Universidad De Valladolid

Escuela De Ingeniería Informática

TRABAJO FIN DE GRADO

Grado En Ingeniería Informática

Mención En Tecgnología De La Información

Training4Players: basket coaching software

Alumno: Daniel Pariente Bermejo
Tutor/es/as: Joaquín Adiego Rodríguez
Natalia Martín Cruz

A mi familia y amigos

Agradecimientos

Tras muchos meses de trabajo, hoy llega el día en que finalizo mi grado en ingeniería informática. Han sido unos años inolvidables, con momentos complicados que me han hecho madurar como persona. He conocido a gente que en un principio eran compañeros, y que en este largo recorrido han terminado siendo amigos.

Hoy puedo decir que cumplo un sueño, del que estoy realmente orgulloso. Desde aquí, quiero dar las gracias a una serie de personas, que han hecho posible que logre finalizar mis estudios.

En primer lugar, quiero dar las gracias a mis padres. Ellos han estado día tras día aguntando los malos momentos y celebrando los buenos, por lo que para ellos va mi primer agradecimiento en esta sección.

Los siguientes a los que quiero dar las gracias, son a mis compañeros de clase. Con ellos he compartido mil historias, días de risas, nervios y sufrimiento, que han acabado de la mejor manera posible, en una gran amistad.

A Maria, por apoyarme en estos últimos años, y confiar más en mí que yo mismo. Por nuestras celebraciones cuando aprobaba asignaturas, por nuestras apuestas en las notas.....por esto y por más te doy las gracias.

Por último, quiero dar las gracias Joaquín y Natalia. Creo que han sido dos tutores ejemplares y me han aconsejado en todo lo que han podido adecuadamente.

De corazon, Muchas gracias a todos!

Daniel Pariente

Valladolid 03/03/2021

Resumen

En el presente documento, se expone el proyecto de desarrollo de una aplicación Android que permita al usuario consultar todos los datos estadísticos relativos a la NBA, y en base a esos datos, implementar algoritmos que sugieran conclusiones sobre la competición . Además, la aplicación será capaz de predecir el compartamiento de partidos futuros de manera eficiente, tomando como base, datos estadísticos del pasado.

Por otro lado, no solo se centra en cuestiones estadísticas, sino que , nos mostrará las noticias relativas a la competición actualizadas al minuto. Para finalizar, como curiosidad la aplicación es capaz de localizar en Google Maps, el estadio de cada uno de los equipos de la NBA, y el usuario podrá ver perfectamente vía satélite, la localización del pabellón en cuestión.

Dicho proyecto, surge como una colaboración con la facultad de Económicas, en la que se precisa de una app, que proporcione estadísticas sobre los equipos y jugadores para analizar su rendimiento a lo largo de los últimos años. Para garantizar un correcto estudio de esta liga, se proporciona la posibilidad de consultar todos los datos estadísticos, desde el año 1995 hasta la actualidad . Se elige este periodo de tiempo debido a que en el año 2000, se produce un cambio en la reglamentación de la competición y se quiere ver como afectan estos cambios a los equipos de la liga.

La aplicación leerá los datos directamente de dos APIs diferentes, y los mostrará al usuario tras realizar un procesamiento de datos y operaciones (si fuera necesario). De esta manera los datos siempre estarán actualizados. La aplicación contiene la estructura preparada para la utilización de una base datos del tipo Firebase. Respecto a la interfaz del usuario me he centrado en que sea una aplicación eficiente, robusta y fácil de usar. En las próximas secciones explicaré con detalle los modelos seguidos, diagramas..etc.

Abstract

In this document, the development project of an android application is exposed, which allows the user to consult all the statistical data related to the NBA, and based on these data, implement algorithms that suggest conclusions about the competition. In addition, the application will be able to predict the behavior of future matches, based on statistical data from the past, quite efficiently.

On the other hand, it not only focuses on statistical issues, but will show us the news related to the competition updated to the minute. Finally, as a curiosity, the application is capable of locating in google maps, the stadium of each of the Nba teams, and the user will be able to see perfectly via satellite the location of the pavilion in question.

This project arises as a collaboration with the Faculty of Economics, in which an app is required, which provides statistics on the teams and players, to analyze their performance over recent years. To guarantee a correct study of this league, the possibility of consulting all statistical data, from 1995 to the present, is provided. This period of time is chosen, because in the year 2000, there is a change in the regulations of the competition and we want to see how these changes affect the teams in the league.

The application will read the data directly from two different APIs, and will show it to the user after performing data processing and operations (if necessary). In this way the data will always be updated. The application contains the structure prepared for the use of a Firebase-type database. Regarding the user interface, I have focused on making it an efficient, robust and easy-to-use application. In the next sections I will explain in detail the following models, diagrams ... etc.

Índice

Índice de figuras

Índice de cuadros

1. Introducción

En la sociedad actual, el uso de aplicaciones móviles está adquiriendo un enfoque totalmente generalizado . La mayoría de personas, tienen un smarthphone y eso implica el uso de aplicaciones. La utilización de estas apps, tienen como fin facilitar la vida de las personas, pueden ser usadas para consultar información, entretener o realizar algún tipo de transacción.

Actualmente, una parte del mercado de estas aplicaciones, la constituyen las apps deportivas. Suelen estar orientadas mayoritariamente, a aficionados que quieren ver las últimas novedades de los equipos, consultar algún dato en concreto o ver la situación actual de la competición. También, podemos encontrar un enfoque mas "profesional", en el que se sitúan las aplicaciones deportivas que son utilizadas por el cuerpo técnico de los equipos, y que en base a estadísticas de eventos pasados, pueden predecir el compartamiento futuro de sus equipos, lo que es de gran utilidad para ellos.

En nuestro caso la aplicación constituye una mezcla entre ambos enfoques. Por un lado, podremos consultar todas las estadísticas relativas a la competición (jugadores, rendimiento, equipos, partidos..etc) de la liga NBA desde el año 1995, hasta la actualidad. Todos los datos se muestran totalmente actualizados, y no es necesario esperar a que finalice por ejemplo una jornada completa para consultar las nuevas estadísticas. Por otro lado, se tiene un enfoque mas profesional, en el que se han desarrollado algoritmos que permiten predecir el resultado de los encuentros, o aconsejar al entrenador que jugador es el más idóneo para ocupar un puesto que no es el suyo.

Este rendimiento "inmediato", lo conseguimos gracias a la lectura de los datos de dos APIs diferentes. El usuario elige un dato a consultar de una temporada determinada, y la aplicación se encarga de concatenar la url necesaria para obtener esos datos. Realiza una consulta para obtener un archivo en formato JSON de los datos pedidos por el usuario, y los trata con operaciones de transformación a texto en plano, o operaciones aritméticas en algunos casos (depende de lo que el usuario quiera consultar).

Por último, una vez que el usuario ve por pantalla los datos estadísticos, el sistema le da la opción de exportar los datos en formato CSV. Simplemente tendrá que dar al botón "exportar" y la app generará un archivo CSV compatible con Excel, y se guardará en el directorio que elija. Esta función, está pensada para que los archivos generados en la app se utilicen en otras herramientas más potentes de tratamiento de datos estadísticos.

En cuanto al futuro de este proyecto, se podrían continuar añadiendo algunas funciones que pueden ser interesantes para el usuario, como pueden ser un analisis más profundo de cada jugador. La app proporciona todos los porcentajes de tiro que tiene un jugador, pero se podría por ejemplo, añadir desde que zonas tira mejor, donde falla más tiros, donde realiza la mayoría de tiros etc.. . Una mejora que siempre estará presente es la de mejorar los algoritmos de predicción de resultados, metiendo más variables, para así aumentar la eficacia de los mismos.

1.1. Origen del proyecto

El proyecto surge con la idea de estudiar el rendimiento de equipos y jugadores en la liga y predecir su comportamiento futuro. A partir del año 2000, se establecen nuevas normas en la competición y es por eso que vamos a proporcionar datos estadísticos de la liga desde 1995 hasta la actualidad, para ver como han afectado estas nuevas normas a la competición.

Al principio, se valoró la opción de realizar el estudio de alguna liga de fútbol, pero ya existen bastantes estudios sobre este deporte y preferimos realizar uno de baloncesto. Quizá realizar la app de baloncesto es más complicado, porque hay menos información que en otros deportes.

Esto último, fue uno de los primeros problemas que encontré, utilicé varias semanas para investigar que APIs eran las más adecuadas para utilizar en la aplicación. No fue sencillo ya que la mayoría de las APIs son de pago, y las que son gratis no son muy completas. Finalmente opté por combinar varias Apis, y de esta manera conseguí tener una cantidad de datos bastante amplia.

1.2. El proyecto

Este proyecto tiene como objetivo desarrollar una aplicación para dispositivos móviles bajo el sistema operativo Android, permitiendo la correcta visualización de estadísticas y pronosticos de los partidos de la liga NBA. También se pretende que su funcionamiento sea preciso en tablets.

Para cumplir este cometido, se diseñarán multiples ventanas que permitan la correcta visualización de los datos. Se hará hincapié en la facilidad de uso de la aplicación, buscando en todo momento un diseño minimalista. Además, se intentará tener la mayor fluidez posible en la carga de datos.

Para lograr una correcta fluidez y una correcta transicción entre ventanas, he intentado desarrollar un código lo más eficiente posible. En lineas generales, las transicciones entre ventanas serán siempre fluidas, al igual que el tiempo de carga de datos, aunque este última, dependerá del estado y velocidad del servidor de la API que estemos leyendo en ese momento.

Del objetivo principal que he mencionado al comienzo de esta subsección, podemos elaborar una lista de objetivos secundarios:

- Conocer en profundidad el lenguaje Android y su funcionamiento. Profundizar en este lenguaje para mejorar el nivel de las aplicaciones desarrolladas en el futuro.
- Entender el funcionamiento de las APIs que nos brindan los datos estadísticos.
- Aprender a enlazar dichas APIs en nuestra aplicación, y tratar los datos que se leen de la forma más eficiente posible.
- Aprender a elaborar una planificación de un proyecto software.
- Desarrollar una memoria con caracter técnico sobre un proyecto de software.

- Conocer una nueva forma de trabajo a partir de personas que no tienen conocimientos de informática, y que proporcionan unos requisitos funcionales al proyecto.
- Desarrollar una visión centrada en el usuario, combinándola con la visión del desarrollador, para lograr un mejor resultado final.

1.3. Motivación

¿Por qué una aplicación para teléfonos móviles?

LLegados a este punto surge la cuestión de por qué hemos elegido desarrollar una aplicación móvil. En mi opinión, los smarthphones han cambiado la vida de las personas en esta última década. Resulta extraño encontrar en 2021 a personas sin un smarthpone en su poder. Es por esta primera razón, por la que elijo desarrollar una app para dispositivos móviles, porque creo que es una tecgnología que va a llegar a la mayoría de personas.

El concepto de telefono móvil ha cambiado drásticamente con el paso de los años. Su origen surge con la necesidad de realizar llamadas telefónicas para establecer comunicación entre individuos. Este enfoque ha cambiado radicalmente, convirtiendo a los telefónos móviles en una ventana permanentemente abierta para rescatar cualquier tipo de información, dando lugar al nacimiento de los smarthphones.

El uso de los smartphones se ha visto incrementado año tras año. En parte, este crecimiento se debe a la utilización de internet como herramienta global de consulta de información. En la siguiente imagen lo vemos claro, en los últimos años se ha incrementado el uso de servicio de datos exponencialmente, mientras que el de voz se mantiene constante:

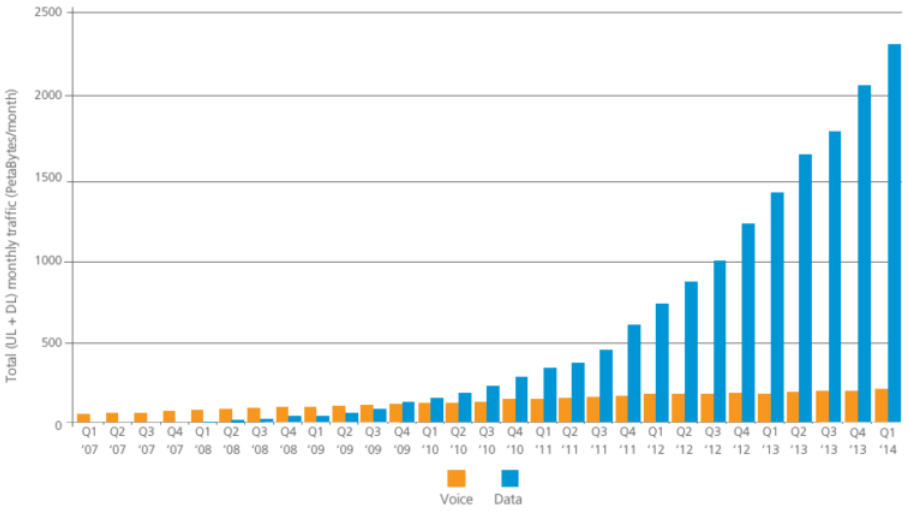

Figura 1: Comparativa crecimiento voz y datos

En la ilustración anterior, vemos la irrefutable prueba de que los dispositivos móviles son utilizados mayoritariamente para navegar por internet, mientras que las llamadas quedan en un segundo plano.

La comodidad que prestan estos dispositivos es total, llegando a causar en muchos casos una sensación de dependencia que puede desembocar en adicciones a este tipo de dispositivos[1].

La venta de smarthphones sigue incrementandose año tras año. Actualmente ya hemos superado los 3000 millones de usuarios de este tipo de dispositivos. Los países de China, la India y Estados Unidos son los que más usuarios aportan a esta estadística, superando todos ellos los 100 millones de usuarios.

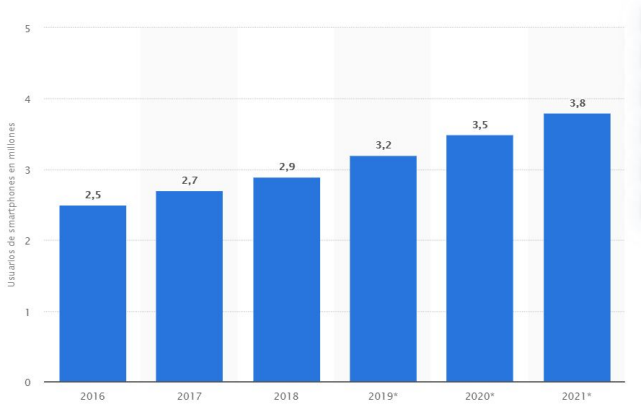

Figura 2: Crecimiento de usuarios que usan smartphones(en millones)

Los fabricantes que más dispositivos aportan son las marcas Samsung, Apple y Huawei. Estos fabricantes constituyen el 50 por ciento de ventas de smartphones en todo el mundo. [2]

¿Por qué la Nba y no de otras ligas?

La Nba es la mejor liga baloncestística de todo el mundo. Cada partido en esta competición se convierte en un auténtico "show"que produce visualizaciones en todos los lugares del mundo. Los mejores patrocinadores y marcas se pelean por aparecer en los descansos de los partidos de esta liga. El logo de la NBA es una de las marcas más conocidas de todo el mundo. Por todo lo que mueve este deporte y en concreto esta liga, se ha decidido enfocar la aplicación a esta competición.

Figura 3: Logo de la NBA.

En un primer momento se intentó hacer también de la liga española de baloncesto. El problema es que en este deporte toda la información y estudios estadísticos están centrados en la Nba, y no es fácil encontrar tanta información de otras ligas. En nuestro caso, para leer datos de las APIs, solo podíamos conseguir datos estadísticos de la Nba.

Si bien es cierto que existen aplicaciones que muestran estadísticas de otras ligas de baloncesto, estas aplicaciones tienen fines económicos para sus creadores, y no les importa invertir cierta cantidad de dinero para obtener la información que necesitan. No es nuestro caso, ya se trata de un proyecto docente y no tenemos ninguna intención de ganar dinero. Dicho esto, como se puede imaginar, efectivamente se pueden obtener datos de otras ligas (incluida la española), pero en este caso tendríamos que realizar una remuneración mensual/trimestal, y en este proyecto se pretende que todo se realice a coste cero.

Otra opción que se planteó fue utilizar la conocida técnica Web scrapping[3], que consiste en extraer información de páginas web estudiando la arquitectura de dicho lugar. Es una técnica que podríamos haber usado, pero que en este proyecto no ha sido aplicada por la dificultad de mantenimiento. En concreto, la app que se ha desarrollado se pretende que se use para estudios estadísticos en los siguientes meses, y sí por ejemplo en este tiempo cambia la estructura de alguna web de donde nuestra app lee datos, esta

dejaría de funcionar.

¿Por qué Android?

Cuando hablamos de Android, estamos refiriéndonos al sistema operativo más utilizado en smartphones en todo el mundo. Pertenece a Google, y desde hace años está en constante evolución. Una de las características principales de este software, al igual que Linux, es que se trata de un sistema operativo de código abierto, lo que permite a desarrolladores dar un extenso abanico de posibilidades a la hora de añadir funcionalidades al sistema.

Otro de los puntos fuertes de Android, es que es un sistema operativo gratis y multiplataforma, lo que quiere decir que puede ser instalado de manera gratuita en prácticamente cualquier dispositivo. En la primera versión de este software, podíamos encontrar ya los conocidos servicios de Google, un navegador Web compatible con HTML y XHTML (que además ofrecía la función de zoom integrado y permitía ver múltiples páginas en diferentes ventanas). Pero lo que le iba a alzar a la cima a este sistema, sería la posterior integración de Android Market, el famoso almacen de aplicaciones que hoy en día conocemos como Play Store.[4]

Algunas de las novedades que trajo la primera versión para Android fueron:

- *Ventana de notificación desplegable:* Fue la principal novedad y caracteristica de este software. Desde un principio, se tuvo claro que querían un sistema de notificaciones que permitiese en todo momento dar visibilidad en pantalla a la nueva información que entraba al usuario.
- *Widgets en la pantalla de inicio:* Conforman la esencia de Android. Desde sus incios dieron la posibilidad de implementar en la ventana principal del dispositivo numerosos widgets con distintas funcionalidades.
- *Integración de Gmail:* La primera versión de Android contenía el mejor sistema de visualización de correo visto hasta la fecha. Esto fue gracias, en parte, al apoyo de Gmail a POP e IMAP.
- *Android Market:* Como hemos mencionado antes, es la principal característica y revolución de los smartphones. Los primeros solo contenían una fila con unas pocas aplicaciones para descargar. A día de hoy sobran las palabras para describir las dimensiones del actual Play Store.

Por todo esto, se ha escogido la plataforma Android como sistema operativo para desarrollar nuestra aplicación. Otro argumento de mucho peso para escoger esta tecgnología es el conocimiento previo, ya que anteriormente había cursado la asignatura de Sistemas Móviles donde desarrollé también una aplicación en Android.

Llegados a este punto, podemos preguntarnos por qué no se ha planteado la posibilidad de desarrollar la aplicación en el sistema del gran competidor de Android: IOS. El software de Apple abarca prácticamente la otra mitad del mercado de smartphones. A pesar de ser tan conocida como android, y tener también unas características muy atractivas, no fue una opción que nos planteásemos.

La programación en IOS era totalmente desconocida para mi, se debe de realizar en lenguaje Objetive C o en el propio lenguaje lanzado por Apple (Swift). Sin embargo, en Android lo he programado en

Java que es el lenguaje que más he usado en estos últimos años. Por otro lado, se necesita un ordenador de Apple para desarrollar aplicaciones en IOS y no disponíamos de uno. Se podía haber usado algun sistema de virtualización pero según tengo entendido la experiencia no era la mejor.

Figura 4: Widgets Android.

2. Entorno Tecgnológico

2.1. Software Utilizado

2.1.1. Android Studio

Android Studio [5] es el entorno de desarrollo(IDE) oficial para la programación de aplicaciones Android. Está basado en el software IntelliJ IDEA de JetBrains. Fue presentado el 16 de mayo de 2013 en la conferencia Google I/O, y reemplazó al que hasta ese momento era el entorno de desarrollo oficial para Android, el entorno de desarrollo Eclipse.

Su primera versión estable fue publicada en diciembre de 2014. En cuanto a los sistemas operativos compatibles, funciona en las plataformas Microsoft Windows, MacOS y GNU/Linux.
Desde hace poco tiempo, Google anunció que kotlin es el lenguaje recomendado para desarrollar aplicaciones Android, aunque Android Studio es compatible con otros lenguajes de programación como por ejemplo Java y C++ [5].

Android Studio está en constante evolución, y cada versión incluye nuevas funcionalidades enfocadas a mejorar la experiencia del desarrollador en el software. Entre las funcionalidades de la última versión estable destacamos las siguientes[6]:

- Un sistema de compilación flexible basado en Gradle.
- Herramientas Lint para detectar problemas de rendimiento, usabilidad, compatibilidad de versiones y otros problemas.
- Integración de ProGuard y funciones de firma de aplicaciones.
- Un editor de diseño enriquecido, que permite a los usuarios arrastrar y soltar componentes de la interfaz de usuario.
- Un emulador rápido y cargado de funciones.
- Soporte integrado para Google Cloud Platform, que permite la integración con Firebase Cloud Messaging

Figura 5: Logo del software Android Studio.

2.1.2. Retrofit

Retrofit es un cliente de servidores REST para Android y Java desarrollado por Square. Destaca por su sencillez y nos brinda un amplio abanico de opciones a la hora de realizar peticiones a servidores del tipo GET, POST, PUT, PATCH, DELETE y HEAD. Además, nos va a permitir gestionar diferentes tipos de parámetros, parseando automáticamente la respuesta al tipo de datos que más nos convenga en cada caso.

Esta biblioteca la he utilizado mucho para transformar los datos que recogía de las APIs. Concretamente, traía a la aplicación los datos en formato JSON, y a partir de los métodos de Retrofit, convertía los JSON en objetos Java que definía previamente.

Su funcionamiento es bastante sencillo e intuitivo. Si leemos su documentación[7], vemos que el único requisito que tenemos que cumplir es dar los permisos necesarios a la aplicación para conectarse a internet, y añadir un par de dependencias necesarias. Con eso, ya podemos empezar a manipular url para sacar

datos de servidores de APIs.

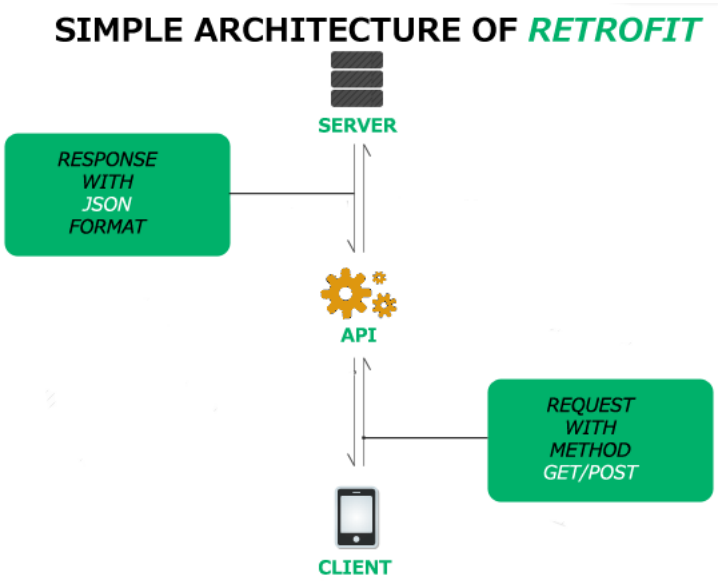

Figura 6: Esquema del funcionamiento de la librería Retrofit.

2.1.3. Glide

Se trata de otra librería muy utilizada en Android. Sirve para descargar imágenes, videos o gifs de servidores remotos a partir de una fuente. Es muy útil, y nos simplifica mucho el trabajo ya que no tenemos que preocuparnos de administrar conexiones, codificar datos o manejar hilos.

Esta librería la he utilizado en casi todas las ventanas de la aplicación para los escudos de los equipos y las imágenes de cada jugador, dándole un toque mas profesional a la aplicación con prácticamente tres lineas de código[8].

Figura 7: Logo de la librería Glide.

2.1.4. Overleaf

Es una herramienta en linea que permite la edicción de documentos en Latex. He utilizado esta aplicación para elaborar el presente documento. A nivel de presentación, permite realizar diseños más elegantes que los editores habituales.

2.1.5. Photoshop

Es un software que nos permite editar imagenes y cambiar sus parámetros por defecto. La he utilizado básicamente para elaborar el logo de la aplicación.

Figura 8: Logo de la aplicación.

3. Hosts Utilizados

Los dispositivos utilizados para el desarrollo de la aplicación han sido un ordenador portátil y dos dispositivos móviles Android. Se han utilizado varios dispositivos Android a modo de prueba, intentando siempre que sean de diferentes tamaños de pantalla y con distintos procesadores para comprobar el correcto funcionamiento de la aplicación en diferentes entornos. Las características de los dispositivos son las siguientes:

Dispositivo	Ordenador Portátil
Marca	Lenovo
Modelo	IdeaPad 3
Procesador	Intel(R) Core(TM) i5-1035G1 CPU
Memoria Ram	8GB
Tarjeta Grafica	Intel(R) UHD Graphics
Pantalla	1920 x 1080 (32bits)
Sistema Operativo	Windows 10 Home

Cuadro 1: Ordenador Portátil Lenovo

Este ha sido el ordenador donde he programado la aplicación. No he tenido ningún problema a la hora de mover el software Android Studio y he podido utilizarlo con fluidez. La única pega, es que no he conseguido utilizar el simulador de Android Studio, para probar la aplicación según la iba desarrollando (no sé la causa, debería funcionar). Pero no ha sido molestia, ya que conectaba un telefóno movil al ordenador y hacía las pruebas sin ningún tipo de problema.

Dispongo también de un ordenador portátil Accer de hace 6 años con procesador AMD antiguo. Intenté usar Android Studio con él y la experiencia fue muy mala, con constantes congelamientos de pantalla y cierres inesperados, por lo que ha sido de agradecer tener un ordenador más moderno y potente dónde realizar el proyecto.

Dispositivo	Teléfono Móvil
Marca	Xiaomi
Modelo	Mi 10 Lite
Procesador	Octa-core Max 2.4GHz
Memoria Ram	6GB
Almacenamiento	128 GB
Pantalla	6.57”2400×1080
Sistema Operativo	Android 10.0

Cuadro 2: Teléfono Móvil Xiaomi

Este es mi móvil personal dónde he probado la aplicación. Se trata de un smartphone de última generación, con buen procesador y gran pantalla, que nos permite comprobar perfectamente el funcionamiento de la aplicación. La experiencia con él ha sido muy buena. He notado que carga los datos un poco mas rápido que el otro dispositivo dónde hice las pruebas, y que describiré a continuación.

Dispositivo	Teléfono Móvil
Marca	Huawei
Modelo	MYA-L11
Procesador	Quad-core Max 1.4GHz
Memoria Ram	2GB
Almacenamiento	16 GB
Pantalla	4.7”720×1280
Sistema Operativo	Android 6.0

Cuadro 3: Teléfono Móvil Huawei

Este otro dispositivo, a nivel de rendimiento, es mucho menos potente que el anterior. Posee peor procesador y menos capacidad de almacenamiento. En las pruebas realizadas, se ha comportado mejor de lo que esperaba. Sí que he notado que el tiempo de carga de los datos es algo mayor, pero la diferencia es casi inapreciable. Probar la app en este dispositivo me ha servido sobre todo para ver que tal se adaptan los elementos de la aplicación a una pantalla más pequeña.

4. API Rest

Como he mencionado al principio del presente documento, he utilizado dos APIs de las que he consumido los datos. La primera de la que voy hablar se llama SportsDataIO.

Figura 9: API sportsdataIO.

Se trata de una API muy completa (la más grande que he encontrado) que nos da todo tipo de datos de los principales deportes existentes. Posee prácticamente todas las estadísticas que se pueden sacar de la Nba. El problema que tiene es que la mayoría de sus datos están codificados a no ser que pagues una cuota anual. Aún así, de aqui he sacado bastantes datos, y lo he combinado con otra API que explicaré al final de esta sección. Lo primero que debemos de hacer al consumir una API es estudiar la estructura de los datos. En nuestro caso, el servidor SportsDataIO, nos presentaba los datos en Json, en forma de un arreglo con otros objetos Json dentro. Sabiendo esto, preparamos el entorno para recibir estos datos. En concreto de esta API he obtenido los siguientes datos:

- Listado de los equipos de la Liga, con sus escudos, división y conferencia a la que pertenecen.
- De los equipos, además se han obtenido el número de victorias y derrotas, que llevan en la temporada actual. A partir de este dato, se ha elaborado un algoritmo que nos calcula el puesto en la clasificación de un determinado equipo.
- Información del próximo partido al que se enfrenta un equipo. Se incluye datos como por ejemplo la fecha del encuentro o el rival de esa jornada.
- Estadisticas de un equipo en la temporada actual. Para otras temporadas anteriores esta información es de pago y la obtendremos de la otra API.
- Los estadios de cada equipo. Se ha podido obtener también su dirección, capacidad, y las coordenadas. Con este último dato, he podido geolocalizar en Google Maps la ubicación exacta del estadio.
- Las noticias relacionadas con jugadores, equipos, fichajes etc..
- Los jugadores pertenecientes a cada equipo y sus datos personales. Se incluyen foto de perfil, posición y sueldo.

La segunda API de dónde he sacado el resto de datos es Balldontlie. Se trata de una API menos completa que la primera pero que proporciona datos muy interesantes. Destaca por su sencillez y por ser totalmente gratuita.

De Balldontlie he sacado básicamente las estadísticas de los jugadores y equipos. No es una consulta que se realice directamente, ya que al ser dos APIs diferentes, los jugadores tienen identificadores diferentes, y tendremos que realizar una transformación para saber que jugador/equipo de SportsDataIO es en Balldontie.

Los datos que nos da esta api están un poco mal presentados y desordenados, es quizá por esto, que las estadísticas que nos proporciona son gratuitas, a diferencia de otras APIs dónde estos mismos datos son de pago.

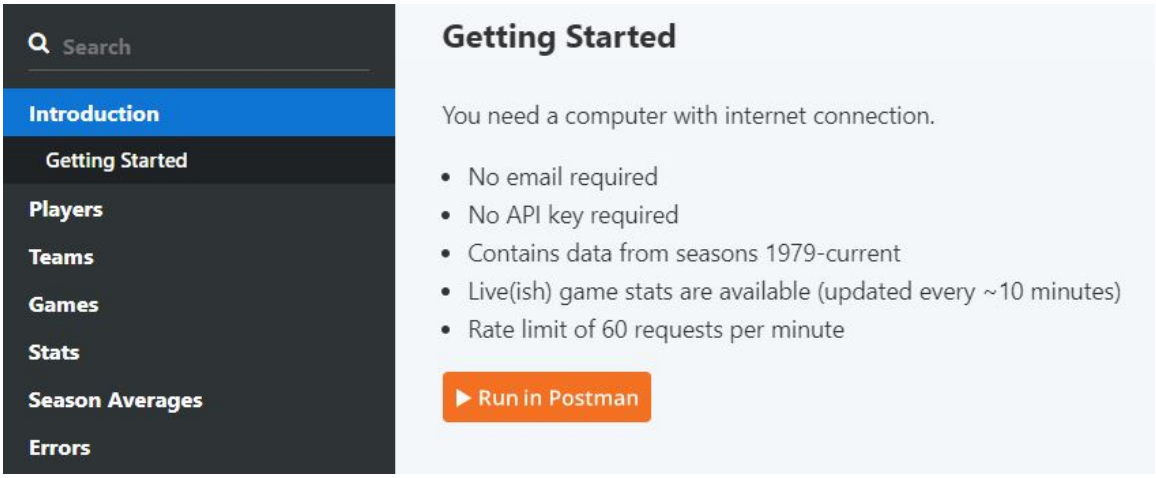

Figura 10: Pantalla de incio de Balldontlie.

Dicho esto, realizando las transformaciones oportunas y ordenado los datos es una API muy útil de la que he sacado bastantes datos:

- Estadisticas generales de un jugador en una temporada (1995-2021). De aqui obtenemos la media de puntos , rebotes, asistencias, porcentajes de tiro, etc.. de los jugadores en la temporada seleccionada.
- Partidos jugados por un jugador en una temporada (1995-2021).
- Estadísticas generales de un jugador en un determinado partido.
- Partidos jugados por un equipo en una temporada (1995-2021).
- Del dato anterior, obtenemos las estadisticas generales de un equipo en temporadas anteriores (información que en la otra API era de pago).

5. Organización del proyecto

5.1. Introducción

En esta sección se va a explicar como se ha estructurado el proyecto. Vamos a describir los usuarios participantes en el proyecto y el alcance del mismo.

5.2. Propósito

El propósito de la organización del proyecto, es obtener una visión global del mismo. El plan de desarrollo de software, proporciona un manual teórico, que permite tener el control sobre el proyecto. Este manual, es susceptible a cambios y mejoras, que permitan tener una mejor visión del proyecto. La parte del análisis de especificaciones técnicas, permiten explicar que métodos se han utilizado para elaborar el plan de desarollo.

5.3. Usuarios

El resultado final de este proyecto es una aplicación Android orientada a dos tipos de usuarios. Por un lado, encontramos al usuario que tiene una base de conocimientos técnicos, y que usará la aplicación para estudiar los datos estadísticos que se le proporcionan. Por otro lado, encontramos usuarios que utilizarán la aplicación con fines informativos, a modo de "curiosear"los datos de la liga e ir siguiendo jornada a jornada a sus equipos/jugadores favoritos.

En otro enfoque, encontramos los usuarios que han participado en el proyecto. Por un lado estaría el alumno, que tiene un rol de desarrollador, y por el otro lado estarían los tutores cuyas labores son las de dirigir y orientar el proyecto. Los describimos a continuación:

- *Alumno:* Es el encargado de llevar a cabo el proyecto. Sus funciones son la de desarrollar la aplicación a partir de unas especificaciones proporcionadas por los tutores, y la de realizar la documentación del proyecto.
- *Tutor 1 :* Este primer tutor tiene conocimientos técnicos Android. Será el encargado de resolver las dudas técnicas que se tengan sobre la aplicación y elegir los mejores caminos (a nivel técnico) para resolver las especificaciones iniciales.
- *Tutor 2 :* Este segundo usuario es el que determina las especificaciones y funciones que tiene la aplicación. Aporta por un lado una visión de usuario común, que utiliza la aplicación para entretenimiento, por lo que decide que interfaz de usuario es la más adecuada. Por otro lado, aporta una visión técnica, ya que va a usar la aplicación para estudios estadísticos. Este segundo tutor no tiene ningún tipo de conocimiento informático.

5.4. Alcance

Para hablar del alcance del proyecto deberemos conocer previamente los objetivos del mismo. En esta subsección definiremos todo lo necesario para alcanzar la aplicación final, describiendo las características del proyecto, sus limitaciones y la metedología utilizada.

5.4.1. Objetivos

En cuanto a los objetivos del proyecto, podemos dividir esta sección en diferentes tipos de objetivos, ya que vamos a tener objetivos para los dos tipos de usuarios que hemos explicado. El objetivo principal del proyecto es desarrollar una aplicación bajo el sistema operativo Android, que permita al usuario consultar los datos estadísticos relativos a la liga NBA. Además, la aplicación será capaz de realizar pronósticos de los partidos de cada jornada de manera eficiente. La plataforma permitirá visualizar los datos de las temporadas 1995-2021. En concreto, se permitirá al usuario conocer las actuaciones de equipos y jugadores, en cada uno de los partidos de la temporada seleccionada en el rango temporal descrito anteriormente (1995-2021). Este objetivo, está propuesto para usuarios que quieran informarse sobre su equipo/jugador favoritos. También podrán consultar otro tipo de información, como pueden ser los estadios, su localización geográfica, clasificaciones y noticias.

Por otro lado, tenemos los objetivos orientados a usuarios con conocimientos técnicos. En este tipo de objetivos destacamos la capacidad de la aplicación para generar archivos en formato CSV (compatible con Excel), con los datos necesarios que precise el usuario. Otra funcionalidad que proporcionamos a nivel técnico para este tipo de usuarios, es la capacidad de calcular la eficiencia que tiene un jugador en una temporada. Este valor, se explicará mas adelante con detalle, se calcula a partir de operaciones matemáticas de los aspectos positivos y negativos que tiene un jugador en el partido, y nos indica la relevancia que tiene en el juego durante una temporada para su equipo.

5.4.2. Limitaciones del proyecto

El proyecto tiene las siguientes restricciones:

- *Limitaciones económicas:* Como hemos comentado al principio del documento, estamos ante un proyecto con un presupuesto de 0 euros, por lo que se precisa que las herramientas utilizadas para llevarlo a cabo sean de software libre. Esto obviamente limita las funcionalidades de la aplicación, pero combinando diferentes herramientas y exprimiendo las versiones libres, hemos podido llevar a cabo un proyecto muy funcional.

- *Limitaciones de integrantes:* Otra de las restricciones que encontramos en este proyecto es la falta de personal. Contamos con un alumno que es el desarrollador y dos tutores que supervisan el proyecto, cada uno en su campo de competencias. Si se hubieran tenido más integrantes, concretamente más desarrolladores, se podría haber conseguido un proyecto más potente en todos los sentidos.

- *Limitaciones de conocimientos:* La experiencia que traía para desarrollar aplicaciones, era la de haber realizado una aplicación sencilla en Android en una asignatura del grado hace aproximadamente un año. En ese pequeño proyecto aprendí nociones básicas, por lo que he tenedio que leer mucha documentación de internet para elevar el nivel de la aplicación de este proyecto . Aún así, es evidente que no se ha podido alcanzar un nivel profesional , pero sí que se ha logrado que las funcionalidades e interfaz de usuario sean las más próximas a este enfoque.

- *Limitaciones temporales:* Este proyecto como trabajo de fin de grado está programado para su realización en unas 300 horas. Si se hubieran tenido más horas para su realización, se podría haber mejorado el resultado final. Aún así, se ha superado el número de horas programado inicialmente como estimación y eso ha ayudado a que la aplicación final sea más completa.

5.4.3. Características del Proyecto

En una visión globalizada, las características principales del proyecto son las siguientes:

- La duración del proyecto se estima en unas 300 horas. En este caso, se ha superado ampliamente ese intervalo de tiempo.
- El desarrollo del proyecto es llevado exclusivamente por parte del alumno. Los tutores son los encargados de la supervisión y aprobación del mismo.
- Todo el proyecto esta documentado en la presente memoria, dónde se especifica claramente el proceso de desarrollo y especificaciones del mismo.
- Es imprescindible que los datos de la aplicación estén actualizados al instante de ser consultados. No vale que se actualizan al finalizar la jornada, ya que hay mucha descompensación en el número de encuentros que juega cada equipo.
- Las específicaciones inciales son susceptibles de cambios, y pueden ser modificadas de acuerdo con las recomendaciones de los tutores.

5.4.4. Metodología Software

El proceso de desarrollar un proyecto de software no es sencillo y no se realiza directamente. Para su correcta elaboración, se precisan una sería de metodologías que faciliten el trabajo al desarrollador. Existen muchas metodologías, las más conocidas son las metodologías ágiles, que como su propio nombre indica, sirven para agilizar las etapas para el desarrollo de software. Las metodologías ágiles se caracterizan por utilizar un sistema de trabajo iterativo o en cascada. Este término destaca por ir presentando al cliente un MVP (Mínimo Producto Viable) cada vez con más funciones hasta que se llegue a la versión final del producto.

Dentro de la metodologías ágiles, existen muchos tipos diferentes. La metodología que se ha utilizado para este proyecto es la metodología Kanban. Su origen se sitúa en Japón, con la producción de la conocida marca de coches Toyota. Esta metodología se caracteriza por utilizar tarjetas para gestionar tareas de manera visual. La clave de utilizar Kanban es que permite saber que tareas se están realizando en el momento, cuáles están pendientes de realizar y cuáles se han finalizado.

La distribución de la información se distribuye de manera sencilla, haciendo tantas columnas como estados tenga el proyecto. Como podemos ver, se trata de una metodología muy sencilla para integrar en un proyecto. Además de su sencillez, en Kaban debemos respetar las siguientes reglas para su correcto funcionamiento:

- **Visualizar frecuentemente el tablero de trabajo:** Esto nos ayudará a saber que tareas tenemos que hacer en cada momento, y cuáles estan finalizadas, dándonos así una perspectiva global del desarrollo del proyecto.
- **Regular la carga de trabajo en cada tarea:** Se deben establecer soluciones viables para cada funcionalidad del proyecto. Una solución demasiado compleja o con mucha carga de trabajo puede llevar a un mal resultado de la misma.

- **Facilidad de lectura:** La utilización de colores nos hará enteneder rápidamente la dificultad de la tarea.
- **Seguimiento temporal:** Deberemos controlar si se están siguiendo las plazos de realización de tareas y entregas para saber el estado en el que se encuentra el proyecto.

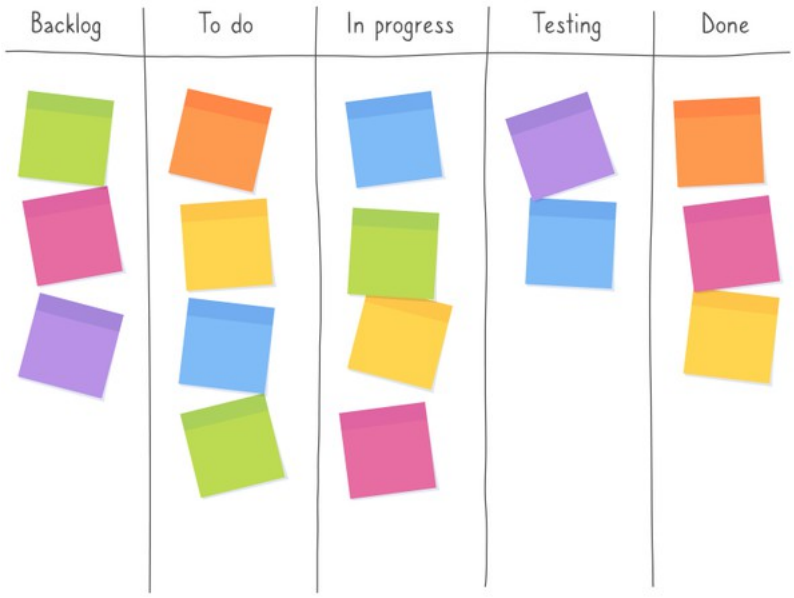

Figura 11: Metodología Kanban.

En la ilustración, se puede observar como debería quedar la organización del proyecto con esta metodología. Es bastante fácil de llevar a cabo y en nuestro caso lo hemos hecho con post-its de colores que simulan perfectamente las tarjetas visuales que carcterizan a la técnica de Kanban. La experiencia ha sido muy buena y creo que es recomendable usar alguna metedología para este tipo de proyectos.

6. Planificación con Kanban

En esta sección se va a explicar cada una de las etapas por las que ha pasado el proyecto con la metodología de Kanban. Hablaremos de los roles que desempeñan los miembros de este proyecto en la metodología, el orden temporal, las especificaciones de cada etapa y el crecimiento progresivo de la aplicación en cada fase del proyecto.

6.1. Roles

Vamos a especificar los roles[13] que se han llevado a cabo en el proyecto . En Kanban, destacamos dos roles diferentes. En primer lugar, tenemos el conocido como Service Request Manager, que es el encargado de analizar las necesidades de los clientes, y por tanto en nuestro proyecto va a definir las especificaciones que tendrá la aplicación. En las reuniones de proyecto, es la persona encargada de ordenar los elementos del trabajo y ordenar prioridades. También se le puede conocer con el nombre de Product Owner.

En segundo lugar, encontramos al Service Delivery Manager, que es el encagado de llevar a cabo el flujo de trabajo, entregando los trabajos y permitiendo llevar una planificación temporal correcta con el resto del equipo. En este rol, se suelen encontrar personas con conocimientos técnicos que solventan las necesidades de los clientes a partir de unas especificaciones definidas por el Service Request Manager.

Para este proyecto, los miembros del mismo han asumido los siguientes roles:

Nombre	Rol
Joaquín Adiego Rodriguez	Service Request Manager
Natalia Martín Cruz	Service Request Manager
Daniel Pariente Bermejo	Service Delivery Manager

Cuadro 4: Roles del Proyecto

Al tratarse de un Trabajo de Fin de Grado, no se dispone de un amplio equipo de personal técnico, sino que se limita a un alumno encargado de desarrollar el proyecto y a dos tutores que supervisan. Aún así, se tiene una persona más en el proyecto de lo normal, ya que estamos acostumbrados en estos casos, a un tutor y un alumno, sin embargo, aquí tenemos a dos tutores, lo que es de agradecer.

6.2. Planificación temporal

Una vez definidos los roles dentro del proyecto y aplicando la metedología explicada, debemos de hacer lo mismo con respecto a las reuniones de trabajo. Existen una serie de tipos de reuniones predefinidas en Kanban[14] que deben realizarse periódicamente. Las reuniones que hemos utilizado en este proyecto son las siguientes:

Reunión	**Descripción**
Reunión inicial	Primera reunión dónde se plantea la idea inicial del proyecto y unas especificaciones bases.
Reunión de reposición	Esta reunión sirve para definir la priorización de las próximas tareas a realizar. Se analizan las dependencias, riesgos y otros aspectos que sean relevantes para la futura tarea. Tras esta reunión se deberá actualizar la tarjetas de próximas tareas.
Daily Kanban	Se trata de una reunión que se realiza diariamente frente al tablero de Kanban. Es una reunión breve en el que se analizan las tareas del día anterior y se afrontan las tareas del día actual.
Revisión de entrega del servicio	Se analizan las tareas entregadas y se comparan con las expectativas. Se estudian posibles mejoras, se identifican desviaciones y causas.

Cuadro 5: Reuniones del Proyecto

Aplicando el método de Kanban, se ha ido realizando un trabajo de manera continua, no se han utilizado plazos o "sprints" como por ejemplo ocurre en la metodología Scrum. Siguiendo las bases de este método, hemos dividido el proyecto en tareas, cada una de ellas con una dificultad y duración diferentes. En la próxima sección dividiremos el proyecto en etapas, donde explicaremos de forma detallada cada una de las tareas llevadas a cabo, con la situación que en ese momento tenía el tablero de Kanban.

El proyecto como veremos próximamente, ha sido dividido en etapas y planificado de acuerdo con la metodología de Kanban. A continuación, podemos ver el calendario de etapas de acuerdo con el orden temporal prefijado para su realización en este curso 2020-2021:

Etapa	Fecha
Etapa 1	15/11/2020 - 20/12/2020
Etapa 2	20/12/2020 - 27/02/2021
Etapa 3	27/02/2021 - 01/04/2021
Etapa 4	01/04/2021 - 28/04/2021
Etapa 5	28/04/2021 - 30/05/2021

Cuadro 6: Planificación del Proyecto

6.3. Estimación de Costes

En este proyecto, no se puede realizar una estimación real de los costes, ya que se trata de un trabajo entre el alumno y los tutores que carece de remuneración en relación al trabajo realizado. Aún así, vamos a hacer un cálculo hipotético del costo del trabajo realizado. Lo primero que tenemos que mirar, es la carga temporal que ha abarcado el proyecto. Regularmente, podemos decir que se ha trabajado de lunes a viernes, aunque esporádicamente se ha utilizado algún fin de semana para seguir avanzando en el proyecto. Haciendo un promedio diario, podemos afirmar que se han empleado aproximadamente 3 horas diarias , por lo que podemos calcular las horas totales de la siguiente manera:

- **5 días a la semana x 3 horas por día x 28 semanas = 420 horas**

Como se puede comprabar la suma total de horas del proyecto supera el número total de horas estimado para la realización del Trabajo de fin de Grado, que se estipula en 300 horas.

Realizando una hipotética estimación de costes en el caso de tratarse de un proyecto real, deberemos de tener en cuenta el sueldo medio de un ingeniero informático por horas en España[36]. Dicha remuneración, está actualmente establecida en 11,4 euros/ hora , por lo que haciendo un cálculo rápido, la remuneración a recibir por parte del alumno por la realización del proyecto sería de:

- **420 h x 11,4 euros/h = 4788 euros**

En cuanto al coste del software utilizado ha sido nulo. Todas las herramientas han sido utilizadas en su versión gratuita, provocando en algunos casos un aumento de la dificultad de desarrollo del proyecto. Se tuvieron que buscar alternativas para mantener el proyecto a coste cero con el mejor resultado posible.

7. Etapas del proyecto

En esta sección vamos a explicar las diferentes etapas por las que ha pasado el proyecto. Aplicando la metodología de Kanban, vamos a ir obteniendo diferentes versiones de la aplicación progresivamente según vayamos cumpliendo tareas.

Como explicamos en la anterior sección, en la figura 11 podemos ver las diferentes clasificaciones que pueden tener las tareas para el proyecto. Vamos a comenzar a explicar cada uno de estos estados y posteriormente analizaremos cada fase por la que ha pasado la aplicación:

- **Backlog :** Aquí vamos a colocar las especificaciones iniciales que va a tener la aplicación. Es susceptible de cambios, ya que se pueden añadir o eliminar funcionalidades según se vaya viendo la viabilidad de las tareas.
- **To do :** Son las tareas pendientes de realizar. Nos ayuda a tener una prespectiva sobre las próximas acciones que tenemos que llevar a cabo dentro del proyecto en los días posteriores.
- **In progress :** Se trata de las tareas que se están realizando en ese momento. Es conveniente poner un límite de tareas que se pueden realizar simultáneamente.
- **Testing :** En este apartado vamos a poner a prueba las tareas que hayamos completado. Se recomienda realizar una batería de pruebas para asegurar el correcto funcionamiento de la nueva funcionalidad añadida.
- **Done :** Una vez que hayamos completado una tarea y nos hayamos asegurado que ha pasado todos los test de pruebas, entonces añadiremos esa función a este apartado.

7.1. Etapa 1: Especificaciones iniciales

Aqui se da oficialmente comienzo al proyecto. Atrás quedan los mensajes intercambiando con los tutores para ponernos de acuerdo con la temática de la aplicación. En estos mensajes se discutía simplemente la temática de la app, ya que teníamos claro que se quería hacer una aplicación para dispositivos móviles en Android. Dicho esto, una primera idea que surgió fue la de una aplicación Android para las principales ligas de fútbol. En un primer momento, se aceptó la propuesta pero rápidamente nos dimos cuenta de que era algo de lo que ya había muchas aplicaciones e incluso algún trabajo de fin de grado de otros años por lo que se optó por buscar otro deporte. Se planteó la idea de hacerlo de las principales ligas de baloncesto. La temática nos gustó a todos por lo que empezamos a trabajar.

7.1.1. Buscando APIs

Con la temática bastante definida, como primera tarea tenía que buscar como leer los datos de las ligas. Mis tutores, me aconsejaron buscar alguna API, para evitar tener que recurrir a la técnica “web scrapping”, ya que era una técnica muy dificultosa y difícil de mantener. Para que se entienda mejor, vamos a explicar lo que es una API. Explicado con nuestras palabras, es un servidor de donde leemos datos que transformamos para su posterior utilización en la aplicación. El trabajo de encontrar APIS para las diferentes ligas no fue nada sencillo.

Antes de empezar a utilizar cualquier API, primero investigaba sobre ella. En concreto, miraba la estructura de sus datos, su funcionamiento, requisitos y si era gratuita. A partir de este momento, ya tenía definido un pequeño tablero de Kanban con unas pocas tareas que daban inicio al proyecto:

Backlog	**To do**	**In progress**	**Testing**	**Done**
Realizar una aplicación que muestre los principales datos de la liga Nba.	-	Recopilar las APIs más apropiadas para el proyecto.	Comprobar la correcta lectura de la API elegida	-

Cuadro 7: Etapa 1. Primera tablero de Kanban

Tras una búsqueda exhaustiva en la que casi todas las APIs eran descartadas por ser de pago, encontré SportsdataIO, Balldontie y RapidApi. Esta última, nos daba datos de equipos, jugadores y de partidos. Se trataba de un API bastante completa y con una muy buena valoración entre la comunidad de internet. Además, te proporcionaba directamente el código que tenías que poner dependiendo del lenguaje que utilizarás, lo que es de especial interés.

Duración de la tarea: 1 semana.

Figura 12: API RapidApi.

¿Por qué no se terminó usando en el proyecto?

En primer lugar, se descartó porque aunque ofertaba un plan gratuito con un número limitado de consultas, tenías que registrar una tarjeta de crédito para poder utilizarlo. Realmente esto no es el principal problema, pero ya supone una pequeña traba ya que SportsdataIO de la que hemos hablado anteriormente, era mas completa y no te pedía registrar ningun dato bancario.

Un segundo problema, era la limitación en el número de consultas. Proporcionaba una cuota máxima

de 100 consultas diarias. Esto era problemático porque en fases de desarrollo del proyecto habría días que podría superar ese límite y no estaba por la labor de estar pendiente de si superaba el limite de cuota o no.

7.1.2. Probando la API

Finalmente se comenzó a utilizar la API SportsdataIO. Su utilización no fue inmediata ni mucho menos. Esta API nos devuelve los datos en formato JSON. El lenguaje JSON es bastante sencillo e intuitivo[15]. A continuación, voy a hacer una pequeña introducción a este lenguaje.

Introducción al lenguaje JSON

JSON es un lenguaje perfecto para el intercambio de datos. Es fácil de entender para los humanos y extremadamente sencillo de procesar para las máquinas. En JSON los datos están distribuidos en dos estructuras distintas:

- *Colección de pares nombre-valor :* Conocido coloquialmente en la mayoría de los lenguajes como "objeto".
- *Lista ordenada de valores :* Es lo que conocemos como arreglo o array en la mayoría de los lenguajes de programación.

Estas dos estructuras son universales, ya que todos los lenguajes de programación de una manera u otra los soportan, por lo que resulta lógico que JSON al ser un intercambiador de datos independiente del lenguaje de programación, utilice estas dos estructuras de datos.

En cuanto a la estructura de un objeto JSON, todos comienzan con una llave de apertura y termina con una llave de cierre. Cada nombre es seguido por dos puntos y los pares nombre/valor están separados por coma:

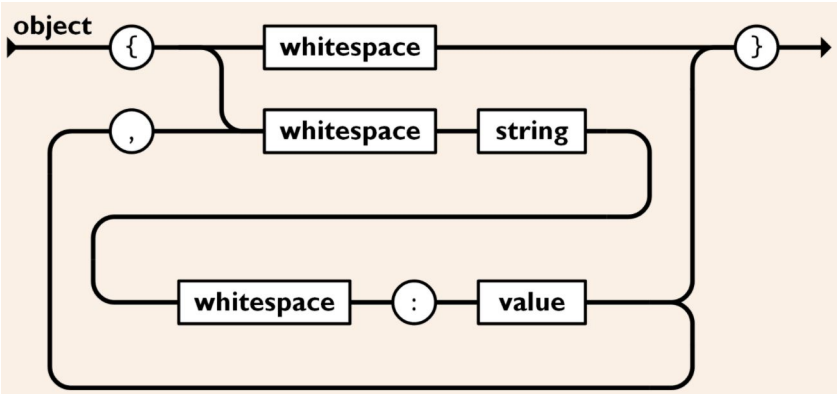

Figura 13: Estructura Objeto JSON.

En cuanto a los Array en JSON, como en todos los lenguajes sirven para almacenar una colección de datos. Es muy común en JSON utilizar Arrays que contengan objetos JSON en su interior. Los arreglos empiezan por corchete en el lado izquierdo y termina con otro corchete en el lado derecho. Los valores se separan por comas:

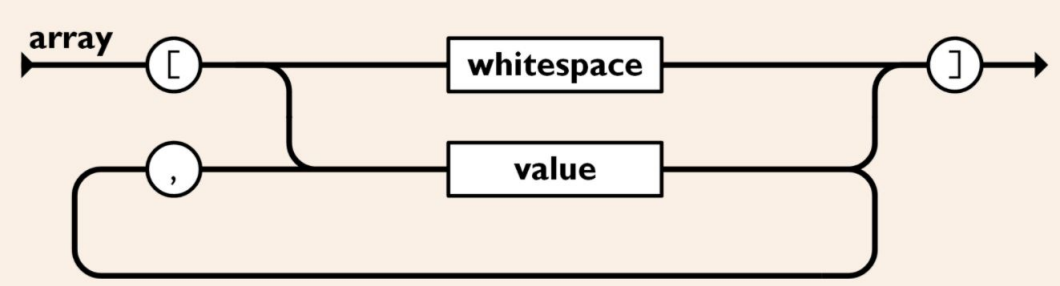

Figura 14: Estructura Array JSON.

Una vez que ya tenemos las nociones básicas para entender el lenguaje JSON, vamos a explicar las pruebas que hicimos para leer datos de la API SportdataIO. Lo primero que hicimos fue comprobar que nos mostraba bien los datos, para comprobarlo lo hicimos de dos maneras:

- **Directa :** Esta es la primera prueba que hice para ver los datos y la más inmediata. Me cree una cuenta y se me proporcionó una API key. Esta API key no es mas que un código alfanumérico que concatenado con una url nos permite visualizar los datos. Combinando la url con mi API key pude visualizar correctamente los datos que estaba demandando. En la siguiente imagen, podemos ver un ejemplo en el que se nos muestra la información de un equipo de la liga. En amarillo aparece resaltado mi API key. Podemos observar también las dos estructuras típicas del lenguaje JSON. En este caso he realizado una consulta a la API que me devuelve un array con muchos objetos, cada uno de ellos es un equipo de la NBA. En la imagen solo se ve un equipo a modo de ejemplo. La estructura de la respuesta es la explicada al principio de la sección, comenzando con un corchete para el array y luego una llave de apertura y cierre para cada objeto(equipo NBA). Dentro del objeto, podemos ver los atributos que contiene (identificador, ciudad, nombre, estadio, conferencia.. etc) y un enlace a un archivo SVG que nos permitirá poner el escudo a cada uno de los equipos. De esta manera tan inmediata se comprobó que la API funcionaba correctamente, y podíamos pasar a la siguiente fase, que consistía en probarla con las librerías especificas de Android.

Figura 15: Consulta a la API SportsdataIO.

- **Indirecta:** Esta prueba es mas compleja, pero también más importante, ya que vamos a mostrar los datos directamente en un dispositivo móvil. Utilizaremos la librería Retrofit que nos proporciona Android y que nos permite leer correctamente objetos JSON. Este proceso se conoce como REST API, y es muy utilizado en las aplicaciones Android. Lo primero que hay que hacer es definir una interfaz que contendrá la estructura de datos de la API y la forma de tratarlos. En dicha interfaz deberemos indicar la forma de tratar esos datos. En nuestro caso, le indicaremos que se trata de un Array de objetos JSON.

  Una vez preparada la interfaz, deberemos crear un objeto Retrofit y utilizar la libreria GsonConverterFactory que actúa como un convertidor de JSON a objetos Java. Con este método, pudimos leer todos los atributos de un objeto JSON y comprobar mediante logs que la app esta leyendo los datos adecuadamente.

  Duración de la tarea: 2 semanas.

7.1.3. Integrando los datos

Tras comprobar que la Api funcionaba correctamente y que los datos se visualizaban correctamente en la aplicación, la siguiente tarea que tenía que realizar era hacer visuales a esos datos. En concreto, quería mostrar una ventana con todos los equipos de la liga. Con este planteamiento actualizamos el tablero de Kanban nuevamente:

Backlog	**To do**	**In progress**	**Testing**	**Done**
Realizar una aplicación que muestre los principales datos de la liga Nba.	Dar funcionalidad al pulsar a cada equipo, control de eventos.	Mostrar todos los equipos con sus datos en la pantalla de la aplicación.	Comprobar la correcta visualización de todos los eventos.	✓Recopilar las APIS más apropiadas para el proyecto.

Cuadro 8: Etapa 1. Segundo tablero de Kanban

Una vez que hemos conseguido leer los datos en la aplicación, en esta tarea tenemos que mostrarlos correctamente al usuario.

Realmente, esta tarea es la más compleja e importante de la aplicación, porque vamos a seguir el mismo método para mostrar jugadores, partidos etc..

Exige una mayor inversión temporal, ya que deberemos usar estructuras de datos del tipo LiveData[15], para que los datos estén actualizados en todo momento. Dicha estructura era totalmente desconocida por mí y ha sido muy usada en la aplicación para mostrar equipos, jugadores y partidos.

A continuación, mostramos un esquema del funcionamiento de este tipo de objetos Java:

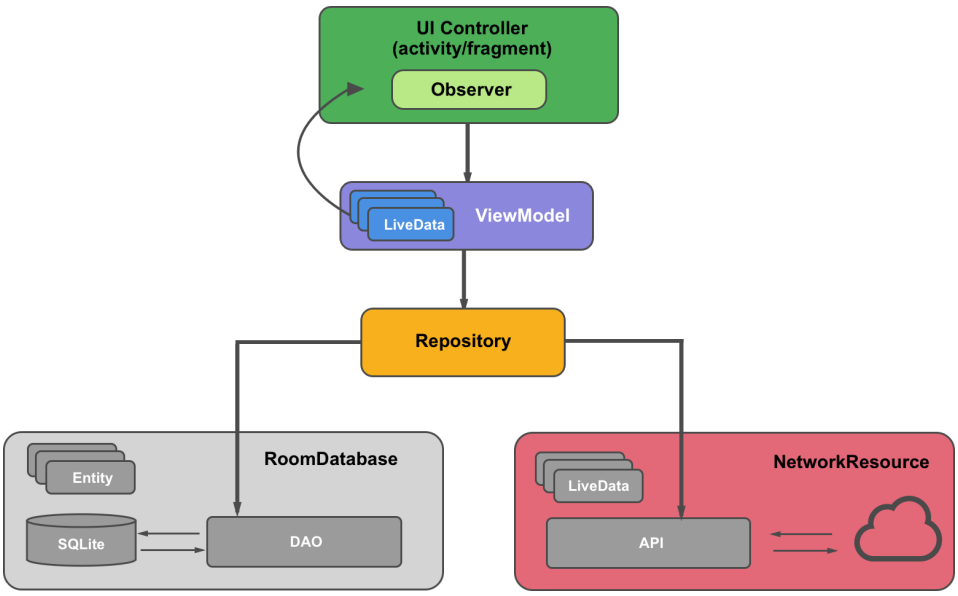

Figura 16: Esquema de LiveData.

En el esquema podemos ver que tenemos una API de donde consumimos la información. Esta información la almacenados en la clase ViewModel, que al contener estructuras del tipo LiveData permite acceder a todas las clases de la aplicación a la información almacenada. Por último, tenemos el observador que notifica a toda la aplicación de cualquier cambio realizado en el objeto del tipo LiveData. Con esta estructura podremos ir creando dinamicamente los equipos, jugadores o partidos e ir mostrándolos al usuario con todos sus datos.

Duración de la tarea: 4 semanas.

Tras la finalización de todas las tareas de la etapa 1, mostramos una tabla a modo resumen que recoge la temporalidad de cada una de las tareas:

Tarea	**Duración**
Búsqueda de APIs	1 semana
Pruebas APIs	2 semanas
Integración en la App	4 semanas
Total	**7 semanas**

Cuadro 9: Resumen Etapa 1

7.2. Etapa 2: Iteración entre ventanas

Una vez que tenemos una versión de la aplicación en la que leemos correctamente los datos de la API (listado de equipos), el siguiente paso corresponde a asignar eventos al pulsar a cada uno de los equipos. Además, siguiendo con el manejo de eventos entre ventanas, se implementará un menú inferior que nos redirija a diferentes ventanas. Estas dos tareas constituyen la etapa 2 del desarrollo de la aplicación.

7.2.1. Añadiento Eventos

El objetivo de esta tarea es redirigir al usuario a una nueva ventana que muestre información del equipo seleccionado. En este caso, realizar un redirecionamiento al pulsar un equipo es sencillo, lo más complicado en este proceso, es conocer que equipo ha pulsado el usuario y en base a esto realizar las acciones pertinentes. Con esta nueva información, actualizamos el tablero de Kanban con las nuevas tareas para realizar:

Backlog	**To do**	**In progress**	**Testing**	**Done**
Realizar una aplicación que muestre los principales datos de la liga Nba.	- Desarrollar un menú inferior que maneje la sucesión de eventos.	- Dar funcionalidad al pulsar a cada equipo, control de eventos.	- Comprobar la correcta transicción al elegir un equipo - Comprobar la correcta visualización de todos los eventos.	✓Recopilar las APIS más apropiadas para el proyecto. ✓Mostrar todos los equipos con sus datos en la pantalla de la aplicacion.

Cuadro 10: Etapa 2. Primer tablero de Kanban

Dicho esto, cuando el usuario pulsa un equipo, pretendemos que se abra otra ventana dónde mostremos la plantilla con los jugadores de ese equipo. Se pretende que para cada jugador, se muestre una foto de perfil que permita confirmar la identidad del propio individuo, su posición en la cancha, y a modo de referencia del reconocimiento que tiene en la liga, el sueldo que percibe.

Para conocer todos estos datos necesitamos realizar una nueva consulta a la API, indicando el equipo que ha elegido el usuario y utilizar el procedimiento explicado anteriormente en el esquema de funcionamiento de los objetos del tipo LiveData. Cuando el usuario pulsa sobre un equipo, a vistas del usuario abrimos una nueva ventana con los jugadores de ese equipo. En la perspectiva del desarrollador enviamos a la clase responsable de crear la nueva ventana los datos del equipo. Entre los datos que mandamos destacamos la clave del equipo, que no es más que una abreviatura del nombre del equipo con el que lo identificamos y la foto que contiene el escudo del equipo. Con estos datos, la nueva clase realiza una consulta a la API y por medio de la clave del equipo obtenemos todos los jugadores del equipo y los mostramos en pantalla.

Este proceso no fue muy complejo, el único problema fue entender la sintaxis de la bilioteca Retrofit para meter parámetros a las url al realizar las consultas a la API. Una vez que conseguimos utilizar la clave del equipo como parámetro en la url, se mostraban sin problemas todos los jugadores de cada uno de los equipos.

Duración de la tarea: 2 semanas.

7.2.2. Menu inferior

Una vez que tenemos la ventana principal de la aplicación, dónde el usuario puede seleccionar caulquier equipo y ver sus datos, la siguiente tarea es crear un pequeño menú en la parte inferior que permita redirigirnos a distintas pestañas, dónde veremos diferentes tipos de información pero siempre relacionadas con el equipo elegido por el usuario. Como explicaré a continuación, esta ha sido una tarea complicada y mientras pulía su correcto funcionamiento, se ha ido trabajado simultáneamente en mejorar la estética de las ventanas descritas anteriormente.

El menú inferior va a constar de cuatro pestañas diferentes. En cada una de ellas tendremos un icono

ilustrativo y un nombre que definen la temática de los datos del equipos que se van a mostrar. Las cuatro pestañas elegidas son las siguientes:

- *Plantilla:* En esta pestaña podremos consultar los jugadores del equipo seleccionado. Es la pestaña inicial que se muestra al pulsar en un equipo.
- *Estadísticas:* Muestra las estadísticas generales del equipo para una determinada temporada.
- *Pizarra:* Sirve para que la app, a partir de un algortimo, nos proporcione en base a estadísticas pasadas, que jugador es el ídoneo para sustituir a otro en una posición que no es la suya por defecto.
- *Estadio:* Porporciona información acerca del estadio del equipo. En esta información se mostrarán detalles sobre su posición geográfica.

Cuando seleccionamos una de estas pestañas, el icono se oscurece y se resalta el nombre de la pestaña para indicar al usuario que opción ha elegido. Para realizar las pruebas de funcionamiento teníamos preparadas 3 ventanas una para cada pestaña del menu inferior con algun texto identificativo, así podíamos comprobar el correcto funcionamiento al cambiar de pestañas. Estas transicciones dieron problemas, básicamente porque al cambiar de una pestaña a otra teníamos que indicar a la app que pestaña estaba elegida, deseleccionar las otras y pasar todos los datos a la nueva ventana. Mientras trabajaba en esto para no quedarme muy atascado iba mejorando la estética de las otras ventanas.

Backlog	**To do**	**In progress**	**Testing**	**Done**
Realizar una aplicación que muestre los principales datos de la liga Nba.	Desarrollar ventanas para las pestañas del menu inferior.	-Desarrollar menù inferior. -Trabajar en la estética de las ventanas.	Probar la transicción entre ventanas al cambiar de pestañas en el menu inferior.	✓Recopilar las APIS más apropiadas para el proyecto. ✓Mostrar todos los equipos con sus datos en la pantalla de la aplicación ✓Dar funcionalidad al pulsar a cada equipo, control de eventos

Cuadro 11: Etapa 2. Segundo tablero de Kanban

Trabajando en los diseños

El proceso de desarrollo del menú inferior no fue sencillo, llegando en muchas ocasiones a quedarme "atascado", es por esta razón que me pareció una buena idea combinar el desarrollo del menú inferior, con la mejora de la estética de la aplicación.

El diseño de la aplicación es una de las partes más importantes en el desarrollo de apps Android. No nos tenemos que ir muy lejos para comprobarlo. Por ejemplo, si nos fijamos en una de las grandes marcas de telefonía, como puede ser Apple, vemos que exige que todas las aplicaciones que estén en su mercado(App Store) pasen una serie de filtros que garanticen la calidad a nivel de diseño de la app[17]. En el caso de nuestro proyecto nos hemos basado en los diseños de esta marca. Se ha buscado en todo momento seguir una corriente minimalista, que busque la sencillez en el diseño de las ventanas.

Entrando en detalles, a nivel de diseño, en todas las ventanas de la aplicación se han empleado tonos blancos y grises, buscando en la medida de lo posible la sencillez de la aplicación. En este enfoque minímalista, incluimos también el logotipo de la app, diseñado por nosotros mísmos.

Para terminar, como mostraremos en las siguientes secciones, se ha optado por utilizar un sistema de tarjetas para mostrar los datos. Este aspecto sirve para diferenciar cada dato y le da un toque más realista a la app. En nuestro caso, por ejemplo en el listado de equipos, cada equipo será una tarjeta con unos margenes definidos y en cuyo interior se muestran datos sobre el equipo en cuestión.

Figura 17: Ejemplo de una tarjeta de un equipo de la liga.

Duración de la tarea (Desarrollo menú inferior + Estética) : 3 semanas.

7.2.3. Ventanas del menú inferior

Una vez que tenemos un menú inferior totalmente funcional, en el que hemos probado el correcto cambio entre ventanas al pulsar las diferentes pestañas, el siguiente paso es trabajar en las nuevas ventanas. Se irá haciendo progresivamente en el orden que marcan las pestañas del menú. Con la ventana de "Plantilla" terminada, habrá que centrarse en las tres restantes, comenzando por la de "Estadísticas", continuando por la de "Pizarra" y finalizando con la de "Estadio". Un primer tablero de Kanban en esta tarea sería el siguiente:

Backlog	**To do**	**In progress**	**Testing**	**Done**
Realizar una aplicación que muestre los principales datos de la liga Nba.	- Ventana Pizarra. - Ventana Estadio.	Desarrollar ventana Estadísticas.	Comprobar progresivamente ventanas asociadas a menú inferior.	✓Recopilar APIS. ✓Mostrar listado de equipos. ✓Control de eventos. ✓Menú inferior

Cuadro 12: Etapa 2. Tercero tablero de Kanban

Ventana Estadísticas Equipo

En esta ventana se van a mostrar las estadísticas completas del equipo seleccionado para la temporada actual. Pondremos el logo del equipo, cuya imagen es enviada a todas las clases Java que intervienen en el proceso hasta llegar a la nueva ventana. Los datos estadísticos que aportamos para cada equipo para la temporada actual son los siguientes:

- *Victorias:* Número total de victorias en la temporada.
- *Derrotas:* Número total de derrotas en la temporada.
- *Racha:* Partidos consecutivos ganados o perdidos. Se designa por W y un número una racha de victorias, por ejemplo W3 indica que la racha es de 3 victorias seguidas. Para las derrotas se aplica el mismo procedimiento pero con la letra L.
- *%Vict:* Es el porcentaje de victorias en relación al número de partidos jugados.
- *Pts +:* Media de puntos anotados en cada partido.
- *Pts -:* Media de puntos que le anotan en cada partido.
- *Ranking:* Posición en la clasificación.
- *Liderato:* Victorias que le faltan para llegar al liderato de la clasificación.
- *VictLocal:* Victorias en casa.
- *VictFuera:* Victorias fuera de casa.

- *DerrLocal:* Derrotas en casa.
- *DerrFuera:* Derrotas fuera de casa.

Todos estos datos los obtenemos de la API SportsdataIO porque son datos de la temporada actual. Si quisieramos obener datos de otra temporada, deberemos recurrir a la API Balldontie.

Esta tarea ha sido bastante sencilla, simplemtente teníamos que leer los datos de la API directamente sin tener que recurrir a objetos del tipo LiveData, lo que simplificaba bastante el proceso.

Duración de la tarea: 1 semana.

7.2.4. Ventana Pizarra. Algoritmo de elección

La siguiente ventana en la que tenemos que trabajar es la que hemos denominado "Pizarra". Este nombre se ha asignado debido a que en esta pestaña vamos a poder realizar una labor de entrenadores, de ahí que se llame "Pizarra" refiriendose a la típica pizarra que tienen los entrenadores. La idea de esta funcionalidad, es proveer al usuario de un método de sustitución de jugadores que estén lesionados o sean baja para el partido. En otras palabras, el usuario elegirá una posición (Base, Escolta, Alero, Ala-Pivot, Pivot) y la aplicación le dirá en base a datos estadísticos de la temporada actual, quién es el jugador más idóneo para cubrir esa posición, sin ser esta su posición habitual durante la temporada.

Esta función puede ser muy útil para determinados partidos en los que en alguna posición por determinadas circunstancias (lesión, enfermedades o sanción) no tengamos jugadores. El usuario tan solo con elegir la posición le dará el nombre del jugador que podría sustituir a los jugadores lesionados con un argumento estadístico.

Esta tarea es algo más compleja, ya que tenemos que leer datos de la API y operar con ellos. Desarrollaremos un algortimo que a partir de los datos estadísticos, sepa decirnos que jugador es más apto para cada una de las posiciones.

Explicación de Algoritmo de elección.

Como hemos comentado, el primer paso es que el usuario/entrenador elija una posición. Una vez que tenemos este dato, ya podemos empezar a trabajar. Para cada posición, el algoritmo realiza unos cálculos diferentes pero hay un proceso que es común sea cual sea la opción elegida. Este camino común, consiste en mirar a todos los jugadores del equipo que ocupen una posición diferente a la elegida por el usuario. Esto es bastante lógico, ya que solo deberemos de tener en cuenta los jugadores que no ocupen la posición que se elije, ya que se supone que son jugadores que no están acostumbrados a jugar en otra posición pero que estadisticamente se demuestra que pueden desempeñar ese rol perfectamente.

Una vez que tenemos la lista de estos jugadores, ya sí que distinguimos entre posiciones que variable analizar y las operaciones a reliazar sobre ellas. Vamos a explicar lo que hace el algoritmo para cada posición en la cancha de baloncesto:

- **Base:** Los bases son jugadores que por lo general tienen un gran manejo de balón y visión del juego. Estas dos facetas les permiten ser el "cerebro" del equipo y marcar los ritmos de los partidos. Además, son jugadores de baja estatura y ágiles por lo que ya nos podemos imaginar que a los pivots los tenemos que descartar de esta ecuación. Teniendo en cuenta esto, hemos creado un ratio Asistencias/Pérdidas. El algoritmo, obtiene la lista de jugadores que no son bases ni pivots y cálcula este ratio para cada uno de ellos. El jugador con mejor ratio, es el elegido por el algoritmo y mostrado al usuario.

- **Alero:** Los aleros son jugadores con un buen tiro de dos puntos. Suelen caracterizarse por hacer penetraciones a canasta que terminen en bandejas o mates. Suelen ser jugadores altos que habitualmente suman una buena cantidas de rebotes. El algoritmo en este caso coge la lista de jugadores que no sean ni aleros ni pivots. Se evitan coger Pivots porque son jugadores grandes y lentos que no pueden en ningún caso desempeñar la labor de los aleros. Dicho esto, se crea un ratio que suma el número de rebotes/10 + el porcentaje de tiro de dos puntos. El valor mayor obtenido entre todos los jugadores nos indica el jugador más idóneo para jugar en la posición de alero.

- **Escolta:** Los escoltas son los jugadores con mejor tiro exterior del equipo. Son los encargados de intentar anotar triples y canastas decisivas que decanten los partidos. Suelen ser jugadores anotadores que meten bastantes puntos por partido. Como en las otras posiciones el algoritmo coge la lista de jugadores que no son escoltas. Pero en este caso, es algo más complejo, ya que primeramente se mira que haya realizado una cantidad de lanzamientos razonable y acto seguido se comprueba su porcentaje de tres. Para terminar se mira el número de puntos que anota ese jugador por partido. Con estas condiciones evitamos coger a jugadores que en toda la temporada hayan tirado uno o dos triples (no son tiradores) y por suerte hayan anotado esos tiros. Las tres variables que se analizan garantizan que el nombre del jugador proporcionado por algortimo sea el de un jugador anotador y buen tirador desde la línea de tres.

- **Pivot:** Son los jugadores más altos del equipo, los que mejor defienden, y por tanto los que más rebotes y tapones consiguen. El ratio que utiliza el algoritmo es obtener un valor a partir de la suma de rebotes y tapones de cada jugador por partido. Aqui no es necesario "vetar" posiciones (a excepción de los propios pivots) ya que por lo general un base por ejemplo que es de estatura baja no va a tener un buen promedio de rebotes y tapones ya que los jugadores más altos que él lo superarán en la mayoría de los casos.

- **Ala-Pivot:** Son jugadores similares a los pivots pero con mejor porcentaje de tiro. En este caso vamos a utilizar la misma condición que con los pivots, aplicando el ratio tapones+ rebotes pero en este caso exceptuamos a pivots y ala-pivots por lo que garantizamos que el jugador seleccionado sea un alero con una buena capacidad para defender y anotar desde cerca de la zona, que es exactamente lo que buscamos en esta posición.

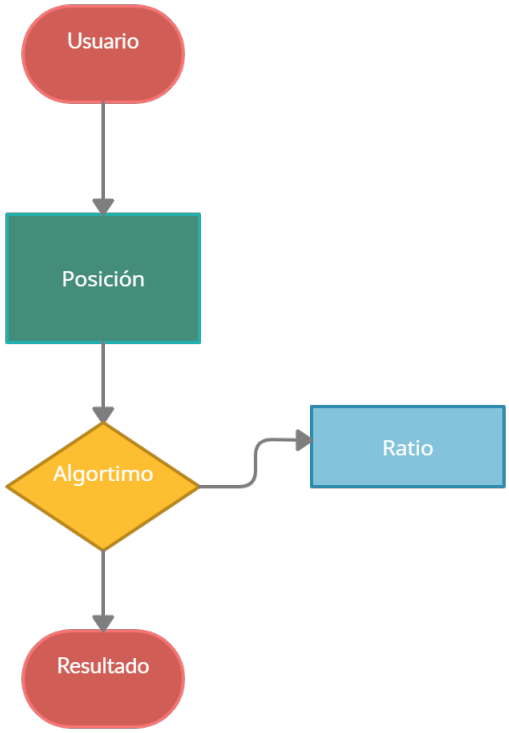

Figura 18: Esquema de algortimo de elección.

Elaborar este algoritmo fue costoso, en cuanto a dificultad de implementación y a como dar con la fórmula que nos porporcionase unos resultados correctos. Debido a que para cada posición en el juego utilizabamos un ratio diferente, y a las pruebas que ello conlleva, esta tarea se alargó más de lo previsto.

Duración de la tarea: 4 semanas

7.2.5. Ventana Estadio

Esta ventana es la última pestaña correspondiente al menú inferior. Es una tarea que se divide en dos subtareas. Por un lado, tenemos que leer la información del estadio del equipo que haya elegido el usuario y mostrarlo por pantalla. Esta es una tarea "sencilla" en el sentido de que tenemos que seguir el mismo proceso para leer información de la API, como hemos hecho otras veces. La segunda subtarea, consiste en mostrar la posición geográfica del estadio a través de Google Maps. Esta tarea lógicamente es más costosa y no se ha realizado nada parecido anteriormente en el proyecto, por lo que nos llevó más tiempo. Vamos a analizar por separado ambas subtareas:

- *Mostrando información del estadio:* Como en otras ocasiones, pasamos entre las diferentes clases Java la clave del equipo (a modo de identificador) que ha seleccionado el usuario. Con ello, hacemos la consulta a la API y mostramos la información del estadio. Entre los datos que proporcionamos, destacan el nombre completo del estadio, la dirección, la localidad y la capacidad.

  Con la capacidad de cada estadio, hemos realizado un pequeño cálculo que tiene que ver con la presión que puede llegar a ejercer el público en dicho estadio. Dicho cálculo consiste en hacer un promedio de los espectadores que acuden a cada uno de los estadios. Con este valor promedio, podemos comparar si el estadio elegido por el usuario tiene mayor tasa de capacidad o menor que la media de los estadios de la liga. Si el estadio posee una mayor capacidad, se advierte que la presión ejercida por el público en los partidos de ese estadio puede favorecer al equipo local, ya que habrá una mayor cantidad de espectadores en relación al número de afcionados a los que están acostumbrados los jugadores en el resto de estadios de la liga.

- *Geolocalizando el estadio:* Esta opción se ha implementado porque le va a dar un toque más profesional a la aplicación. En la práctica, no es una función de mucha utilidad, pero sí lo es a modo de curiosidad, ya que puedes indagar la zona donde se sitúa cada estadio sin problemas en un mapa bastante visual.

  Para llevar a cabo la geolocalización, solo necesitamos dos parámetros, la latitud y altitud. Estos dos parametros lógicamente los leemos de la API. Para mostrar el mapa al usuario utilizaremos los servicios de Google. Siguiendo las instrucciones para desarrolladores [20], añadimos las dependecias, implementamos los métodos necesarios y elegimos el tipo de mapa que queremos mostrar en nuestra aplicación.

Duración de la tarea: 1 semana.

Tras la finalización de todas las tareas de la etapa 2, mostramos una tabla a modo resumen que recoge la temporalidad de cada una de las tareas:

Tarea	Duración
Control de eventos	2 semanas
Menu inferior + Estética	3 semanas
Ventana Estadísticas Equipo	1 semana
Ventana Pizarra	4 semanas
Ventana Estadio	1 semana
Total	**11 semanas**

Cuadro 13: Resumen Etapa 2

7.3. Etapa 3: Añadiendo nuevas funcionalidades

En esta etapa utilizaremos los datos que nos proporciona la segunda API Balldontie. LLegados a este punto, tenemos una versión de la aplicación bastante completa y funcional. Para comenzar, implementaremos una pantalla que muestra las estadísticas de un jugador al pulsarlo. En esa misma ventana, añadiremos un botón que nos permite ir a otra ventana que nos muestre los partidos que ha jugado dicho jugador. Por último, implementaremos eso para cualquier temporada desde el año 1995 hasta la actualidad.

7.3.1. Estadísticas del Jugador

La primera tarea que tenemos que llevar a cabo en esta tercera etapa, es mostrar las estadísticas de cada jugador. Cuando el usuario pulsa en un jugador de un equipo, con el control de eventos haremos que se abra una nueva ventana dónde veremos las estadísticas de la temporada actual. Esto a priori, podemos pensar que es sencillo porque sólo deberíamos leer los datos de la API Balldontie como hemos hecho en otras ocasiones. Sin embargo, tenemos un problema ya que al seleccionar un jugador, dicho jugador está siendo mostrado con sus datos a través de una consulta a la API SportsdataIO y su identificador no se corresponde con el que tiene en la API Balldontlie, que es la API que tenemos que usar para mostrar sus estadísticas. Para solucionar esto, deberemos de realizar una conversión de datos como explicaremos en la proxima sección.

Backlog	To do	In progress	Testing	Done
Realizar una aplicación que muestre los principales datos de la liga Nba.	-Estadísticas de Jugador. -Partidos de Jugador -Añadir más temporadas.	Control de eventos al pulsar en un jugador, mostrar sus estadísticas en la temporada actual.	Comprobar la correcta visualización de las estadísticas individuales de cada jugador.	✓Recopilar APIS. ✓Mostrar listado de equipos. ✓Control de eventos. ✓Menú Inferior. ✓Ventanas Menú inferior

Cuadro 14: Etapa 3. Primer tablero de Kanban

Fusionando APIs

Como hemos explicado, mostrar las estadísticas de la temporada actual, no es un proceso inmediato. El proceso es algo complejo, por lo que va a ser explicado de manera general sin entrar en demasiados detalles técnicos.

Lo primero que hacemos cuando el usuario elige un jugador es recoger el nombre y apellidos del jugador, concatenarlos y guardarlos en una variable. A continuación, realizamos una consulta a la API Balldontie utilizando como parámetro el nombre y apellidos del jugador obteniendo el identificador numérico del jugador seleccionado. Una vez que tenemos dicho identificador , hacemos otra consulta a la misma API esta vez tomando como parámetro el identificador obtenido anteriormente y el año de la temporada.

Lo ideal hubiera sido que en ambas APIs los jugadores tuvieran el mismo identificador y así no habría que realizar esta transformación. También nos hubiera ahorrado trabajo que la API Balldontie a partir de un nombre ya nos mostrase las estadisticas del jugador pero esto no era posible, se precisaba explicitamente el identificador del jugador por lo que no nos quedaba más remedio que llevar a cabo este laborioso proceso.

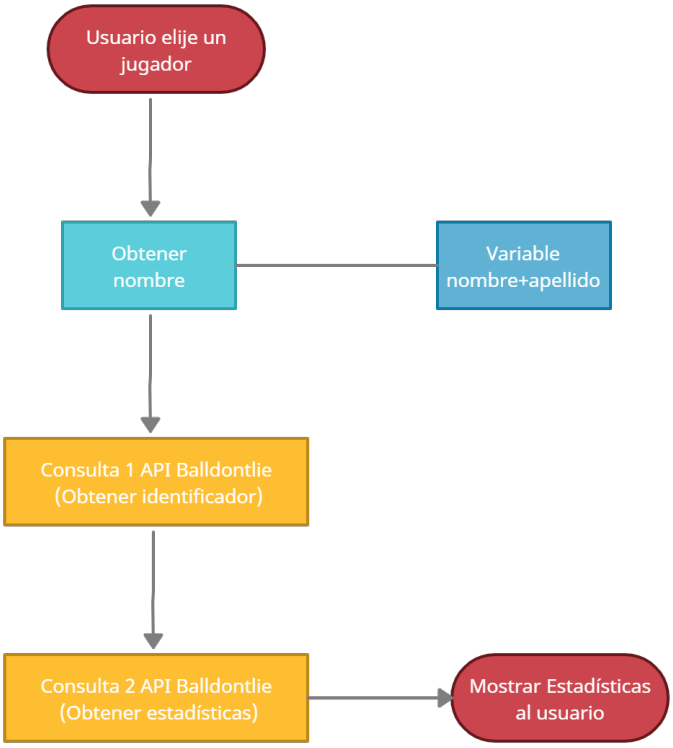

Figura 19: Esquema de fusión de APIs.

Los test de esta tarea fueron de gran importancia ya que en el proceso intervenían muchas variables y hubo que realizar muchas pruebas hasta que se comprobó que todo funcionaba de manera correcta. Una vez que se consiguió esto, se procedió a implementar las temporadas desde 1995. Mostrar otras temporadas fue el menor de los problemas ya que en la ventana el usuario tenía un spinner[21] donde elegía la temporada que quería visualizar.

Datos mostrados y variable eficiencia

Las variables estadísticas de un jugador para una temporada en concreto determinan su actuación personal a lo largo de la temporada. A partir de estos datos, hemos creado una variable llamada "eficiencia", que por medio de un número, nos indica como de relevante es un jugador para el juego de su equipo. Las variables que mostramos al usuario son son las siguientes:

- *Partidos:* Muestra el número de partidos jugados.
- *Minutos:* Se trata del número promedio de minutos que juega el jugador en cada uno de los partidos.
- *Puntos:* Promedio de puntos por partido.
- *Fgm:* Promedio de número de tiros de campo anotados.
- *Fga:* Promedio de número de tiros de campo intentados.
- *Ftm:* Promedio de número de tiros libres anotados.
- *Fta:* Promedio de número de tiros libres intentados.
- *Fg3m:* Promedio de número de tiros de tres anotados.
- *Fg3a:* Promedio de número de tiros de tres intentados.
- *Rebotes:* Promedio de rebotes por partido.
- *RebotesO:* Promedio de rebotes ofensivos por partido.
- *RebotesD:* Promedio de rebotes defensivos por partido.
- *Robos:* Promedio de robos por partido.
- *Asistencias:* Promedio de asistencias por partido.
- *Tapones:* Promedio de tapones por partido.
- *Pérdidas:* Promedio de pérdidas por partido.
- *Fg_pct:* Porcentaje de tiros de campo anotados.
- *Ft_pct:* Porcentaje de tiros libre anotados.
- *Fg3_pct:* Porcentaje de tiros de tres anotados.
- *Efc:* Eficiencia de un jugador en el juego.

La variable eficiencia

Se trata de una variable que hemos creado a partir de las demás. Su principal cometido es ser un comparador estadístico del valor general de los jugadores. La eficiencia tiene en cuenta las aspectos ofensivas del juego del jugador (puntos, asistencias, rebotes ofensivos)y las defensivas (robos, tapones, rebotes defensivos) [22]. Su fórmula es la siguiente:

$$(PTS + REB + AST + ROB + TAP - TCFallados - TLFallados - PÉR)/PJ$$

Dicha fórmula se dió a conocer por el periodista deportivo de Kansas City y estatista Martin Manley (1953–2013).

Los equipos utilizan esta variable para ver cómo es el rendimiento de sus jugadores. Los equipos que quieren optar al título deberan tener al menos dos jugadores con una eficiencia mayor de 20. En nuestra aplicación hemos denotado la eficiencia en tres colores:

- **Verde:** Si la eficiencia adquiere un valor mayor de 15 aparecerá el color verde. Se considera que la media de jugadores de la liga tienen unos valores de eficiencia en torno a 15 por lo que un valor mayor que este lo consideramos muy óptimo.
- **Naranja:** Si el valor de eficiencia está entre 10-15 aparecerá en color anaranjado. Esto significa que el jugador está un poco por debajo de la media del resto de jugadores, aunque aún así puede ser un jugador importante que aporte rendimiento al equipo.
- **Rojo:** Si el valor de la eficiencia aparece en rojo será una mala señal para ese jugador. Significa que la variable tiene un valor por debajo de 10, lo que se traduce en que ese jugador aporta muy poco al rendimiento del equipo y se puede prescindir de él.

Duración de la tarea: 2 semanas.

Añadiendo partidos

Una vez que tenemos una ventana dónde vemos las estadísticas individuales del jugador seleccionado para la temporada actual, el siguiente paso es implementar un boton que nos abra otra ventana dónde veamos los partidos que ha jugado hasta la fecha. Esta tarea tuvo como único problema ordenar los partidos jugados temporalmente, ya que el resto de proceso para implementarlo fue similar a otras tareas ya realizadas.

Backlog	**To do**	**In progress**	**Testing**	**Done**
Realizar una aplicación que muestre los principales datos de la liga Nba.	Añadir todas las temporadas a las estadísticas equipo.	Implementar visualización de partidos jugados a cada jugador para cada temporada.	Comprobar que los partidos mostrados coinciden temporalmente con el calendario real de la competición.	✓Recopilar APIS. ✓Mostrar listado de equipos. ✓Control de eventos. ✓Menú Inferior. ✓Ventanas Menú inferior ✓Estadísticas jugador.

Cuadro 15: Etapa 3. Segundo tablero de Kanban

El mostrar todos los partidos jugados por un jugador en la temporada seleccionada no fue un problema. Si tenemos en cuenta que el usuario en la pantalla por medio del spinner elegía la temporada, sólo teníamos que añadir un botón que al ser pulsado hiciese la consulta a Balldontlie, tomando como parámetro la temporada seleccionada. Efectivamente aparecían correctamente los partidos disputados en esa temporada, pero de aquí surgían dos problemas:

- *Partidos desordenados:* Incomprensiblemente todos los partidos aparecían desordenados. No tenía mucho sentido mostrar los partidos en un orden aleatorio al usuario, porque si estábamos mostrando esta información era para hacernos una idea de la evolución que ha llevado el jugador en cada uno de los partidos progresivamente. Realizar la ordenación temporal no fue sencillo, ya que no aparecía la fecha de los encuentros en ninguno de los atributos de la API. Me dí cuenta de que cada partido tenía un atributo que era un número a modo de identificador como ocurría en los jugadores. En estos últimos, esos números no tenían relación pero en este caso sí que la había. En concreto, dichos identificadores adquirían valores mayores cuándo el partido era más reciente y valores menores cuando eran más antiguos, es decir, el identificador era mayor cuánto más reciente era el partido. Con esta información, ordenabamos correctamente todos los partidos de cada jugador y los mostrabamos al usuario.
- *Logos equipos:* No teníamos los escudos de cada equipo, porque no podíamos hacer como otras veces que el usuario pinchaba un equipo o jugador y pasabamos las imágenes entre clases hasta llegar a la ventana dónde queríamos mostrarlas. Todos los equipos contaban con un identificador,por lo que teníamos que comprobar el identificador de cada equipo, y cargarlo previamente a la aplicación. Ya sabiendo cuál es el identificador de cada equipo, para cada partido lo comprobamos y cargamos la imagen correspondiente. Esta tarea no llevo prácticamente tiempo realizarla debido al conocimiento previo y experiencia al realizar tareas similares.

Duración de la tarea: 1 semana.

Añadiendo temporadas a las estadísticas de equipos

Para finalizar esta etapa, añadimos el resto de temporadas a la ventana de estadísitcas de equipo. Recordemos que solo podíamos visualizar los datos de la temporada actual. Con Balldontlie podemos visualizar el resto de temporadas también. El procedemiento es exactamente el mismo que hemos realizado anteriormente. Un Spinner para que el usuario elija la temporada, y realizar la consulta con la selección del usuario como parámetro variable.

Control de errores y fin de etapa.

Para finalizar esta sección deberemos controlar un error que es fácil que suceda y que al usuario se le puede ocurrir fácilmente. Recordemos que podemos mostrar datos para equipos y jugadores desde el año 1995. En el caso de los equipos, no existe ningún problema ya que los 30 equipos de la liga han participado en la competición desde antes del año 1995. Pero en el caso de los jugadores, como nos podemos imaginar no ocurre eso. ¿Que pasaría si pedimos estadísticas de una temporada en la que un jugador ni siquiera había debutado? . La respuesta es que si solicitamos esa información, la pantalla se quedará sin datos dándo sensación de bloqueo. Además si el usuario pulsa el botón para ver los partidos disputados de ese jugador, para una temporada conflictiva, no ocurrirá nada. Este es un caso que hay que controlar. Para solventarlo, lo que hacemos es comprobar si la respuesta a la consulta está vacía o no. Si está vacía significa que es una temporada conflictiva para el jugador, en la que no había debutado aún. Detectando de esta manera dichas situaciones mostramos un mensaje emergente al usuario, diciendo que no hay datos para la temporada seleccionada. Los campos dónde aparecen cada estadística del jugador aparecen con el texto "Sin datos". Por último, en estas situaciones bloqueamos el botón de partidos para que el usuario no pueda pulsarlo apareciendo en un color grisaceo degradado. Cuándo el usuario selecciona otra temporada de la que se tengan datos la consulta se hace normalmente, se muestran las estadísticas y el botón de partidos vuelve a estar totalmente habilitado.

Duración de la tarea : 1 semana.

Esta etapa contuvo una menor carga temporal básicamente porque como hemos dicho el proceso de realización de las tareas era similar al realizado en las anteriores etapa. Para finalizar vemos un resumén de esta tercera etapa:

Tarea	**Duración**
Estadísticas Jugador	2 semanas
Partidos Jugador	1 semana
Temporadas y control de errores	1 semana
Total	**4 semanas**

Cuadro 16: Resumen Etapa 3

7.4. Etapa 4: Menu principal y generación de archivos CSV

Tras completar las primeras etapas hemos obtenido una versión de la aplicación entregable y con una amplia funcionalidad. Debido a que se tiene un buen margen temporal aún, vamos a introduccir otra etapa en la que implementemos más funciones. En esta última etapa, nos centraremos en crear un menú principal, que nos permita acceder a las últimas noticias de la liga, a la clasificación y a una ventana que sea capaz de realizar pronósticos de los partidos futuros. Para concluir y por petición de uno de los tutores, se implementará una opción que exporte las estadísticas en archivos con formato CSV.

7.5. Menú principal y pantalla de carga

Antes de implementar el menú principal de la aplicación, se integró una pantalla de carga. Este elemento es imprescindible, y aparece en prácticamente todas las aplicaciones . La Splash Screen [23], es un elemento indisepensable en la aplicación y tiene dos funciones muy importantes. Por un lado tiene un cometido estético, que le da un toque de profesionalidad y seriedad a la aplicación. En el otro extremo, el tiempo que aparece dicha pantalla sirve para realizar "trabajos en la sombra". Dichas tareas consisten en cargar datos para agilizar el funcionamiento de la aplicación.

Backlog	**To do**	**In progress**	**Testing**	**Done**
Realizar una aplicación que muestre los principales datos de la liga Nba.	-Ventana Noticias -Ventana Clasificación -Ventana Pronósticos	Integración de Splash Screen y menú principal	Comprobar la correcta reedireción del menú principal y que la splash screen carga en segundo plano los datos.	✓Recopilar APIS. ✓Mostrar listado de equipos. ✓Control de eventos. ✓Menú Inferior. ✓Ventanas Menú inferior ✓Estadísticas jugador ✓Partidos y control de errores

Cuadro 17: Etapa 4. Primer tablero de Kanban

Esta tarea fue bastante sencilla y ya la había realizado en la asignatura de "Sistemas Móviles" por lo que no se gastó demasiado tiempo en su realización. En un primer momento, se implementó la Splash Screen con una duración predeterminada de 4 segundos. En este caso tenía únicamente una funcionalidad estética, para que el usuario al abrir la aplicación no se encontrase directamente el menú y tuviese una pantalla previa de carga a modo de presentación. Posteriormente, se eliminó esa condición de 4 segundos, y se determinó que el tiempo que iba a estar la pantalla de carga iba a ser el correspondiente en realizar las operaciones asignadas. Dichas operaciones consisten en consultas a alguna API para obtener datos, y transformarlos para reducir los tiempos de carga dentro de la aplicación. Por tanto, el tiempo de vida de la Splash Screen

es variable y depende del estado del servidor de la API, aunque siempre suele estar en torno a 4-6 segundos.

Una vez que tenemos este importante elemento de la aplicación, proceederemos a implementar un menú que nos conduzca a diferentes pestañas. La primera versión de menú, consistió en una ventana con 4 botones, cada uno de ellos nos reedirigía a las ventanas de Noticias, Clasificación, Pronósticos y Equipos. Posteriormente, se mejoró el diseño dejando un aspecto mucho más profesional con el uso de Cardviews, que nos permiten implementar un diseño basado en tarjetas [24]. Por último, hicimos 4 ventanas con un mensaje para comprobar que el menú nos reedirigía correctamente a cada una de las nuevas ventanas.

Duración de la tarea: 1 semana.

Ventana Noticias

Aquí comenzamos con la implementación de las ventanas del menú. Para mostrar las noticias de la liga, recurriremos a la primera API con la que hemos trabajado (SportsDataIO). Tenemos que hacer exactamente el mismo proceso que para mostrar el listado de equipos, y que cuando se pulse se muestre información asociada a ese elemento. Dicho esto, mostramos por medio de una consulta las últimas noticias de la liga con un título y la fecha de publicación. Si pinchamos en la noticia veremos el contenido de la misma y la fuente. La realización de esta tarea fue casi inmediata y no hubo ningún tipo de complicación.

Duración de la tarea: 3 días.

Ventana Clasificación

En esta ventana se va a mostrar una clasificación general de la liga. En ella mostraremos la posición que ocupa cada equipo, su logo y otros datos estadísticos que describiremos más adelante. La obtención de una tabla clasificatoria para los equipos participantes de la liga no es un proceso inmediato que nos de la API con una consulta, sino que tendremos que realizar varias operaciones para obtener el resultado final deseado.

Backlog	**To do**	**In progress**	**Testing**	**Done**
Realizar una aplicación que muestre los principales datos de la liga Nba.	Ventana Pronósticos.	Realizar una clasificación para la liga.	Cmprobar la correcta posición de cada equipo en la clasificación y sus datos estadísticos en la competición.	✓Etapa 1 ✓Etapa 2 ✓Etapa 3 ✓Menú principal y Splash Screen ✓Ventana Noticias

Cuadro 18: Etapa 4. Segundo tablero de Kanban

La Api SportsdataIO, nos brinda el listado completo de equipos con el número de victorias, derrotas y porcentaje de victorias que llevan hasta la fecha actual en la presente temporada. No nos proporciona una clasificación por lo que tenemos que crearla nosotros mismos. El atributo que vamos a usar para hacer esta clasificación va a ser el porcentaje de victoria. En la NBA no se dan puntos por victoria o empate

como pasa en el fútbol. Aquí es muy común que unos equipos lleven más partidos disputados que otros, por lo que la mejor medida es mirar el porcentaje de victorias de cada equipo respecto a las partidos jugados. Dicho esto, sin entrar en detalles del proceso seguido, hacemos un listado de los equipos ordenados ascendentemente por su porcentaje de victoria. De esta manera, conseguimos una clasificación completa de todos los equipos en la que mostramos los siguientes datos:

- *Posición:* Esto no es un atributo de la API, sino que es la posición que ocupa el equipo ordenado en el Array de datos. Le sumamos 1 al valor de este índice porque se empieza en la posición 0 y entonces tenemos como resultado la posición real que ocupa el equipo en la liga.
- *Nombre del equipo:* Es el nombre completo del equipo que muestra la ciudad de la que es representante.
- *Partidos jugados:* No es tampoco un atributo, pero lo obtenemos por medio de la suma de victorias y derrotas.
- *Victorias:* El número de victorias que lleva el equipo actualmente.
- *Derrotas:* Son las derrotas que acumula el equipo hasta la fecha.
- *Porcentaje:* Es el porcentaje de victorias que lleva el equipo en la temporada. Sirve para ordenar los equipos en la clasificación.
- *Logo:* No es ningún dato, simplemente el escudo de cada equipo como se viene realizando en la mayoría de las clasificaciones deportivas.

Esta tarea dio algun problema, porque en el proceso de crear la clasificación nos salían equipos repetidos y tuvimos que implementar un método de comprobación que eliminase dichos equipos. Salvo por este inconveniente, por lo general, la tarea no fue muy complicada y se pudo realizar satisfactoriamente.

Duración de la tarea: 1 semana.

Ventana Pronósticos

Esta tarea no fue muy complicada. Se dividió en dos etapas, la primera de ellas consistía en el diseño de la ventana. Se habilitó un Spinner, para elegir el equipo del que queríamos visualizar el pronóstico. Cuando el usuario elige el equipo y pulsa el botón, se hace una consulta con el nombre del equipo como parámetro. Entonces obtenemos la fecha y el rival del partido.

A partir de aqui, ponemos dos botones, uno para la heurística de predicción, que es el modelo predictivo desarrollado por el alumno y otro para el modelo predictivo estadístico. La siguiente fase, solo consistiría en integrar las fórmulas de los dos modelos en la aplicación, proceso que se explicará mejor en las próximas secciones.

Duración de la tarea: 1 semana

Para finalizar vemos un resumen de esta cuarta etapa:

Tarea	**Duración**
Menú principal y pantalla de carga	1 semana
Ventana Noticias	3 días
Ventana Clasificación	1 semana
Ventana Pronósticos	1 semana
Total	**3 semanas**

Cuadro 19: Resumen Etapa 4

7.6. Etapa 5 : Scripts para la obtención de datos

En esta última etapa del proyecto, se llevará a cabo el desarrollo de un script que automatice la obtención de todos los datos relativos a los partidos desde 1995 hasta la actualidad. Como se puede intuir, se trata de un gran volumen de datos, por lo que no puede realizarse de manera manual.

7.7. Estructura del archivo y enfoque de la tarea

Una vez que conseguí tener la aplicación prácticamente terminada, mi tutora me mandó recopilar datos de la API en formato CSV para crear un modelo de predicción de partidos, en base al análisis de datos estadísticos pasados.

La estructura que se me especificó para el archivo, iba a consistir en tener prácticamente toda la información de cada partido en cada línea del archivo en formato CSV. Dicho esto, cada fila tipo excel, contendría el nombre del jugador, sus estadísticas en el partido, el equipo para el que jugó, las estadísticas de su equipo(posición en el ranking y racha con la que llegaba), las mismas estadísticas para el equipo rival y el resultado final del encuentro. Una vez que tenemos clara la estructura, podemos empezar a trabajar con el siguiente tablero de Kanban como referencia:

Backlog	**To do**	**In progress**	**Testing**	**Done**
Realizar una aplicación que muestre los principales datos de la liga Nba.	Comprobar porcentaje de acierto del futuro modelo obtenido a partir del archivo de datos creado	Generar un archivo csv con la estructura de datos especificada por mi tutora	Comprobar vericidad del archivo creado	✓Etapa 1 ✓Etapa 2 ✓Etapa 3 ✓Menú principal y Splash Screen ✓Ventana Menú principal

Cuadro 20: Etapa 5. Primer tablero de Kanban

Tras entender la petición de nuestra tutora, tenemos que analizar la estrategia a seguir para obtener el archivo. Una primera idea que lleve a cabo fue, iterativamente, realizar consultas cambiando jugadores y temporadas en la aplicación, pero el intento fue fallido porque Android realizaba las peticiones de manera asíncrona, y nosotros necesitamos un proceso puramente secuencial en las consultas. Probé diferentes maneras, con hilos independientes, cada uno para una consulta diferente pero no terminó de funcionar. El problema era que no había entendido bien a mi tutor, y este script se debería de hacer fuera de Android de manera independiente, ya que este sistema operativo dificultaba mucho esta tarea.

Cambiando la estrategia, lo que hice fue elegir el entorno de desarrollo Eclipse Kepler [31]. Aquí fui haciendo pruebas de menor a mayor complejidad. Uno de los principales problemas que encontré es que la API tenía un límite de 60 consultas por minuto. Para solventarlo implementé un contador de consultas, y cuando detectaba que llegaba a 59 le mandaba esperar 1 minuto. Pasado este tiempo, retomaba el trabajo y seguía realizando consultas.

El script comenzaba por el año 1995 e iba jugador por jugador obteniendo todos los partidos. Obtener todos los partidos de un jugador, equivalía a una consulta y eran aproximadamente 493 jugadores a analizar por temporada. Teniendo en cuenta que tenemos que analizar 25 temporadas, el número de consultas se dispara. El script tardó aproximadamente 7 horas en realizar la tarea.

Con este script obtuvimos todos los datos que me pedía mi tutora , a excepción de la racha y clasificación de los equipos. Esto no lo podíamos obtener en este script ya que cada jugador no juega todos los partidos y no podemos saber la racha real y ranking con el que llegaban los equipos. Para solucionarlo, desarrollé otro script, con base al anterior, pero en vez de ver los partidos de los jugadores, veía los partidos de los equipos. Miraba partido por partido, comprobaba si se había ganado o perdido, y actualizaba la racha. El ranking lo obtenía como porcentaje de victoria, mirando el numero de victorias respecto al número de partidos jugados.

Llegados a este punto, tenemos dos archivos excel, uno con los partidos de los jugadores y otro con los partidos de los equipos. Entonces , nos preguntaremos, ¿como podemos relacionar las rachas de los equipos del segundo archivo con las rachas de los equipos de los jugadores del primer archivo? . La respuesta está en que tenemos una variable que es un identificador de partidos, y que es común entre ambos archivos. Con esta variable, podremos saber las rachas en el primer archivo gracias a los comandos de Excel. En concreto, utilicé el comando BUSCARV, que nos permite buscar datos entre varias hojas excel a partir de una serie de condiciones. Este fue el camino más sencillo que se me ocurrió para crear el archivo, que nos servirá para crear un modelo predictivo de partidos.

La tarea se complicó bastante al principió porque estaba enfocandolo mal, pero luego una vez que se cambió de estrategia, se avanzo más rapidamente en la consecución de los objetivos propuestos.

Duración de la tarea: 4 semanas

Profundizando en los scripts

Vamos a explicar más detalladamente la estructura y el funcionamiento de los scripts. Dichos programas han sido desarrollados en lenguaje Java, ya que es el lenguaje de programación que he utilizado en la aplicación y en el que tengo más experiencia.
Para llevar a cabo la creación de los archivos en formato CSV se ha utilizado la librería Apache Commons CSV de Java.

Figura 20: Logo de la librería Apache Commons.

Esta librería, como bien dice su manual de uso, "nos proporciona una interfaz simple para leer y escribir archivos CSV de varios tipos". Es muy fácil de usar, siguiendo la guía de usuario [33], podremos comenzar a utilizarla en el script para crear la cabecera del archivo CSV o para escribir datos en cada fila durante el proceso de generación del archivo.

El uso de la librería Apache Commons ha sido clave para llevar a cabo el proceso de obtención del archivo final. A parte de esto, me gustaría destacar, otro elemento que ha sido muy importante para el correcto funcionamiento de los scripts. Se trata del manejo de hilos en Java. Todo programa informático consta de una serie de hilos de ejecución donde albergan los procesos. Dichos hilos pueden estar en diferentes estados (nuevo, muerto, preparado, bloqueado..etc)

Figura 21: Estados de los hilos.

Para estos dos programas, ha sido fundamental "dormir" el hilo principal del script. Gracias a esto, cuando nos aproximábamos al límite de consultas, dormíamos el programa durante 1 minuto. Esto lo he logrado con el método, "Thread sleep()", al cual le pasamos como parámetro el numero de milisegundos que queremos tener el programa en espera. Este método, pone a el hilo principal de ejecución del programa en estado de bloqueo. Cuando pasa el número de milisegundos que hemos indicado, el hilo se reanuda y el programa vulve a ejecutarse en el punto en el que se había bloqueado.

Por último, quiero destacar el tratamiento de objetos JSON. En anteriores secciones, hemos explicado la estructura de este tipo de objetos Java. En estos scripts, por cada consulta obtenemos un objeto JSON. A partir de este objeto, obtenemos su array de datos que contiene a su vez varios objetos JSON. Estos objetos, a su vez, contienen una serie de atributos que son los datos que leemos y escribimos en el archivo CSV. En los dos scripts utilizados, hemos extraido los siguientes JSONObject:

- *home_team:* Se trata de un objeto que contiene todos los datos relativos al equipo que juega como local en el partido.
- *visitor_team:* Se trata de un objeto que contiene todos los datos relativos al equipo que juega como visitante en el partido.
- *player:* Aquí se incluyen datos sobre un jugador concreto como pueden ser nombre, apellidos, posicion, peso, altura..etc
- *team:* Objeto que contiene el nombre completo del equipo y su identificador.
- *game:* Resume todo lo que ha pasado en el partido, incluyendo el resultado del encuentro y los puntos anotados por cada equipo.

Estos objetos, van incluidos dentro del objeto principal de la consulta, que contiene las estadisticas completas del jugador en el partido.
Para acabar con esta sección, vamos a presentar en forma de pseudocódigo los dos scripts utilizados. El pseudocódigo, se utiliza para hacer más sencilla la compresión de la estructura utilizada en el código de los dos scripts.

El pseudocódigo es común a los dos script. La única diferencia entre ambos, son los atributos que se leen y escriben para el archivo final, pero el proceso es común en ambos casos. Se comienza con una inicialización de un ArrayList que nos permite tener guardados todos los equipos con su identificador correspondiente. Seguidamente se crea la cabecera del archivo CSV, en la que se encuentran las columnas que va a presentar nuestro archivo. Para acabar leeremos y escribiremos en el archivo CSV los datos necesarios:

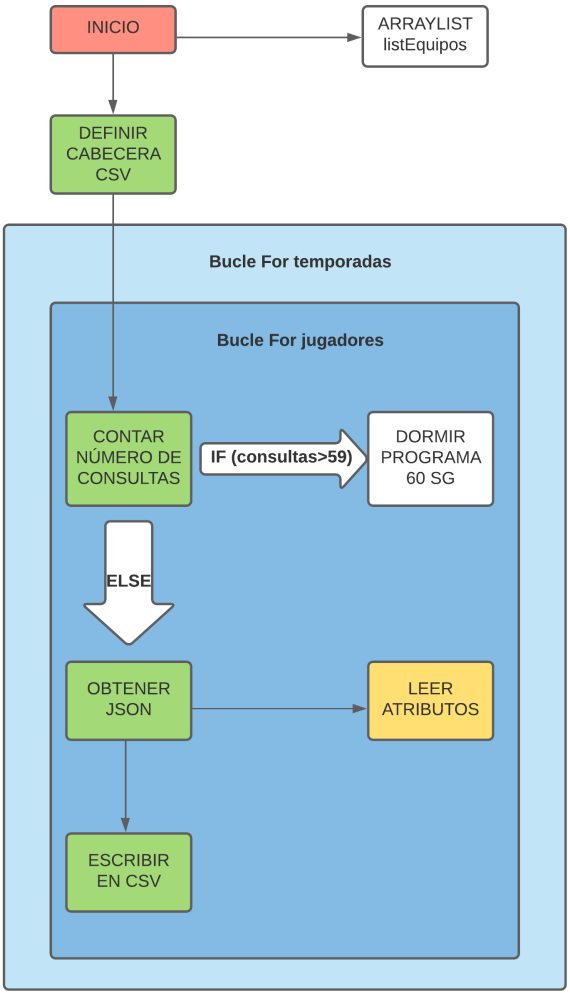

Figura 22: Pseudocódigo del algoritmo 1.

Estudio de la complejidad del algoritmo

En esta sección, vamos ha realizar un análisis de la eficiencia del algoritmo utilizado para generar el archivo en formato CSV. Para ello, nos vamos a centrar en el parámetro o funciones de coste conocidas como complejidad temporal. Esta variable la podemos definir como el tiempo que tarda en ejcutarse el algoritmo. Es una medida sencilla que nos permite observar como crece el tiempo de ejecución conforme lo hace el tamaño de las entradas.

Para medir la complejidad temporal, vamos a utilizar la notación asintótica (O grande) [34]. Esta notación permite expresar de manera clara la relación de la que hablabamos antes entre el tiempo de ejecución y el tamaño de las entradas. La notación es sencilla, se escribe una O grande seguida de paréntesis, dónde encontramos una variable matemática que describe la complejidad temporal en función de n, donde n es el tamaño de la entrada. Entre los casos más típicos encontramos las siguientes:

- O(1): Orden constante
- O(log n): Orden logarítmico
- O(n): Orden lineal
- O(n log n): Orden cuasi-lineal
- $O(n^2)$: Orden cuadrático
- $O(n^3)$: Orden cúbico
- O(na): Orden polinómico
- O(2n): Orden exponencial
- O(n!): Orden factorial

Nuestro algoritmo lo clasificamos dentro del orden cuadrático. Es decir, que podemos representar su complejidad temporal por medio de la expresión $O(n^2)$. Esto quiere decir que el tiempo de ejecución es proporcional al cuadrado del tamaño de la entrada. La razón por la que nuestro algoritmo tiene este tipo de complejidad, está en el doble bucle que se utiliza. Aqui tenemos un bucle para las temporadas y anidado a él otro para cada jugador. Lo mismo ocurre en el segundo script pero en este caso los bucles son para temporadas y equipos en vez de jugadores. Las operaciones que se realizan en el interior de estos bucles son de orden constante O(1), lo que quiere decir que el tiempo de ejecución en esta parte no se ve afectado por el tamaño de la entrada.

Para terminar, vamos a realizar el cálculo del tiempo de ejecución total que ha empleado el algoritmo en cada uno de los dos scripts. El primer script (jugadores), tiene que analizar 493 jugadores en 25 temporadas distintas, lo que se resume en 493 x 25= 12.325 consultas. Cada consulta, cuesta aproximadamente un segundo en ser procesada al servidor. Por lo que tenemos 12.325 sg= 205 minutos= 3,42 horas. Este cálculo de tiempo es sin pausas, hay que recordar que cada 59 consultas deberemos de hacer una pausa de 1 min. Por lo tanto, tenemos que realizar 12.325/59 = 209 pausas o lo que es lo mismo añadir 209 minutos, que sumados a los 205 minutos que teníamos sin pausas hacen un total de 414 minutos que son aproximadamente 7 horas.

En el caso del segundo script el proceso es identico sólo que analizamos equipos en vez de jugadores, y pasamos de 493 jugadores a 30 equipos. Operando análogamente, obtenemos un total de 25 temporadas x 30 equipos = 750 consultas, a 1 segundo por consulta tenemos 750 segundos que son 12,5 minutos de trabajo sin pausas. Si calculamos el número de pausas necesarias, tenemos que hay que hacer 750 consultas/59= 12,71 pausas de 1 minuto. Este tiempo de pausa sumado al tiempo de trabajo que habíamos calculado anteriormente nos da un resultado de tan sólo 25 minutos.

Si comparamos el tiempo de ejecución de ambos scripts vemos las consecuencias del orden temporal cuadrático $O(n^2)$. A entradas pequeñas como ocurre en el segundo script tendremos tiempos de ejecución bajos, pero si incrementamos el tamaño de las entradas el bucle anidado hará que se dispare el tiempo de ejecución final.

8. Modelo predictivo

8.1. Introducción

En esta sección, vamos a comprobar la eficacia que adquirieron los dos modelos predictivos de la aplicación. Por un lado, analizaremos la heurística de predicción. El alumno ha propuesto una heurística capaz de predeccir partidos, en base a estadísticas pasadas de la actual temporada . El nombre de heurística, nos indica que se trata de un conjunto de técnicas o métodos creados por el alumno para predecir succesos. En este caso, se va a utlizar una fórmula para predeccir partidos. En el otro extremo, se ha desarrollado un modelo más profesionalizado de predicción, en el que se han analizado la tendencia de los resultados dados en la liga, y que son usados para predecir sucesos futuros, este modelo lo hemos llamado modelo predictivo estadístico. Para comprobar la eficacia de ambos modelos se han hecho dos test de 15 partidos escogidos de manera aleatoria para cada uno de los dos modelos. Para terminar, al final de este apartado, analizaremos como de realista son estos modelos y realizaremos una comparación entre ambos.

8.2. Heurística Predictiva

8.2.1. Fórmula utilizada

La ecuación utilizada para hacer los pronósticos, es el elemento más importante para realizar predicciones futuras. En nuestro caso, hemos analizado 4 factores que creemos que son los más determinantes en los partidos, por orden de importancia tenemos:

- ***Clasificación:*** El primero de los factores es la posición en el ranking de cada equipo. Este parámetro tenemos claro que va a ser el que más peso tenga en la fórmula, porque la posición en la clasificación de un equipo es el factor que marca la regularidad a lo largo de la temporada. A la hora de analizar a los dos equipos que se enfrentan en un partido, la posición en la clasificación nos indica como ha sido su rendimiento a lo largo de la temporada. Por estas razones, en nuestra fórmula este factor va adquirir una mayor importancia a la hora de dar la probabilidad final de victoria de cada uno de los dos equipos que se enfrentan.

  El factor posición en el ranking tiene un valor ponderado del 45 %. Es un valor decreciente, en el sentido de que el primer equipo en la clasificación recibirá la totalidad del valor de este parámetro, el segundo recibirá menos y así sucesivamente a medida que sigamos descendiendo en el puesto clasificactorio. Todos los parámetros de la fórmula final están calculados sobre uno, por lo que el primer clasificado recibirá en este apartado una nota de 0,45 y el último 0,0. Los equipos que no sean ni el primero ni el último en la clasificación recibirán la nota correspondiente a la siguiente ecuación:

  $$0{,}45-(0{,}0150 * n)$$

  siendo n el puesto en la clasificación que ocupa el equipo en ese momento. El valor de 0,0150 no se ha escogido al azar, sino que se obtiene de dividir 0,45 entre los 30 equipos que componen la clasificación. De esta manera, podemos observar que en la fórmula 0,45-(0,0150 x n) cuanto mayor sea el valor de n menor nota recibirá en este apartado, que es justo lo que estamos buscando.

- ***Racha:*** Del anterior parámetro, podemos pensar que puede ocurrir la situación de que un equipo este en una posición muy alta en el ranking, pero que en la actualidad esté viviendo un momento

malo a nivel de juego. Estos equipos pueden que estén viviendo del pasado, al haber conseguido muchas victorias en partidos anteriores pero que ahora estén perdiendo todos los partidos y engañen al algortimo en base a sus viejos resultados . De esta situación nace el segundo parámetro de la fórmula que es la racha de cada uno de los equipos. Para conocer la racha de cada equipo lo hemos representado con números enteros. Un valor por ejemplo de +2 indica que ese equipo lleva dos partidos seguidos ganando, mientras que un valor de -3 indica que lleva 3 partidos consecutivos perdiendo. Para asignar una valoración a este parámetro hemos acotado los valores al intervalo [-5,+5] por lo que los equipos que tengan una racha mejor de +5 recibirán la máxima puntuación independientemente de lo holgados que hayan sobrepasado los +5 de racha. En el lado opuesto, ocurrirá lo mismo, los equipos con una racha de -5 o peor no recibirán nada de puntuación.

Consideramos este factor muy importante, y le asignamos un peso de un 25 % en la fórmula final. La aplicación, cuando llega a esta parte de la fórmula mira si los equipos tienen una racha mayor/igual que 5 o menor/igual que -5 y les asignan una puntuación de 0,25 y 0 respectivamente (puntuación sobre la unidad). Si no se cumplen las dos situaciones anteriores se aplica la siguiente fórmula:

$$0{,}25-[(5 - n)*0{,}027]$$

Siendo n la variable entera que indica la racha del equipo. El valor de 0,027 es el valor mínimo para una constante de 3 decimales que aporta algo de puntuación al peor de los casos que es una racha de -4 (sin contar rachas de -5 o peores que aportan 0 de puntuación en este apartado).

Si analizamos la fórmula podemos ver que los casos de rachas positivas dan un puntuaje superior a 0.1 y en el segundo mejor caso (racha +4) superior a 0.2,pero una vez que entramos en rachas negativas por pequeñas que sean la puntuación se sitúa por debajo de 0,1 castigando a los equipos que llegan en peor dinámica a los encuentros.

- ***Local o Visitante:*** Otro de los factores que creemos que es importante considerar es si el equipo disputa el partido como local o como visitante. Hay equipos que se hacen muy fuertes en su estadio y otros que cuando juegan a domicilio les cuesta más ganar, por lo que es un factor que debe de entrar en el análisis predictivo.

  En orden de importancia ocupa la tercera posición de nuestras variables predictivas y le vamos a dar un peso del 20 % en la fórmula final. Aqui el cálculo es más sencillo, simplemente miramos que equipo juega de local y cuál de visitante. Una vez que sabemos eso, la aplicación comprueba todos los partidos jugados por los dos equipos en la temporada y saca el porcentaje de victoria como local y visitante. Una vez que tenemos esos datos, simplemente a cada uno de los dos equipos le aplicamos el porcentaje de victorias de local/visitante (según corresponda) sobre 0,2 y tenemos la puntuación que se llevan en este apartado.

- ***Puntuación media:*** Para terminar tenemos la última variable predictiva que es el puntuaje medio de ambos equipos. Se debe de tener en cuenta ya que si por ejemplo un equipo promedia una menor cantidad de puntos anotados respecto al de su rival se prevé que en ese partido va a tener que anotar más canastas de las habituales para ganar.

A esta variable la hemos asignado un peso del 10 % y es un peso que al ser bajo se ha decidido que se tome en su totalidad o que no se sume ningun puntuaje en este aportado. Su cálculo es también sencillo, se miran todos los partidos disputados por el equipo hasta la fecha y se hace la media de los puntos anotados por el equipo. Para finalizar se comparan los puntos medios anotados por ambos equipos y el que mayor puntuación media tenga se le asigna 0,1 y a su rival 0,0.

Con estas 4 variables obtenemos la fórmula final predictiva desarrollada por el alumno:

*0,45 % * clasificación+ 0,25 % * racha+ 0,20 %* local/visitante+ 0,1 % puntuaje medio*

Esta fórmula se ha obtenido por medio del método ensayo y error[36]. En cuanto a los porcentajes, se han ido realizando pruebas y variando los porcentajes hasta que se ha llegado a la fórmula final anteriormente descrita.

Afinando la fórmula

A estas alturas nos podremos haber dado cuenta de un pequeño detalle y es que hay variables predictivas en las que la ponderación no se reparte entre ambos equipos sino que los dos equipos pueden sumar una gran cantidad de porcentaje en la probabilidad final. Se puede dar el caso de que dos equipos cuenten con un buen puesto clasificatorio y por ejemplo una buena racha. En estos casos, añadiendo la puntuación de las dos variables restantes, podemos obtener dos probabilidades (una por cada equipo) que entre ambas sumen más del 100 %. Para solventar este problema vamos a tomar como referencia la suma de ambos porcentajes. Aplicando las siguiente fórmulas hemos calculado las probabilidades finales para cada equipo:

- *Porcentaje Total= Porcentaje Equipo 1+ Porcentaje Equipo 2*
- *Porcentaje Equipo 1= (Porcentaje Equipo 1 * 100)/Porcentaje Total*
- *Porcentaje Equipo 2= (Porcentaje Equipo 2 * 100)/Porcentaje Total*

De esta manera siempre tenemos unos porcentajes finales razonables y cuya suma no superan nunca el 100 %.

8.2.2. Predicciones

A continuación, vamos a exponer los partidos que han servido de análisis para el modelo predictivo. Lo presentamos en forma de tabla, en la que en cada partido podemos vemos la probabilidad de victoria que nos daba el modelo y el resultado final del encuentro. Con el resultado final del partido, podremos ver si ha habido mucha diferencia en el partido entre los dos equipos, y concluir si la predicción era buena o no:

Local	Visitante	% Local	% Visitante	Puntuación	Resultado	Aproximación
Toronto	Cavaliers	68,2 %	31,8 %	112-96	**Acierto**	Muy buena
Nets	Dallas	59,1 %	40,9 %	98-115	**Fallo**	Mala
Spurs	Clippers	29,9 %	70,1 %	101-134	**Acierto**	Excelente
Bulls	Detroit	67,4 %	32,6 %	100-86	**Acierto**	Muy Buena
Portland	Pelicans	65,0 %	35,0 %	101-93	**Acierto**	Buena
Nuggets	Celtics	79,2 %	20,8 %	87-105	**Fallo**	Muy mala
Lakers	Bucks	49,8 %	50,2 %	113-106	**Fallo**	Buena
Thunder	Knicks	32,8 %	67,2 %	97-119	**Acierto**	Muy Buena
Jazz	Kings	81,4 %	18,6 %	128-115	**Acierto**	Buena
Minnesota	76ers	17,1 %	82,9 %	113-122	**Acierto**	Buena
Warriors	Heat	54,5 %	45,5 %	120-112	**Acierto**	Buena
Houston	Hornets	30,2 %	69,8 %	97-122	**Acierto**	Excelente
Suns	Grizzlies	74,3 %	25,7 %	128-97	**Acierto**	Excelente
Orlando	Wizards	30,2 %	69,8 %	116-131	**Acierto**	Muy Buena
Hawks	Pacers	51,8 %	48,2 %	129-117	**Acierto**	Buena

Cuadro 21: Modelo Predictivo Alumno. Test 1

Local	Visitante	% Local	% Visitante	Puntuación	Resultado	Aproximación
Toronto	Heat	46,4 %	53,6 %	102-111	**Acierto**	Buena
Nets	Celtics	54,9 %	45,1 %	123-95	**Acierto**	Buena
Spurs	Pacers	40,2 %	59,8 %	109-94	**Fallo**	Mala
Bulls	Thunder	45,7 %	54,3 %	125-127	**Acierto**	Buena
Portland	Nuggets	33,2 %	66,8 %	105-106	**Acierto**	Buena
Nuggets	Portland	51,0 %	49,0 %	111-106	**Acierto**	Buena
Lakers	Hornets	60,1 %	39,9 %	101-93	**Acierto**	Buena
Thunder	Houston	40,9 %	59,1 %	106-136	**Acierto**	Muy Buena
Jazz	76ers	52,0 %	48,0 %	123-131	**Fallo**	Mala
Minnesota	Heat	45,0 %	55,0 %	119-111	**Fallo**	Mala
Warriors	Grizzlies	60,0 %	40,0 %	103-111	**Fallo**	Mala
Houston	Wizards	25,2 %	74,8 %	107-88	**Fallo**	Mala
Suns	Cavaliers	76,9 %	23,1 %	119-113	**Acierto**	Buena
Orlando	DAllas	28,1 %	71,9 %	124-130	**Acierto**	Buena
Hawks	Clippers	30,4 %	69,6 %	110-119	**Acierto**	Buena

Cuadro 22: Modelo Predictivo Alumno. Test 2

8.2.3. Análisis Resultados

Se han analizado 30 muestras de la liga regular. El modelo ha acertado un total de 22 encuentros suponiendo el 22/30= 73,3 % de acierto. Me gustaría destacar la columna "Aproximación", en ella podemos ver que el modelo sólo ha realizado una aproximación "muy mala" en el partido Nuggets vs Celtics. Este fallo se debe a que el modelo daba muy favorito a uno de los equipos y en el encuentro el equipo rival ganó muy holgadamente. Aunque en la tabla pongamos que es una predicción muy mala si lo analizamos no lo es. El modelo predijo como ganador a los Nuggets de manera bastante holgada, la razón es que este equipo venían con una racha de 11 partidos seguidos ganando y los celtics con 5 seguidos perdiendo, además los Nuggets habían ganado el 59 % de los partidos disputados por el tan solo 45 % de victorias de los Celtics, por lo que el pronóstico no era malo, fue una sopresa de partido. Podemos encontrar también varias predicciones que se las cataloga de excelentes. Estas predicciones son aquellas en las que el modelo predice con una alto porcentaje que uno de los equipos va a hacerse con la victoria, y el resultado final nos muestra no solo que ha conseguido esa victoria, sino que la diferencia con el equipo rival se corresponde a la diferencia predicha entre ambos equipos.

8.3. Modelo Predictivo Estadistico

Este es el modelo más importante de los dos ya que ha sido desarrollado teniendo en cuenta las estadísticas de cada jugador en cada encuentro en los últimos 20 años. Estas estadísticas son introduccidas en un potente software, que analiza la relación que tienen las estadísticas individuales de los jugadores en un partido con la repercusión que tienen en el resultado final. De esta manera se pueden establecer relaciones acción-suceso con bases estadísticas. Este modelo predictivo permite predecir dos tipos de variables:

- *Puntos obtenidos por el equipo:* Este modelo predictivo es capaz de calcular los puntos que anotará cada equipo en un partido. Se han analizado 234.000 muestras dándonos un desajuste de 10,2851 puntos por partido. Es decir, que el modelo se equivoca por norma general en un intervalo aproximado de [-10,10] puntos. Para calcular la puntuación de cada equipo, vamos a necesitar las estadísticas de los jugadores de cada equipo o lo que es lo mismo, las estadísticas medias del equipo hasta el momento actual. El puntuaje de los partidos se calcula a partir de las siguientes variables regresadas:

  - Asistencias: Número de asistencias a canasta que promedia un equipo.
  - Tapones: Número de bloqueos o tapones que realiza un equipo por partido.
  - Rebotes Defensivos: Número de rebotes capturados en defensa por un equipo.
  - Rebotes Ofensivos: Número de rebotes capturados en ataque por un equipo.
  - Robos: Son los robos de balón que realiza un equipo por partido.
  - Pérdidas: Número de balones que pierde un equipo por partido.
  - Racha: Variable entera que indica la racha con la que llega un equipo a un partido.
  - Racha Rival: Variable entera que indica la racha con la que llega el rival al partido.

  Estas variables constituyen prácticamente la totalidad de las estadísticas que componen un partido de baloncesto. El modelo desarrollado por el alumno consideraba sólo 4 parámetros para predecir los partidos, este en cambio utiliza 8, lo que se traduce en un estudio mucho más completo de los partidos.Vamos a ver como utilizamos estas 8 variables para realizar los pronósticos, hemos utilizado la siguiente fórmula:

*Puntuaje Partido= Constante_Puntuación+Asistencias*Coef_Asistencias+Tapones*Coef_Tapones+ RebotesD*Coef_RebotesD+RebotesO*Coef_RebotesO+Robos*Coef_Robos+Pérdidas*Coef_Pérdidas+ Racha*Coef_Racha+ Racha Rival*Coef_RachaRival.*

Cada variable tiene asignada una constante. Dichas constantes han sido obtenidas del estudio de las 234.000 partidos, en los que se ha logrado obtener una relación entre las variables y la puntuación final del partido. El resultado de dicho estudio tiene como consecuencia los siguientes valores para las 8 constantes descritas anteriormente:

Constante	**Valor**
Constante Fórmula	104,7354
Asistencias	0,625208
Tapones	0,0729081
Rebotes Defensivos	0,1252698
Rebotes Ofensivos	-0,2431569
Robos	0,2170984
Pérdidas	1,063205
Racha	0,1050802
Racha Rival	-0,1238877

Cuadro 23: Valor de las constantes para calcular la puntuación

En cuanto a la integración de esta fórmula a la app, podemos decir que ha sido una integración sencilla. Se han definido las constantes y con ellas se opera convenientemente siguiendo la fórmula anteriormente explicada.

- *Probabilidad de victoria:* El segundo dato que nos proporciona el modelo es la probabilidad de victoria de cada equipo. Es el mismo dato que el utilizado en la herística de predicción desarrollado por el alumno pero utilizando fórmulas diferentes. En esta ocasión se van a utilizar las mismas variables que hemos usado para calcular la puntuación de los equipos, lo que cambia es el valor de las constantes. Al aplicar la fórmula no obtendremos directamente la probabilidad de victoria del equipo, sino que obtendremos el valor de y^*, cuyo valor debe de ser consultado en la tabla de la función de distribución acumulada de una distribución normal:

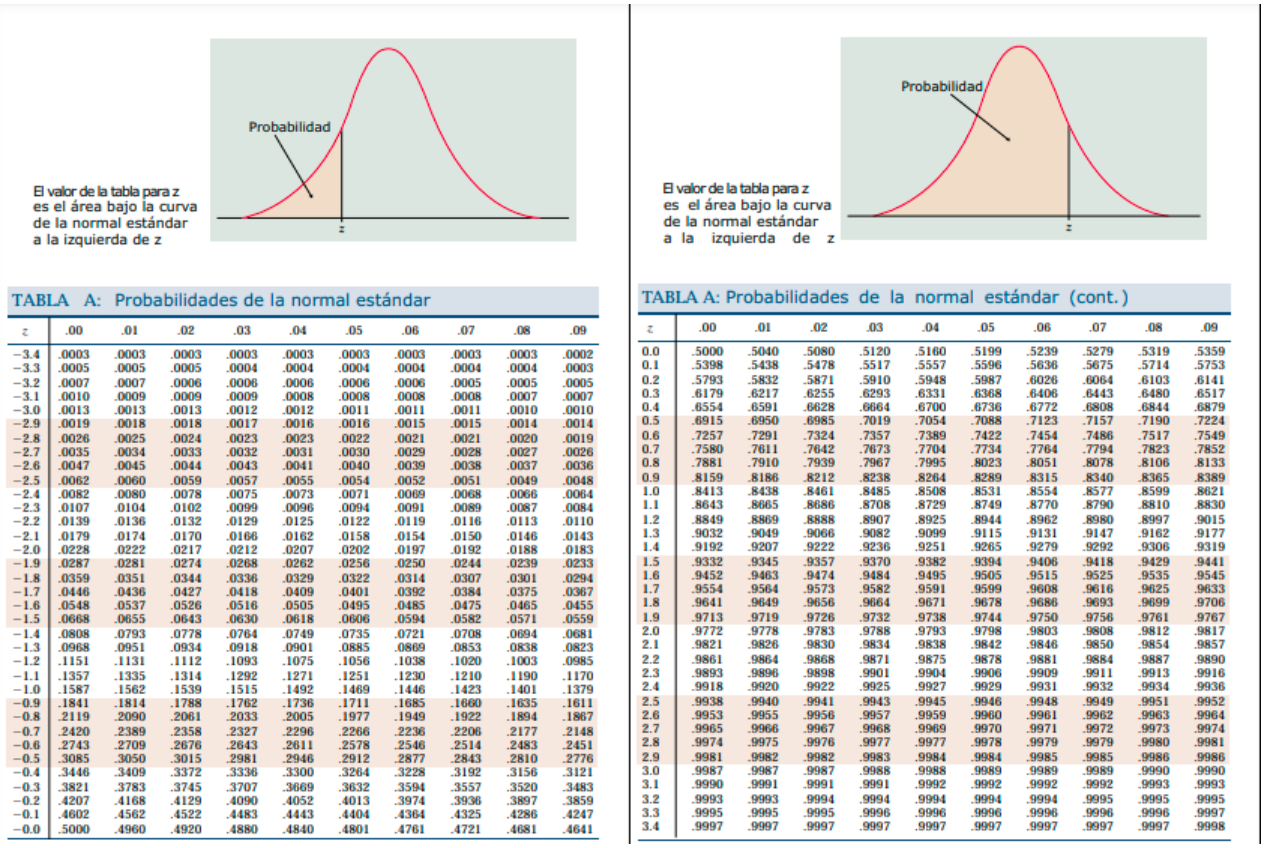

TABLA A: Probabilidades de la normal estándar

z	.00	.01	.02	.03	.04	.05	.06	.07	.08	.09
−3.4	.0003	.0003	.0003	.0003	.0003	.0003	.0003	.0003	.0003	.0002
−3.3	.0005	.0005	.0005	.0004	.0004	.0004	.0004	.0004	.0004	.0003
−3.2	.0007	.0007	.0006	.0006	.0006	.0006	.0006	.0005	.0005	.0005
−3.1	.0010	.0009	.0009	.0009	.0008	.0008	.0008	.0008	.0007	.0007
−3.0	.0013	.0013	.0013	.0012	.0012	.0011	.0011	.0011	.0010	.0010
−2.9	.0019	.0018	.0018	.0017	.0016	.0016	.0015	.0015	.0014	.0014
−2.8	.0026	.0025	.0024	.0023	.0023	.0022	.0021	.0021	.0020	.0019
−2.7	.0035	.0034	.0033	.0032	.0031	.0030	.0029	.0028	.0027	.0026
−2.6	.0047	.0045	.0044	.0043	.0041	.0040	.0039	.0038	.0037	.0036
−2.5	.0062	.0060	.0059	.0057	.0055	.0054	.0052	.0051	.0049	.0048
−2.4	.0082	.0080	.0078	.0075	.0073	.0071	.0069	.0068	.0066	.0064
−2.3	.0107	.0104	.0102	.0099	.0096	.0094	.0091	.0089	.0087	.0084
−2.2	.0139	.0136	.0132	.0129	.0125	.0122	.0119	.0116	.0113	.0110
−2.1	.0179	.0174	.0170	.0166	.0162	.0158	.0154	.0150	.0146	.0143
−2.0	.0228	.0222	.0217	.0212	.0207	.0202	.0197	.0192	.0188	.0183
−1.9	.0287	.0281	.0274	.0268	.0262	.0256	.0250	.0244	.0239	.0233
−1.8	.0359	.0351	.0344	.0336	.0329	.0322	.0314	.0307	.0301	.0294
−1.7	.0446	.0436	.0427	.0418	.0409	.0401	.0392	.0384	.0375	.0367
−1.6	.0548	.0537	.0526	.0516	.0505	.0495	.0485	.0475	.0465	.0455
−1.5	.0668	.0655	.0643	.0630	.0618	.0606	.0594	.0582	.0571	.0559
−1.4	.0808	.0793	.0778	.0764	.0749	.0735	.0721	.0708	.0694	.0681
−1.3	.0968	.0951	.0934	.0918	.0901	.0885	.0869	.0853	.0838	.0823
−1.2	.1151	.1131	.1112	.1093	.1075	.1056	.1038	.1020	.1003	.0985
−1.1	.1357	.1335	.1314	.1292	.1271	.1251	.1230	.1210	.1190	.1170
−1.0	.1587	.1562	.1539	.1515	.1492	.1469	.1446	.1423	.1401	.1379
−0.9	.1841	.1814	.1788	.1762	.1736	.1711	.1685	.1660	.1635	.1611
−0.8	.2119	.2090	.2061	.2033	.2005	.1977	.1949	.1922	.1894	.1867
−0.7	.2420	.2389	.2358	.2327	.2296	.2266	.2236	.2206	.2177	.2148
−0.6	.2743	.2709	.2676	.2643	.2611	.2578	.2546	.2514	.2483	.2451
−0.5	.3085	.3050	.3015	.2981	.2946	.2912	.2877	.2843	.2810	.2776
−0.4	.3446	.3409	.3372	.3336	.3300	.3264	.3228	.3192	.3156	.3121
−0.3	.3821	.3783	.3745	.3707	.3669	.3632	.3594	.3557	.3520	.3483
−0.2	.4207	.4168	.4129	.4090	.4052	.4013	.3974	.3936	.3897	.3859
−0.1	.4602	.4562	.4522	.4483	.4443	.4404	.4364	.4325	.4286	.4247
−0.0	.5000	.4960	.4920	.4880	.4840	.4801	.4761	.4721	.4681	.4641

TABLA A: Probabilidades de la normal estándar (cont.)

z	.00	.01	.02	.03	.04	.05	.06	.07	.08	.09
0.0	.5000	.5040	.5080	.5120	.5160	.5199	.5239	.5279	.5319	.5359
0.1	.5398	.5438	.5478	.5517	.5557	.5596	.5636	.5675	.5714	.5753
0.2	.5793	.5832	.5871	.5910	.5948	.5987	.6026	.6064	.6103	.6141
0.3	.6179	.6217	.6255	.6293	.6331	.6368	.6406	.6443	.6480	.6517
0.4	.6554	.6591	.6628	.6664	.6700	.6736	.6772	.6808	.6844	.6879
0.5	.6915	.6950	.6985	.7019	.7054	.7088	.7123	.7157	.7190	.7224
0.6	.7257	.7291	.7324	.7357	.7389	.7422	.7454	.7486	.7517	.7549
0.7	.7580	.7611	.7642	.7673	.7704	.7734	.7764	.7794	.7823	.7852
0.8	.7881	.7910	.7939	.7967	.7995	.8023	.8051	.8078	.8106	.8133
0.9	.8159	.8186	.8212	.8238	.8264	.8289	.8315	.8340	.8365	.8389
1.0	.8413	.8438	.8461	.8485	.8508	.8531	.8554	.8577	.8599	.8621
1.1	.8643	.8665	.8686	.8708	.8729	.8749	.8770	.8790	.8810	.8830
1.2	.8849	.8869	.8888	.8907	.8925	.8944	.8962	.8980	.8997	.9015
1.3	.9032	.9049	.9066	.9082	.9099	.9115	.9131	.9147	.9162	.9177
1.4	.9192	.9207	.9222	.9236	.9251	.9265	.9279	.9292	.9306	.9319
1.5	.9332	.9345	.9357	.9370	.9382	.9394	.9406	.9418	.9429	.9441
1.6	.9452	.9463	.9474	.9484	.9495	.9505	.9515	.9525	.9535	.9545
1.7	.9554	.9564	.9573	.9582	.9591	.9599	.9608	.9616	.9625	.9633
1.8	.9641	.9649	.9656	.9664	.9671	.9678	.9686	.9693	.9699	.9706
1.9	.9713	.9719	.9726	.9732	.9738	.9744	.9750	.9756	.9761	.9767
2.0	.9772	.9778	.9783	.9788	.9793	.9798	.9803	.9808	.9812	.9817
2.1	.9821	.9826	.9830	.9834	.9838	.9842	.9846	.9850	.9854	.9857
2.2	.9861	.9864	.9868	.9871	.9875	.9878	.9881	.9884	.9887	.9890
2.3	.9893	.9896	.9898	.9901	.9904	.9906	.9909	.9911	.9913	.9916
2.4	.9918	.9920	.9922	.9925	.9927	.9929	.9931	.9932	.9934	.9936
2.5	.9938	.9940	.9941	.9943	.9945	.9946	.9948	.9949	.9951	.9952
2.6	.9953	.9955	.9956	.9957	.9959	.9960	.9961	.9962	.9963	.9964
2.7	.9965	.9966	.9967	.9968	.9969	.9970	.9971	.9972	.9973	.9974
2.8	.9974	.9975	.9976	.9977	.9977	.9978	.9979	.9979	.9980	.9981
2.9	.9981	.9982	.9982	.9983	.9984	.9984	.9985	.9985	.9986	.9986
3.0	.9987	.9987	.9987	.9988	.9988	.9989	.9989	.9989	.9990	.9990
3.1	.9990	.9991	.9991	.9991	.9992	.9992	.9992	.9992	.9993	.9993
3.2	.9993	.9993	.9994	.9994	.9994	.9994	.9994	.9995	.9995	.9995
3.3	.9995	.9995	.9995	.9996	.9996	.9996	.9996	.9996	.9996	.9997
3.4	.9997	.9997	.9997	.9997	.9997	.9997	.9997	.9997	.9997	.9998

Figura 23: Tabla de distribución normal.

Como hemos comentado anteriormente, la fórmula es similar a la utilizada para obtener la puntuación de los partidos pero requiere una transformación final, ya que obtenemos un valor que deberá de ser buscado correctamente en la tabla mostrada anteriormente. La fórmula utilizada es la siguiente:

$y^* = Constante\_Puntuación + Asistencias * Coef\_Asistencias + Tapones * Coef\_Tapones + RebotesD * Coef\_RebotesD + Rebotes0 * Coef\_RebotesO + Robos * Coef\_Robos + Pérdidas * Coef\_Pérdidas + Racha * Coef\_Racha + Racha\ Rival * Coef\_RachaRival.$

El valor de las constantes en este caso es el siguiente:

Constante	**Valor**
Constante Fórmula	-0,0776508
Asistencias	0,0347215
Tapones	0,0441346
Rebotes Defensivos	0,0349014
Rebotes Ofensivos	-0,0360219
Robos	0,0300956
Pérdidas	-0,0790446
Racha	0,0206968
Racha Rival	-0,0212921

Cuadro 24: Valor de las constantes para calcular la probabilidad de victoria

El valor obtenido en la fórmula, se consulta en la tabla de la distribución normal, buscando la parte de las unidades y décimas en la primera columna. En la fila correspondiente al valor de la columna buscamos el valor de las centésimas. El resultado será la probabilidad normal del suceso.

8.3.1. Predicciones

A continuación, vamos a presentar los partidos que se han utilizado para probar la eficacia del modelo. Los partidos han sido los mismos que se han usado en el modelo desarrollado por el alumno para así poder hacer una comparación entre ambos modelos.

Como en el anterior modelo, presentamos los resultados en forma de tabla, pero añadimos dos nuevas columnas. La columna "Pred Pts" nos indica la predicción del resultado del encuentro. La columna colindante [-10,10] nos indica si la predicción en la puntuación del partido tiene una desviación respecto al resultado final comprendido en ese intervalo:

Local	Visitante	% Local	% Visitante	Pts	Pred Pts	[-10,10]	Resultado	Aprox
Toronto	Cavaliers	58,32 %	41,68 %	112-96	106-104	SI	**Acierto**	Muy buena
Nets	Dallas	51,93 %	48,07 %	98-115	105-104	NO	**Fallo**	Mala
Spurs	Clippers	37,35 %	62,65 %	101-134	103-106	NO	**Acierto**	Excelente
Bulls	Detroit	52,46 %	47,54 %	100-86	106-105	NO	**Acierto**	Buena
Portland	Pelicans	59,7 %	40,3 %	101-93	106-104	NO	**Acierto**	Buena
Nuggets	Celtics	62,76 %	37,24 %	87-105	106-102	NO	**Fallo**	Muy mala
Lakers	Bucks	50,86 %	49,14 %	113-106	105-104	SI	**Acierto**	Buena
Thunder	Knicks	37,07 %	62,93 %	97-119	103-107	NO	**Acierto**	Muy Buena
Jazz	Kings	71,06 %	28,94 %	128-115	108-102	NO	**Acierto**	Buena
Minnesota	76ers	20,22 %	79,78 %	113-122	100-109	NO	**Acierto**	Buena
Warriors	Heat	49,54 %	50,46 %	120-112	104-105	NO	**Fallo**	Mala
Houston	Hornets	48,32 %	51,68 %	97-122	104-105	NO	**Acierto**	Buena
Suns	Grizzlies	52,38 %	47,62 %	128-97	105-104	NO	**Acierto**	Buena
Orlando	Wizards	34,37 %	65,63 %	116-131	103-107	NO	**Acierto**	Muy Buena
Hawks	Pacers	54,58 %	45,42 %	129-117	106-104	NO	**Acierto**	Buena

Cuadro 25: Modelo Predictivo Estadístico. Test 1

Local	Visitante	% Local	% Visitante	Pts	Pred Pts	[-10,10]	Resultado	Aprox
Toronto	Heat	52,18 %	47,82 %	112-96	105-104	SI	**Acierto**	Buena
Nets	Celtics	50,76 %	49,24 %	98-115	106-104	NO	**Fallo**	Mala
Spurs	Pacers	51,88 %	48,12 %	101-134	106-102	NO	**Fallo**	Mala
Bulls	Thunder	47.76 %	52,24 %	100-86	103-105	NO	**Fallo**	Mala
Portland	Nuggets	36,28 %	63,72 %	101-93	103-107	NO	**Fallo**	Mala
Nuggets	Portland	52,13 %	47,87 %	87-105	105-104	NO	**Fallo**	Mala
Lakers	Hornets	51,73 %	48,27 %	113-106	105-104	SI	**Acierto**	Buena
Thunder	Rockets	42,99 %	57,01 %	97-119	104-106	NO	**Acierto**	Buena
Jazz	76ers	47.71 %	52,29 %	128-115	106-108	NO	**Fallo**	Mala
Minnesota	Heat	35,79 %	64,21 %	113-122	103-107	NO	**Acierto**	Buena
Warriors	Grizzlies	59,02 %	40,98 %	120-112	106-104	NO	**Acierto**	Buena
Houston	Wizards	27,71 %	72,29 %	97-122	102-109	NO	**Acierto**	Excelente
Suns	Cavaliers	57,37 %	42,63 %	128-97	106-103	NO	**Acierto**	Muy Buena
Orlando	Dallas	26,29 %	73,71 %	116-131	102-109	NO	**Acierto**	Muy Buena
Hawks	Clippers	36,94 %	63,06 %	129-117	103-106	NO	**Fallo**	Mala

Cuadro 26: Modelo Predictivo Estadístico. Test 2

8.3.2. Análisis de Resultados

El modelo ha acertado 20/30 resultados lo que supone un 66,6 % de acierto. Es un porcentaje que nos indica cierta fiabilidad. Esta fiabilidad se demuestra analizando los partidos en los que fallado. A excepción del partido que enfrentaba a los Nuggets vs Celtics, en el que ya hemos explicado, que cualquier modelo predictivo daría como ganador a los Nuggets por el historial de ambos equipos, el resto de partidos fallados la mayoría son pronósticos muy igualados. En la tabla de pruebas, se escogieron los partidos aleatoriamente, y algunos son al inicio de la temporada lo que pone en apuros al modelo a la hora de predecirlos. Otros partidos que ha fallado, han sido por enfrentarse dos equipos que llegaban con una dinámica muy mala ó dos equipos que eran realmente buenos, por lo que ya nos imaginamos lo dificil que es predecir en este tipo de situaciones.

El modelo también ha obtenido varios "Excelentes" en la columna de "Aproximación". Algunos partidos a priori son fáciles de predecir, como por ejemplo el Spurs vs Clippers, en el que ambos modelos tenían claro que había un equipo muy superior al otro. En cuanto a la predicción de el puntuaje en los partidos, no podemos decir que haya sido bueno, pero es que predecir este tipo de variables es extremadamente dificíl. La mayoría de reultados, se han salido del intervalo propuesto con el error asumido que esperábamos, pero creemos que es normal equivocarse en este tipo de pronósticos.

8.4. Predicciones en PlayOff

Los PlayOff de la NBA constituyen la fase final de la competición. Se enfrentan los 8 mejores equipos de cada conferencia, por lo que nivel de los encuentros se eleva. Las eliminatorias o series se juegan en un formato al mejor de 7 partidos, en el que se tienen que ganar 4 partidos para clasificarse para la siguiente ronda. El equipo que posea la ventaja de campo en cada eliminatoria disputará los partidos 1, 2, 5 y 7 como local, mientras que el resto de partidos se jugará en el pabellón del equipo contrario. Se establece como equipo con ventaja de campo al que haya tenido mejor balance en liga entre los 2 contendientes de una eliminatoria. En el momento en que un equipo gana 4 partidos, se clasifica para la siguiente ronda de eliminatoria, sin jugar obligatoriamente los 7 partidos programados.

Prueba 1: Serie Hawks vs 76ers (1-1).

Hawks ganó el primer partido de la serie en el campo de los 76ers. Se mostró superior durante todo el partido llegando a tener una ventaja de 20 puntos de diferencia máxima y empezando el útlimo cuarto aventajando en 16 puntos a su rival. Un último gran cuarto de los 76 ers les hicieron soñar con la remontada que finalmente no pudo ser ejecutada.

El segundo partido se volvía a jugar en casa de los 76ers y esta vez la victoria sería para los 76 ers que conseguían de esta manera igualar la serie. De esta manera nos enfrentamos a un pronóstico que debería de ser muy igualado. El partido se jugará en casa de los Hawks, los cuales han ganado 13 partidos seguidos en Atlanta desde la derrota por 120-109 ante los Milwaukee Bucks el 15 de abril.

Con este pequeño análisis la heurística predictiva del alumno predice que los 76ers tienen un 57,5 % de probabilidades de ganar mientras que los Hawks cuentan con un 42,5 % de opciones. El modelo predictivo estadístico augura que los Hawks se harán con la victoria en un partido de lo más igualado dándole un 51,5 % de probabilidades de ganar por un 48,5 % que dan a los 76 ers. Parece que el pronóstico del

segundo algoritmo predice un partido muy igualado, donde el factor cancha puede ser muy determinante, ya que los Hawks encadenan un gran número de victorias consecutivas en sus partidos de local.

Resultado:

Victoria de los 76ers por 127-111. Acierta la heurística predictiva del alumno y falla el modelo predictivo estadístico. Fue un partido muy igualado en el que un gran 3 cuarto de los 76ers decantó la balanza a favor del equipo de Philadelphia.

Prueba 2: Serie Bucks vs Nets (1-2).

Un duelo en el que a priori todas las casas de apuestas darían como favorito a los Nets. En estos pronósticos se incluyen nuestro dos modelos. Después de ganar los dos primeros partidos los Nets, el tercer encuentro se lo llevaron los Bucks por una diferencia mínima de 3 puntos.

El cuarto partido, será un duelo decisivo, la heurística predictiva del alumno nos dice que los Nets se llevarán el encuentro con una probabilidad del 58,7 % y el modelo estadístico cree que será un duelo más igualado otorgando un 55 % de probabilidades a los Nets de llevarse el encuentro. Ambos modelos apuntan al mismo favorito y con razón, ya que los Nets se han mostrado superiores en los dos primeros partidos de la serie y en el tercero perdieron por una mínima diferencia.

Resultado:

Victoria de los bucks por 107-96. Fue una sorpresa para todos este resultado, buena parte de la culpa la tiene la lesión en el segundo cuarto de Irving, una de las estrellas de los Nets que sumada a la baja del nueve veces All-Star y MVP de 2018 James Harden, han hecho que los modelos fallen en su pronóstico.

Prueba 3: Serie Clippers vs Jazz (0-1).

Los Jazz, el mejor equipo de la liga juega en casa contra los Angeles Clippers en el segundo partido de la serie. El primero de los partidos fue ganado por los Jazz en un partido de lo más igualado que se decidió en la última jugada del encuentro.

En este segundo partido se enfrentan de nuevo en casa de los Jazz, y los dos modelos apuntan al mismo vencedor: Utah Jazz. La heurística predictiva del alumno predice que se llevaran el partido con cierta holgura, pronósticando un 62,2 % de opciones a los Jazz y un 37,8 % de opciones a Clippers. El modelo predictivo estadístico preveé un partido muy igualado, otorgando a los Jazz un 53,5 % de probabilidades de victoria por un 46 % de opciones a Clippers. Apostar a una victoria de los Jazz es de lo más lógico y más viendo el historial de partidos durante la temporada de este equipo, sin embargo, en frente tienen un rival muy bueno, y que en el primer partido a punto estuvo de dar la sorpresa.

Resultado:

Victoria de los Jazz por 117-111. Otro partido muy igualado dónde pudo ganar cualquiera. Finalmente los dos modelos acertaron, siendo más preciso el modelo predictivo estadístico que predijo un partido

igualado, por contra del modelo del alumno, que esperaba una victoria clara de los Jazz.

Prueba 4: Serie Denver vs Suns (0-1).

Otro partido que se prevé muy igualado entre dos buenos equipos. Suns ha quedado en segundo lugar en la clasificación de la liga y Denver quinto, por lo que ya sabemos que son dos equipos de muy alto nivel.

El primero de los partidos se lo llevaron los Suns holgadamente con una diferencia de 17 puntos. Todo hace indicar que volverán a repetir victoria en su campo. La heurística predictiva del alumno nos dice esta vez que los Suns ganarán el partido con una probabilidad del 60,4 % por un 39,6 % que otorga a Denver. El modelo predictivo estadístico como nos tiene acostumbrados en estos partidos, prevee un duelo más igualado, dándo como favorito a los Suns con un 53,5 % de opciones frente al 46,5 % que le otorga a Denver.

Resultado:

Victoria con autoridad de los Suns por 123-98. Acertaron ambos modelos pero estuvo más fino el modelo del alumno demostrando la superioridad de los Suns frente a Denver.

8.5. Comparación entre ambos modelos

Tras realizar todas las pruebas descritas, podemos concluir que son dos modelos muy fiables a la hora de predecir partidos. En términos estadísticos, el modelo del alumno acertó un total de 22 de 30 posibles partidos, lo que supone un 73´3 % de acierto. Por otro lado, el modelo predictivo estadístico acertó un total de 20/30 resultados suponiendo el 66,6 % de acierto. Si comparamos ambos modelos, el modelo del alumno tiene un rendimiento en 22/20= 1,1 veces mejor que el modelo estadístico. Son rendimientos similares y muy buenos que nos dan cierta seguridad a la hora de hacer pronósticos.

A la hora de comparar ambos modelos, lo primero que nos llama la atención son lo ajustados que son los pronósticos en el modelo estadístico para los partidos de PlayOff. En cambio, esto para los partidos de la liga regular no pasa tanto. La razón de este suceso, es que en el modelo predictivo estadístico es muy importante las rachas de los dos equipos que se enfrentan, y marcan la diferencia, ya que las demás estadísticas suman muchos puntos para ambos equipos porque todos los equipos tienen buenos jugadores que aportan porcentaje a las probabilidades de ganar. En los partidos de PlayOff, las rachas son más pequeñas, porque al acabar la temporada comienzan de 0, ya que los PlayOff son prácticamente una competición a parte. En los pronósticos de la liga regular hay partidos en los que también da pronosticos igualados, esto puede deberse a que sean partidos de principio de temporada con rachas con valor 1 ó 0, lo que un pronóstico igualado es muy razonable, o que por otro lado los dos equipos lleguen muy igualado al partido.

En cuanto a la variable puntuación de este modelo, podemos entender que se trata de una variable muy complicada de predecir, se puede orientar si se van a meter muchos puntos o va a ser un partido de pocos puntos pero es una verdadera lotería ya que depende del día que tengan los jugadores, lesiones, objetivos en juego..etc.

En cuanto a la heurística predictiva propuesta por alumno, en la liga regular tiene un acierto realmente sorprendente. La razón de esto es que la mayoría de la probabilidad de victoria que otorga a los equipos

está directamente relacionada con el puesto clasificatorio y la racha. Tiene mucho sentido, porque cuando se enfrentan dos equipos que se llevan muchos puestos en la clasificación la lógica nos dice que algo tiene el equipo mejor clasificado para sacarle tantos puestos en el ranking al otro. Y parece ser que la liga sigue esta lógica, eso sí, el modelo tiene sus limitaciones. Sufre cuando se intenta pronosticar partidos de inicio de temporada,en el que ambos equipos tienen un puesto similar, da un pronostico igualado y a veces falla. La mayoría de los partidos en que ha fallado en los test expuestos en tablas, ha sido por esto, partidos del inicio de la competición en el que los equipos no estaban asentados. Los partidos en los que se enfrentan dos equipos con estadísticas similares también son los más propensos a fallar, pero esto es lógico si dos equipos tienen estadísticas similares, el pronostico será igualado y podrá ganar cualquiera.

9. Análisis de Riesgos

9.1. Introducción

El concepto de riesgo se define como "Evento o circunstancia cuya probabilidad de ocurrencia es incierta, pero que, en caso de aparecer, tiene un efecto (positivo o negativo) sobre los objetivos de un proyecto." Cualquier proyecto independientemente de su tamaño y complejidad, contiene en mayor o menor medida una serie de problemas. Por lo que deberemos de considerar como tratar estos inconvenientes. Cualquier elemento que se salga de la planificación inicial del proyecto, supone un riesgo y debe de ser contrarrestado con un plan de riesgos.

El plan de riesgos, es una metodología utilizada para acotar al máximo los posibles problemas que aparezcan en el proyecto. Las reuniones con todos los miembros del proyecto, son una de las mejores maneras de analizar los posibles riesgos y buscar soluciones que amortiguen su impacto en el proyecto.

A continuación, vamos a analizar las causas más cómunes. Podemos clasificar los principales riesgos en cuatro tipos [25]:

- *Riesgos derivados del cliente:* El cliente es el primer riesgo potencial de cualquier proyecto. En este apartado, destacan las expectativas poco realistas que esperan los usuarios en el resultado final del proyecto.
- *Riesgos derivados de la empresa suministradora de servicios:* Un primer riesgo que encontramos está en el lado de una empresa externa. En nuestro caso, estos riesgos derivan de las empresas encargadas de proporcionarnos los datos estadísticos para la aplicación.
- *Riesgos derivados de la planificación:* La planificación es uno de los aspectos más importantes para cualquier proyecto. Se debe definir claramente los plazos y los encargados de la realización de cada una de las fases. Una deficiente planificación, provocará riesgos que pueden poner en duda la viabilidad del proyecto.
- *Riesgos derivados del producto:* Existen una gran variedad de riesgos que surgen directamente del proceso de desarrollo del producto. Hasta que los miembros del equipo no se meten de lleno en el proceso de desarrollo de las tareas, no saben los verdaderos riesgos y si las soluciones pensadas son viables.

9.2. Objetivo del Plan de Riesgos

El plan de riesgos constituye una descripción de responsabilidades y actividades relacionadas con la gestión de riesgos. Su finalidad es definir claramente las responsabilidades y riesgos que surgan en el proyecto. Por lo general, el plan de riesgos tiene los siguientes objetivos:

- *Identificar, analizar y cuantificar posibles riesgos que puedan aparecer durante el desarrollo de un proyecto software.*
- *Desarrollar respuestas adecuadas para los posibles riesgos.*
- *Monitorizar el transcurso de un proyecto para evaluar el estado de los riesgos y actuar en consecuencia.*

9.3. Identificar Riesgos

La identificación de Riesgos es un proceso clave para conseguir que el resultado final del proyecto sea el esperado. Una mala identificación de riesgos hará que nos encontremos con imprevisto a lo largo del proceso de desarrollo. Si hemos detectado correctamente estos riesgos y hemos elaborado un plan de contigencia adecuado podremos reducir el impacto que tendrían inicialmente. Existen numerosas técnicas de identificación, destacando las siguientes:

- Investigación en los informes del proyecto.
- LLuvia de ideas en las reuniones de planificación.
- Técnica Delphi[27].
- Método DAFO[28].
- Diagrama de Ishikawa [29].

En este proyecto se va a utilizar la lluvia de ideas como método base para la identificación de posibles riesgos. Se pedirá opinión a los tutores de las posibles amenazas que pongan en duda la viabilidad del proyecto.

9.4. Gestión de Riesgos

Tras explicar los principales tipos de riesgos que podemos encontrar en un proyecto software, vamos a enumerar todos aquellos que hemos detectado en el presente proyecto:

R01	Inexperiencia del alumno
Causa	Carencia de conocimientos
Consecuencia	Fallos en el proyecto
Prioridad	ALTA
Probalidad	BAJA
Impacto	MEDIO
Plan de acción	Consultar documentación o tutores
Plan de contingencia	Analizar los nuevos conocimientos adquiridos

Cuadro 27: Riesgo 01

R02	Mala planificación temporal
Causa	No se cumplen los plazos establecidos
Consecuencia	Retraso en la finalización del proyecto
Prioridad	ALTA
Probalidad	MEDIA
Impacto	ALTO
Plan de acción	Añadir más temporalidad a las fases del proyecto
Plan de contingencia	Hacer un seguimiento exhaustivo del proyecto

Cuadro 28: Riesgo 02

R03	Reducir costes en el proyecto
Causa	Proyecto con presupuesto reducido
Consecuencia	Bajada de nivel de prestaciones
Prioridad	ALTA
Probalidad	MEDIA
Impacto	ALTO
Plan de acción	Realizar los test necesarios y pruebas de calidad
Plan de contingencia	Buscar otros caminos a coste cero

Cuadro 29: Riesgo 03

R04	Pérdida de material
Causa	Pérdida de trabajo hecho en el proyecto
Consecuencia	Volver a realizar el trabajo perdido
Prioridad	ALTA
Probalidad	BAJA
Impacto	ALTO
Plan de acción	Volver a hacer el material perdido
Plan de contingencia	Copias de seguridad

Cuadro 30: Riesgo 04

R05	Mayor inversión de recursos
Causa	Mayor esfuerzo en una tarea que el esperado
Consecuencia	No se cumplen lo plazos esperados
Prioridad	ALTA
Probalidad	MEDIA
Impacto	ALTO
Plan de acción	Replanificar proyecto
Plan de contingencia	Planificación cuidadosa

Cuadro 31: Riesgo 05

R06	Aprendizaje de nuevas tecnologías
Causa	Aumento de tiempo para entender herramientas software
Consecuencia	Retraso entrega de tareas
Prioridad	ALTA
Probalidad	MEDIA
Impacto	ALTO
Plan de acción	Replanificar proyecto
Plan de contingencia	Consultar documentación

Cuadro 32: Riesgo 06

R07	Herramientas Software inadecuadas
Causa	Las herramientas para el desarrollo del proyecto no funcionan como se esperaba
Consecuencia	Retraso entrega de tareas
Prioridad	ALTA
Probalidad	BAJA
Impacto	ALTO
Plan de acción	Utilizar otras herramientas
Plan de contingencia	Documentarse antes de elegir las herramientas

Cuadro 33: Riesgo 07

R08	App final no se ajusta a la sociedad actual
Causa	No se conoce las expectativas que tienen los usuarios en este tipo de aplicaciones
Consecuencia	Rediseño de la aplicación. Mas versiones
Prioridad	BAJA
Probalidad	BAJA
Impacto	CRÍTICO
Plan de acción	Cambio en el diseño final
Plan de contingencia	Sondear a usuarios previamente

Cuadro 34: Riesgo 08

R09	Nuevas funcionalidades en la App
Causa	Los tutores exigen implementar nuevas opciones en la App
Consecuencia	Rediseño de proyecto
Prioridad	ALTA
Probalidad	MEDIA
Impacto	MEDIO
Plan de acción	Nuevo listado de requisitos
Plan de contingencia	Clarificar al inicio el listado de requisitos definitivo

Cuadro 35: Riesgo 09

R10	Falta de información del proyecto a los tutores
Causa	No se realizan las suficiones reuniones
Consecuencia	Posibilidad de no realizar tareas correctamente
Prioridad	ALTA
Probalidad	BAJA
Impacto	MEDIO
Plan de acción	Solicitar reuniones más frecuentemente
Plan de contingencia	Desarrollar un plan de seguimiento más robusto

Cuadro 36: Riesgo 10

R11	Baja disponibilidad de algún miembro
Causa	No se tiene el tiempo suficiente para llevar a cabo el proyecto
Consecuencia	Incumplimiento de planificación temporal
Prioridad	ALTA
Probalidad	BAJA
Impacto	MEDIO
Plan de acción	Planificación más realista
Plan de contingencia	Ajustar previamente disponibilidad de los miembros

Cuadro 37: Riesgo 11

R12	Dispositivos Defectuosos
Causa	Avería en algún dispositivo de desarrollo
Consecuencia	Paralización del proyecto
Prioridad	ALTA
Probalidad	BAJA
Impacto	ALTA
Plan de acción	Arreglar dispositivos
Plan de contingencia	Tener segundas unidades (redundancia)

Cuadro 38: Riesgo 12

10. Requisitos

10.1. Introducción

Los requisitos son las propiedades que deben caracterizar al producto final como consecuencia del desarrollo de un proyecto. El incumplimiento de los requisitos, es uno de los principales problemas en el desarrollo de proyectos, en los que habitualmente no se cumplen las expectativas del usuario.

En la actualidad, la gestión de requisitos cobra una importancia aún mayor. Un proceso de selección de requisitos insuficiente puede desembocar en un producto final defectuoso. El proceso de selección de requisitos se basa en documentar y analizar los deseos, necesidades y expectativas de un grupo de usuarios y transformarlos en requisitos en el proyecto.

Existen dos clases de requisitos conocidos como requisitos de usuario y requisitos de sistema:

- *Requisitos de usuario:* Suelen ser declaraciones en lenguaje natural y multiples diagramas de los servicios del sistema y de los limites sobre los que trabaja.
- *Requisitos de sistema:* Es documentación detallada que describe los servicios del sistema. Constituye un contrato entre el cliente y el desarrollador. Debe de ser una especificación completa y consistente del sistema

Todos las clases de requisitos se clasifican en esos dos categorías. A parte de estos dos categorías, exiten numeros tipos de requisitos para un proyecto software. Nosotros nos vamos a centrar en 3 tipos, que son los típicos y más útiles en la gestión de requisitos en un proyecto: requisitos funcionales, requisitios no funcionales y requisitos de información.

10.2. Requisitos Funcionales

Los requisitos funcionales son aquellos que describen las propiedades y servicios que el sistema debe proporcionar a los usuarios. Describen el funcionamiento del sistema, los servicios que debe de proporcionar, como debe reaccionar a entradas normales y ante situaciones particulares.

En cuanto a los requisitos funcionales de usuario solemos encontrarnos frases muy generales sobre lo que el sistema debería de hacer. Son los objetivos del sistema. Por otro lado encontrados los requisitos funcionales del sistema, que deben de describir los servicios que provee el sistema con todo detalle (incluyendo casos de uso).

Los requisitos funcionales de usuario y de sistema de este proyecto se recogen en la siguiente tabla:

ID	Requisito	Descripción
RF01	Visualizar noticias	El sistema deberá permitir visualizar el listado de noticias.
RF02	Consultar Clasificación	El sistema deberá permitir consultar la clasificación de la liga.
RF03	Consultar equipos	El sistema deberá permitir visualizar el listado de equipos de la liga.
RF04	Ver próximo partido	El sistema deberá permitir consultar el próximo partido de un equipo de la liga.
RF05	Ver pronóstico	El sistema deberá permitir visualizar el pronóstico de los próximos partidos de la liga.
RF06	Ver jugadores	El sistema deberá permitir visualizar los jugadores pertenecientes a cualquier equipo de la competición.
RF07	Consultar estadísticas de jugador	El sistema deberá permitir visualizar las estadísticas de cualquier jugador.
RF08	Filtrar temporada	El sistema deberá permitir consultar las estadísticas de la temporada especificada por el usuario.
RF09	Visualizar partidos de jugador	El sistema deberá permitir visualizar los partidos disputados por un jugador.
RF10	Visualizar estadísticas equipo	El sistema deberá permitir visualizar las estadísticas de cualquier equipo de la liga.
RF11	Consultar sustituto	El sistema deberá permitir visualizar el sustituto más idóneo para un puesto elegido por el usuario.
RF12	Consultar estadio	El sistema deberá permitir visualizar la información relativa a los estadios de la liga.
RF13	Geolocalizar estadio	El sistema deberá permitir visualizar la localización de los estadios de la liga.
RF14	Sincronización de datos	El sistema deberá permitir la correcta sincronización de datos.
RF15	Exportar datos	El sistema deberá permitir la exportación de las estadisticas visualizadas por el usuario.
RF16	Informar de error	El sistema deberá informar al usuario de que no dispone de datos para la consulta solicitada .

Cuadro 39: Requisitos Funcionales

10.3. Requisitos No Funcionales

Los requisitos no funcionales expresan otras caracteristicas y restriciones que tiene que presentar el sistema para alcanzar el éxito. Por regla general, este tipo de requisitos suelen tener características relacionadas con el rendimiento, temporalidad, usabilidad etc.

Como hemos dicho, definen propiedades emergentes del sistema y pueden también especificar la utilización de alguna herramienta en particular, como por ejemplo, un lenguaje de programación o un método de desarrollo. Este tipo de requisitos pueden ser más críticos que los funcionales:

- *Si un requisito funcional no se cumple, el sistema puede utilizarse parcialmente.*
- *Si un requisito no funcional no se cumple, el sistema puede quedar inutilizado.*

Podemos clasificar los requisitos no funcionales en requisitos del producto, requisitos de la organización y requisitos externos. A continuación, en el cuadro 42, podemos visualizar la tabla los requisitos no funcionales que se han extraído de este proyecto.

10.4. Requisitos de Información

Los requisitos de información son formas especializadas de requisitos que permiten expresar los datos que albergará el sistema. A continuación, en el cuadro 43, podemos ver todos los requisitos de información del proyecto.

ID	**Requisito**	**Descripción**
RNF01	Usabilidad	La aplicación debe caracterizarse por unas vistas sencillas e intuitivas que faciliten la usabilidad al usuario.
RNF02	Compatibilidad	La aplicación deberá de ser compatible con todos los dispositvos con el SO Android instalado.
RNF03	Eficiente	Los tiempos de respuesta deberán de ser lo más reducidos posibles, intentando en todos los casos que se situeen por debajo de 4 sg.
RNF04	Datos actualizados	La aplicación mostrará las estadísticas actualizadas en todo momento.
RNF05	Lectura APIS	La aplicación leerá los datos de las APIS Sportsdataio y Balldonltlie.
RNF06	Formato CSV	La aplicación exportará los datos en formato CSV para su posterior lectura por parte del usuario.
RNF07	Garantía de la información	El sistema proporcionará datos contrarrestados y fiables de fuentes autorizadas .
RNF08	Fácil de recordar	El funcionamiento de la aplicación debe de ser fácil de recordar.
RNF09	Robustez	La aplicación deberá ser robusta y tener un control de errores que informe al usuario cuando haya algún tipo de problema.

Cuadro 40: Requisitos No Funcionales

ID	Requisito	Descripción
RI01	Noticias	Titular, fecha, contenido y fuente.
RI02	Equipos	Conferencia, división, escudo y jugadores.
RI03	Jugadores	Nombre, apellidos, posición , sueldo y foto.
RI04	Estadísticas Jugadores	Partidos jugados, minutos, puntos, porcentajes de tiro, rebotes, asistencias, robos, tapones , pérdidas y eficiencia .
RI05	Estadísticas Equipos	Victorias, derrotas, racha, porcentaje, puntos a favor, puntos en contra, ranking, victorias hasta el liderato, victorias en casa, victorias a domicilio, derrotas en casa , derrotas a domicilio.
RI06	Partidos	Minutos, puntos, porcentajes de tiro, rebotes, asistencias, robos, tapones ,pérdidas.
RI07	Estadio	Nombre, dirección, localidad y capacidad .
RI08	Clasificación	Posición, nombre del equipo, partidos jugados, victorias, derrotas y porcentaje de victoria .
RI09	Pronóstico	Rival, fecha, posición ranking y probabilidad de victoria .

Cuadro 41: Requisitos de Información

11. Plan Software

11.1. Introducción

En esta sección se va a realizar un análisis completo sobre el plan software del proyecto. Se explicarán los actores que participan en el sistema, los casos de uso que llevan acabo estos actores y el modelo de dominio del sistema.

11.2. Actores

En este proyecto el sistema contará únicamente con un actor que será el usuario que utilice la aplicación para consultar los datos. Es cierto que como hemos comentado en otras secciones, podemos diferenciar a dos tipos de usuarios, por un lado los de caracter técnico, que utilicen las opciones de la app para exportar datos, y por el otro lado los usuarios que usan la app para fines informativos. Como ambos roles pueden ser desempeñados por cualquier usuario, ya que no hay restricciones en cuanto al uso de las funcionalidades de la app, solo representaremos a este actor en el sistema:

Actor	**Usuario**
Descripción	Se trata de cualquier usuario que acceda a visualizar datos en la app.
Características	Se caracteriza por poder utilizar todas las funciones que brinda la app.
Relaciones	Sin relaciones, no hay más usuarios en el sistema.
Referencias	-

Cuadro 42: Actores del sistema

11.3. Diagrama de casos de uso

En esta sección, se proporcionará los principales diagramas de caso de uso que explican el funcionamiento del sistema. A continuación, vemos el diagrama principal de casos de uso que define al sistema:

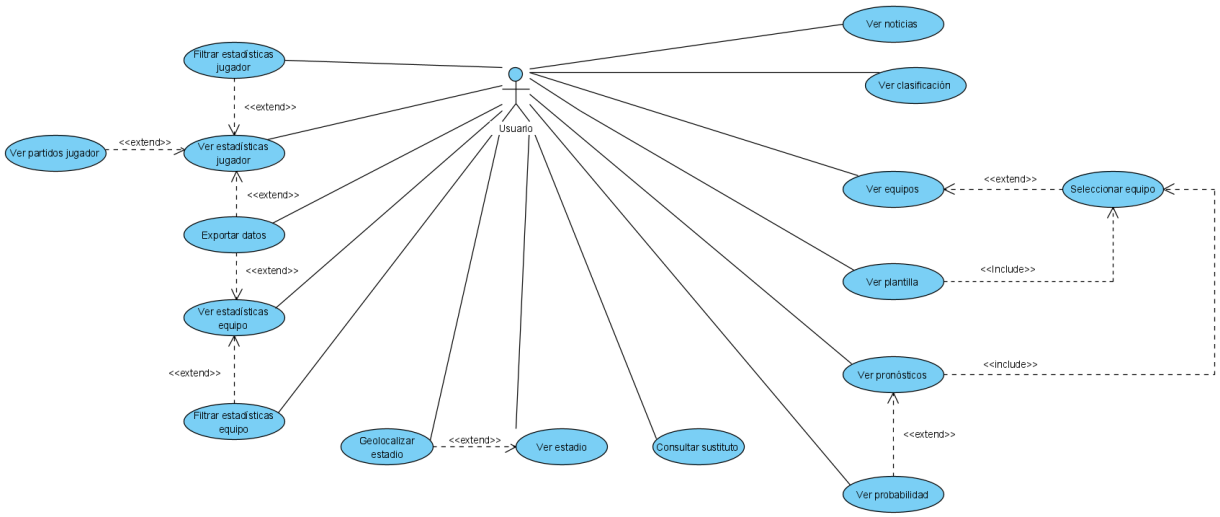

Figura 24: Esquema de casos de uso.

En la figura 21 se muestran las diferentes acciones que puede realizar el usuario en la aplicación. El usuario podrá acceder a todas las funciones que proporciona el sistema una vez que inicia la aplicación y se carguen los datos.

11.4. Descripción casos de uso

Ver Noticias

CU-01	**Ver noticias**
Actor	Usuario
Tipo	Secundario
Precondición	Ninguna
Postcondición	Se muestran las últimas noticias de la liga.
Propósito	Mostrar las últimas novedades de la competición.
Resumen	El usuario desea ver las noticias de la liga NBA.

Cuadro 43: CU-01: Ver noticias

CU-01	**Curso Normal**
Acción del actor	**Acción del sistema**.
1. El usuario selecciona el icono de noticias	2. El sistema realiza la consulta correspondiente a la API
4. El usuario ve la lista de noticias	3. El sistema carga las noticias

Cuadro 44: CU-01: Curso Normal

CU-01: Curso Alternativo
2a. Si no se esta conectado a internet no se mostrarán datos. Se informará al usuario de que no existe dicha conexión.
2b. Si no existen noticias, se informará al usuario de que no existen noticias actualmente.

Cuadro 45: CU-01: Curso Alternativo

Ver clasificación

CU-02	**Ver clasificación**
Actor	Usuario
Tipo	Secundario
Precondición	Ninguna
Postcondición	Se muestra la clasificación actualizada.
Propósito	Mostrar la clasificación de la liga.
Resumen	El usuario desea ver la clasificación de la NBA.

Cuadro 46: CU-02: Ver clasificación

CU-02	**Curso Normal**
Acción del actor	**Acción del sistema**.
1. El usuario selecciona el icono de clasificación	2. El sistema realiza la consulta pertinente a la API.
5. El usuario ve la clasificación	3. El sistema ordena los equipos conforme al porcentaje de victoria
-	4. El sistema carga la clasificación

Cuadro 47: CU-02: Curso Normal

CU-02: Curso Alternativo
2a. Si no hay conexión a internet se informa al usuario de que se necesita conexión a una red para mostrar los datos.

Cuadro 48: CU-02: Curso Alternativo

Ver equipos

CU-03	**Ver equipos**
Actor	Usuario
Tipo	Secundario
Precondición	Ninguna
Postcondición	Se muestra la lista de equipos
Propósito	Mostrar el listado de equipos de la liga
Resumen	El usuario quiere visualizar los equipos que participan en la liga

Cuadro 49: CU-03: Ver equipos

CU-03	**Curso Normal**
Acción del actor	**Acción del sistema**.
1. El usuario selecciona el icono de equipos	2. El sistema realiza la consulta a la API
4. El usuario ve el listado de equipos	3. El sistema carga el listado de equipos

Cuadro 50: CU-03: Curso Normal

CU-03: Curso Alternativo
2a. Si no hay conexión a internet se informa al usuario de que se necesita conexión a una red para mostrar los datos.

Cuadro 51: CU-03: Curso Alternativo

Ver plantilla

CU-04	**Ver plantilla**
Actor	Usuario
Tipo	Secundario
Precondición	Haber elegido previamente un equipo
Postcondición	Se muestra el listado de jugadores para el equipo seleccionado
Propósito	Mostrar listado de jugadores
Resumen	El usuario quiere ver los jugadores de un equipo de la liga

Cuadro 52: CU-04: Ver plantilla

CU-04	**Curso Normal**
Acción del actor	**Acción del sistema**.
1. El usuario selecciona ver plantilla	2. El sistema realiza la consulta a la API
4. El usuario ver el listado de jugadores	3. El sistema carga los datos

Cuadro 53: CU-04: Curso Normal

CU-04: Curso Alternativo
2a. Si no hay conexión a internet se informa al usuario de que se necesita conexión a una red para mostrar los datos.

Cuadro 54: CU-04: Curso Alternativo

Ver pronóstico

CU-05	**Ver pronóstico**
Actor	Usuario
Tipo	Secundario
Precondición	Ninguna
Postcondición	Mostrar pronóstico para un partido
Propósito	Hacer pronóstico de un partido
Resumen	El usuario quiere saber lo que va a ocurrir en un partido futuro

Cuadro 55: CU-05: Ver pronóstico

CU-05	**Curso Normal**
Acción del actor	**Acción del sistema**.
1. El usuario selecciona la opción de realizar pronóstico	2. El sistema realiza la consulta a la API y carga los datos necesarios para realizar el pronóstico.
3. El usuario selecciona el equipo del que quiere saber el pronóstico	4. El sistema consulta cuál es el rival y la fecha del encuentro para dicho equipo.
6. El usuario ve el próximo partido	5. El sistema carga los datos

Cuadro 56: CU-05: Curso Normal

CU-05: Curso Alternativo
2a. Si no hay conexión a internet se informa al usuario de que se necesita conexión a una red para mostrar los datos.
4a. Si no hay conexión a internet se informa al usuario de que se necesita conexión a una red para mostrar los datos.

Cuadro 57: CU-05: Curso Alternativo

Ver probabilidad

CU-06	**Ver probabilidad**
Actor	Usuario
Tipo	Secundario
Precondición	Haber seleccionado un equipo para realizar pronóstico
Postcondición	Mostrar probabilidad de victoria de un equipo
Propósito	Enseñar un porcentaje de victoria de un equipo para un partido.
Resumen	El usuario quiere ver la probabilidad de victoria que tiene un equipo en determinado partido.

Cuadro 58: CU-06: Ver probabilidad

CU-06	**Curso Normal**
Acción del actor	**Acción del sistema**.
1. El usuario selecciona la opción de ver probabilidad	2. El sistema realiza los cálculos necesarios con los datos obtenidos previamente
4. El usuario visualiza la probabilidad de victoria	3. El sistema comprueba si la probabilidad es mayor o menor que el 50 % y le asigna un color

Cuadro 59: CU-06: Curso Normal

CU-06: Curso Alternativo
2a. Si no hay conexión a internet se informa al usuario de que se necesita conexión a una red para mostrar los datos.

Cuadro 60: CU-06: Curso Alternativo

Consultar sustituto

CU-07	**Consultar sustituto**
Actor	Usuario
Tipo	Secundario
Precondición	Haber seleccionado previamente un equipo
Postcondición	Mostrar el sustituto idóneo para una posición en un equipo
Propósito	En contrar el mejor jugador para sustituir a otro.
Resumen	El usuario quiere saber quién es el jugador idóneo para sustituir a otro en una posición no habitual para él

Cuadro 61: CU-07: Consultar sustituto

CU-07	**Curso Normal**
Acción del actor	**Acción del sistema**.
1. El usuario selecciona la opción pizarra	3. El sistema realiza la consulta a la API para obtener los datos y realizar los cálculos.
2. El usuario selecciona la posición que quiere cubrir y pulsa el botón “procesar”	4. El sistema comprueba dependiendo de la posición elegida qué jugadores tienen mejor ratio.
6. El usuario visualiza el jugador elegido por el sistema	5. El sistema obtiene el nombre y ratio del jugador más afin

Cuadro 62: CU-07: Curso Normal

CU-07: Curso Alternativo
2a. Si no hay conexión a internet se informa al usuario de que se necesita conexión a una red para mostrar los datos.

Cuadro 63: CU-07: Curso Alternativo

Ver estadio

CU-08	**Ver estadio**
Actor	Usuario
Tipo	Secundario
Precondición	Haber seleccionado un equipo de la liga
Postcondición	Mostrar datos relativos al estadio
Propósito	Informar de las carcaterísticas del estadio
Resumen	El suaurio quiere informarse sobre el estadio de un equipo

Cuadro 64: CU-08: Ver estadio

CU-08	**Curso Normal**
Acción del actor	**Acción del sistema**.
1. El usuario selecciona la opción de ver estadio	2. El sistema a partir del equipo seleccionado previamente realiza la consulta a la API
4. El usuario visualiza los datos relativos al estadio	3. El sistema carga los datos

Cuadro 65: CU-08: Curso Normal

CU-08: Curso Alternativo
2a. Si no hay conexión a internet se informa al usuario de que se necesita conexión a una red para mostrar los datos.

Cuadro 66: CU-08: Curso Alternativo

Geolocalizar estadio

CU-09	**Geocalizar estadio**
Actor	Usuario
Tipo	Secundario
Precondición	Haber seleccionado un estadio previamente
Postcondición	Mostrar mapa ubicando el estadio
Propósito	Geolocalizar un estadio de un equipo
Resumen	El usuario quiere saber dónde se sitúa el estadio de un equipo

Cuadro 67: CU-09: Geolocalizar estadio

CU-09	**Curso Normal**
Acción del actor	**Acción del sistema**.
1. El usuario solicita localizar el estadio	2. El sistema obtiene las coordenadas mediante una consulta a la API
4. El usuario visualiza el mapa con un marcador en el estadio	3. El sistema con las coordenadas geolocaliza el estadio y carga el mapa

Cuadro 68: CU-09: Curso Normal

CU-09: Curso Alternativo
2a. Si no hay conexión a internet se informa al usuario de que se necesita conexión a una red para mostrar los datos.

Cuadro 69: CU-09: Curso Alternativo

Ver estadísticas equipo

CU-10	**Ver estadísticas equipo**
Actor	Usuario
Tipo	Secundario
Precondición	Haber seleccionado previamente un equipo
Postcondición	Mostrar las estadísticas del equipo
Propósito	Recopilar y calcular variables estadísticas del equipo
Resumen	El usuario quiere saber las estadísticas de un equipo en una temporada

Cuadro 70: CU-10: Ver estadísticas equipo

CU-10	**Curso Normal**
Acción del actor	**Acción del sistema**.
1. El usuario elige ver las estadísticas de un equipo para la temporada actual	2. El sistema consulta a la API las estadísticas del equipo
4. El usuario visualiza las estadísticas del equipo de la actual temporada	3. El sistema calcula las estadísticas necesarias.

Cuadro 71: CU-10: Curso Normal

CU-02: Curso Alternativo
2a. Si no hay conexión a internet se informa al usuario de que se necesita conexión a una red para mostrar los datos.
2b. Si no hay datos para la temporada actual se muestra un mensaje de error la usuario

Cuadro 72: CU-10: Curso Alternativo

Ver estadísticas jugador

CU-11	**Ver estadísticas jugador**
Actor	Usuario
Tipo	Secundario
Precondición	Haber seleccionado un jugador previamente
Postcondición	Mostrar estadísticas del jugador seleccionado
Propósito	Calcular estadísticas de un jugador para la temporada actual
Resumen	El usuario quiere visualizar las estadísticas de un jugador para la temporada actual

Cuadro 73: CU-11: Ver estadísticas jugador

CU-11	**Curso Normal**
Acción del actor	**Acción del sistema**.
1 El usuario elige visualizar las estadísticas de un jugador	2. El sistema consulta a la API los datos
4. EL usuario visualiza los datos	3. El sistema carga los datos

Cuadro 74: CU-11: Curso Normal

CU-11: Curso Alternativo
2a. Si no hay conexión a internet se informa al usuario de que se necesita conexión a una red para mostrar los datos.
2b. Si no hay estadísticas para la temporada actual se informa al usuario del error.

Cuadro 75: CU-11: Curso Alternativo

Ver partidos jugador

CU-12	**Ver partidos jugador**
Actor	Usuario
Tipo	Secundario
Precondición	Haber seleccionado un jugador previamente
Postcondición	Mostrar partidos jugados de un jugador
Propósito	Para la temporada actual mostrar partidos jugados por el jugador seleccionado
Resumen	El usuario quiere ver los partidos disputados por un jugador en la actual temporada

Cuadro 76: CU-12: Ver partidos jugador

CU-12	**Curso Normal**
Acción del actor	**Acción del sistema**.
1. El usuario selecciona la opción ver partidos	2. El sistema consulta a la API la lista de partidos para la actual temporada
4. El usuario visualiza los partidos jugados por el jugador	3. El sistema ordena los partidos cronológicamente

Cuadro 77: CU-12: Curso Normal

CU-12: Curso Alternativo
2a. Si no hay conexión a internet se informa al usuario de que se necesita conexión a una red para mostrar los datos.
2b. Si no hay partidos disputados para la temporada actual se informa al usuario del error.

Cuadro 78: CU-12: Curso Alternativo

Filtrar estadísticas equipo

CU-13	**Filtrar estadísticas equipo**
Actor	Usuario
Tipo	Secundario
Precondición	Haber elegido previamente visualizar estadísticas de un equipo
Postcondición	Filtrar estadísticas equipo por temporada
Propósito	Seleccionar una temporada para ver las estadísticas del equipo
Resumen	El usuario quiere elegir la temporada en la que ver las estadísticas de su equipo

Cuadro 79: CU-13: Filtrar estadísticas equipo

CU-13	**Curso Normal**
Acción del actor	**Acción del sistema**.
1. El usuario selecciona la temporada	3. El sistema realiza la consulta a la API a partir del equipo y año solicitados
2. El usuario solicita la información	4. El sistema carga los datos
5. El usuario visualiza las estadísticas	-

Cuadro 80: CU-13: Curso Normal

CU-13: Curso Alternativo
2a. Si no hay conexión a internet se informa al usuario de que se necesita conexión a una red para mostrar los datos.
2b. Si no hay estadísticas para la temporada seleccionada se informa al usuario del error.

Cuadro 81: CU-13: Curso Alternativo

Filtrar estadísticas jugador

CU-14	**Filtrar estadísticas jugador**
Actor	Usuario
Tipo	Secundario
Precondición	Haber elegido previamente visualizar estadísticas de un jugador
Postcondición	Filtrar estadísticas de un jugador por temporada
Propósito	Seleccionar una temporada para ver las estadísticas del jugador elegido
Resumen	El usuario quiere elegir la temporada en la que ver las estadísticas de su jugador seleccionado

Cuadro 82: CU-14: Filtrar estadísticas jugador

CU-14	**Curso Normal**
Acción del actor	**Acción del sistema**.
1. El usuario selecciona la temporada	3. El sistema realiza la consulta a la API a partir del jugador y año solicitados
2. El usuario solicita la información	4. El sistema carga los datos
5. El usuario visualiza las estadísticas	-

Cuadro 83: CU-14: Curso Normal

CU-14: Curso Alternativo
2a. Si no hay conexión a internet se informa al usuario de que se necesita conexión a una red para mostrar los datos.
2b. Si no hay estadísticas para la temporada seleccionada se informa al usuario del error.

Cuadro 84: CU-14: Curso Alternativo

Exportar datos

CU-15	**Exportar datos**
Actor	Usuario
Tipo	Secundario
Precondición	Haber visualizado datos previamente
Postcondición	Generar archivo CSV con los datos visualizados previamente por el usuario en la ubicación deseada
Propósito	Exportar datos estadísticos a una ubicación
Resumen	El usuario desea guardar los datos que esta visualizando en formato CSV

Cuadro 85: CU-15: Exportar datos

CU-15	**Curso Normal**
Acción del actor	**Acción del sistema**.
1. El usuario solicita exportar los datos que está visualizando	2. El sistema recopila los datos y genera un archivo en formato CSV
4. El usuario elige la ubicación	3. El sistema pregunta al usuario la ubicación donde quiere guardar el archivo
-	5. El sistema guarda el archivo en la ubicación elegida e informa al usuario

Cuadro 86: CU-15: Curso Normal

CU-15: Curso Alternativo
2a. Si no se consigue generar el archivo CSV se informa al usuario del error al usuario y el caso de uso queda sin efecto.
5a. Si no hay espacio suficiente o no se puede acceder a la ubicación especificada se informa del error al usuario y el caso de uso queda sin efecto

Cuadro 87: CU-15: Curso Alternativo

11.5. Modelo de dominio

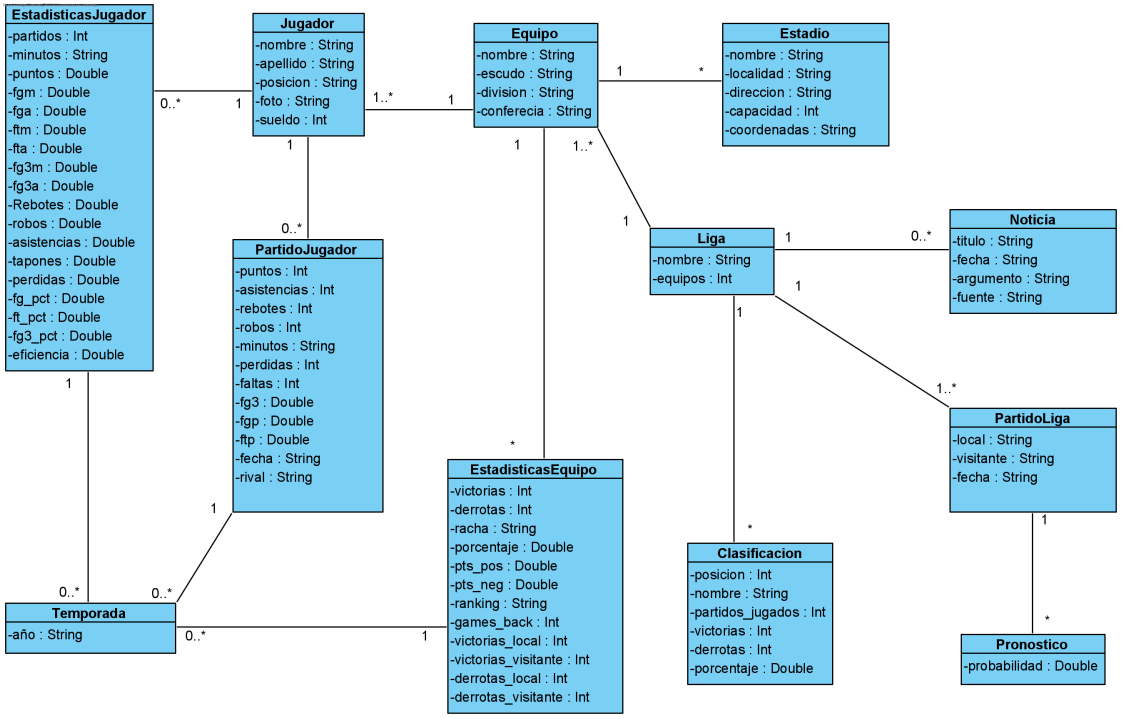

Figura 25: Modelo de Dominio.

11.6. Descripción clases del modelo de dominio

Para terminar, vamos a describir las clases que componen el modelo de dominio anteriormente mostrado:

Liga

Se trata de una clase que representa la liga. En nuestro caso representa a la NBA. A continuación, podemos ver sus atributos:

- *nombre:* Es el nombre de la competición
- *equipos:* Se trata del número de equipos que pertenecen a la liga

Equipo

Se trata de una clase que representa a cada uno de los equipos de la liga. A continuación, podemos ver sus atributos:

- *nombre:* Es el nombre del equipo
- *escudo:* Es el logo del equipo
- *división:* División a la que pertenece el equipo
- *conferencia:* Conferencia a la que pertenece el equipo

Estadio

Se trata de una clase que representa al estadio de cada uno de los equipos. A continuación, podemos ver sus atributos:

- *nombre:* Es el nombre del estadio
- *dirección:* Es la dirección completa dónde se encuentra el estadio
- *localidad:* La localidad a la que pertenece el estadio
- *capacidad:* Es el número de espectadores máximos que pueden entrar al estadio
- *coordenadas:* Conjunto de coordenadas que nos permiten geolocalizar al estadio

Noticia

Se trata de una clase que representa las noticias asociadas a cada uno de los equipos. A continuación, podemos ver sus atributos:

- *titulo:* Titular de la noticia

- *fecha:* Fecha de publicación de la noticia
- *argumento:* Contenido de la noticia
- *fuente:* Es el sitio de donde se ha obtenido la noticia

Clasificación

Se trata de una clase que representa la clasificación de la liga. A continuación, podemos ver sus atributos:

- *posición:* Es la posición que ocupa cada equipo en el ranking
- *nombre:* Es el nombre de cada equipo
- *partidos_jugados:* Es el número de partidos jugados por un equipo
- *victorias:* Es el número de victorias de cada equipo
- *derrotas:* Es el número de derrotas de cada equipo
- *porcentaje:* Es el porcentaje de victorias de cada equipo

PartidoLiga

Se trata de una clase que representa los partidos entre dos equipos en la liga. A continuación, podemos ver sus atributos:

- *local:* Es el equipo que disputa como local el encuentro
- *visitante:* Es el equipo que disputa como visitante el partido
- *fecha:* Es la fecha en la que se va a disputar el partido

Pronostico

Se trata de una clase que representa el pronóstico para los partidos de la liga. A continuación, podemos ver su único atributo:

- *probabilidad:* Se trata de la probabilidad de victoria del equipo elegido para el próximo encuentro

EstadisticasEquipo

Se trata de una clase que representa las estadísticas de cada equipo en la liga. A continuación, podemos ver su único atributo:

- *victorias:* Número de victorias del equipo
- *derrotas:* Número de derrotas del equipo
- *racha:* Racha actual del equipo
- *porcentaje:* Porcentaje de victoria
- *puntos_pos:* Promedio de puntos a favor anotados por partido
- *puntos_neg:* Promedio de puntos en contra anotados por partido
- *ranking:* Posición que se ocupa en la clasificación
- *games_back:* Número de partidos restantes hasta el liderato
- *victorias_local:* Número de victorias jugando como local
- *victorias_visitante:* Número de victorias jugando como visitante
- *derrotas_local:* Número de derrotas jugando como local
- *derrotas_visitante:* Número de derrotas jugando como visitante

Jugador

Se trata de una clase que representa a cada jugador de los equipos de la liga. A continuación, podemos ver sus atributos:

- *nombre:* Es el nombre completo del jugador
- *apellido:* Es el apellido del jugador
- *posicion:* Abreviatura de la posición en la que juega
- *foto:* Foto de perfil del jugador
- *sueldo:* Sueldo total(todos los años) firmado por el jugador

PartidoJugador

Se trata de una clase que representa los partidos disputados por cada jugador en la liga. A continuación, podemos ver sus atributos:

- *puntos:* Puntos anotados en el partido por el jugador
- *asistencias:* Asistencias realizadas en el partido por el jugador
- *rebotes:* Rebotes capturados por el jugador en el partido

- *robos:* Robos realizados en el partido por el jugador
- *minutos:* Minutos jugados por el jugador en el partido
- *perdidas:* Pérdidas realizadas por el jugador en el partido
- *faltas:* Faltas cometidas por el jugador en el partido
- *fg3:* Porcentaje de tiro de 3 del jugador en el partido
- *fgp:* Porcentaje de tiros de campo del jugador en el partido
- *ftp:* Porcentaje de tiros libres del jugador en el partido
- *fecha:* Fecha del partido
- *rival:* Rival contra el que se enfrentó el jugador en el partido

EstadísticasJugador

Se trata de una clase que representa las estadísticas de un jugador en la liga. A continuación, podemos ver sus atributos:

- *partidos:* Número de partidos disputados por el jugador
- *minutos:* Promedio del número de minutos que disputa el jugador
- *puntos:* Promedio del número de puntos que anota el jugador
- *fgm:* Número de tiros de campo anotados por el jugador
- *fga:* Número de tiros de campo intentados por el jugador
- *ftm:* Número de tiros libres anotados por el jugador
- *fta:* Número de tiros libres intentados por el jugador
- *fg3m:* Número de tiros de tres anotados por el jugador
- *fg3a:* Número de tiros de tres intentados por el jugador
- *rebotes:* Promedio del número de rebotes capturados por el jugador
- *robos:* Promedio del número de robos realizados por el jugador
- *asistencias:* Promedio del número de asistencias realizadas por el jugador
- *tapones:* Promedio del número de tapones realizados por el jugador
- *perdidas:* Promedio del número de pérdidas efectuadas por el jugador
- *fg_pct:* Porcentaje de tiros de campo del jugador

- *ft_pct:* Porcentaje de tiros libres del jugador
- *fg3_pct:* Porcentaje de tiros de tres del jugador
- *eficiencia:* Eficiencia del jugador en la temporada

Temporada

Se trata de una clase que representa año de la temporada de la liga. A continuación, podemos ver su único atributo:

- *año:* Es el año de la temporada

12. Manual de Usuario

12.1. Introducción

En esta sección se expondrá al usuario una guía para entender el funcionamiento de la aplicación desarrollada. Se proporcionará ilustraciones de todas las pantallas de la aplicación con el fin de facilitar el aprendizaje de la app.

Las imágenes proporcionadas han sido seleccionadas del dispositivo Xiaomi mi 10 Lite, por lo que su aspecto pueden diferir en otros dispositivos con pantallas de tamaño variable. Independientemente de esto, el funcionamiento de la aplicación será el mismo sea cuál sea el tamaño de la pantalla.

12.2. Icono de la aplicación, pantalla de carga y menú principal

El icono de la aplicación se ha diseñado con el software Photoshop. Como en todas las ventanas, se ha utilizado un diseño muy minimalista. La apariencia del icono es la siguiente:

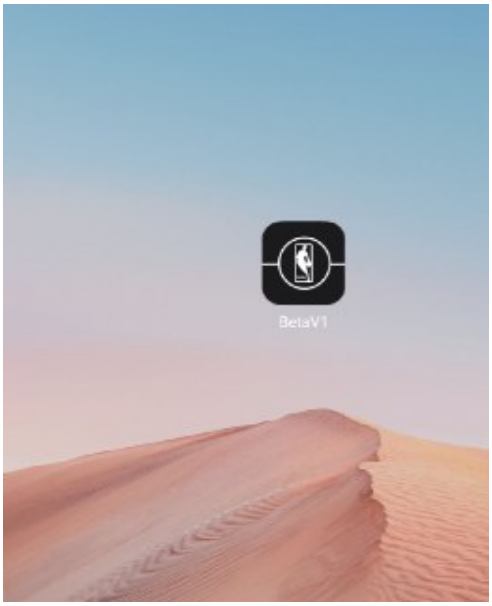

Figura 26: Icono de la aplicación.

Una vez que pulsamos el icono entramos de lleno en la aplicación. Lo primero que aparece tras pulsar el icono es una pantalla de carga a modo de presentación, con una duración variable de segundos. La pantalla de carga tiene la siguiente apariencia:

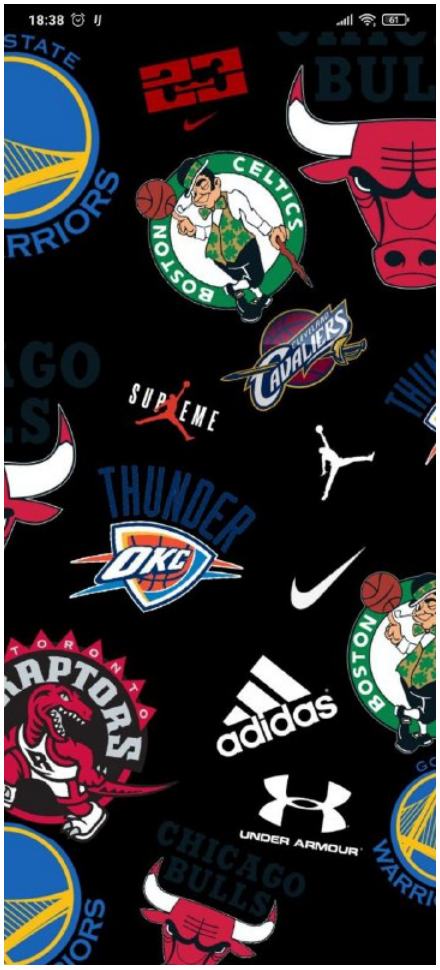

Figura 27: Pantalla de carga.

Esta pantalla de carga sirve a modo de presentación, pero también para realizar cálculos que agilicen la carga de ventanas. Una vez que se realizan los cálculos, se lleva al usuario al menú principal. Dicho menú contiene cuatro opciones: Equipos, noticias, clasificación y pronósticos:

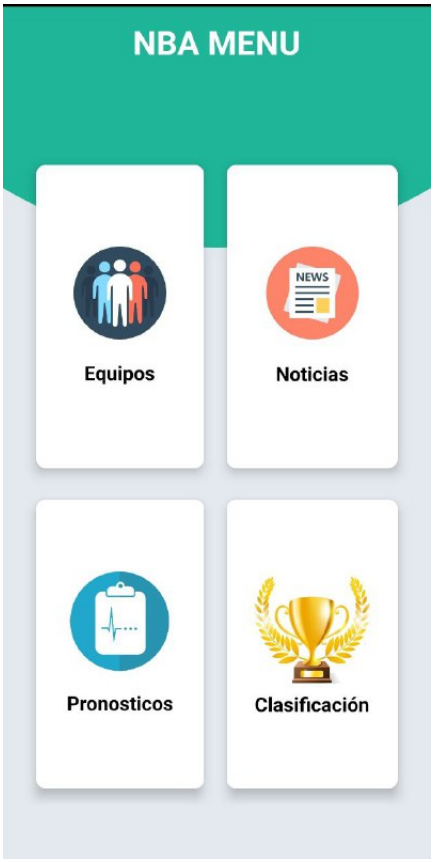

Figura 28: Menú principal.

En este punto, el usuario tiene las cuatro opciones mostradas en la ilustración anterior. Los nombres de las opciones mostradas son totalmente descriptivos. Vamos a ir describiendo poco a poco cada una de las cuatro ventanas disponibles del menú principal.

Si pulsamos el botón noticias, nos abrirá una ventana con los títulos y fechas de las útlimas noticias de la liga. Si volvemos a pulsar en una de las noticias, se expanderá dicha noticia mostrando los detalles de la misma. Podremos ver la fuente de dónde sale el artículo y el argumento de la noticia con más detalles. Podemos verlo en las siguientes dos imágenes:

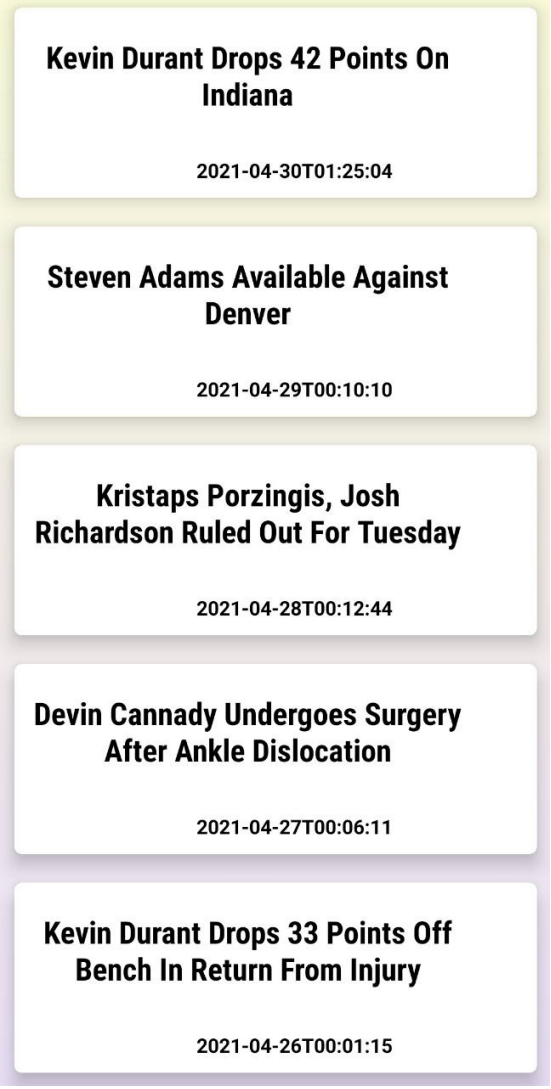

Figura 29: Lista de noticias

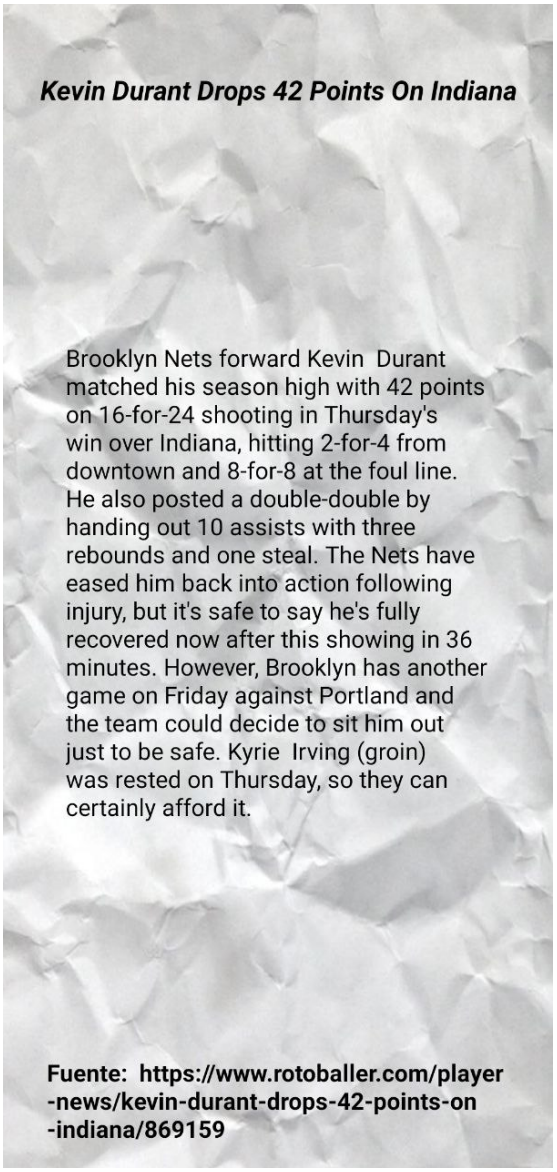

Figura 30: Detalles de una noticia

Otra de las opciones que podemos consultar es la clasificación de la liga. Si pulsamos el icono se nos abrirá una ventana con la clasificación actualizada. En ella podemos ver la posición de cada equipo, su escudo, nombre, partidos jugados, partidos ganados/perdidos y el porcentaje de partidos ganados.

Pos	Nombre	PJ	V	D	%V
1.	Utah Jazz	62	45	17	0.726
2.	Phoenix Suns	62	44	18	0.71
3.	Brooklyn Nets	63	43	20	0.683
4.	Los Angeles Clippers	64	43	21	0.672
5.	Denver Nuggets	63	42	21	0.667
6.	Philadelphia 76ers	62	41	21	0.661
7.	Milwaukee Bucks	62	38	24	0.613
8.	Los Angeles Lakers	62	36	26	0.581
9.	Dallas Mavericks	62	35	27	0.565
10.	New York Knicks	63	35	28	0.556
11.	Portland Trail Blazers	62	34	28	0.548

Figura 31: Clasificación de la liga.

La siguiente ventana que vamos a explicar es la ventana pronósticos. Aquí, deberemos elegir en el desplegable el equipo del que queremos visualizar el próximo partido. Una vez elegido nuestro equipo pulsamos el botón “procesar” y veremos la fecha y el rival al que se enfrenta. También podemos ver la posición en la clasificación que ocupan cada uno de los dos equipos.

Para terminar con esta ventana, como su propio nombre indica, podremos ver un pronóstico para el partido. Si hemos seguido correctamente los pasos veremos habilitado un nuevo botón "ver probabilidad de victoria". Si lo pulsamos nos mostrará en porcentaje cien la probabilidad que tiene cada equipo de ganar el encuentro en base a estadísticas pasadas.

Figura 32: Ventana Pronósticos.

La última opción del menú principal, es la ventana "equipos". Esta es la pestaña principal de la aplicación que nos permite interactuar con los principales equipos de la liga. Si pulsamos en su icono iremos directamenta al listado de equipos, mostrándonos la división y conferencia a la que pertenecen.

Figura 33: Listado de equipos de la liga.

Si pulsamos en uno de los equipos seremos reedirecionados a la primera pestaña del menu inferior negro, que es la que hemos denominado como "Plantilla". En esta ventana, veremos el listado de jugadores actuales que están jugando en el equipo seleccionado. Además, podremos saber su nombre completo, posición en la que juegan y salario.

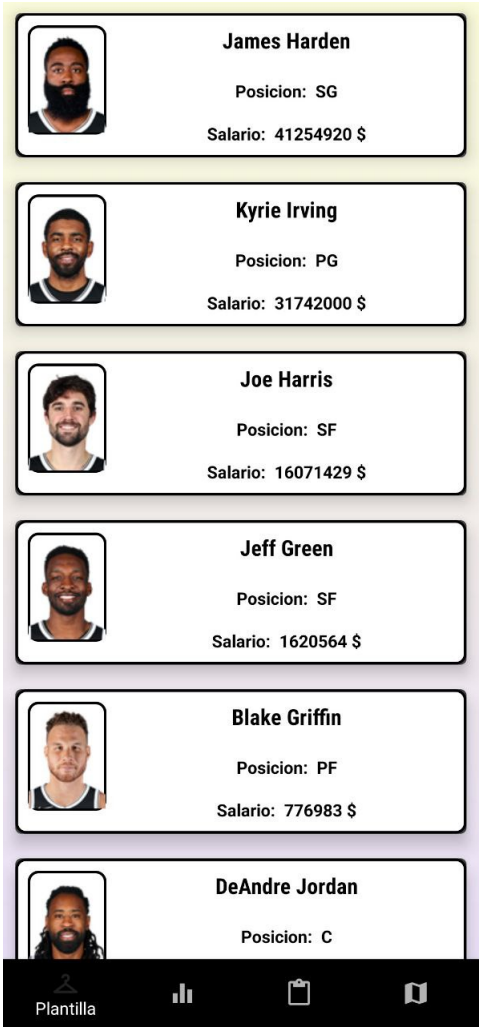

Figura 34: Listado de jugadores de un equipo.

Si pulsamos en cualquier jugador podremos visualizar todas las estadísticas promedio que lleva ese jugador en la actual temporada. En dicha ventana podremos elegir en el desplegable la temporada de la que queremos visualizar las estadísticas. Si elegimos una temporada de la que el servidor tiene datos, veremos habilitado un botón que nos permite ver que ha hecho el jugador en cada uno de los partidos .

Figura 35: Ventana estadísticas jugador.

Figura 36: Ventana de partidos jugados.

Si por otro lado elegimos una temporada de la que el servidor no tiene datos, aparecerá el botón "partidos" bloqueado y la aplicación no nos permitirá pulsar al botón para visualizar los partidos hasta que no eligiesemos una temporada con datos.

Figura 37: Error, sin datos para la temporada seleccionada.

Para acabar con la primera pestaña del submenú inferior (Pestaña Plantilla), en la ventana de partidos que hemos comentado anteriormente, tenemos la opción de exportar todos los partidos a una ubicación del dispositivo en formato CSV.

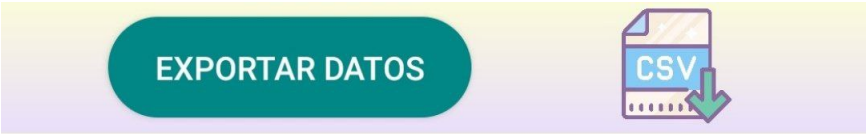

Figura 38: Botón para exportar datos CSV.

La segunda pestaña del submenú inferior es la ventana "Estadísticas". Esta pestaña nos muestra las estadísticas totales del equipo elegido para una temporada. En el desplegable podremos elegir la temporada de la que queramos visualizar los datos. Podremos consultar las victorias, derrotas, racha, porcentaje de victoria, puntos a favor promedio, puntos en contra promedio, las victorias de local/visitante, las derrotas local/visitante, posición en la clasificación y las victorias necesarias para alcanzar el liderato. Por último, debajo de las estadísticas tenemos un botón para exportar los datos consultados.

Figura 39: Ventana Estadísticas.

La tercera pestaña, es la ventana "Pizarra". Esta orientada a entrenadores, y como hemos explicado sirve para encontrar el mejor sustituto en un equipo a partir de una posición elegida previamente. El usuario deberá elegir una de las 5 posiciones predefinidas en los partidos de baloncesto en el desplegable. Una vez que se elige la posición si pulsa el botón "procesar" aparecerá un mensaje con el jugador elegido y el ratio utilizado para su elección.

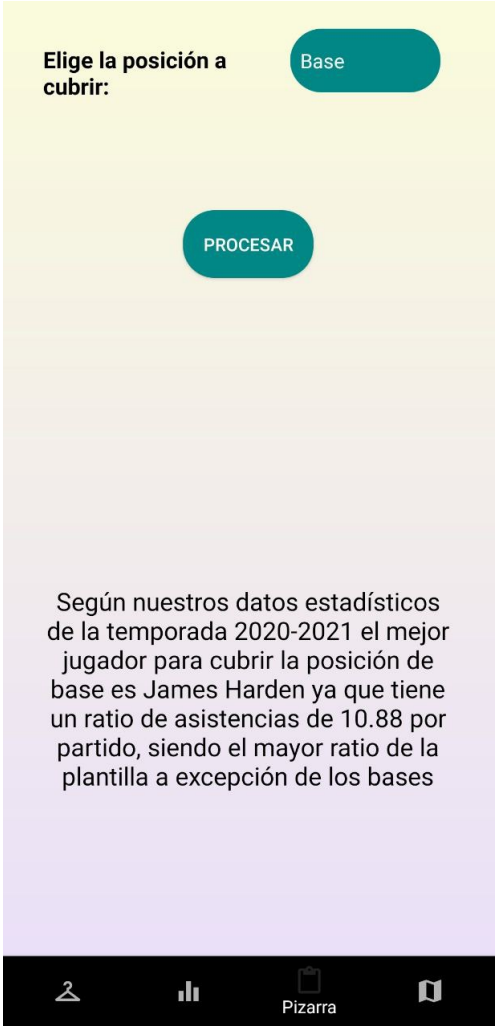

Figura 40: Ventana Pizarra

La última pestaña que tenemos que comentar es la ventana "Estadio". Al seleccionar esta ventana veremos los datos relativos al pabellón del equipo. Entre estos datos destacan el nombre completo del estadio, la dirreción dónde se encuentra, la localidad y la capacidad. Debajo de estos datos la aplicación nos dirá si este estadio tiene una capacidad por encima o por debajo que la media de los estadios de la liga.

Figura 41: Ventana Estadio

Para terminar, si pulsamos el botón "Ver mapa" se geolocalizará el estadio. En el mapa veremos el marcador con el nombre del estadio.

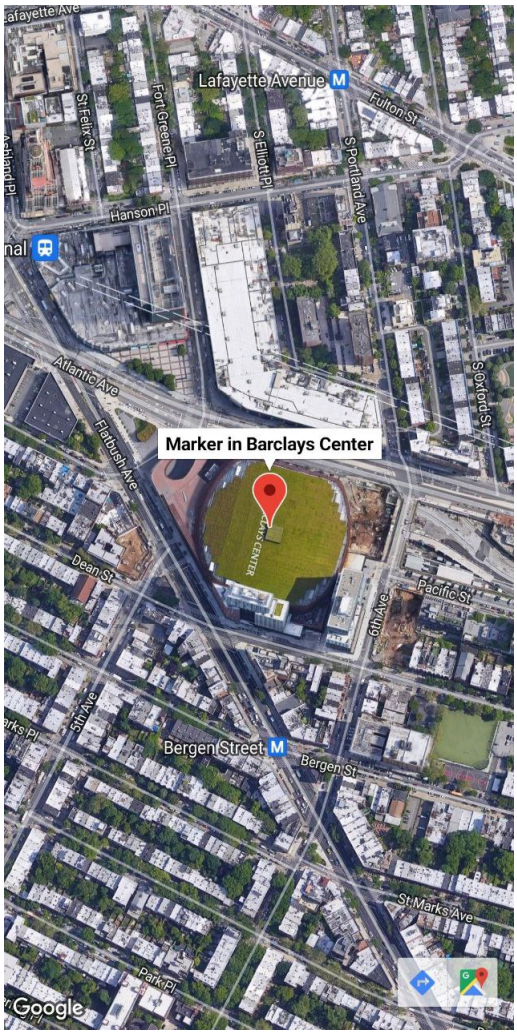

Figura 42: Geolocalización de un estadio.

13. Conclusiones y Posibles Mejoras

En esta sección se analizarán los puntos positivos obtenidos en el proyecto y los aspectos que podrían ser mejorados. Entre estos últimos, se propondrán un listado de mejoras que puede resultar muy beneficiosas para el proyecto, y que no se han podido implementar por cuestiones temporales.

13.1. Conclusiones Del Proyecto

Tras varios meses de trabajo y esfuerzo se ha conseguido finalizar el proyecto. A partir de este momento, me gustaría comentar varios aspectos que quiero destacar de este trabajo.

Para empezar, estoy satisfecho de haber elegido esta temática, trabajar durante meses en lo que te gusta es una gran satisfación y ayuda a realizar mejor las tareas. En mi caso, programar en lenguaje Android, es una de las cosas que más me ha gustado en la carrera universitaria y no me arrepiento de haber escogido esta temática para mi proyecto fin de carrera. He profundizado en el lenguaje Android y he comprobado lo que es estar durante mucho tiempo investigando y aprendiendo nuevas técnicas en este mundillo. Claramente, me va a servir para decidir si tras finalizar mis estudios quiero orientar mi carrera laboral al mundo Android o no. Esta es una de mis primeras conclusiones tras la finalización de mi trabajo fin de grado, la experiencia vivida al desarrollar una aplicación móvil en Android.

Otro aspecto que he aprendido, es el hecho de trabajar con personas que no posean conocimientos informáticos. Al final uno de los objetivos de este trabajo, era cumplir los requisitos que nos proponía el cliente. En muchas ocasiones, las peticiones que quería el usuario para la aplicación, eran complicadas de implementar, pero no lo eran tanto a ojos del usuario. Esto me hizo cambiar mi forma de pensar y de explicar las cosas, en vistas al cliente. Al final la compenetración fue muy buena. Los conocimientos estadísticos que yo no poseía y que fueron aportados por mi segundo tutor sirvieron para hacer un modelo predictivo de resultados en los partidos, que otorga un extra a la aplicación muy interesante.

Para la obtención de los datos necesarios para la construcción de un modelo predictivo de partidos utilicé un par de scripts escritos en lenguaje java. Mi tutora me pidió un archivo en formato CSV que recogiera todos los partidos de los jugadores entre las temporadas 1995 y 2020. La obtención de esta cantidad tan grande de datos se debía de automatizar. Es una tarea que nunca había hecho anteriormente y fue novedosa para mi. Al principio fue algo complicado, por el hecho de tener que entender el proceso a seguir, pero en base a ir haciendo pequeñas pruebas y aumentando poco a poco la complejidad conseguí el resultado esperado. De esta forma, logré que el script recogiera automáticamente los datos del servidor y creará el excel definitivo. Este archivo contendría mas de 230.000 filas y tardó en crearlo aproximadamente unas 7 horas. Es un trabajo que no hubiera sido viable hacerlo de manera manual, al ser una cantidad tan elevada de datos. Este trabajo de automatización me pareció muy interesante y posteriormente comprobé que se lleva a cabo en muchas empresas informáticas, por lo que me resultó muy enriquecedor.

Para finalizar esta sección, como conclusión final, creo que he completado correctamente mi primer proyecto software completo. Nunca antes había desarrollado un proyecto tan grande y me enorgullece haberlo logrado. A pesar de haber conseguido el objetivo de desarrollar una aplicación con los requisitos iniciales propuestos, creo que lo más importante son los conocimientos nuevos adquiridos. Al final, todo se resume en eso, aprender cada vez más cosas y disfrutar en el proceso.

13.2. Conclusiones Predicciones NBA

Tras un analisis estadístico de los últimos 20 años de la competición, podemos sacar una serie de conclusiones:

- La liga Nba es una competición que en su fase regular no es muy complicada de predecir, conociendo un poco el historial y tendencias de los equipos se puede obtener unos pronósticos bastante fiables. Como hemos comentado anteriormente, los PlayOff son la fase de la liga más complicada de predecir debido a la igualdad de los equipos que se enfrentan, y a la no tan ilustrativa variable racha que en esta ocasión nos proporciona menos información . Por otro lado, como ya sabemos el deporte no son matemáticas, el último clasificado puede ganar al primero, esto no es muy habitual en la NBA, pero en muchas ocasiones, los equipos son capaces de desafiar a la ciencia poniéndola en duda con sus resultados. En el caso de la NBA, las predicciones no son del todo fáciles, entran en juego muchos factores, se juegan demasiados partidos y los equipos dosifican dependiendo de su situación en la clasificación y de sus objetivos en la temporada. El hecho de ser un deporte en el que juegan 5 vs 5 hace que las estrellas en este deporte cobren aún más importancia, siendo totalmente decisivos y cambiando drásticamente los resultados de los partidos dependiendo de si juegan o no los encuentros.
- Los modelos predictivos funcionan realmente bien, consiguiendo un elevado porcentaje de acierto. Sufren más al predecir partidos tempranos de la liga, y tienen limitaciones, como por ejemplo, que no tienen en cuenta los lesionados para cada partido o los objetivos en juego del equipo, que son dos factores muy importantes a la hora de analizar encuentros. El modelo predictivo del alumno tiene mayor tasa de acierto, debido a que la liga regular de la Nba tiene una tendencia clara a dar más opciones de victoria a los equipos mejor clasificados. Esto parece lógico, pero no lo es tanto en otros deportes, por ejemplo en fútbol, la liga Inglesa es una liga muy irregular en la que no es extraño que los equipos de las últimas posiciones de la tabla ganen a los primeros clasificados. La razón es la igualdad de la mayoría de los equipos de la liga, cosa que no ocurre en la NBA. Aqui es mucho más extraño que los últimos clasificados ganen a los primeros, ya que los equipos están mas descompensados que en otros deportes, como en el ejemplo que hemos puesto anteriormente de la liga inglesa de fútbol.

13.3. Propuestas de mejora

En esta sección, vamos a proponer una serie de mejoras que en base al transcurso del desarrollo de la aplicación se ha observado que pueden ser beneficioasas para el resultado final.

Añadir más ligas

Ya explicamos en otras secciones la dificultad de esto debido a que es un proyecto a coste cero y que no había APIs que nos proporcionasen datos de otras ligas de manera gratuita. Aumentando la complejidad, con la técnica Web Scrapping, es posible obtener datos de otras ligas. Creo que es una mejora que haría a la app mucho más completa, por lo que la primera propuesta va para este mejora.

Añadir nuevos modelos predictivos

Es una mejora muy interesante, porque diferentes modelos predictivos nos van a proporcionar diferentes tipos de prediciones. Sería de gran utilidad tener varios tipos de predicciones para los partidos y realizar una comparación para ver cuál es el modelo que se ajusta mejor a la realidad.

Incluir notificaciones

Si se incluyerá un sistema de notificaciones que nos avisará cuando hay nuevos datos en la aplicación, esta, adquiriría un toque profesional. Además, sería de gran utilidad para el usuario y creo que incitaría a utilizar aún mas la app.

Sistema de favoritos

Como ocurre en otras aplicaciones deportivas, una función muy útil, sería la de poder escoger equipos como favoritos. De esta manera, el usuario nada mas abrir la aplicación podría ver directamente los datos de su equipo u equipos favoritos.

Incluir más deportes

Es una mejora algo compleja, porque supone un enorme trabajo añadir más deportes. Eso sí, provocaría un gran incremento de usuarios a la hora de utilizar la aplicación y el resultado final sería el de una app lo más completa posible.

Incluir otros idiomas

La versión actual de la aplicación sólo incluye el idioma castellano. Incluir más lenguajes sería una gran mejora y nos permitiría abarcar un mayor número de usuarios.

A. Anexo I: Soporte Digital

El contenido del archivo DatosTFGDanielParienteBermejo.zip contiene todos los archivos esenciales del proyecto. Entre estos archivos destacamos los siguientes:

- *Trainning4Players.zip:* Es un archivo comprimido en cuyo interior podemos encontrar el proyecto de ANdroid Studio. Se incluyen todas las clases que hacen funcionar a la aplicación.
- *Jugadores.java:* Es el script encargado de generar el archivo excel que alberga todos los partidos de los jugadores entre las temporadas 1995-2020.
- *Equipos.java :* Es el script encargado de generar el archivo excel que alberga todos los partidos de los equipos entre las temporadas 1995-2020.
- *jugadores_partidos.xlsx:* Archivo excel generado por el script Jugadores.java
- *equipos_partidos.xlsx:* Archivo excel generado por el script Equipos.java
- *modelos.xlsx:* Archivo excel que contiene el modelo predictivo estadístico.
- *Trainning4PLayers.apk :* Es el ejecutable de la aplicación Android, directamente preparado para ser instalado en cualquier dispositivo Android.

Referencias

[1] *Uso del ancho de banda en el mundo*
MYCOMPUTER.COM
Recuperado a 06/03/2021,
de https://www.muycomputer.com/2014/06/27/ancho-banda-mundo/

[2] *Una década con smartphones*
LAVANGUARDIA.COM
Recuperado a 06/03/2021,
de https://www.lavanguardia.com/tecnologia/20170226/42274940927/diez-anos-smartphones-cambiado-vida.html

[3] *Adicción a los smartphones*
PSICOADAPTA.ES
Recuperado a 06/03/2021,
https://www.psicoadapta.es/blog/adiccion-al-movil-nomofobia/

[4] *Usuarios con smartphones*
STATISTA.COM
Recuperado a 06/03/2021,
https://es.statista.com/estadisticas/636569/usuarios-de-telefonos-inteligentes-a-nivel-mundial/

[5] *Técnica Web Scrapping*
ANTEVENIO.COM
Recuperado a 06/03/2021,
https://www.antevenio.com/blog/2019/03/que-es-el-web-scraping-y-para-que-sirve/

[6] *Origen de Android*
HISTINF.BLOGS.UPV.ES
Recuperado a 06/03/2021,
https://histinf.blogs.upv.es/2012/12/14/android/

[7] *Introducción a Android Studio*
DEVELOPER.ANDROID.COM
Recuperado a 11/03/2021,
https://developer.android.com/studio/intro?hl=es-419

[8] *Características de Android Studio*
ES.WIKIPEDIA.ORG
Recuperado a 11/03/2021,
https://es.wikipedia.org/wiki/Android_Studio

[9] *¿Que es Retrofit?*
SQUARE.GITHUB.IO
Recuperado a 12/03/2021,
https://square.github.io/retrofit/

[10] *Biblioteca Glide*
GITHUB.COM
Recuperado a 12/03/2021,
https://github.com/bumptech/glide

[11] *Metodología Kanban*
VIEWNEXT.COM
Recuperado a 22/03/2021,
https://www.viewnext.com/kanban-desarrollo-software/

[12] *Metodologías Ágiles*
DIGITAL55.COM
Recuperado a 22/03/2021,
https://www.digital55.com/desarrollo-tecnologia/mejores-metodologias-agiles-creacion-software/

[13] *Roles en Kanban*
IPMOGUIDE.COM
Recuperado a 23/03/2021,
https://ipmoguide.com/kanban-roles/

[14] *Reuniones típicas de Kanban*
ITNOVE.COM
Recuperado a 23/03/2021,
https://itnove.com/blog/kanban/equipos/las-cadencias-de-kanban/

[15] *Tipos JSON*
JSON.ORG
Recuperado a 26/03/2021,
https://www.json.org/json-es.html

[16] *Objetos LiveData*
DEVELOPER.ANDROID.COM
Recuperado a 29/03/2021,
https://developer.android.com/topic/libraries/architecture/livedata?hl=es-419java

[17] *Diseños de Apliciones*
ACTUALIDADIPHONE.COM
Recuperado a 02/04/2021,
https://www.actualidadiphone.com/la-importancia-del-buen-diseno-de-una-aplicacion/

[18] *Documentación Google Maps*
DEVELOPERS.GOOGLE.COM
Recuperado a 05/04/2021,
https://developers.google.com/maps/documentation/android-sdk/start?hl=es-419

[19] *Spinner Android*
DEVELOPERS.ANDROID.COM
Recuperado a 06/04/2021,
https://developer.android.com/guide/topics/ui/controls/spinner

[20] *La variable eficiencia*
ES.WIKIPEDIA.ORG
Recuperado a 07/04/2021,
https://es.wikipedia.org/wiki/Eficiencia_(baloncesto)

[21] *¿Qué es un Splash Screen?*
ELANDROIDELIBRE.ELESPANOL.COM
Recuperado a 07/04/2021,
https://elandroidelibre.elespanol.com/2015/10/splash-screens-que-son-y-para-que-sirven.html

[22] *Cómo crear un diseño basado en tarjetas*
DEVELOPER.ANDROID.COM
Recuperado a 08/04/2021,
https://developer.android.com/guide/topics/ui/layout/cardview?hl=es

[23] *Principales tipos de Riesgos*
SEMANTIC-SYSTEMS.COM
Recuperado a 12/04/2021,
https://www.semantic-systems.com/semantic-noticias/articulos-tecnologicos/gestion-de-riesgos-en-proyectos-de-implantacion-de-software/

[24] *Gestión de Riesgos*
OCW.UNICAN.ES
Recuperado a 12/04/2021,
https://ocw.unican.es/pluginfile.php/1408/course/section/1803/tema7-gestionRiesgos.pdf

[25] *Técnica Delphi*
EMPRENDEPYME.NET
Recuperado a 12/04/2021,
https://www.emprendepyme.net/que-es-el-metodo-delphi.html

[26] *Método DAFO*
BLOG.RIEUSSET.ES
Recuperado a 12/04/2021,
https://blog.rieusset.es/2017/10/09/identificacion-de-riesgos-y-oportunidades-en-rieusset/

[27] *Diagrama de Ishikawa*
BLOGDELACALIDAD.COM
Recuperado a 12/04/2021,
https://blogdelacalidad.com/como-utilizar-el-diagrama-de-ishikawa-para-identificar-riesgos/

[28] *Requisitos de un proyecto*
OBSBUSINESS.SCHOOL
Recuperado a 13/04/2021,
https://www.obsbusiness.school/blog/requisitos-de-un-proyecto-evita-las-dudas

[29] *Software Eclipse Kepler*
ES.WIKIPEDIA.ORG
Recuperado a 11/05/2021,
https://es.wikipedia.org/wiki/Eclipse_(software)

[30] *Sueldo medio Ingeniero Informático*
JOBTED.ES
Recuperado a 13/05/2021,
https://www.jobted.es/salario/ingeniero-inform %C3 %A1tico

[31] *Guía de usuarios de la librería Apache Commons*
COMMONS.APACHE.ORG
Recuperado a 31/05/2021,
https://commons.apache.org/proper/commons-csv/user-guide.html

[32] *Notación Asintótica*
ITNUEVOLAREDO.EDU.MX
Recuperado a 02/06/2021,
http://www.itnuevolaredo.edu.mx/takeyas/apuntes/Matematicas
%20para %20Computacion/Apuntes/Notacion %20O %20grande.pdf

[33] *Método Ensayo y error*
ES.WIKIPEDIA.ORG
Recuperado a 12/06/2021,
https://es.wikipedia.org/wiki/Ensayo_y_error