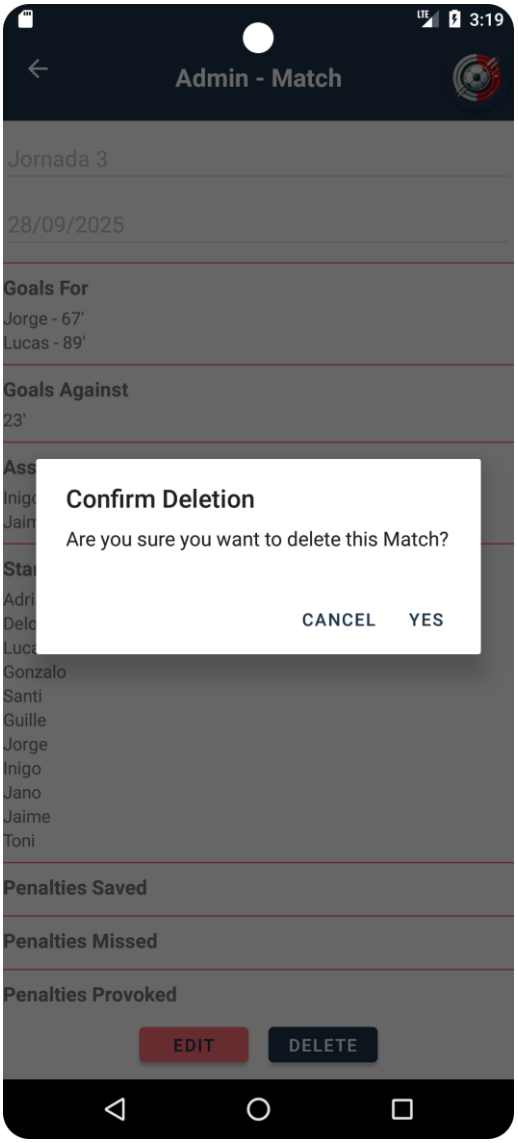


Figura 45: Captura de pantalla de eliminar partido

- Si se selecciona la opción de editar, se accede a una vista similar a la de creación, pero con los datos precargados. En esta vista se puede modificar cualquier dato identificativo del partido y, además, se pueden restablecer (con el botón del icono de la basura) y volver a introducir las listas de eventos como goles, asistencias o cambios. Una vez realizada la edición, se pulsa "Save" y los cambios quedan aplicados (ver Figura 46).

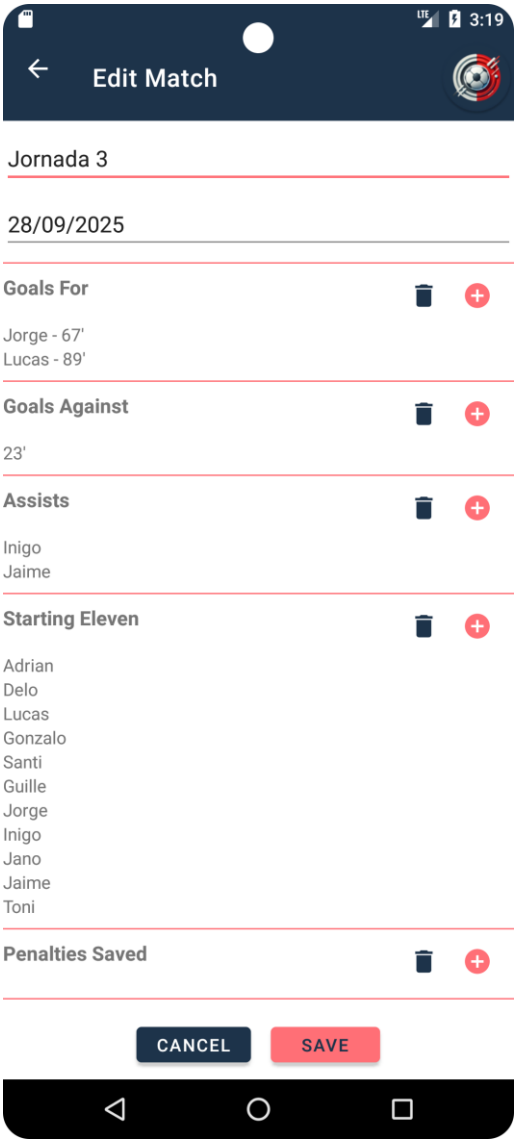


Figura 46: Captura de pantalla de edición de partido

Esta sección "Admin", permite a los usuarios administradores tener un control total sobre la estructura de la competición: temporadas, equipos, usuarios y partidos, gestionando de forma sencilla toda la información desde la propia aplicación. Los permisos están bien definidos en la base de datos para que sólo este tipo de usuarios puedan hacer modificaciones sobre la misma.

Capítulo 7 - Evaluación y validación del sistema

El presente capítulo tiene como objetivo evaluar y validar el sistema desarrollado, comprobando que cumple con los requisitos establecidos y ofrece una experiencia satisfactoria a los usuarios. Para ello, se han llevado a cabo diferentes tipos de pruebas que permiten verificar el correcto funcionamiento de las funcionalidades implementadas, la coherencia de los datos y la usabilidad general de la aplicación.

En este capítulo se muestran las pruebas realizadas para comprobar que la aplicación funciona como debe y cumple con los requisitos establecidos. El objetivo es verificar que las funcionalidades principales se ejecutan correctamente, que los datos son coherentes y se guardan y visualizan correctamente, y que la experiencia de uso resulta clara y sencilla para los usuarios.

Primero se presentan las pruebas funcionales, donde se revisa que las acciones básicas de la aplicación (crear temporadas, añadir equipos y usuarios, registrar partidos o consultar estadísticas), se realizan sin errores. Después se describe la comprobación del flujo de datos, asegurando que la información introducida queda almacenada correctamente en la base de datos y aparece de forma coherente en las pantallas correspondientes.

A continuación, se explican las pruebas de usabilidad con usuarios reales, cuyo propósito es medir la facilidad de uso, la claridad de la interfaz y el grado de satisfacción durante la interacción con el sistema. Finalmente, se recogen los problemas detectados durante las pruebas, así como las soluciones aplicadas para resolverlos y garantizar la estabilidad y fiabilidad de la aplicación.

7.1 Pruebas funcionales

Metodología utilizada

Las pruebas se han ido realizando a medida que se desarrollaban los diferentes módulos de la aplicación. Primero se hacían las pruebas unitarias sobre cada módulo para comprobar que funcionaba de forma independiente y, más adelante, se pasaba

a las pruebas de integración para asegurar que los nuevos módulos se conectaban correctamente con los que ya estaban implementados.

Casos de prueba

Los casos de prueba realizados corresponden directamente con los definidos en los casos de uso. En concreto, se han comprobado las funcionalidades de inicio de sesión, registro de administradores, todas las funcionalidades de administrador de creación de temporadas, equipos, usuarios y partidos, visualización de estadísticas personales, visualización de estadísticas generales y consulta de partidos.

Resultados

Los resultados obtenidos no siempre fueron los esperados. En varias ocasiones, los problemas detectados no estaban en la funcionalidad individual de cada módulo, sino en la integración entre ellos. Una vez identificados estos errores de conexión, se aplicaron las correcciones necesarias hasta lograr que todas las funcionalidades trabajasen de forma conjunta y cumpliesen con los requisitos establecidos.

7.2 Comprobación del flujo de datos (introducción y visualización)

Entrada de datos

Para comprobar la correcta entrada de datos en el sistema, se han realizado distintas comprobaciones antes de guardar los mismos en el sistema, para que sean correctos o en algunos casos coherentes. Empezando por la comprobación de los datos de inicio de sesión, o la creación de los partidos, que no se pueden crear si un equipo no tiene once jugadores. Se comprueba también, cuando un administrador desea crear un nuevo usuario, que no haya uno existente con el mismo correo para otro administrador, lo que causaría un problema de base. Estas pruebas han permitido verificar que los datos introducidos son correctos.

Validación de almacenamiento

Una vez introducidos los datos, se comprobó que quedaban almacenados de forma correcta en la base de datos de Firebase. Se revisó que cada elemento (temporadas, equipos, usuarios y partidos) se guarda en su rama o ramas

correspondientes, con el tipo de dato correcto, manteniendo la coherencia con la estructura diseñada para la aplicación.

Validación de visualización

Una vez introducidos los nuevos registros en la base de datos, se verificó que los datos se mostraban de manera coherente en cada una de las vistas. Es decir, por una parte, que los datos en el sistema estuviesen ordenados correctamente según hubiesen sido creados en Firebase y, por otra, que al obtener los datos en los diferentes fragmentos, la información se mostrase correctamente en cada una de las vistas (por ejemplo que se mostrase el nombre del jugador y no su ID a pesar de que en el flujo del sistema se estuviese trabajando con identificadores).

Ejemplos concretos

Un ejemplo concreto y claro de este proceso es la introducción de un partido con varios goles. Al registrar los goleadores y los minutos, fue necesario comprobar que sus estadísticas personales aumentaban en consecuencia, además de los datos del propio partido, y que las diferentes tablas de clasificación reflejaban los cambios en las diferentes métricas del equipo. Con ejemplos de este tipo, se confirmó que el flujo de datos, desde la entrada inicial hasta la visualización final, funcionaba de manera correcta y completa.

7.3 Pruebas de usabilidad con usuarios reales

Metodología y descripción de tareas

Para llevar a cabo las pruebas de usabilidad se seleccionaron tres administradores y alrededor de doce usuarios normales. Los administradores representaban el perfil encargado de la gestión de temporadas, equipos, usuarios y partidos, mientras que los usuarios simulaban el papel de los jugadores dentro de la aplicación, utilizando las funcionalidades destinadas a la consulta de estadísticas.

El administrador debía registrarse, iniciar sesión, crear temporada, equipo y usuarios, y posteriormente crear uno o más partidos para ese equipo.

El usuario debía iniciar sesión, visualizar las estadísticas personales y generales de cada uno de los equipos de los que formaba parte, así como ir a comprobar que podía ver la información relativa a los partidos de cada uno de sus equipos.

Observaciones

Durante las pruebas intermedias se detectaron algunos problemas de carga en determinadas pantallas, lo que ralentizaba la experiencia de uso. Estos fallos se corrigieron implementando soluciones que mejoraron la fluidez general de la aplicación. Más allá de estos inconvenientes puntuales, los participantes encontraron la interfaz intuitiva y no tuvieron dificultades para completar las tareas, ya que la navegación resultó clara en la mayoría de los casos.

En general, la aplicación fue percibida como estable, funcional y agradable de utilizar, además de intuitiva, que era uno de los objetivos principales.

Mejoras propuestas

Algunos usuarios sugirieron la incorporación de la funcionalidad Fantasy, que permitiría dar un componente de juego adicional dentro de la aplicación. Esta propuesta se considera interesante de cara a futuras versiones, ya que en un principio ya se pensaba implementar porque aportaría un extra de interacción y entretenimiento.

7.4 Problemas encontrados y soluciones aplicadas

Incidencias principales

Durante el desarrollo y la validación del sistema se identificaron principalmente dos tipos de problemas:

- La sincronización de datos entre la base de datos y el sistema.
- La visualización inmediata de cambios en la aplicación tras realizar alguna operación modificadora (por ejemplo, crear un usuario y que no aparezca automáticamente en la lista de jugadores).

Análisis de problemas

El primer problema, relacionado con la sincronización, se debía a que en algunos casos los registros no se obtenían correctamente desde la base de datos o bien, en

sentido contrario, no se introducían de forma correcta. Esto provocaba inconsistencias en los datos mostrados en la aplicación. Se detectó sobre todo durante las fases de pruebas funcionales y flujo de datos, cuando al comparar la información introducida con la almacenada se apreciaban discrepancias.

El segundo problema tenía que ver con la actualización en tiempo real de la interfaz. Aunque los datos se guardaban correctamente en la base de datos, en ocasiones no se reflejaban de inmediato en las vistas. Esto obligaba a recargar la pantalla o navegar fuera y volver a entrar para que los cambios se mostrasen.

Soluciones implementadas

Para resolver la falta de sincronización, se revisó módulo por módulo la conexión con la base de datos, asegurando que tanto la lectura como la escritura se hiciesen de forma correcta y consistente. Firebase funciona con un sistema de introducción de datos basado en nombres de variables y métodos. Por lo tanto, una vez entendida esta parte, se aplicaron ajustes en los métodos encargados de cargar e introducir información para garantizar que siempre se utilizaban y guardaban los valores adecuados.

En el caso de la actualización inmediata de los datos, se introdujo el uso de LiveData y MutableLiveData[27], lo que permitió que los cambios en la base de datos se reflejasen automáticamente en la interfaz sin necesidad de recargar. Además, se implementó el módulo DataManager como punto central de la gestión de datos, encargado de mantener actualizado la información cada vez que se realizaba un cambio en el sistema.

Gracias a estas soluciones, la aplicación ofrece ahora una experiencia consistente, evitando que los usuarios tengan que realizar acciones adicionales para ver reflejados los cambios. Estas medidas mejoraron de forma significativa la estabilidad y fiabilidad del sistema, y han permitido que el sistema cumpla de manera más sólida con los requisitos de actualización en tiempo real y coherencia de datos.

Capítulo 8 - Conclusiones y futuras versiones

En este último capítulo se presentan las conclusiones generales del trabajo, valorando los objetivos cumplidos y reflexionando sobre la experiencia personal durante el desarrollo. Además, se incluyen propuestas de mejora y posibles ampliaciones para futuras versiones de la aplicación. Estas mejoras buscan añadir nuevas funcionalidades, y reforzar la utilidad y el componente lúdico de la herramienta.

8.1 Valoración de los objetivos alcanzados

El desarrollo de esta aplicación ha supuesto un reto tanto a nivel técnico como organizativo. Desde el inicio se partía de una idea clara sobre cómo estructurar los datos y se planteó un desarrollo modular. Sin embargo, durante el proceso fue necesario realizar ajustes en la organización de los datos, lo que implicó reestructurar tareas y replantear ciertos plazos.

A pesar de estos cambios, se ha conseguido entregar una versión robusta, coherente y con datos consistentes. La aplicación permite al usuario interactuar con la información de manera sencilla y satisfactoria, cumpliendo con los objetivos marcados en cuanto a funcionalidad y usabilidad. Además, se ha logrado establecer una base sólida y escalable sobre la que se podrán implementar nuevas funcionalidades en el futuro.

8.2 Conclusión personal del trabajo

A nivel personal, este proyecto ha supuesto un aprendizaje profundo no solo en el desarrollo técnico de una aplicación completa, sino también en la gestión de problemas reales como la organización de datos, la integración entre módulos y la conexión con una base de datos en la nube.

Uno de los aspectos más desafiantes fue garantizar que la comunicación entre la aplicación y la base de datos fuese eficiente y estable. No se trataba únicamente de conectar ambos sistemas, sino de asegurar que los datos se transfiriesen y actualizaran correctamente en todo momento. Este punto resultó clave para la coherencia y

persistencia de la información introducida por los administradores, especialmente en lo referente a partidos y estadísticas.

Finalmente, puedo concluir que el resultado es una herramienta funcional, útil y con un alto potencial de mejora y crecimiento. El trabajo realizado ha dado lugar a un sistema que cumple con la mayoría de los objetivos planteados inicialmente y que, además, abre la puerta a futuras ampliaciones que podrían hacer la aplicación todavía más atractiva. Igualmente, puedo decir que con esta versión se cubren todas las necesidades que inicialmente quería que la aplicación cubriese, y por tanto haré uso de ella en mi equipo, ya que realmente me parece útil.

8.3 Propuestas de mejora y ampliación futura

Aunque la aplicación es completamente operativa en su versión actual, existen varias ideas que podrían enriquecer el sistema y aportar nuevas formas de interacción para los usuarios. Entre ellas destacan la inclusión de un modo Fantasy, que se incluyó en el planteamiento inicial pero finalmente no se pudo implementar a tiempo, la posibilidad de generar informes estadísticos más avanzados y la implementación de herramientas de gamificación.

8.3.1 Implementación del modo Fantasy

Una de las ampliaciones más interesantes sería la inclusión de un modo Fantasy dentro de la aplicación. En un principio se iba a implementar, pero por cómo fue yendo el desarrollo de la aplicación, se descartó para la primera versión. Este modo permitiría a los usuarios crear, en cada jornada, un equipo reducido de tres jugadores cuyo rendimiento en ese fin de semana marcaría la puntuación Fantasy obtenida por el usuario, y se añadiría a una clasificación aparte dedicada exclusivamente a esta modalidad. De esta manera, se crearía una nueva funcionalidad dentro de la aplicación que añadiría un componente lúdico e interactivo, que fomenta aún más el uso de la herramienta.

8.3.2 Generación de informes estadísticos avanzados

Actualmente la aplicación ofrece estadísticas simples, lo cual resulta útil, pero todavía básico. Una posible mejora sería implementar un sistema de informes avanzados que permitiera analizar tenencias, comparar jugadores a lo largo de la temporada o incluso generar gráficos detallados. Este tipo de funcionalidades ofrecerían un valor añadido tanto a entrenadores como a jugadores, facilitando un análisis más profundo del rendimiento.

8.3.3 Gamificación con premios y logros

Otra línea de ampliación futura sería la incorporación de logros y recompensas virtuales. Por ejemplo, premios por alcanzar un número determinado de goles, asistencias o partidos jugados. Este enfoque no sólo aportaría motivación extra a los usuarios, sino que también aumentaría la retención y el uso continuado de la aplicación, reforzando el componente social y de juego.

8.3.4 Implementación de un módulo Social

Otra ampliación interesante sería la creación de un apartado Social, pensado como una sección de noticias o actualizaciones que combinen las distintas funcionalidades de la aplicación. Este espacio serviría para generar publicaciones automáticas a partir de los datos registrados en cada jornada, ofreciendo a los usuarios un entorno más dinámico e interactivo. Algunos ejemplos de estas publicaciones podrían ser:

- "Pedro ha superado este fin de semana a Andrés en minutos jugados esta temporada."
- "Iñigo se coloca como nuevo pichichi del equipo tras esta jornada."
- "El equipo de la semana está formado por Andrés, Lucas y Guille."

Además, se podrían incorporar notificaciones para esta sección, lo que aportaría un extra de captación de atención de los usuarios, así como un componente más social y motivador que refuerza el sentido de comunidad y competencia entre ellos.

Capítulo 8 - Conclusions and future versions

In this final chapter, the general conclusions of the work are presented, evaluating the objectives achieved and reflecting on the personal experience during the development. In addition, proposals for improvement and possible extensions for future versions of the application are included. These improvements aim to add new functionalities and to strengthen the usefulness and the playful component of the tool.

8.1 Evaluation of the objectives achieved

The development of this application has represented a challenge both at the technical and organizational level. From the beginning, a clear idea of how to structure the data was established, and a modular development was proposed. However, during the process, it was necessary to make adjustments in data management, which implied the reassignment of tasks and the reconsideration of certain deadlines.

Despite these changes, it has been possible to deliver a robust, coherent version with consistent data. The application allows users to interact with the information in a simple and satisfactory way, fulfilling the objectives set in terms of functionality and usability. In addition, a solid and scalable foundation has been established, on which new functionalities will be able to be implemented in the future.

8.2 Personal conclusion of the project

At a personal level, this project has represented deep learning, not only in the technical development of a complete application but also in the management of real problems such as data organization, module integration, and connection with a cloud database.

One of the most challenging aspects was ensuring that communication between the application and the database was efficient and stable. It was not only a matter of connecting both systems but of guaranteeing that data were transferred and updated correctly at all times. This aspect proved to be key for the coherence and persistence of the information entered by the administrators, especially concerning matches and statistics.

Finally, it can be concluded that the result is a functional and useful tool, with high potential for improvement and growth. The work carried out has resulted in a system that meets most of the objectives initially set and that, in addition, opens the door to future extensions that could make the application even more attractive. It can also be stated that with this version all the needs initially intended for the application are covered, and therefore it will be used within my team, as it proves to be genuinely useful.

8.3 Proposals for improvement and future extensions

Although the application is fully operational in its current version, several ideas exist that could enrich the system and provide new forms of interaction for users. Among them, the inclusion of a Fantasy mode, which was considered in the initial plan but could not be implemented on time, the possibility of generating more advanced statistical reports, and the implementation of gamification tools stand out.

8.3.1 Implementation of Fantasy mode

One of the most interesting extensions would be the addition of a Fantasy mode within the application. Initially, it was planned to be implemented, but due to the course of the development process, it was discarded for the first version. This mode would allow users to create, each matchday, a reduced team of three players whose performance that weekend would determine the Fantasy score obtained by the user, and it would be added to a separate ranking dedicated exclusively to this modality. In this way, a new functionality would be created within the application, adding a playful and interactive component that further encourages the use of the tool.

8.3.2 Generation of advanced statistics reports

Currently, the application offers simple statistics, which are useful but still basic. A possible improvement would be the implementation of an advanced reporting system that allows the analysis of trends, the comparison of players throughout the season, or even the generation of detailed graphs. This type of functionality would offer added value to both coaches and players, facilitating a deeper analysis of performance.

8.3.3 Gamification with rewards and achievements

Another future extension would be the addition of achievements and virtual rewards. For example, awards for reaching a certain number of goals, assists, or matches played. This approach would not only provide extra motivation for users but would also increase retention and continued use of the application, reinforcing the social and playful component.

8.3.4 Implementation of a Social module

Another interesting extension would be the creation of a Social section, designed as a space for news or updates that combine the different functionalities of the application. This space would serve to generate automatic posts from the data recorded in each matchday, offering users a more dynamic and interactive environment. Some examples of these posts could be:

- "Pedro has surpassed Andrés this weekend in minutes played this season."
- "Iñigo becomes the new top scorer of the team after this matchday."
- "The team of the week is formed by Andrés, Lucas and Guille."

In addition, notifications could be included into this section, providing extra user engagement and adding a more social and motivating component that strengthens the sense of community and competition among players.

Capítulo 9 - Referencias

[1] Fantasy. https://es.wikipedia.org/wiki/F%C3%BAtbol_fantasy

[2] Figma. https://www.figma.com/

[3] Android Studio. https://developer.android.com/studio/intro?hl=es-419

[4] LaLiga Fantasy. https://fantasy.laliga.com/

[5] FotMob. https://www.fotmob.com/

[6] Flashscore. https://www.flashscore.es/

[7] MatchApp. https://matchapp.es/terms/version-web/

[8] SFL Football. https://sflfootball.app/

[9] Prematch. https://www.prematchapp.de/

[10] Mister Fantasy. https://www.misterfantasy.es/

[11] MVVM. https://es.wikipedia.org/wiki/Modelo%E2%80%93vista%E2%80%93modelo_de_vista

[12] Activity. https://developer.android.com/guide/components/activities/intro-activities

[13] Fragment. https://developer.android.com/guide/fragments?hl=es-419

[14] Adapter. https://developer.android.com/topic/libraries/data-binding/binding-adapters?hl=es-419

[15] RecyclerView. https://developer.android.com/develop/ui/views/layout/recyclerview?hl=es-419

[16] Kotlin. https://kotlinlang.org

[17] Firebase. https://firebase.google.com/

[18] Firebase Authenticator. https://firebase.google.com/docs/auth

[19] Firebase Realtime Database. https://firebase.google.com/docs/database

[20] Github. https://github.com/

[21] BottomNavigationView. https://developer.android.com/reference/com/google/android/material/bottomnavigation/BottomNavigationView

[22] View Binding. https://developer.android.com/topic/libraries/view-binding

[23] Toolbar. https://developer.android.com/develop/ui/views/components/appbar/setting-up?hl=es-419

[24] NavController. https://developer.android.com/guide/navigation/navcontroller?hl=es-419

[25] LiveData. https://developer.android.com/reference/android/arch/lifecycle/LiveData

[26] Spinner. https://developer.android.com/develop/ui/views/components/spinner?hl=es-419

[27] MutableLiveData.
https://developer.android.com/reference/android/arch/lifecycle/MutableLiveData

Apéndices

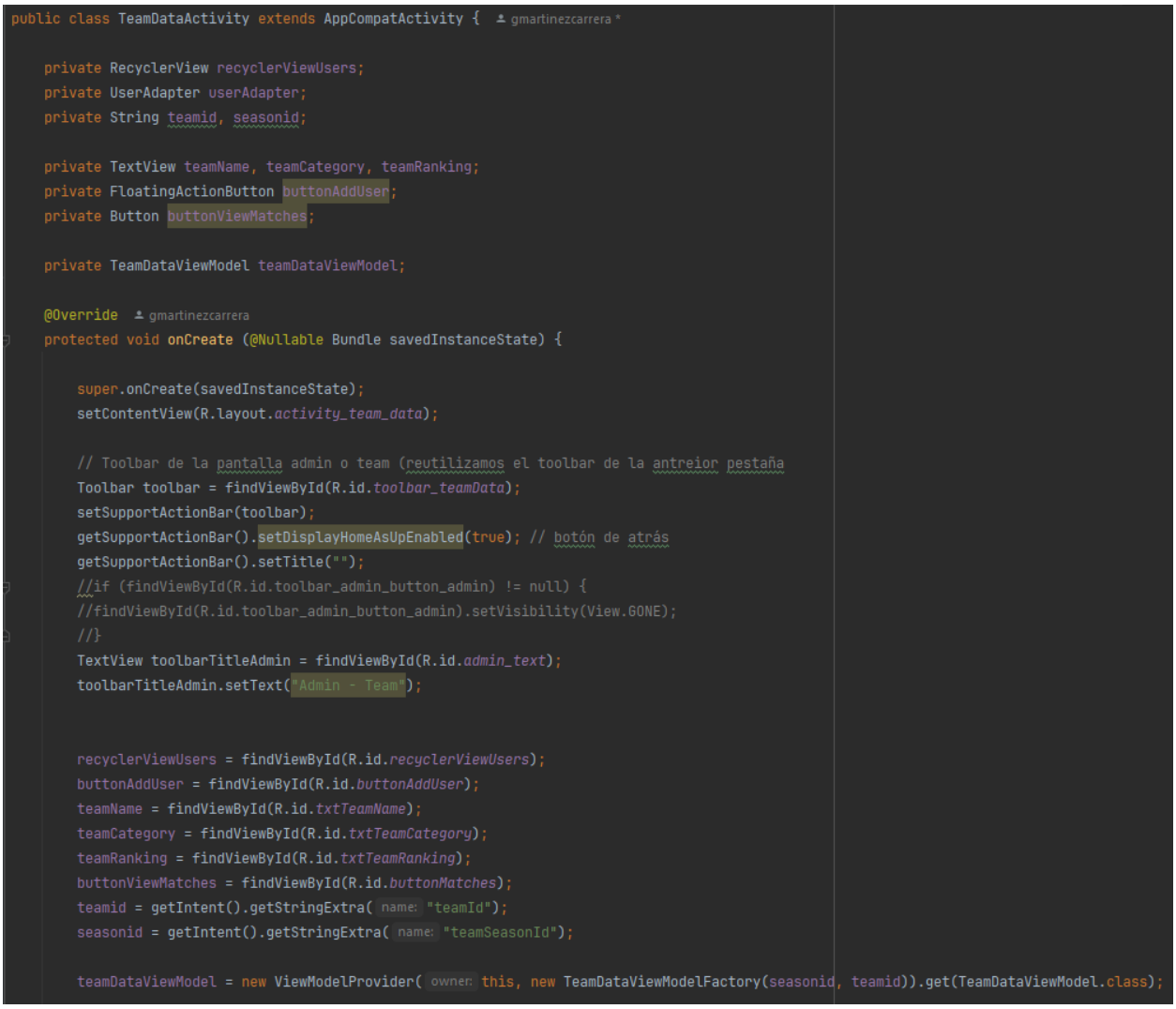
A.1 Apéndice 1 – Componentes de la aplicación

La estructura de la aplicación se basa en una organización modular que separa la interfaz de usuario, la lógica de negocio y los datos. Se utilizan fragmentos para mostrar cada sección funcional y una serie de clases modelo para representar entidades como usuarios, partidos o equipos. Todo el sistema se coordina a través del módulo DataManager, que gestiona la carga, sincronización y acceso a los datos en tiempo real.

Por ejemplo, tenemos por una parte el objeto Team, con la información del equipo, la lista de jugadores y la lista de partidos (ver Figura 47), y por otra, la actividad TeamDataActivity, que es un apartado accesible únicamente para el administrador, desde donde gestionará ese equipo y sus componentes, ya sean los usuarios que lo componen o los partidos que han jugado (ver Figuras 48 y 49).

```
public class Team {  gmartinezcarrera *

    private String idTeam;  2 usages
    private String name;  3 usages
    private String category;  2 usages
    private String leagueRanking;  2 usages
    private List<User> teamUsers;  4 usages
    private List<Match> teamMatches;  6 usages

    private String seasonID;  2 usages

    public Team (String id, String name, String category, String seasonID, String leagueRanking){
        this.idTeam = id;
        this.name = name;
        this.category = category;
        this.seasonID = seasonID;
        this.leagueRanking = leagueRanking;
        this.teamUsers = new ArrayList<>();
        this.teamMatches = new ArrayList<>();

    }
```

Figura 47 : Clase Team



```
public class TeamDataActivity extends AppCompatActivity {  gmartinezcarrera *

    private RecyclerView recyclerViewUsers;
    private UserAdapter userAdapter;
    private String teamid, seasonid;

    private TextView teamName, teamCategory, teamRanking;
    private FloatingActionButton buttonAddUser;
    private Button buttonViewMatches;

    private TeamDataViewModel teamDataViewModel;

    @Override  gmartinezcarrera
    protected void onCreate (@Nullable Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_team_data);

        // Toolbar de la pantalla admin o team (reutilizamos el toolbar de la antreior pestaña
        Toolbar toolbar = findViewById(R.id.toolbar_teamData);
        setSupportActionBar(toolbar);
        getSupportActionBar().setDisplayHomeAsUpEnabled(true); // botón de atrás
        getSupportActionBar().setTitle("");
        //if (findViewById(R.id.toolbar_admin_button_admin) != null) {
        //findViewById(R.id.toolbar_admin_button_admin).setVisibility(View.GONE);
        //}
        TextView toolbarTitleAdmin = findViewById(R.id.admin_text);
        toolbarTitleAdmin.setText("Admin - Team");


        recyclerViewUsers = findViewById(R.id.recyclerViewUsers);
        buttonAddUser = findViewById(R.id.buttonAddUser);
        teamName = findViewById(R.id.txtTeamName);
        teamCategory = findViewById(R.id.txtTeamCategory);
        teamRanking = findViewById(R.id.txtTeamRanking);
        buttonViewMatches = findViewById(R.id.buttonMatches);
        teamid = getIntent().getStringExtra( name: "teamId");
        seasonid = getIntent().getStringExtra( name: "teamSeasonId");

        teamDataViewModel = new ViewModelProvider( owner: this, new TeamDataViewModelFactory(seasonid, teamid)).get(TeamDataViewModel.class);
```

Figura 48 : Clase TeamDataActivity (Parte 1)

```
    userAdapter = new UserAdapter( context: this, new ArrayList<>(), seasonid, teamid);
    recyclerViewUsers.setLayoutManager(new LinearLayoutManager( context: this));
    recyclerViewUsers.setAdapter(userAdapter);

    // se observa equipo
    teamDataViewModel.getTeam().observe( owner: this, team -> {
        if (team != null){
            teamName.setText(team.getTeamName());
            teamCategory.setText(team.getTeamCategory());
            teamRanking.setText(team.getTeamLeagueRanking());
        }
    });

    teamDataViewModel.getUsers().observe( owner: this, users -> {
        if (userAdapter == null) {
            userAdapter = new UserAdapter( context: this, users, seasonid, teamid);
            recyclerViewUsers.setLayoutManager(new LinearLayoutManager( context: this));
            recyclerViewUsers.setAdapter(userAdapter);
        } else {
            userAdapter.setUsers(users);
        }

    });


    buttonAddUser.setOnClickListener(v -> {
        // añadir usuario
        Team team = teamDataViewModel.getTeam().getValue();
        if (team != null){
            if (team.getTeamUsers().size() < 30){
                openWindowToAddUser();
            }else{
                Toast.makeText( context: this, text: "Max number of users for this team reached", Toast.LENGTH_SHORT).show();
            }
        }

    });


    buttonViewMatches.setOnClickListener(v -> {
        Intent intent = new Intent ( packageContext: this, MatchListActivity.class);
        intent.putExtra( name: "teamId", teamid);
        intent.putExtra( name: "teamSeasonId", seasonid);
        startActivity(intent);
        overridePendingTransition(R.anim.slide_in_right, R.anim.slide_out_left);
    });
}
```

Figura 49 : Clase TeamDataActivity (Parte 2)

A.1.1 Fragmentos principales

SocialFragment (Pantalla Inicial)

Este fragmento representa la entrada principal tras el login. El nombre hace referencia a los datos del usuario y sus compañeros, donde se muestra un resumen de los datos más relevantes del usuario autenticado. Se presentarán estadísticas del equipo seleccionado a través de spinners (un mismo usuario puede pertenecer a varios equipos en diferentes temporadas o incluso en la misma), y además un resumen personal de las estadísticas de ese jugador en la temporada y equipo seleccionado.

El fragmento accede a los datos mediante DataManager, que recupera el LoggedInUser y extrae la información estadística desde su objeto User, buscando dentro de la rama que coincida con el equipo y temporada actuales.

StandingsFragment y sus subfragmentos

Este fragmento es el núcleo de las estadísticas generales. Incluye una barra de pestañas (TabLayout) y un sistema de paginado (ViewPager2) que permite navegar entre diferentes tipos de ranking. Está conectado a StandingsPagerAdapter, que instancia cada subfragmento según la posición de la pestaña. Los subfragmentos definidos son:

- GeneralStatsFragment: muestra la tabla general ordenada por puntos. También incluye partidos jugados, titularidades, etc.
- GoalsFragment.java: ordena a los jugadores según el número total de goles.
- AssistsFragment: muestra el ranking de asistencias totales y sus métricas.
- MinutesFragment: clasifica a los jugadores por minutos acumulados en juego.
- GoalsPerMatchFragment: ordena a los jugadores a partir del promedio de goles por partido jugado.
- GoalsAndAssistsFragment: combina ambas métricas de goles y asistencias para ofrecer una clasificación en función de las participaciones totales en goles.

Cada fragmento recibe desde StandingsViewModel la lista de usuarios de la temporada y equipo seleccionados en los spinners. El ViewModel accede a estos datos a través del DataManager y actualiza en tiempo real las listas si se cambian los filtros. Esta lógica está basada en LiveData, de forma que los fragmentos sólo observan y muestran lo que reciben.

ProfileFragment

Este fragmento muestra la información únicamente relativa al usuario que ha iniciado sesión. Comparte diseño con StandingsFragment, reutilizando las mismas pestañas y fragmentos, pero con un cambio importante, sólo mostrando las estadísticas del usuario autenticado. El filtrado se realiza en el ProfileViewModel, donde se recorren los usuarios de la temporada y equipo seleccionados, y filtra por el usuario autenticado.

MyTeamFragment

La pantalla MyTeam, que inicialmente iba a servir para implementar una funcionalidad diferente, se ha construido finalmente para permitir al usuario autenticado consultar, de forma sencilla, los partidos de los equipos de los que forma parte. Se presentará la lista de partidos para la temporada y el equipo elegidos en los desplegables superiores y se podrá seleccionar cada uno de ellos para poder visualizar en detalle cada uno de ellos. El diseño mantiene la coherencia con otras pantallas de la aplicación, reutilizando componentes ya implementados.

A.1.2 Núcleo del sistema: DataManager.java

El DataManager es el encargado de inicializar y mantener en memoria todos los datos relativos al usuario que ha iniciado sesión. Los métodos más importantes son:

- loadInitialData: carga todas las ramas de Firebase y construye los objetos correspondientes en memoria.

- reloadData: permite forzar la recarga de todo el sistema, útil después de crear o editar elementos importantes por el admistrador.
- getLoggedInUser: devuelve el objeto User del usuario autenticado actualmente.

Al cargar los datos, el DataManger convierte cada entrada de Firebase en una instancia de Admin, Season, Team, User o Match, y las organiza en mapas y listas para acceso rápido. De hecho, en esta carga en memoria, los datos sí que se organizan en forma de árbol. Gracias a esto, no es necesario consultar Firebase constantemente: toda la lógica trabaja en memoria y sólo se sincroniza con Firebase cuando hay cambios, o cuando se cierra y vuelve a iniciar sesión.

A.1.3 Módulos de datos

- Admin: tal y como se almacenará en la base de datos, cada objeto Admin guardará la información correspondiente, como el email o su identificador de Firebase, para poder hacer búsquedas internas y hacer las cargas únicamente relativas a la información que debe ver cada usuario.
- Season: las temporadas tienen un identificador único y una lista de equipos además de guardar el identificador de su administrador.
- Team: cada equipo incluye la información relativa al mismo y una lista de jugadores que forman parte de él, aparte de una lista de los partidos que ha jugado dicho equipo.
- User: representa un jugador o un administrador. Contiene su identificador de Firebase para poder hacer búsquedas internas, información relativa al jugador como nombre o posición, y las estadísticas para ese usuario en el equipo y temporada a los que pertenece. Goles, asistencias, puntos o minutos jugados son algunos de los datos que almacena.
- Match: cada partido almacena unos datos identificativos como el nombre o la fecha, y el resto son estadísticas del mismo, como goles, asistencias, once inicial, cambios realizados, goles en contra, etc.

A.1.4 Registro de temporadas, equipos, usuarios y partidos por el administrador

El sistema permite a los administradores gestionar todos los datos estructurales de la aplicación: temporadas, equipos, jugadores y partidos. Esta funcionalidad es accesible desde el botón superior de LobbyActivity, el cual sólo se muestra si el usuario autenticado tiene permisos administrativos (verificados desde DataManager). Desde ahí, se lanza una serie de actividades que permiten llevar el control total del sistema de juego.

Todo el flujo se ha diseñado para ser progresivo, permitiendo construir la estructura desde cero: primero se crean temporadas, luego equipos dentro de cada temporada, y después se introducen jugadores y partidos dentro de cada equipo. Cada acción modificadora está vinculada a la base de datos y a la lógica interna del sistema, por lo que cada vez que se crea, edita o elimina una entidad, se actualiza también la memoria del DataManager con la base de datos.

Las pantallas de gestión son las siguientes: SeasonActivity, TeamActivity, TeamDataActivity, UserDetailActivity, MatchListActivity, MatchCreateActivity, MatchDataActivity y MatchEditActivity.

SeasonActivity

Muestra todas las temporadas creadas por el administrador actual, además de un botón para añadir nuevas. Si no hay ninguna, se muestra únicamente dicho botón. Las temporadas se identifican por un ID y se les asigna un periodo deportivo (septiembre año 1 - junio año 2).

TeamActivity

Una vez seleccionada la temporada, se muestran los equipos asociados. Cada equipo pertenece a una única temporada y se representa con nombre, categoría y posición. Desde esta pantalla también podemos añadir nuevos equipos a esta temporada.

TeamDataActivity

Gestiona los jugadores de un equipo. Se pude visualizar la lista actual de usuarios del equipo, añadir nuevos o acceder a la información de cada uno. Los jugadores se guardan con una clave compuesta que incluye los identificadores de todos sus padres, lo que permite que un mismo usuario esté presente en varios contextos (temporadas y/o equipos).

UserDetailActivity

Permite visualizar la información de los usuarios. Al crear un nuevo jugador, además, se vincula automáticamente a Firebase Authentication, utilizando un email y password introducidos para generar una cuenta real. Además, se actualiza la base de datos con sus estadísticas iniciales, que en ese momento están a cero.

MatchListActivity

Desde el propio equipo, aparte de añadir jugadores, se puede acceder a la lista de partidos jugados por el equipo actual, además del botón para añadir nuevos partidos. En caso de no haber ninguno registrado, sólo aparece este botón.

MatchCreateActivity

Permite registrar un nuevo partido. Desde esta vista se pueden añadir todos los eventos relevantes en un partido: goles a favor (con jugador y minuto), asistencias, goles en contra, penaltis parados, fallados o provocados, once inicial y sustituciones. Todos estos elementos se gestionan mediante spinners y botones dinámicos para añadir estadísticas. Al confirmar, se guarda el partido en ejecución y en Firebase y se actualizan automáticamente las estadísticas de todos los jugadores implicados.

MatchDataActivity

Muestra los detalles de un partido ya creado. No permite edición, simplemente sirve como vista informativa con todos los eventos organizados en secciones. Esta vista dispone de dos botones, uno para eliminar el partido y, por tanto, todas sus estadísticas y las de sus usuarios, y otro para editar los eventos del mismo.

MatchEditActivity

A partir de un partido existente, si se elige la opción de editar, se puede editar cualquier parte del mismo. Incluye botones "Clear" para borrar las listas completas y botones para añadir nuevas. Al guardar, la forma de editar los datos es eliminando el partido anterior (revirtiendo por tanto las estadísticas para sus usuarios) y creando uno nuevo con la misma información y el mismo identificador, de manera que se aplican todos los cambios de forma limpia y persistente.

Todo el sistema de administración se apoya en DataManager, que centraliza la sincronización entre Firebase y la memoria interna. Al realizar cualquier acción modificadora, como crear una temporada, añadir un jugador a un equipo o editar un partido, se actualizan automáticamente las estructuras en memoria y se reflejan los cambios en la base de datos.

A.2 Apéndice 2 – Cálculo de puntos según estadísticas

El cálculo de puntos a partir de las estadísticas depende no sólo de la métrica en sí, sino también de la posición del jugador. Estas posiciones, teniendo en cuenta que las formaciones pueden variar, se han organizado de la siguiente forma:

- Portero
- Defensas: defensas centrales, laterales, carrileros
- Centrocampista: mediocentro defensivo, mediocentro
- Delanteros: mediocentro ofensivo, extremos, delantero

En la Tabla 3 podemos ver cómo se calculan los puntos de los jugadores dependiendo de los diferentes eventos del partido.

	Portero	Defensa	Centrocampista	Delantero
Gol	10 puntos	6 puntos	5 puntos	4 puntos
Asistencia	4 puntos	4 puntos	3 puntos	3 puntos
Titularidad	1 punto	1 punto	1 punto	1 punto
Minutos	1 punto/45 mins en mismo partido	1 punto/45 mins en mismo partido	1 punto/45 mins en mismo partido	1 punto/45 mins en mismo partido
Penalti provocado	2 puntos	2 puntos	2 puntos	2 puntos
Penalti fallado	-3 puntos	-3 puntos	-3 puntos	-3 puntos
Penalti parado	5 puntos	-	-	-
Gol encajado	-1 punto	-1 punto	-1 punto	-1 punto

Tabla 3 : Cálculo de puntos