Trabajo de Fin de Grado

Planificador Torneos Ultimate
Ultimate Tournaments Planner

Romina Alejandra Martín Liberón

La Laguna, 7 de Junio de 2016

D. **José Vicente Blanco Pérez,** con N.I.F. 42.171.808-C profesor Titular de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutor

C E R T I F I C A

Que la presente memoria titulada:

"Planificador Torneos Ultimate."

ha sido realizada bajo su dirección por D. **Romina Alejandra Martín Liberón,** con N.I.F. 54.110.933-Y.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 7 de Junio de 2016.

Agradecimientos

A mi Vicente por su dedicación y paciencia.
A mis profesores por guiarme.
A mi familia por su aliento constante.
Y a mi pareja por su apoyo incondicional.

Licencia

Resumen

La finalidad del presente trabajo ha sido la creación de una aplicación Android que facilite y automatice tareas para el torneo canario de Ultimate Frisbee: Dr. Sand & Mr. Grass. Este torneo tiene una duración de tres días y atrae a gente de toda Europa. El Ultimate Frisbee es un deporte sin contacto y sin árbitros, esto quiere decir que son los propios jugadores los que dictaminan las decisiones, es por esto que es un deporte en el que lo más valorado es el juego limpio.

Este proyecto pretende realizar la gestión de todas las tareas que hasta ahora se llevan a cabo de forma manual durante esta clase de torneos fomentando de esta manera la creación de nuevos torneos. A lo largo del torneo se lleva a cabo la venta de distintos productos con el fin de financiar el propio torneo, esta venta es gestionada por uno de los módulos de la aplicación.

La aplicación implementa, entre otras cosas, mecánicas de Drag & Drop para el menú principal. En cada uno de los módulos se llevarán a cabo distintas operaciones para realizar el seguimiento de los movimientos que se realicen. Se cuenta con una base de datos en local para el módulo de gestionar las compras ofreciendo la posibilidad de exportar los datos y una base de datos en la nube para el almacenamiento del módulo de gestión de partidos y de las operaciones realizadas sobre el mismo.

Para el desarrollo del proyecto se ha requerido un análisis profundo en el que se detalla su estructura la forma en la que se gestionan los datos y las operaciones realizadas sobre los mismos. De igual forma ha sido una oportunidad para conocer Android y profundizar en su estudio y sus funcionalidades que ofrecen una cantidad infinita de posibilidades de crecimiento gracias a todas las facilidades que suministra, a toda la información disponible y la cantidad de elementos que pueden integrarse.

Palabras clave: Android, Ultimate, Frisbee, Aplicación

Abstract

Ultimate Frisbee, also known as Ultimate, is a non-contact team sport which has traditionally relied upon a spirit of sportsmanship, placing the responsibility for fair play on the player. Ultimate is played by players with a flying disc where points are scored by passing the disc to a teammate in the opposing end zone. This sport is mostly played mixed gender. The teams are composed of seven players if the match is going to take place in grass or five players in case it takes place at the beach.

There are over 500 Ultimate official players in Spain. This document aims at providing an Android application responsible for the management of an Ultimate Frisbee tournament called Dr. Sand & Mr. Grass. This tournament takes place in Tenerife and brings together over 150 people from around Europe. It has taken place sixteen times and preparations are under way for the next championship to be held on February 2017. During Dr. Sand & Mr. Grass tournament, the responsible team called Guayota, manages to sell some products as a way to pay for the cost of the competition and to make the team able to travel to other tournaments.

This application seeks to manage purchasing and the operation of matches so it can help to handle the tournamet's tasks easily. All information about each feature and the way it was implemented will be explained throughout this document.

Keywords: *Ultimate Frisbee, App, Android, Application*

Índice General

Índice de Ilustraciones

Índice de tablas

Capítulo 1. Introducción

1.1 Descripción

El Ultimate Frisbee[1] es un deporte que se basa en el juego limpio ya que no tiene árbitros y son los propios jugadores quienes ejercen este rol. El juego consiste en dos equipos de 5 o 7 jugadores cada uno dependiendo si el terreno de juego es playa o césped respectivamente. Cada jugador tendrá diez segundos para hacer un pase a un compañero y cada equipo podrá realizar tantos pases como desee con el objetivo de llegar a una zona en la que en caso de recibir un pase será punto. Si el jugador no realiza el pase durante esos diez segundos la posesión del disco cambia y será el otro equipo quien ataque.

A pesar de no ser un deporte que destaque entre los más conocidos en España hay más de 500 jugadores que practican este deporte. La aplicación desarrollada en este documento se encargará de la gestión del torneo de Ultimate Frisbee Dr. Sand & Mr. Grass, nace en Canarias, reúne a más de 150 personas tanto de España como de Europa y lleva celebrándose, contando la de este año, 16 ediciones. A lo largo de sus tres días de duración el equipo organizador del torneo, Guayota, lleva a cabo la venta de productos de distinta índole para poder financiar el propio torneo y la posible participación del equipo en torneos fuera de la isla.

Es por esto que este proyecto pretende impulsar y facilitar el desarrollo de tareas que hasta ahora se debían llevar a cabo manualmente. Se distinguen dos grandes módulo: Gestión de Partidos y la Gestión de las Compras realizadas durante el torneo con el fin mencionado previamente. A lo largo de este documento se entrará en detalle, se explicará cada funcionalidad y la forma en la que se ha llevado a cabo su implementación.

1.2 Antecedentes

Aprovechando el auge de la tecnología móvil y los dispositivos móviles se abren nuevas posibilidades para el desarrollo de aplicaciones, es por eso que la posibilidad de automatizar y simplificar tareas para los usuarios es cada vez mayor.

En el ámbito deportivo son múltiples las opciones que aparecen cuando se trata de los deportes más comercializados actualmente, la NBA o el fútbol dominan las aplicaciones que ofrecen o monitorizan partidos y sus resultados. Se presenta por tanto una herramienta que permita realizar esta actividad para un deporte minoritario pero en pleno desarrollo.

1.3 Aplicaciones Similares

Existen aplicaciones en el mercado con funcionalidades similares a algunas de las desarrolladas en la aplicación detallada en esta memoria. Destacan entre ellas:

- **UltiAnalytics Frisbee Stats[2]:** aplicación en la que se hace un seguimiento de partidos que se ingresan de forma manual. Ofrece la posibilidad de publicar entradas en la red social Twitter a medida que ocurren puntos o situaciones importantes dentro del partido.

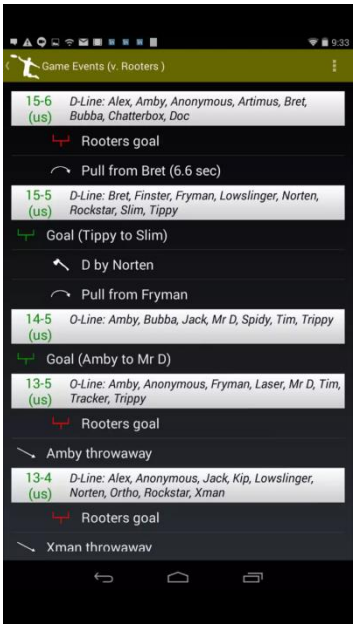


Ilustración 1 Capturas de pantalla UltiAnalytics

- **Organizar torneo, crear liga[3]:** aplicación que permite crear partidos para un torneo y especificar el resultado obtenido en cada partido, ofrece la posibilidad de mostrar la clasificación general de los equipos.

Ilustración 2 Captura Organizar torneo, crear liga

- **NBA app[4]:** a pesar de no ser una aplicación relacionada con el Ultimate Frisbee contiene algunas funcionalidades similares a las que se desean implementar en el proyecto aquí detallado. Entre ellas están las estadísticas de cada jugador, a pesar de que en esta aplicación no se ofrecen estos datos, se lleva a cabo el almacenamiento de los mismo en el modelo de datos con vistas a en un futuro enseñar estos datos.

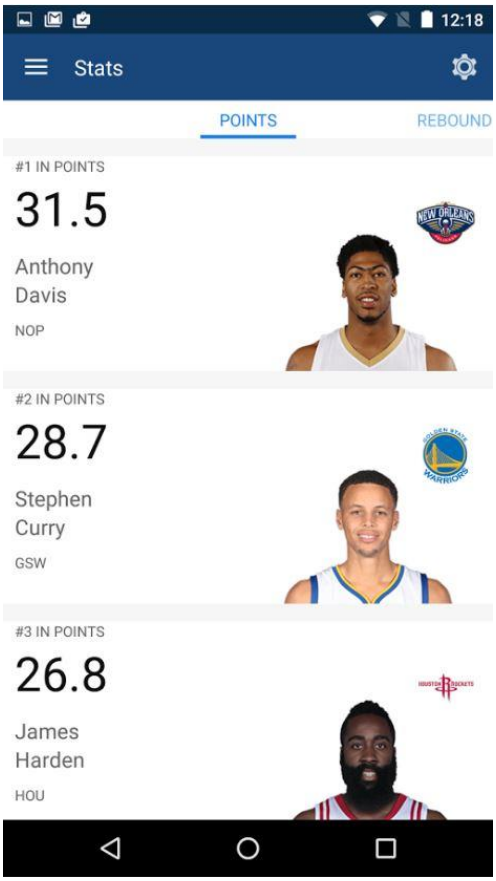


Ilustración 3 Captura NBA app

1.4 Propuesta

La aplicación tiene distinguidas dos grandes funcionalidades visuales a las que se accede mediante un menú principal. Para poder manejar y usar estas funcionalidades se tratan previamente todos los datos necesarios, como lo son los equipos con los jugadores que los conforman y los productos disponibles para el torneo.

- **Gestión de Partidos**

  En este módulo está contemplada la gestión de un partido, existe una lista en la que están todos los partidos por jugar, sobre la cual se puede elegir un partido y proceder a su gestión.

  Al acceder a la gestión de un partido se puede puntuar a cada equipo y asignar el punto al jugador que corresponda, de igual manera en la gestión el partido puede posponerse y reanudarse en el momento

que así sea necesario con el resultado tal y como iba en el momento en que se posponga.

La lista de partidos por jugar se genera siguiendo patrones para que se adapten al formato del torneo y evitando de esta forma la tarea de que se realice manualmente los emparejamientos entre cada equipo. Existe una primera ronda en la que se enfrentan todos los equipos entre sí y una fase final en la que en base a los resultados previos estarán los equipos que se hayan clasificado para ello.

Existe también una lista de partidos finalizados, esta lista contiene los resultados finales de estos partidos, gracias a los datos que se recogen se despliega también la clasificación.

- **Gestión de compras**

Se lleva a cabo la venta de diversos productos con el objetivo de recolectar dinero para financiar el torneo y la participación de Guayota en torneos fuera de las Islas Canarias.

Por tanto en la gestión de compra se despliega la lista de equipos ordenada de forma alfabética y la lista de jugadores pertenecientes a cada equipo. Cada jugador tendrá a su disposición dos opciones: agregar una compra y pagar la deuda acumulada que tenga.

Si la elección es agregar una compra, se desplegarán los distintos productos que pueden añadirse o eliminarse.

Capítulo 2. Objetivos

Este trabajo tiene como objetivo principal la creación de una aplicación para dispositivos móviles con el sistema operativo Android capaz de simplificar y ayudar a gestionar un torneo de Ultimate Frisbee. Esta gestión incluye el seguimiento de los partidos y de las compras que se realicen durante el torneo.

Para conseguir dicho objetivo se pretende el uso de las nuevas tecnologías y de las competencias adquiridas a lo largo del Grado de Ingeniería Informática en la Universidad de La Laguna.

El proyecto incluye, además de la aplicación, objetivos entre los que se pueden destacar:

- Ampliar conocimientos en Android.
- Llevar a cabo el diseño y desarrollo de un proyecto.
- Usar metodologías de trabajo.
- Creación de una memoria técnica sobre la aplicación desarrollada en el presente trabajo.
- Puesta en práctica de los conocimientos obtenidos a lo largo de los estudios del Grado de Ingeniería en Informática.

2.1 Metodología de trabajo

Tras realizar un análisis en profundidad de las distintas opciones para el desarrollo de este proyecto se ha elegido seguir la metodología Agile[5].

Agile se basa en un enfoque iterativo para planificar y guiar los procesos del proyecto, esto significa que la aplicación ha sido completada siguiendo iteraciones. Tras la finalización de cada iteración se ha realizado una

reunión con el tutor del TFG en la que se llevó a cabo una validación de cada sprint[1].

Agile tiene entre sus mayores ventajas la productividad, calidad y seguimiento que se lleva a cabo de los avances del proyecto, para facilitar esta tarea se hizo uso de Trello[2][6]. Gracias a esta herramienta se puede especificar el Product Backlog[3] y llevar un seguimiento práctico y sencillo para el desarrollo del proyecto.

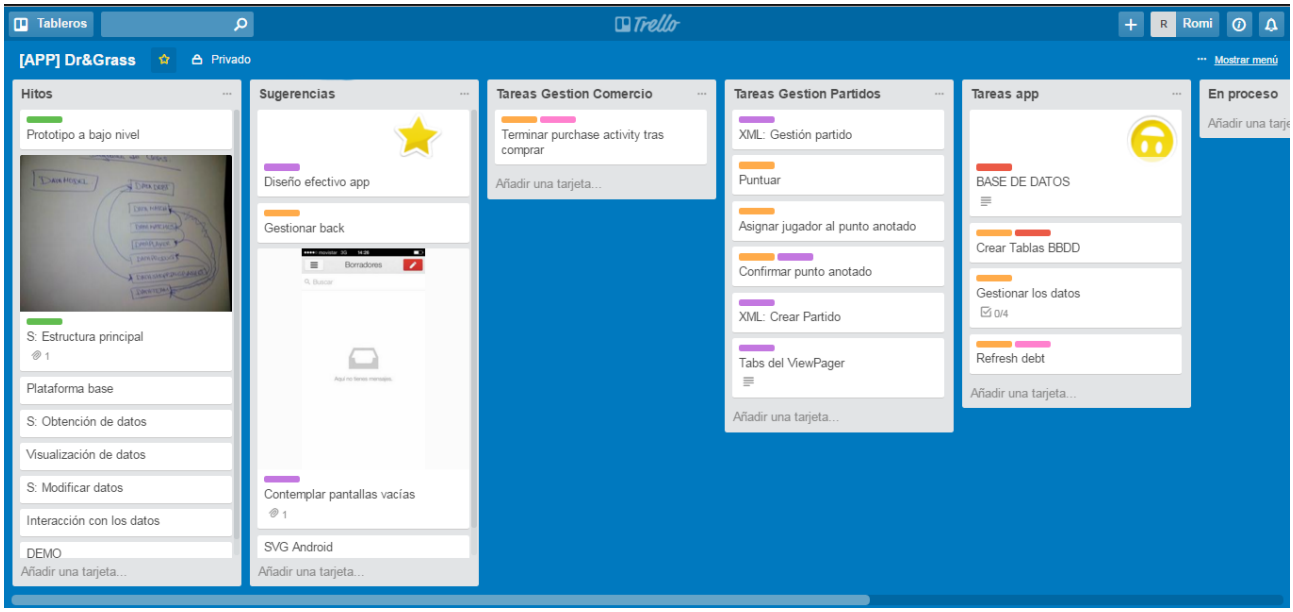


Ilustración 4 Tareas Trello del Proyecto

2.2 Plan de trabajo

Para la aplicación se establecen tres sprints de 14 días de trabajo cada uno:

- Estructura Principal: Sprint en el que se propone el desarrollo de la estructura base de la aplicación, el modelo de datos que se seguirá y el patrón a seguir para el posterior trabajo estos datos.

---

[1] Sprint: iteración que proporciona un incremento de producto que sea potencialmente entregable.

[2] Trello: herramienta que permite organizar proyectos y tareas.

[2] Trello: herramienta que permite organizar proyectos y tareas.

[3] Product Backlog: lista organizada por prioridades de las tareas a realizar.

- Obtención de datos: Sprint para traer los datos desde un JSON[4] e incrustarlos en la estructura principal conseguida anteriormente.
- Modificación de los datos: Sprint en el que, tras su finalización, se puede trabajar con los datos obtenidos desde el JSON.

Con el objetivo de obtener un proyecto bien definido se establecen tres fases: Análisis, Desarrollo y Documentación.

Tarea	Nombre	Tiempo	
1	Estudio de la propuesta	7 días	Incepción
2	Desarrollo de la propuesta	7 días	
3	PROTOTIPO A BAJO NIVEL	HITO	
4	Sprint 1: Estructura principal	14 días	Desarrollo
5	PLATAFORMA BASE	HITO	
6	Sprint 2: Obtención de datos	14 días	
7	VISUALIZACIÓN DE DATOS	HITO	
8	Sprint 3: Modificar los datos	14 días	
9	INTERACCIÓN CON LOS DATOS	HITO	
10	Demo funcional	7 días	
11	Guía de usuario	7 días	Documentación
12	Memoria TFG	14 días	
13	Vídeo de Presentación	7 días	
14	Presentación PowerPoint	7 días	

Tabla 1 Plan de Trabajo

2.2.1 Fase Análisis

Durante la primera fase se realiza un estudio profundo sobre las aplicaciones disponibles para propósitos similares expuestas con anterioridad, de igual forma se examinan las posibilidades de éxito del objetivo del

[4] JSON: **J**ava**S**cript **O**bject **N**otation, es un formato de texto ligero para el intercambio de datos.

proyecto y las tecnologías que se usarán para conseguir dicho objetivo.

Partiendo de los resultados de esta fase se puede llevar a cabo la planificación del proyecto y los mockups[5] de la aplicación cumpliendo de esta manera el primer sprint planteado por la metodología Agile.

2.2.2 Fase Desarrollo

Siguiendo la metodología Agile esta fase será iterativa ya que tiene como objetivo conseguir cumplir cada sprint de los detallados previamente. Al ser una fase que conlleva la implementación de múltiples funcionalidades es en la que más se detallan las tareas con el fin de tener un seguimiento exhaustivo, evitar errores y que la productividad se mantenga en un alto nivel.

El hito de esta fase es obtener un demo funcional de la aplicación en la que todas las funcionalidades detalladas en este documento se lleven a cabo correctamente.

2.2.3 Documentación

En esta fase se ha de generar la documentación completa de la aplicación desarrollada con el fin de proporcionar la información necesaria para comprender el procedimiento del desarrollo del proyecto y de la aplicación implementada.

Se generan informes de todos los aspectos técnicos implicados en el desarrollo de la aplicación y las metodologías seguidas para cumplir con los objetivos planteados.

[5] Mockup: modelo a escala de un diseño utilizado para la demostración del diseño.

Capítulo 3. Desarrollo

3.1 Diseño

3.1.1 Estructura

A lo largo de esta primera fase de diseño se especifica la estructura que sigue el proyecto, la forma en la que está constituido el árbol de directorios lógicos y la profundización en detalle de las funcionalidades de la aplicación.

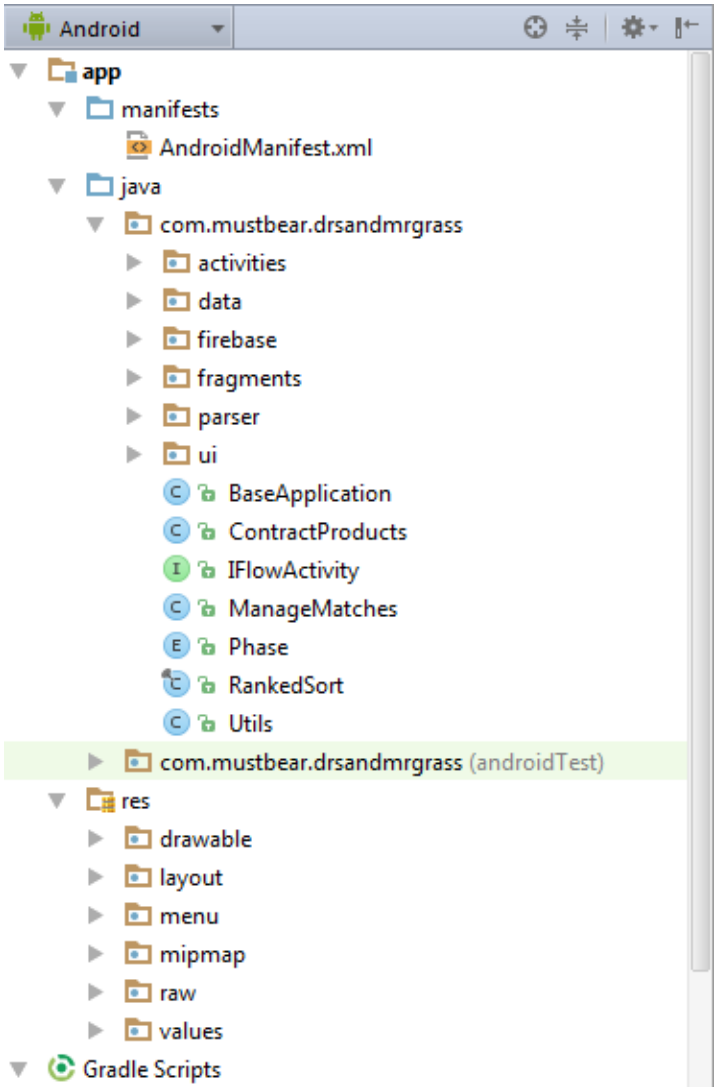


Ilustración 5 Estructura del proyecto

Como el objeto del presente documento no es llevar a cabo un análisis profundo del árbol de directorios que siguen los proyectos desarrollados en Android[6] se comentarán brevemente los más relevantes, en la imagen previa se detalla la estructura que sigue el proyecto:

- **Manifests**: contiene el fichero AndroidManifest que será el encargado de definir la aplicación, desde su nombre hasta la versión mínima de Android para poder ejecutarla.
- **Java**: contiene el código fuente de la aplicación, los ficheros se almacenan en carpetas basándose en el nombre del paquete. En el proyecto se distinguen los paquetes:
  - **Activities**: contiene las clases para la gestión de las *Activities*.
  - **Data**: contiene las clases que se encargan del modelo de datos.
  - **Firebase**: clases para la gestión de las operaciones necesarias para el uso de Firebase.
  - **Fragments**: clase para la gestión de los *fragments* usados a lo largo del proyecto.
  - **Parser**: contiene las clases necesarias para el tratamiento de datos, tanto de llevar a cabo el tratamiento de los ficheros JSON[7] donde vendrán los equipos y los jugadores pertenecientes al mismo. También se tratan los ficheros CSV[8] donde se encuentra la deuda de los equipos.
  - **UI**: clases para el manejo de la interfaz de usuario.
- **Res**: contiene los recursos usados por la aplicación.
  - **Drawable**: imágenes usadas en la aplicación. También puede contener selectores, que son ficheros XML que permiten asignar una imagen u otra a un recurso en función de una condición.
  - **Layout**: contiene ficheros XML con vistas de la aplicación, las vistas permiten configurar las pantallas que componen la interfaz de usuario.
  - **Menu**: contiene ficheros XML con los menús de cada actividad
  - **Mipmap**: carpeta donde se almacenan imágenes en función de la densidad de píxeles, de esta forma la aplicación sigue un diseño más adaptativo.

- **Raw:** ficheros adicionales que no se encuentran en formato XML, en este caso éste directorio contiene los ficheros JSON con los equipos y los jugadores pertenecientes a los mismos.
- **Values:** ficheros XML que indican valores usados en la aplicación, se pueden especificar, entre otros elementos: colores, márgenes y cadenas.

3.1.2 Funcionalidades

La aplicación contiene una base de datos en Firebase[6][9] en la que almacenar los datos y modificarlos según la operación a tratar, a lo largo de la aplicación tal y como se mencionó previamente destacan dos grandes módulos: gestión de compras y gestión de partidos, para cada una se detallan a continuación las operaciones básicas.

Funcionalidades gestión de compras:

- **Lista de participantes:** se debe desplegar una lista con todos los equipos cargados en el modelo de datos, si se selecciona uno de los equipos se debe mostrar la lista de jugadores que pertenezcan al mismo.
- **Gestionar productos disponibles:** cada participante puede adquirir o eliminar cualquiera de los distintos productos ofertados, cada producto que adquiera le acarrea una deuda que puede ser gestionada posteriormente. Se contempla la posibilidad de que se pueda eliminar solo en caso de que el jugador haya adquirido los artículos previamente.
- **Deuda de participantes:** se ha de gestionar la deuda de los participantes mediante su asignación y se ha de ofrecer la posibilidad de realizar el pago de la misma. Se contempla que la deuda solo sea saldada si la misma existe. Tras realizar el pago de una deuda el jugador puede volver a realizar compras ya que la deuda se generará nuevamente.
- **Generar fichero CSV:** el usuario tiene la opción de exportar en un fichero CSV todas las compras

[6] Firebase: Es un proveedor de servicios en la nube.

realizadas durante el torneo para un análisis posterior.

Funcionalidades gestión de partidos:

- **Lista de partidos:** se listan los partidos que van a disputarse y que se han generado de forma automática, se despliegan también los partidos disputados junto con sus resultados.
- **Gestión de partidos:** se suman puntos y se asignan al jugador y equipo correspondiente. Cada partido tiene diversas posibilidades:
  - Finalizar: se comprueba el resultado y se almacena el partido como finalizado.
  - Posponer: se ofrece la posibilidad de pausar el partido y reanudarlo posteriormente con el resultado que existía en ese momento.
  - Cancelar: salir del partido sin realizar ninguna operación en el mismo.
- **Fase final:** se gestionan los resultados de la primera fase y en base a ello se genera una fase eliminatoria.
- **Clasificación:** se genera una lista en la que se despliega en orden la clasificación de los equipos basándose en los resultados obtenidos a lo largo del torneo.
- **Almacenamiento:** los datos referentes a los partidos se gestionarán en la nube gracias a Firebase.

3.2 Análisis

Esta fase arrastra la mayor carga de trabajo ya que abarcó la toma de decisiones sobre los componentes que se usarían y la forma en la que se llevaría a cabo su implementación.

3.2.1 Diseño funcional – Mockups

El proyecto mantiene como objetivo obtener una aplicación para Android que pueda ser instalada en dispositivo con este sistema operativo. Tras un análisis

profundo se llega a la conclusión de que esta aplicación puede desenvolverse mejor y tener un mayor alcance si los dispositivos objetivo son *tablets*. Los *tablets* ofrecen mayor comodidad al usuario para llevar a cabo el control de un partido o la gestión de compras ya que abarca mucha más información y simplifica la visualización de la misma.

El hecho de conocer que los dispositivos a los que se dirige y saber que son *tablets* simplifica las API[7] de Android sobre las que se debe trabajar puesto que todas tienen como versión mínima del sistema operativo la versión 4.0 – Ice Cream Sandwhich. Esto significa que la compatibilidad no supone una complejidad tan grande como podría serlo en caso de ser destinado a cualquier dispositivo móvil.

Lo primero que se encontrará será la imagen de carga de la aplicación, en este caso con imágenes del torneo.

Ilustración 6 Splash Screen de la aplicación

Se diseñó un menú principal en el que se contempla el acceso a los dos grandes módulos que se han distinguido a lo largo de este documento: gestión de compras y gestión de partidos.

[7] API: **A**pplication **P**rogramming **I**nterface, conjunto de subrutinas funciones y procedimientos que ofrece una biblioteca para ser utilizado por otro software como capa de abstracción.

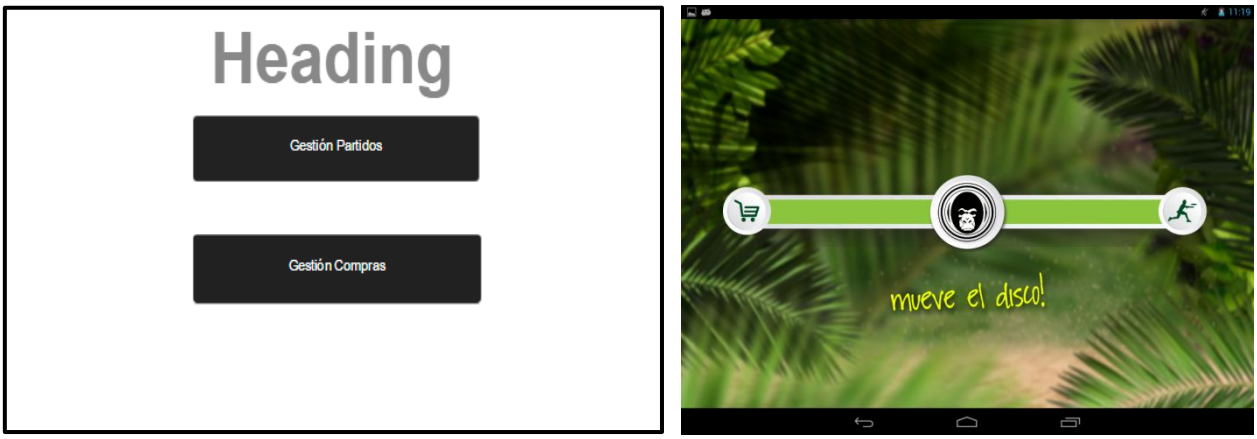


Ilustración 7 Mockup menú inicio

A continuación en caso de acceder a la gestión de partidos se encontrará una lista separada en dos pestañas con los partidos pendientes y los partidos finalizados.

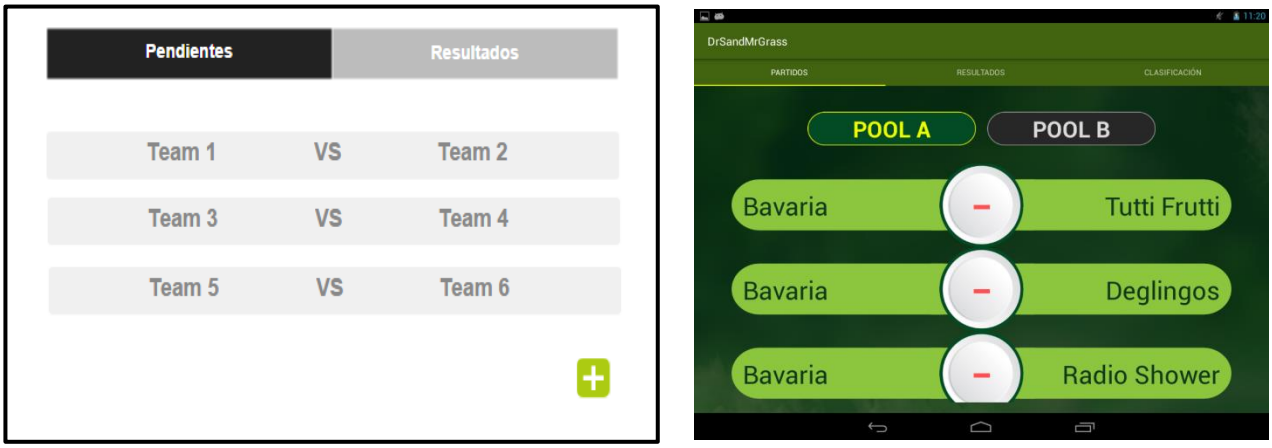


Ilustración 8 Mockup gestión partidos

Dentro de cada partido se lleva a cabo la gestión del mismo, esta gestión incluye la suma o resta de puntos a cada equipo según corresponda y la asignación de puntos al jugador que pertenezca al equipo sobre el que se realice la operación.

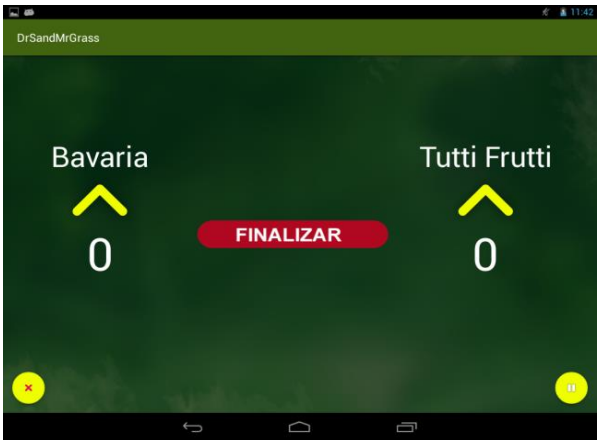


Ilustración 9 Mockup gestionar partido

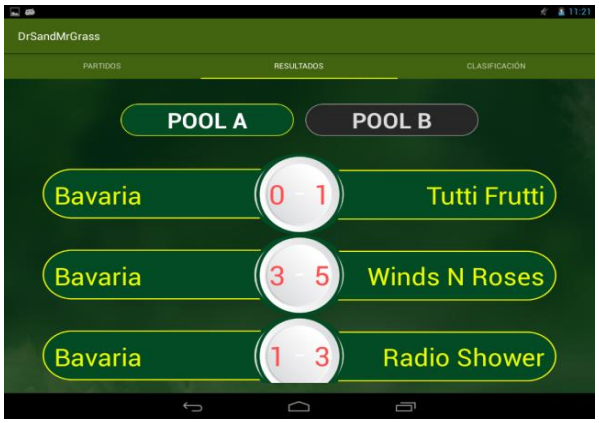


Ilustración 10 Mockup partidos finalizados

Los partidos del torneo están constituidos por partidos distribuidos en dos fases, una primera fase en la que se enfrentan todos los equipos entre sí y una fase posterior en la que se disputan los 8 primeros puestos de forma eliminatoria y el resto enfrentando a los equipos nuevamente entre sí.

Ilustración 11 Fase eliminatoria

Gracias a la gestión que se realiza de los partidos se genera la clasificación tanto de la primera fase como la de la clasificación final del torneo. Esto magnifica las posibilidades en cuanto a estadísticas que el proyecto puede ofrecer.

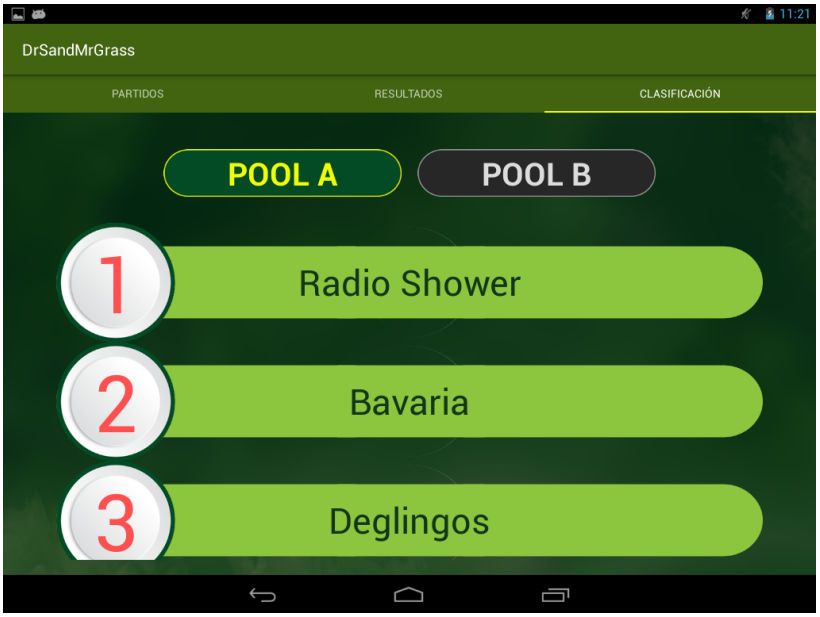


Ilustración 12 Clasificación primera fase

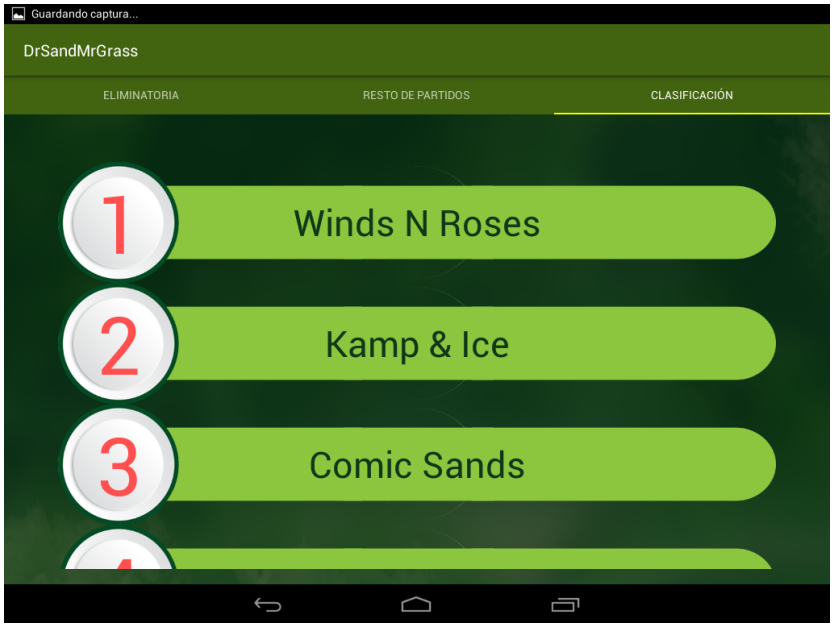


Ilustración 13 Clasificación final

Por otro lado, si la opción elegida es la gestión de compras lo primero que se despliega será la lista de equipos y a su lado los jugadores que pertenezcan al equipo seleccionado. Cada jugador tendrá la deuda junto con las opciones de agregar productos a su historial o saldar la deuda en caso de que ésta exista. En función de la existencia de la deuda del jugador se desplegará de una color u otro, verde en caso de que no tenga deuda acumulada y rojo en caso contrario.

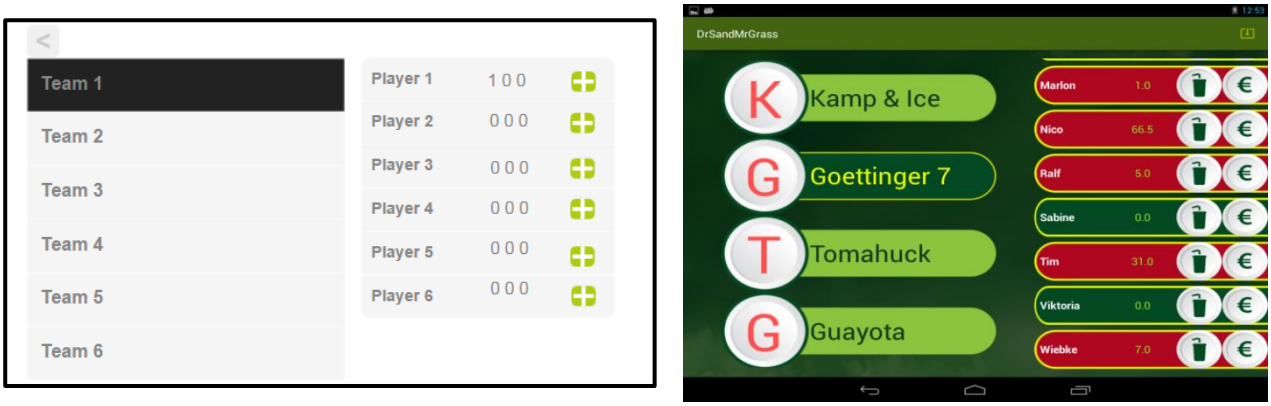


Ilustración 14 Mockup gestión compras

Si se elige la opción de gestionar los productos se accederá a una pantalla en la que se despliegan todos los productos disponibles junto con la opción de agregarlos o eliminarlos de las compras asignadas al jugador. Una vez seleccionados los productos con los que se desea operar y si se deben agregar o eliminar se deberá confirmar la

operación, en esta confirmación aparece un mensaje con la lista de productos seleccionados.

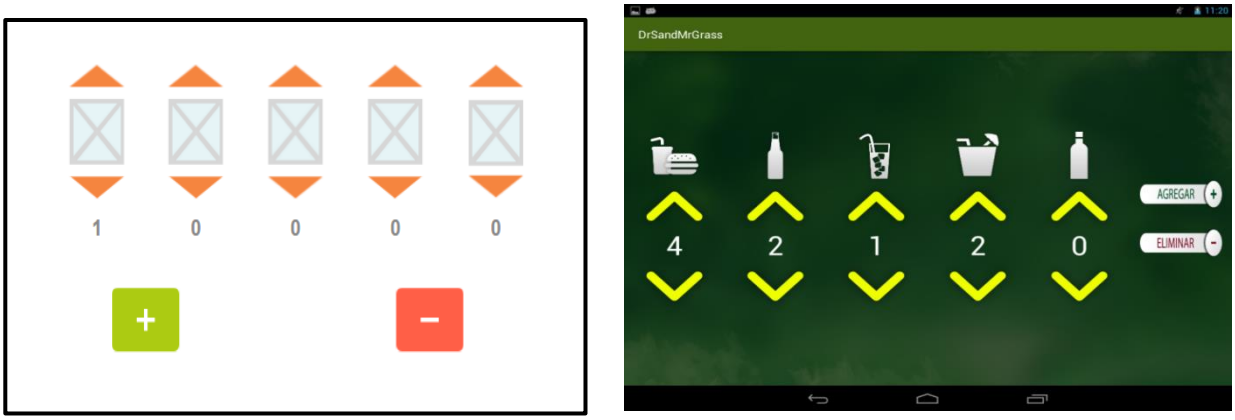


Ilustración 15 Mockup gestión compra

Por cada operación que conlleve modificación alguna en los datos se pide una confirmación al usuario en el que se muestran los datos que cambiará. Por ejemplo, si un usuario ejerce la opción de finalizar un partido se le mostrará un AlertDialog[8] junto con el resultado en ese momento del partido y si desea finalizarlo con dicho resultado. En caso afirmativo procede a finalizar el partido y realizar las acciones que ello conlleve, en caso contrario mantiene al usuario en la gestión del partido.

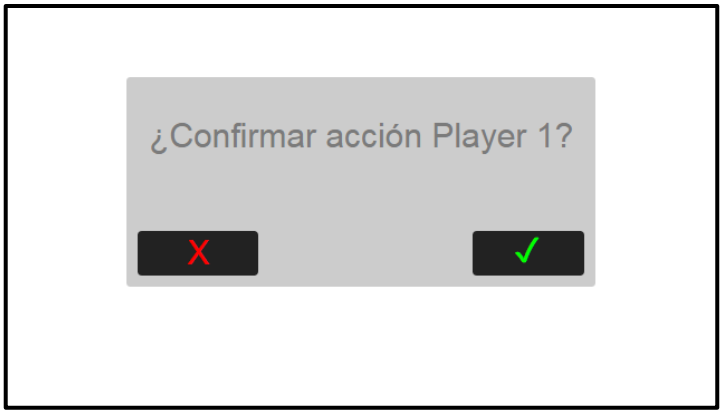


Ilustración 16 Mockup AlertDialog

[8] AlertDialog: notificación de Android que solicita al usuario que confirme una determinada acción.

Ilustración 17 Ejemplo confirmaciones solicitadas

3.2.2 Mecánicas

La aplicación busca simplificar actividades y a partir de ello crear una interfaz natural. En el menú principal se aprovecha el Ultimate Frisbee como base y se usa un disco con la mecánica típica de Drag&Drop para elegir de esta manera el módulo al que se quiere acceder, de esta forma hace que el usuario se sienta bienvenido y con una interfaz con la que puede identificarse. Se encuentra también el uso del elemento de Android ViewPager[9][10], esto permite que se deslice por la pantalla para elegir si lo que se desea ver son los partidos pendientes, los disputados o la clasificación de los equipos.

De igual forma cabe destacar que para generar automáticamente los enfrentamientos se han tenido en cuenta diversos factores. Este torneo tiene características especiales para los enfrentamientos ya que su desarrollo se divide en una fase principal y una eliminatoria. Para la primera fase se desarrolla un algoritmo que genera los enfrentamientos entre todos los equipos. En la fase los factores a tener en cuenta aumentan, se debe tomar la clasificación previa y contemplar de igual manera qué sucede en caso de empate en dicha clasificación. Para este torneo en particular, y por tanto en la aplicación también, se siguen las reglas estipuladas en la normativa de la Federación de Ultimate Frisbee en las acciones a realizar en caso de empate.

[9] ViewPager: vista de Android que permite desplazarse por diferentes menús de la aplicación.

Todas las mecánicas trabajan para el desarrollo de una interfaz práctica, sencilla e intuitiva con el objetivo de facilitar al usuario la simplificación de actividades y la adaptación al uso de la aplicación.

3.3 Implementación

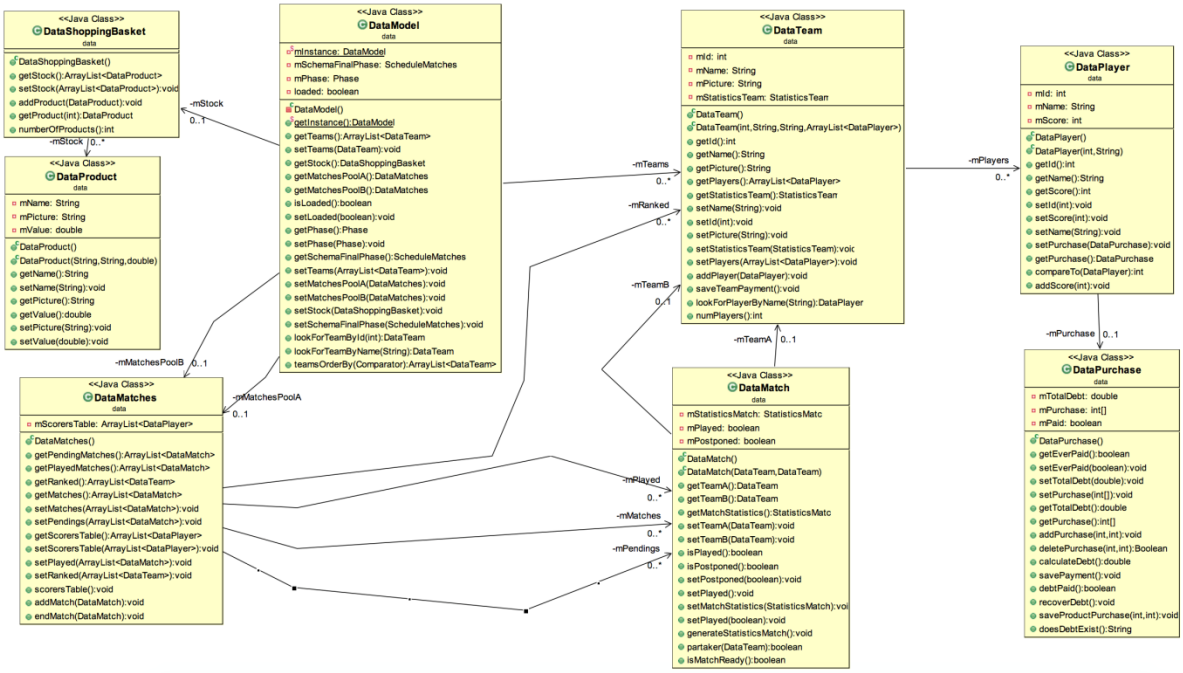


Ilustración 18 Diagrama de clases

3.3.1 Patrones de diseño en Android

En Android puede decirse que la pantalla de una aplicación es una *Activity* que se constituye por una parte lógica y una parte gráfica. En la parte lógica será donde se encuentre el código Java con una clase que se encargara de manipular y colocar el código de la *Activity*. La parte gráfica es un XML que tiene todos los elementos visibles en la pantalla declarados con etiquetas similares a las de una página web.

Una vez conocido que el dispositivo a usarse en este proyecto serían *tablets* los *fragments[11]* se convierten en elementos indispensables, hasta el momento Android trabajaba con las *activities* pero los *fragments* surgen precisamente para solucionar el problema de la adaptación de la interfaz gráfica en las aplicaciones destinadas a

este tipo de dispositivos. Los *fragments* pueden definirse como una porción de la interfaz de usuario que pueden ser añadidos o eliminados de forma independiente al resto de elementos de la *activity*. Esto significa que la pantalla ya no debe ser exclusiva para una funcionalidad sino que permite dividir la interfaz en varias porciones y evitar duplicar código ya que pueden ser reutilizados.

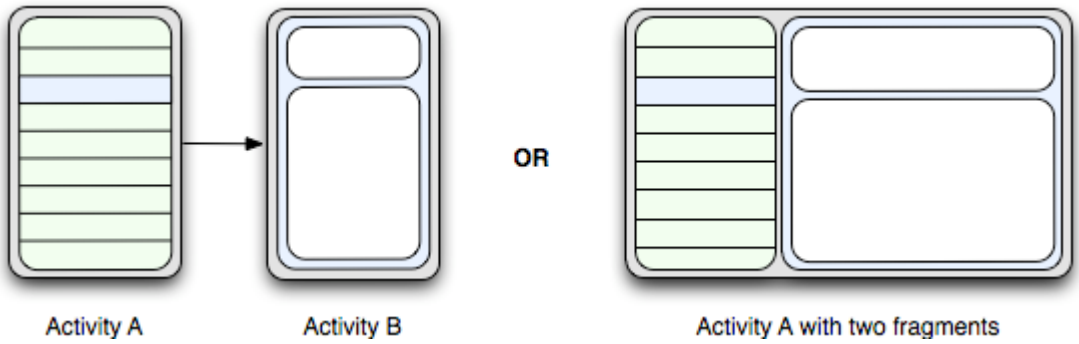


Ilustración 19 Diferencia Activity y Fragments

Este proyecto se desarrolla usando *fragments* en todas sus *activities*, de esta forma se ofrece mayor escalabilidad en caso de querer añadir nuevos componentes. Como se usan a lo largo de todo el proyecto se usará un ejemplo con el fin de ilustrar su funcionamiento y comportamiento. La pantalla de gestión de compras tiene dos *fragments* bien definidos, a la izquierda un *fragment* que contiene la lista de equipos y a la derecha otro *fragment* en el que se encuentra la lista de jugadores, en base al primero el segundo cambiará. Si esto se implementa sin el uso de *fragments* no se podría reutilizar la lista de equipos, que en este caso es reutilizada para la clasificación, tampoco podría usarse en un dispositivo móvil ya que en este caso se requeriría la reestructuración de los elementos.

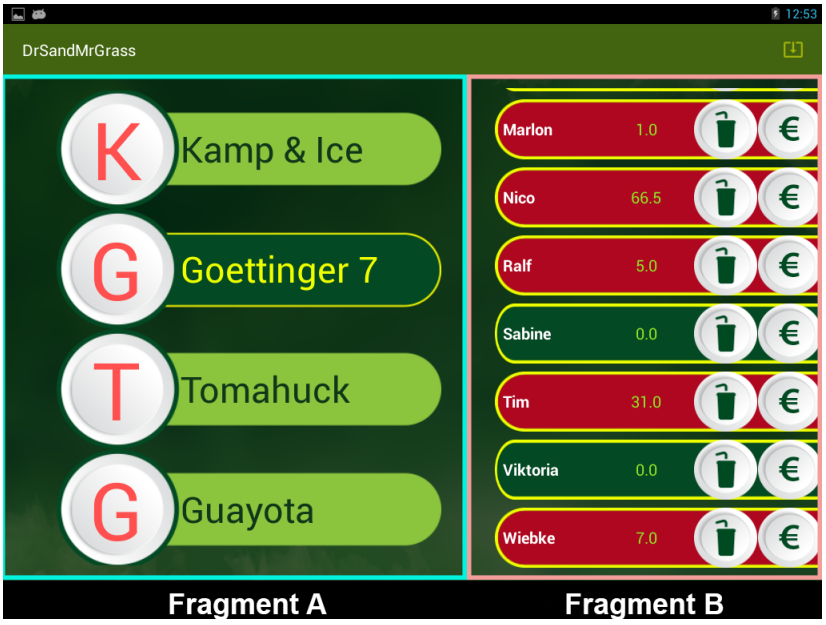


Ilustración 20 Fragments en la aplicación

Cabe destacar que esta estrategia usando los *fragments* y las *activities* da pie al patrón MVC[10]. La analogía aquí presentada da como modelo el objeto que tiene todos los datos de la aplicación, como controlador a la *activity* y por último pero no menos importante la vista como el *fragment*.

3.3.2 Mecánicas

En la aplicación se hace uso de la mecánica Drag&Drop, esta mecánica se usa en la pantalla principal de la aplicación donde se encuentra el menú que gestiona el acceso a cualquiera de los módulos principales de la aplicación.

Para el uso de esta mecánica se implementa un *listener* que detecta cuando la pantalla es tocada y el momento en el que se mueve el objeto por la misma, esto lo hace gracias al onTouch de los MotionEvents de Android. Es necesario detectar también las dimensiones de la pantalla de esta forma en cuanto el *listener* implementado detecte que está en la zona de las opciones lo detecte y lance la *Activity*.

[10] MVC: **M**odelo **V**ista **C**ontrolador

Ilustración 21 Menú con Drag&Drop

Otra mecánica usada en el proyecto es el uso de los ViewPager de Android, el ViewPager es una vista de Android constituida por la serie de elementos que quieran desplegarse, esto permite deslizarse entre los menús con un simple gesto. Esta mecánica puede encontrarse en la pantalla de gestión de partidos donde con el gesto de deslizar cambiamos entre partidos pendientes, los disputados y la clasificación. El uso de este elemento ofrece un aspecto más limpio ya que evita la necesidad de incluir botones o enlaces a los distintos menús de la aplicación.

3.3.3Codificación

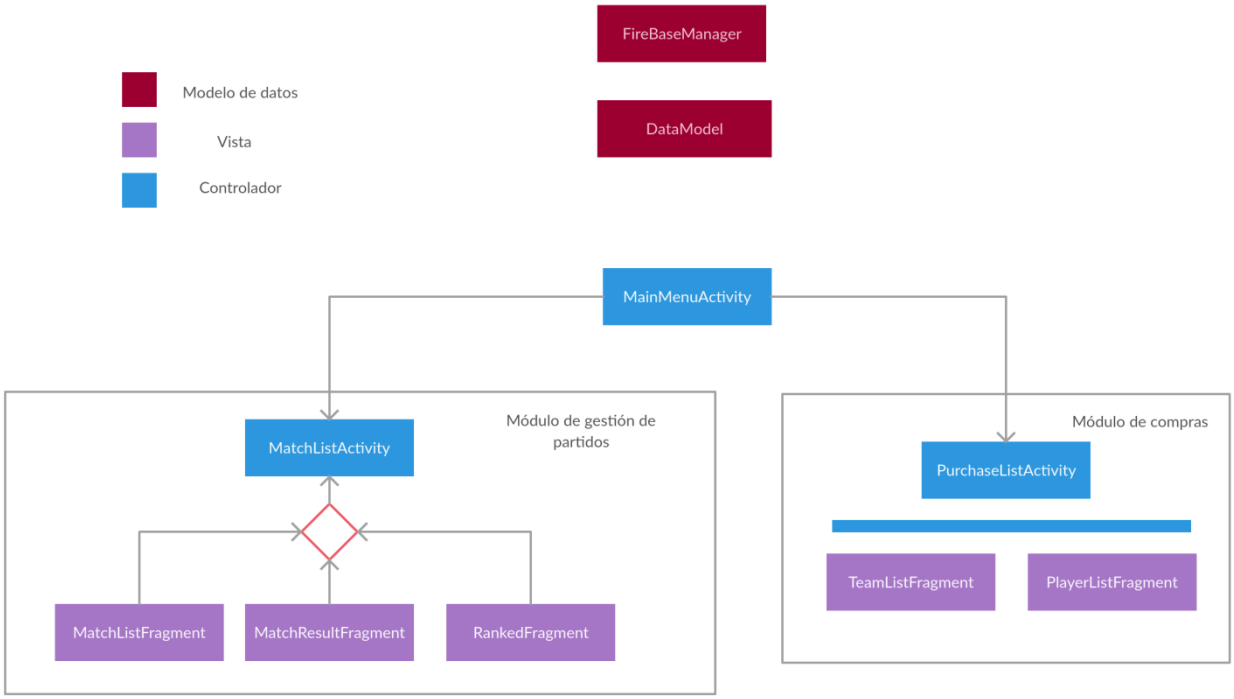


Ilustración 22 UML MVC

En el diagrama anterior se observa la forma en la que se estructura la base de la aplicación y los roles que cumplen cada clase de la estrategia MVC.

- **DataModel**: Es una clase elaborada para que actúe de "almacén" de datos de la aplicación, siguiendo un patrón Singleton para conseguir dicho propósito.
- **MainMenuActivity**: Es la *activity* del proyecto que se encarga de hacer la selección de módulo mediante la mecánica del drag & drop.
- **MatchListActivity**: La *activity* que tiene el control de la gestión de partidos. Posee el ViewPager encargado de mostrar las listas de los partidos. Por tanto se encarga de seleccionar que *fragment* pintará. Contiene a tres y sólo se muestra uno de estos a la vez.
    - MatchListFragment: Pinta la lista de partidos que no están jugados aún.
    - MatchResultFragment: Pinta la lista de partidos jugados.
    - RankedFragment: Pinta la lista de equipos en orden.

- **PurchaseListActivity**: Encargada de mostrar los equipos y de mostrar los jugadores después de seleccionar uno permitiendo gestionar la compra de cada jugador. Por tanto, esta *activity* será la que tiene el control de la gestión de compras. Contiene dos *fragmentos* encargados de la parte visual que en este caso se muestran simultáneamente.
  - TeamListFragment: Pinta una lista de los equipos.
  - PlayerListFragment: Pinta una lista de los jugadores
- **FirebaseManager**: Clase que gestiona la conexión con el servidor proporcionado por el Firebase. Guarda y lee desde la nube.
- **BaseApplication**: Objeto propio de Android que está presente a lo largo de toda la ejecución y ayuda a mantener la instancia de la conexión

3.3.4Calidad de Código

Como se mencionó previamente con el fin de obtener un código comprensible y consistente se hace uso de una convención de código personal, en esta convención destacan:

- Los atributos miembros de una clase son declarados con una "m" por delante, la siguiente letra irá en mayúsculas.
- Se usa CamelCase[11] en la declaración de cualquier elemento.
- Los nombres de las clases deben ser sustantivos, siguen CamelCase también.
- Los nombres de las funciones han de ser verbos.
- Los nombres de las variables deben ser identificativos, no está permitido usar variables nombradas del modo "x,y,z" salvo para recorrer bucles.

[11] CamelCase: estilo de escritura que se aplica a frases o palabras compuestas.

En el análisis realizado por Sonar queda reflejado el número de líneas de código que posee el proyecto que asciende a las 3800 junto con las clases, los métodos y otra información relevante en referencia a la composición del mismo.

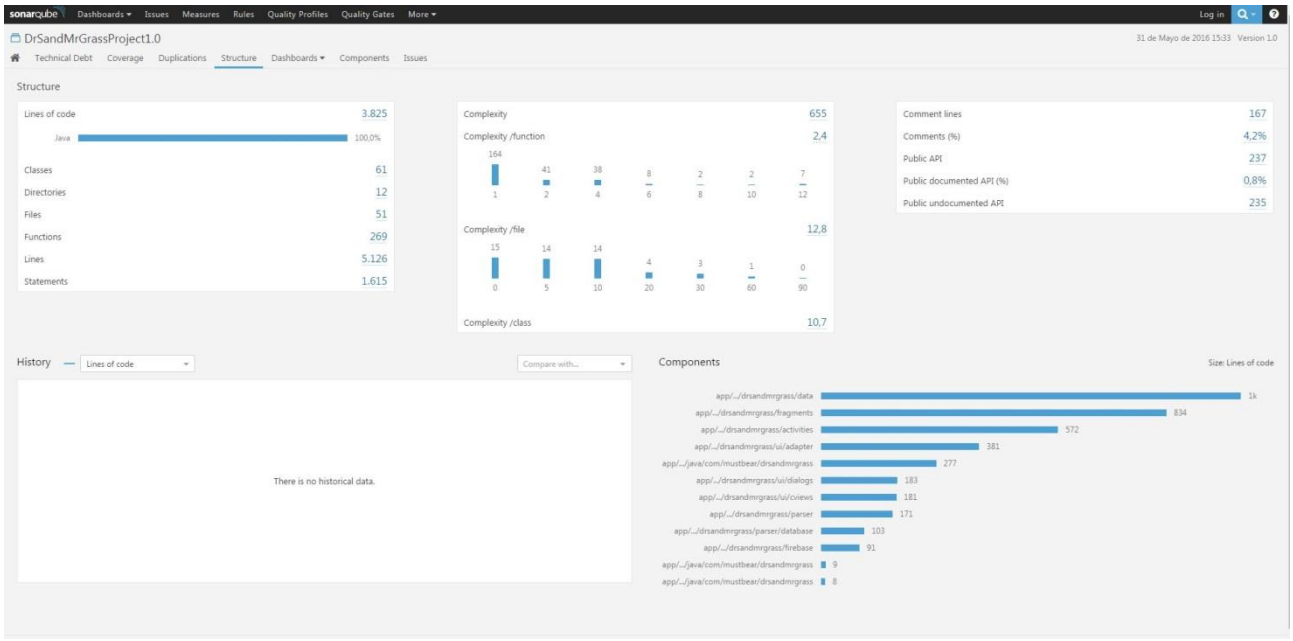


Ilustración 23 Captura código Sonar

De igual forma, Sonar ofrece la calidad del código hasta el momento, donde los archivos que destacan con código duplicado son los ficheros JSON que recibe como entrada la aplicación. La deuda del proyecto dice que se requieren aproximadamente 4 días para resolver todos los conflictos presentes actualmente en la aplicación.

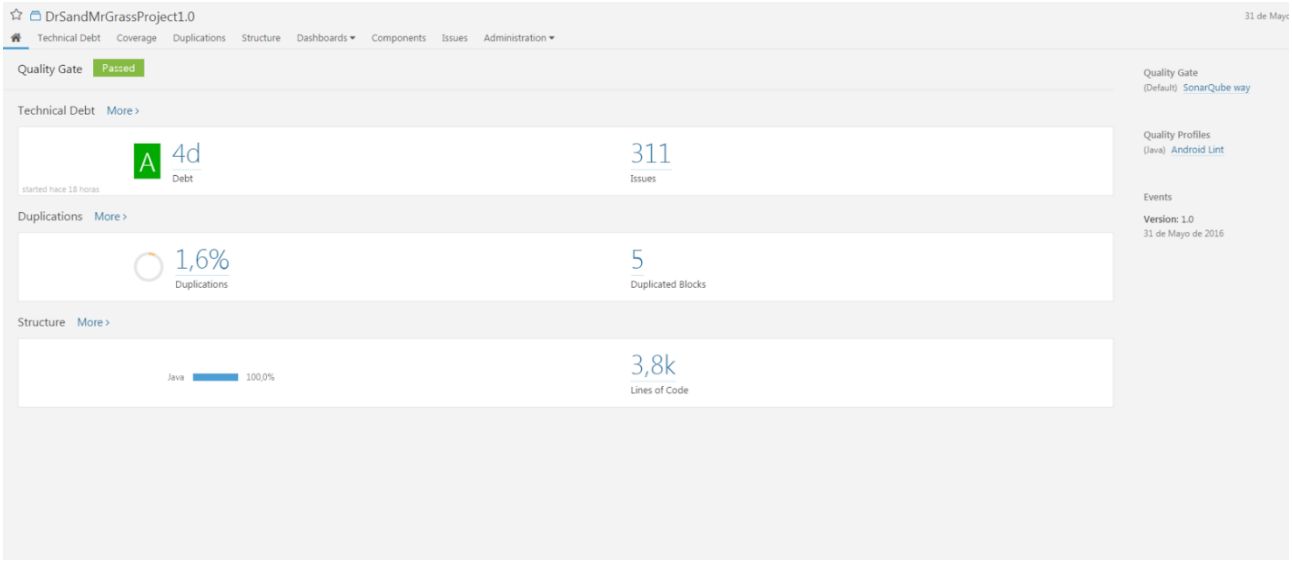
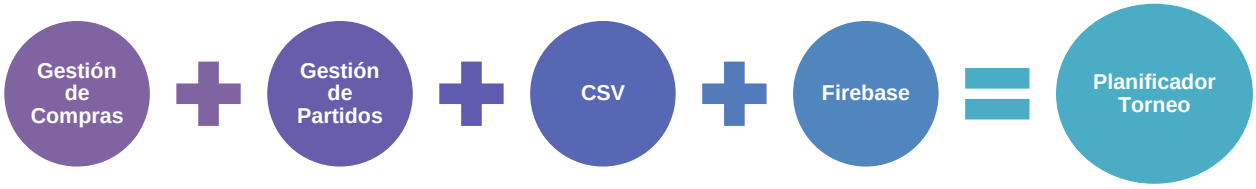


Ilustración 24 Deuda técnica Sonar

3.4 Módulos de la aplicación



Dentro del proyecto se pueden diferenciar cuatro grandes partes, dos de ellas en cuanto a funcionalidad y las otras dos de almacenamiento.

De los módulos que tratan funcionalidades está el de compras, que llevará a cabo el seguimiento y manejo de las compras que realicen los jugadores a lo largo del torneo, y el de partidos que realizará la toma de datos de los partidos que se disputen.

En cambio en los módulos de almacenamiento está por un lado el almacenamiento local que se lleva a cabo mediante el uso de los ficheros CSV, que son cargados una vez se

abre la aplicación, o en la nube llevado a cabo mediante Firebase.

3.5 Documentación

La documentación que se presenta con este informe pretende ser una guía de usuario ya que lleva a cabo un análisis conciso de todos los puntos tratados a lo largo del proyecto. La estructura que se sigue es la que siguen por defecto todos los proyectos Android facilitando así su comprensión. De igual forma el código está documentado y sigue una convención de código personal a lo largo de todo el proyecto con el objetivo de hacer un código comprensible y consistente.

Se presenta a continuación una captura del resultado de ejecutar Sonar[12] sobre el proyecto, esta herramienta permite obtener información del código duplicado, los estándares de codificación y el diseño del software. Gracias a esta herramienta se corrigieron errores entre los que destacan el uso de la herencia entre objetos como por ejemplo se ha de evitar declarar una variable de tipo ArrayList ya que hereda de List, Sonar recomienda que la variable sea declarada de tipo List y que sea en el momento de inicializarla donde se le asigne que es de tipo ArrayList. Ocurre lo mismo con el HashMap y su padre Set.

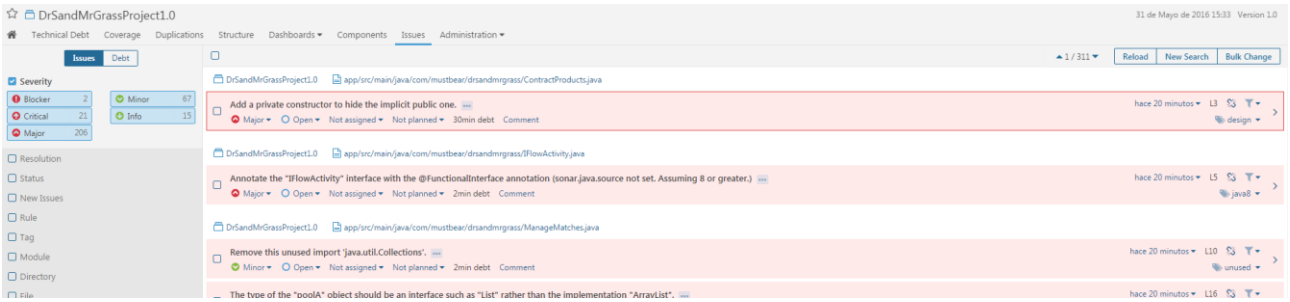


Ilustración 25 Issues Sonar

[12] Sonar: herramienta de análisis de código fuente.

Capítulo 4. Experiencia

4.1 Casos de uso

Desde que se empezase el desarrollo del proyecto detallado en este documento se ha tenido la oportunidad de llevar a cabo dos casos de uso en situaciones reales.

Con motivo de la celebración de la decimosexta edición del torneo Dr Sand & Mr Grass se usó una versión beta de esta aplicación en la que se contó con toda la funcionalidad de la gestión de compras.

El Dr Sand & Mr Grass de la edición del año presente contó con la participación de 180 personas de toda Europa a las que se les gestionaron las compras realizadas durante los 3 días del torneo. A lo largo de estos tres días se registraron más de mil movimientos sin ningún contratiempo.

Gracias a esta primera experiencia se realizaron suficientes pruebas como para afirmar que la aplicación es estable, de igual forma un punto a mejorar que expone este caso de uso es la necesidad de gestionar el almacenamiento de forma más eficiente.

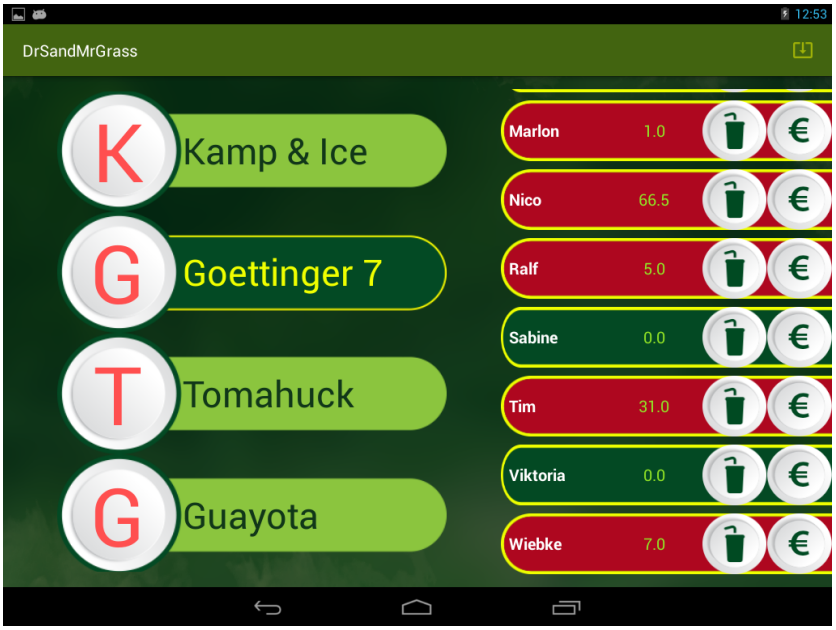


Ilustración 26 Captura del caso de uso

Con motivo de un encuentro canario en el que se reunieron 50 participantes se llevó a cabo la celebración de un torneo en el que se enfrentaron los equipos locales con equipos de Las Palmas y de Lanzarote. Aprovechando la ocasión se llevó a cabo una segunda prueba de la aplicación en una situación real, esta vez con la gestión de partidos incorporada.

Se hizo el seguimiento de 10 partidos disputados a lo largo de un fin de semana, gracias a este seguimiento se descubrieron bugs[13] entre los cuales cabe destacar que no se contemplaba que toda la aplicación estuviese en landscape[14] por lo que habían casos en los que la pantalla rotaba, el otro gran bug que pudo encontrarse fue la imposibilidad de llevar a cabo correctamente la resta de puntos en caso de que se hubiese subido un punto de forma errónea.

[13] Bugs: fallo en un programa que desencadena un resultado indeseado.
[14] Landscape: rotación de la pantalla horizontalmente.

4.2 Experiencia de usuario

Con el objetivo de tener una aplicación testeada lo máximo posible y feedback[15] de la misma a lo largo de los casos de uso mencionados previamente ésta fue usada por distintas personas relacionadas con el Ultimate Frisbee. Podemos destacar entre ellos a miembros pertenecientes a la selección española de Ultimate Frisbee que este año representan a España en el campeonato del mundo en los que se concentrarán más de 100 equipos en las distintas modalidades disponibles.

Estos miembros mencionaron la facilidad de uso de la aplicación y la ayuda que ofrece con todas las tareas que deben llevar a cabo durante la celebración de un torneo. Mencionaron también que no conocen ninguna herramienta que ofrezca la posibilidad de seguir un partido con todos los datos cargados previamente y que de los partidos seguidos puedan obtenerse estadísticas y ofrecerlas a todos los participantes.

Por la parte de gestión de compras se obtuvo un *feedback* muy positivo del que se valoró mucho que la gestión fuese automática y que se gestionase tanto las compras como el pago y la posibilidad de tener un fichero con la deuda de todos los participantes.

[15] Feedback: retroalimentación, mecanismo mediante el cual cierta proporción de la salida de un sistema se dirige a la entrada con objeto de controlar su comportamiento.

Capítulo 5. Herramientas

Se pretende detallar el conjunto de herramientas usadas a lo largo del proyecto para conseguir los objetivos planteados.

5.1 Entorno de desarrollo

5.1.1 GitLab - Control de versiones

El control de versiones es un sistema que permite registrar los cambios realizados sobre archivos a lo largo del tiempo, a lo largo del Grado de Ingeniería en Informática se ha trabajado con múltiples opciones para el control de versiones, en este caso se ha usado GitLab.

Con el fin de que el código mantuviese la calidad y persistencia a errores se realiza un commit[16] tras llevar a cabo cada funcionalidad de forma correcta, de esta forma en caso de un fallo posterior se puede revertir y volver a un estado en el que la aplicación funcione correctamente.

De igual forma se sigue la estrategia de ramificación, es decir, se crean ramas en las que habrán cambios según la funcionalidad o el objetivo que se busque con la modificación del código, se tiene entre otras una rama *development* en la que se lleva a cabo el desarrollo mientras que en la rama *release* estarán las versiones beta del proyecto.

[16] Commit: estado en el que se agregan al repositorio los cambios añadidos previamente.

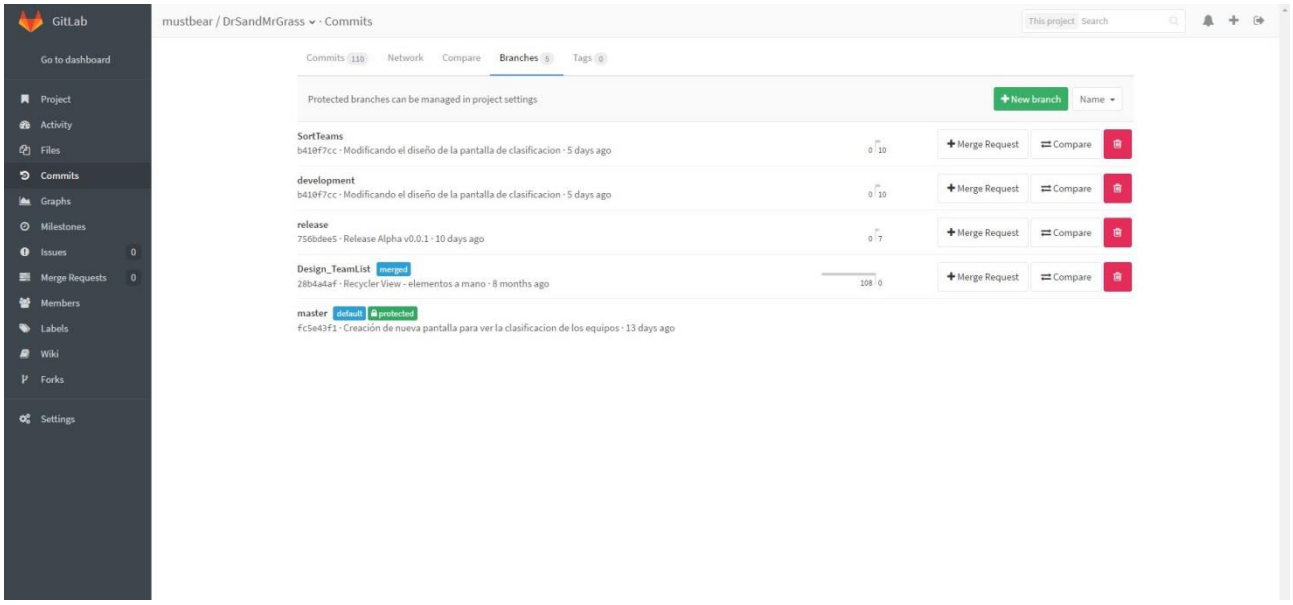


Ilustración 27 Ramas GitLab

5.2 Almacenamiento

5.2.1 Almacenamiento local

Con el objetivo de tener una aplicación estable y disponible para su testeo en el primero de los casos de uso definidos previamente se llevó a cabo la posibilidad del almacenamiento en local. Para conseguir esto se trabajó con el almacenamiento interno del dispositivo.

El módulo de las compras tiene entre sus opciones la capacidad de almacenar los datos de las compras realizadas por todos los equipos y cada jugador del equipo correspondiente. Esto se hace además con el objetivo de tratar los datos y poder ofrecer la deuda a los equipos en las que ésta exista.

Cada vez que el usuario así lo desee se almacenará y exportará en un fichero CSV que se tratará posteriormente para una visualización más sencilla para el usuario.

```
"Equipo","Goettinger 7"
"Nombre","Bebida/Snack","Dr.Beer","Cubata","Cubaton","Ron","Total","Deuda"
"Barbara","0","0","0","0","0","0.0","No"
"Esther","0","0","0","0","0","0.0","No"
"Lennart","0","0","0","0","0","0.0","No"
"Margit","0","0","0","0","0","0.0","No"
"Marie","0","0","0","0","0","0.0","No"
"Marlon","0","0","0","0","0","0.0","No"
"Nico","4","0","0","0","0","4.0","Si"
"Ralf","0","0","0","0","0","0.0","No"
"Sabine","0","0","0","0","0","0.0","No"
"Tim","0","0","0","0","0","0.0","No"
"Viktoria","0","0","0","0","0","0.0","No"
"Wiebke","1","0","0","0","0","1.0","Si"
```

Ilustración 28 CSV generado

Gracias a los conceptos básicos aprendidos sobre el tratamiento de datos y de ficheros en VBA[17] se pueden tratar los ficheros CSV obtenidos previamente, para ello se realiza un programa en VBA.

Esta aplicación ofrece la posibilidad de buscar en el explorador de archivos la carpeta que contenga los ficheros CSV, tras seleccionar el directorio extrae los ficheros CSV que contenga, los analiza y los convierte a un fichero XLSL[18] en el que se visualiza de forma ordenada una lista de los equipos con sus jugadores donde además de la deuda se especifica la compra realizada por cada uno.

En caso de que el jugador no tuviese deuda, bien porque realizó el pago o porque no tiene compra alguna, se resaltará en verde mientras que si tiene estará en rojo. Esto se ha aplicado como prototipo y como solución parcial hasta continuar con las líneas futuras y llevar a cabo el tratamiento de los datos obtenidos.

[17] VBA: **V**isual **B**asic for **A**pplications, lenguaje de macros que se utiliza para programar aplicaciones Windows y que se incluye en varias operaciones Microsoft.
[18] XLSL: extensión de archivo utilizado exclusivamente por Microsoft Excel.

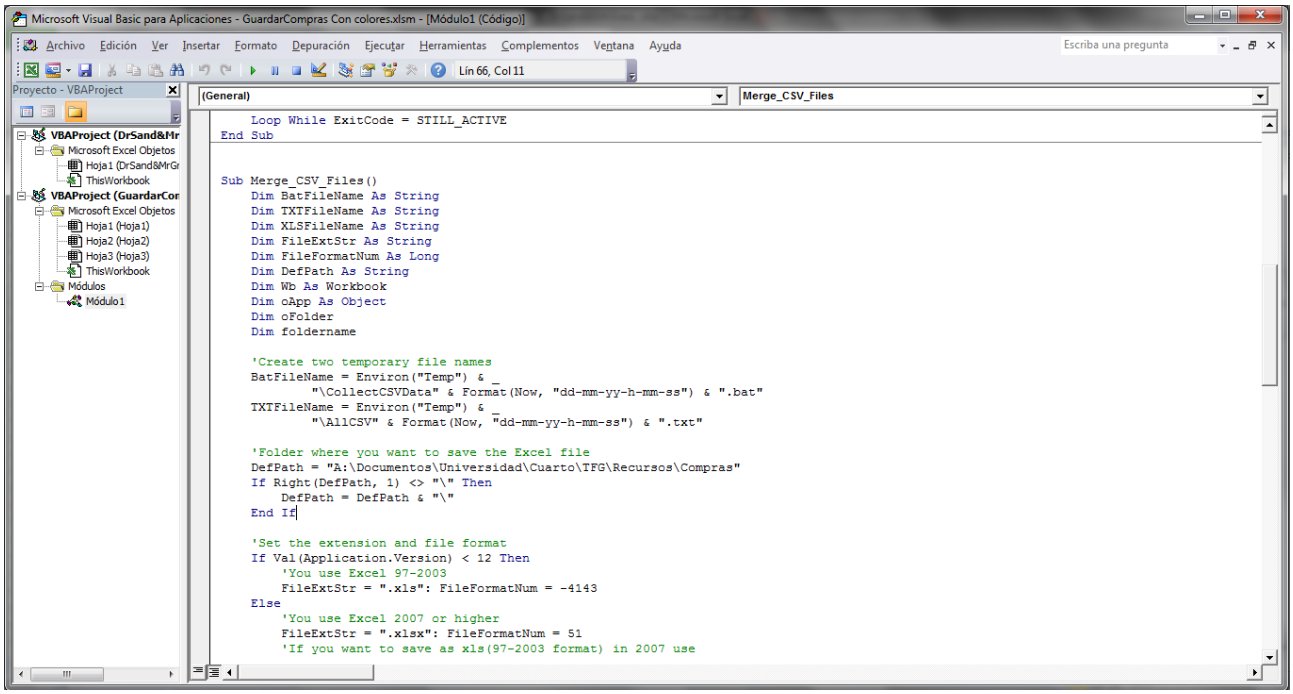


Ilustración 29 Captura código desarrollado en VBA

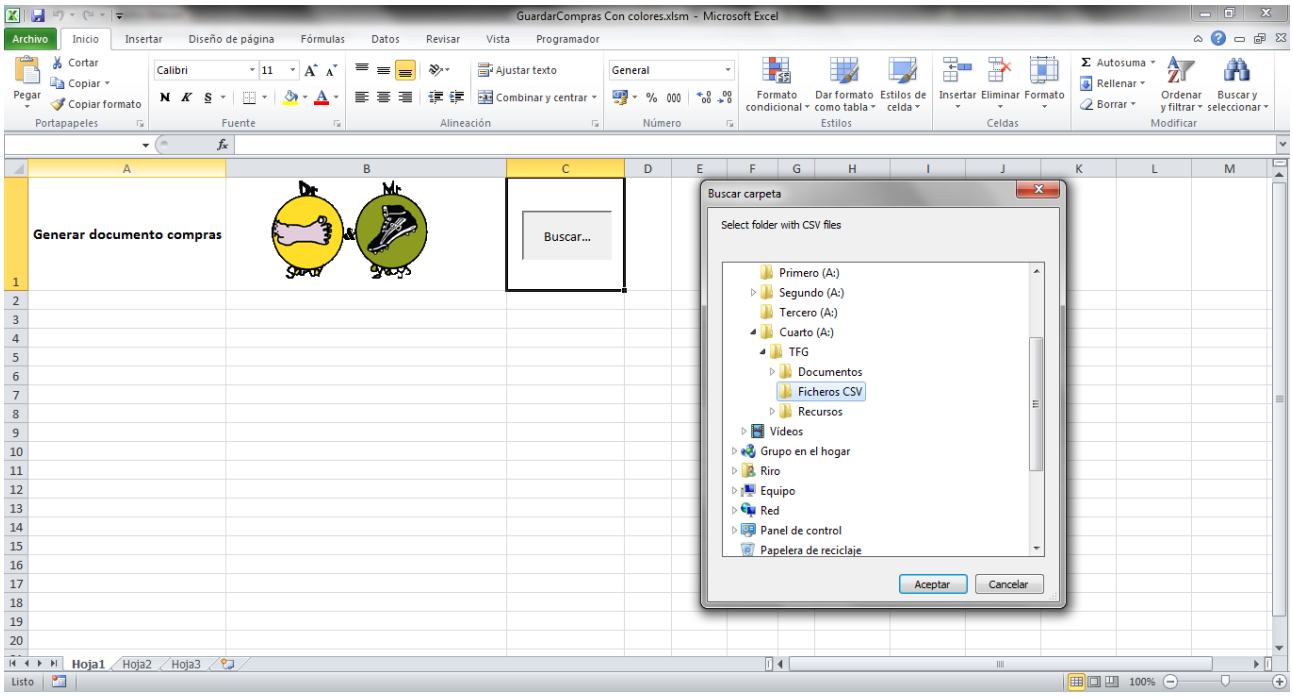
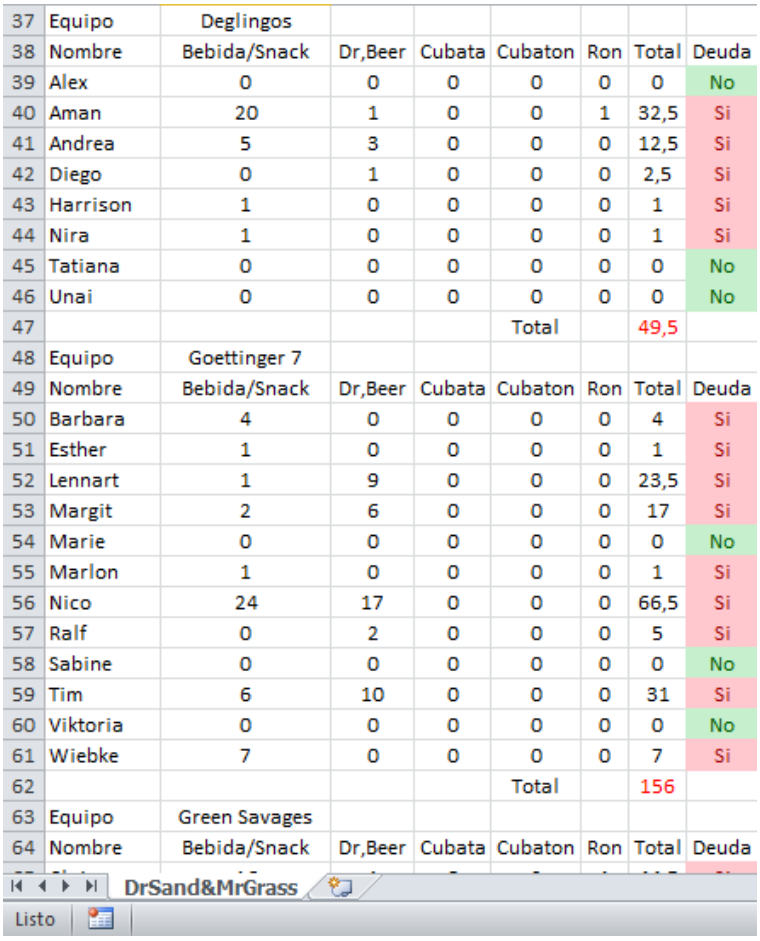


Ilustración 30 Selección directorio para importar CSV



37	Equipo	Deglingos						
38	Nombre	Bebida/Snack	Dr,Beer	Cubata	Cubaton	Ron	Total	Deuda
39	Alex	0	0	0	0	0	0	No
40	Aman	20	1	0	0	1	32,5	Si
41	Andrea	5	3	0	0	0	12,5	Si
42	Diego	0	1	0	0	0	2,5	Si
43	Harrison	1	0	0	0	0	1	Si
44	Nira	1	0	0	0	0	1	Si
45	Tatiana	0	0	0	0	0	0	No
46	Unai	0	0	0	0	0	0	No
47					Total		49,5	
48	Equipo	Goettinger 7						
49	Nombre	Bebida/Snack	Dr,Beer	Cubata	Cubaton	Ron	Total	Deuda
50	Barbara	4	0	0	0	0	4	Si
51	Esther	1	0	0	0	0	1	Si
52	Lennart	1	9	0	0	0	23,5	Si
53	Margit	2	6	0	0	0	17	Si
54	Marie	0	0	0	0	0	0	No
55	Marlon	1	0	0	0	0	1	Si
56	Nico	24	17	0	0	0	66,5	Si
57	Ralf	0	2	0	0	0	5	Si
58	Sabine	0	0	0	0	0	0	No
59	Tim	6	10	0	0	0	31	Si
60	Viktoria	0	0	0	0	0	0	No
61	Wiebke	7	0	0	0	0	7	Si
62					Total		156	
63	Equipo	Green Savages						
64	Nombre	Bebida/Snack	Dr,Beer	Cubata	Cubaton	Ron	Total	Deuda

DrSand&MrGrass

Listo

Ilustración 31 Ejemplo fichero generado

5.2.2 Servicios en la nube

El almacenamiento de la gestión de partidos no se ha tratado con ficheros ya que resulta más conveniente debido a su accesibilidad.

Tras analizar las distintas opciones las más relevantes fueron las descritas a continuación.

I.5.2.2.1 Google Cloud

Es una plataforma en la nube de Google que ofrece servidores con la misma infraestructura que usa Google internamente para sus productos, proporciona servicios a los desarrolladores para facilitar la implementación de aplicaciones sin importar su nivel de complejidad.

En el momento de llevar a cabo el análisis de las distintas alternativas para el almacenamiento en la nube Google Cloud ascendía hasta los 280€ al año

aproximadamente, esto hizo que la opción de usar Google Cloud para este proyecto quedase descartado.

I.5.2.2.2 Amazon AWS

Es una colección de servicios web que en conjunto forman una plataforma de computación en la nube ofrecidas a través de internet mediante Amazon.com y se usa en aplicaciones como Dropbox o Foursquare.

Amazon AWS ofrece bonos para personal educativo, en el que se incluyen estudiantes y docentes que permiten el uso de un servidor de forma gratuita o precios asequibles.

Este servicio proporciona un servidor y la posibilidad de acceder a través de HTTP mediante los protocolos REST[19] y SOAP[20], esto implica la necesidad de bien implementar una API que se comunique y así realizar el tratamiento de datos o llevar a cabo el pago de una API ofrecida por Amazon. Debido a la complejidad de implementar una API y al coste que implicaría usar la proporcionada por Amazon esta opción fue descartada.

I.5.2.2.3 Firebase

Es un proveedor de servicios en la nube que ofrece también servicios de backend. La mayor ventaja que ofrece Firebase es la simplicidad para almacenar y sincronizar los datos en la nube. En la actualización más reciente Firebase y Cloud Google se unieron ofreciendo un servicio mucho más completo y asequible.

Gracias a esta fusión Firebase no se limita al almacenamiento y manejo de datos sino que ofrece Firebase Analytics que es una herramienta de seguimiento y estadísticas desde dónde hacen click los usuarios hasta los momentos en que la aplicación deja de funcionar. La

[19] REST: **R**epresentational **S**tate **T**ransfer, es una interfaz entre sistemas que usa HTTP para obtener datos o indicar la ejecución de operaciones sobre los datos.

[20] SOAP: **S**imple **O**bject **A**ccess **P**rotocol, protocolo estándar que define como dos objetos diferentes pueden comunicarse mediante el intercambio de datos XML.

principal necesidad de esta tecnología es la potencial escalabilidad que proporciona.

Firebase guarda los datos como archivos JSON así que se ha estudiado la mejor forma de almacenaje para poder leer los datos de una manera consistente. Se ha estructurado, con un archivo JSON con tres nodos, de la siguiente manera:

Un atributo 'data' que contiene la información base de la aplicación, productos que están disponibles, equipos, estadísticas de los jugadores.

Otro atributo 'matches' que es el encargado de tener toda la información respecto a los partidos, cuales están jugados, cuales quedan pendientes y cuales han sido los resultados.

Por último un nodo para las opciones de la aplicación, es decir, si están cargados en servidor o no y variables de estado.

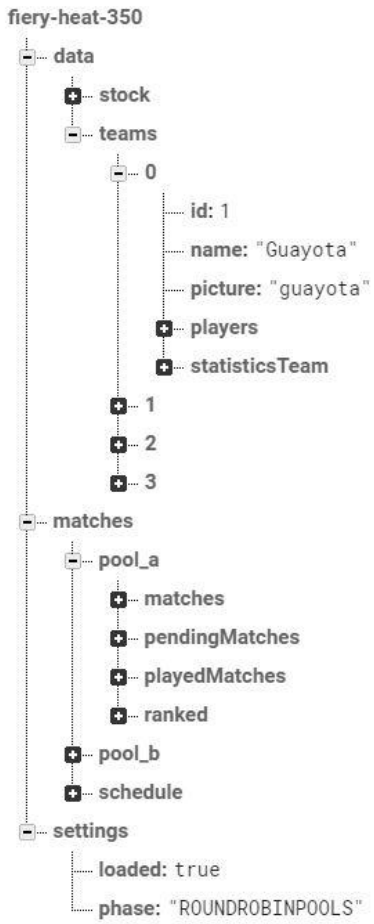


Ilustración 32 Estructura Firebase

5.3 Trello

Tal y como se mencionó previamente con el objetivo de seguir la metodología se estudió la posibilidad del uso de herramientas que facilitasen esta tarea y se eligió Trello.

Trello permite asignar un tablero para el proyecto, dentro de este tablero se crean listas con las distintas funcionalidades o sprints a realizar. Las listas a su vez están formadas por tarjetas, cada tarjeta representa una tarea del conjunto de tareas que forman un sprint.

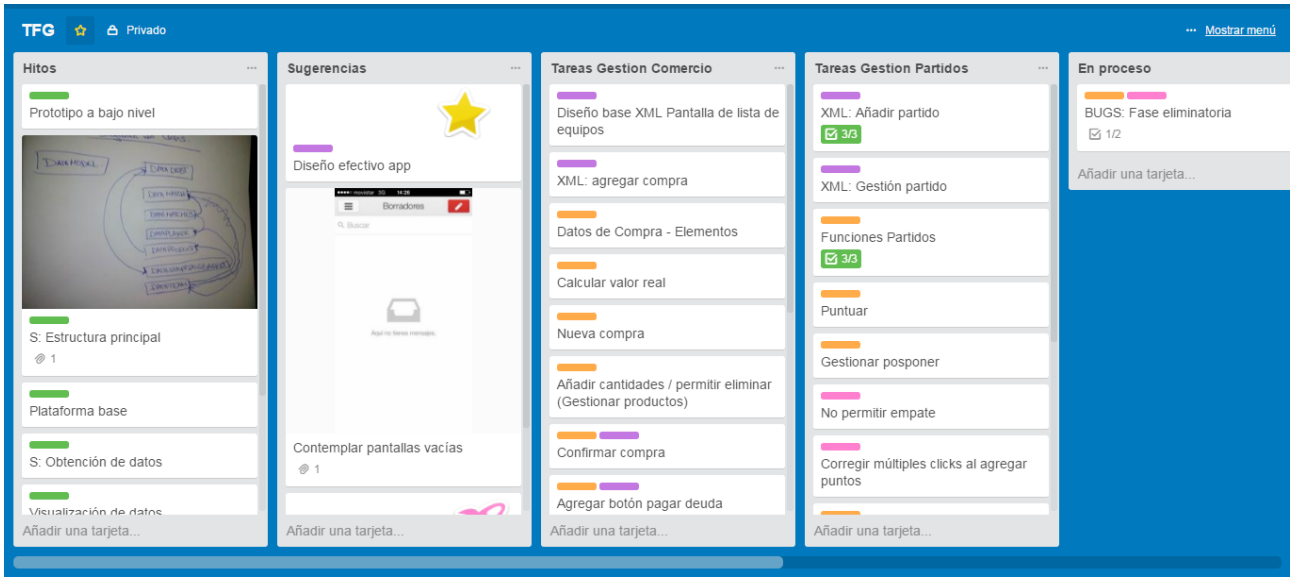


Ilustración 33 Tablero del Proyecto en Trello

Capítulo 6. Conclusiones y líneas futuras

Gracias a la oportunidad de haber realizado distintos casos de uso la visión de futuro de este proyecto crece a medida que se desarrolla. Esto solo es posible gracias a la fase de análisis realizada ya que hizo posible formar una estructura base consistente, que permite trabajar de forma modular y que hace que no se requiera empezar de cero en caso de querer modificar alguna de las funcionalidades existentes.

Esto hace que el abanico de posibilidades sea muy amplio, en base a los partidos y a corto plazo está el uso de la aplicación de forma simultánea mediante múltiples dispositivos ya que la base para conseguirlo está implementada y solo se requiere el controlar Firebase para que permita estas operaciones. También destaca el almacenamiento con una base local de todos los datos y la gestión de llevar a cabo la sincronización de los datos en local con los que se encuentren en Firebase.

Uno de los objetivos para el futuro es también llevar a cabo la implementación de test unitarios para el proyecto y de esta forma simplificar la integración y conseguir una aplicación estable. A su vez se pretende saldar la deuda técnica presente hasta el momento y gracias a Sonar se puede realizar el seguimiento de esta tarea.

De igual forma la línea futura clave y que incrementaría la magnitud del proyecto es la proyección de todos los datos obtenidos y con ello estadísticas tanto para el usuario como para todos los participantes del torneo. Esto se puede enfocar de múltiples formas ya que existe la opción de desplegarlos mediante la página web del equipo encargado del torneo, en este caso Guayota. Otra alternativa es la difusión de los datos

mediante redes sociales o incluso la creación de una aplicación multiplataforma en la que se desplieguen todos los datos y estadísticas tales como pueden ser los máximos goleadores, los equipos que menos puntos reciben, entre otros.

Este proyecto ofrece la posibilidad de involucrarse en una de las tecnologías que mayor crecimiento ha tenido y sigue teniendo actualmente, Android, éste es uno de los motivos que hizo este proyecto tan grande y exhaustivo. Gracias a esto he conocido e intentado seguir nuevas metodologías y tecnologías que se usan actualmente para un buen desarrollo. Cabe destacar de igual manera, que es un proyecto social, que pretende facilitar y automatizar tareas esto significa un gran aporte para la comunidad del Ultimate Frisbee y más especificamente para los organizadores del torneo Dr. Sand & Mr. Grass.

Finalmente solo cabe esperar que este documento haya permitido profundizar el conocimiento del sistema operativo Android y el desarrollo de sus aplicaciones, también que permitiese conocer el que es considerado el deporte más limpio del mundo: Ultimate Frisbee, de igual forma animar a investigar más sobre el tema.

Capítulo 7.
Summary and Conclusions

The analysis phase realized on this project and the chance to get some user experience makes that this application gets many options to grow in different ways. Making the application able to manage multiple matches at the same time would constitute a clear goal in the near future, Firebase makes this possible and the project only needs to set it up to make it since the data model to get this done has the development already set.

Another important goal is to be able and run unit tests for it since it would simplify continuos integration and would make it an stable application. This project also aims to pay it technical debt which Sonar allows to track while it gets satisfied. Anyhow the key to this application will be to be able to provide information and statistics about the managament done during the tournament. This might be done with a website or even a new application to provide information about the top scorers, the matches result and the top scorers teams.

Because of this project I had the chance to study and deepen my understanding of Android and how it works. This also gave me the chance to learn new methodologies and tools for a best code and good development. This project also looks forward to help people and spread the word about the fairest sport as it is Ultimate Frisbee.

Capítulo 8. Presupuesto

8.1 Coste del Proyecto

Llevar a cabo este proyecto se ha dividido en fases tal y como se ha descrito previamente, este proceso ha abarcado seis meses de trabajo en los cuales se han llevado a cabo dichas fases con las horas detalladas a continuación junto con el coste económico donde la remuneración asciende a 15€/hora.

Fases	Duración (h)	Coste(€)
Análisis	90	1350
Desarrollo	30	450
Implementación	250	3750
Total	370	5550

Tabla 2 Coste del Proyecto

8.2 Coste de Explotación

Para una experiencia completa de la aplicación es necesario un tablet que tenga como sistema operativo Android de gama media con el fin de tener un funcionamiento fluido y además la contratación de Firebase como base de datos, cuya cuota mensual ronda los 23€, a continuación se estipula su coste para una contratación anual.

Elemento	Coste(€)
Tablet de gama media	150
Almacenamiento Firebase	275
Total	425

Tabla 3 Coste de Explotación

Apéndice A.
Enlaces de Interés

A.1. Repositorio de Código del Proyecto

https://gitlab.com/mustbear/TFG-Gestion_Torneo/

A.2. Diseño de Mockups funcional

http://ninjamock.com/s/HRTDD

A.3. Pantallazos a tamaño completo

https://drive.google.com/folderview?id=0B5x6ASFUooTiMERycGFVZFNfMlE&usp=sharing

A.4. Diagramas a tamaño completo

https://drive.google.com/open?id=0B5x6ASFUooTicDQ3amp3WUhpc2s

A.5. Capturas calidad de código a tamaño completo

https://drive.google.com/open?id=0B5x6ASFUooTiNmotOXhrX3lJVlU

Bibliografía

[1] **Ultimate Frisbee.**

https://es.wikipedia.org/wiki/Ultimate_(deporte)

[2] **UltiAnalytics Frisbee Stats:**
https://play.google.com/store/apps/details?id=com.summithillsoftware.ultimate&hl=es

[3] **Organizar torneo, crear liga:**
https://play.google.com/store/apps/details?id=com.mileyenda.manager&hl=es

[4] **NBA app:**
https://play.google.com/store/apps/details?id=com.nbaimd.gametime.nba2011&hl=es

[5] **Agile Development Guide.**

http://www.i2btech.com/blog-i2b/tech-deployment/para-que-sirve-el-scrum-en-la-metogologia-agil/

[6] **SDK Android.**

https://developer.android.com/studio/index.html

[7] **Android JSON Parsing Tutorial.**
http://www.androidhive.info/2012/01/android-json-parsing-tutorial/

[8] **OpenCSV parse Library.**

http://opencsv.sourceforge.net

[9] **Firebase.**

https://firebase.google.com/

https://firebase.google.com/docs/android/setup

[10] **Documentación ViewPager.**
https://amatellanes.wordpress.com/2013/05/25/android-ejemplo-de-viewpager-en-android-parte-1/

[11] **Android Development Guide - Fragments**
http://www.sgoliver.net/blog/curso-de-programacion-android/indice-de-contenidos/