UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

Graduado en Ingeniería del Software

SportU: Aplicación móvil para fomentar el deporte y la socialización en ciudades inteligentes

SportU: Mobile application for promoting sport and socialization in smart cities

Realizado por
Lorenzo Toro Gálvez

Tutorizado por
José Carlos Canal Velasco

Departamento
Lenguajes y Ciencias de la Computación
UNIVERSIDAD DE MÁLAGA

MÁLAGA, septiembre de 2022

UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADUADO EN INGENIERÍA DEL SOFTWARE

SportU: Aplicación móvil para fomentar el deporte y la socialización en ciudades inteligentes

SportU: Mobile application for promoting sport and socialization in smart cities

Realizado por
Lorenzo Toro Gálvez

Tutorizado por
José Carlos Canal Velasco

Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA
MÁLAGA, SEPTIEMBRE DE 2022

Fecha defensa: octubre de 2022

Resumen

En la actualidad, las grandes empresas se lucran gracias a los datos recopilados de sus usuarios, convirtiéndose en propietarias de estos y empleándolos para su beneficio propio. De este modo, dichos usuarios no disponen de control sobre los mismos y, por tanto, no tienen la capacidad de decidir para qué fines va a ser utilizada su información.

Este Trabajo de Fin de Grado pretende centrarse en el modelo de computación social distribuida para dar solución al problema expuesto, enfocándose así en el empoderamiento de los usuarios de sus propios datos. Para abordar este modelo, se va a desarrollar una aplicación móvil centrada en la realización de actividad deportiva en ciudades inteligentes, como Málaga.

La aplicación móvil ofrece a los usuarios disponer de un perfil personal únicamente almacenado en su teléfono móvil. Los usuarios también pueden consultar las instalaciones deportivas de Málaga y su afluencia en tiempo real, y registrar entrenamientos tras pasar un determinado tiempo en la ubicación correspondiente ejercitándose. Del mismo modo, un usuario puede enviar solicitudes de amistad a otros, convirtiéndose en amigos y pudiendo acceder así a su perfil y a toda su actividad deportiva. Además, SportU anima a combatir el sedentarismo en la población ya que los usuarios ganan puntos y compiten al realizar actividad física.

Palabras clave: avatar digital, computación móvil, sedentarismo, actividad deportiva, SportU

Abstract

Nowadays, large companies profit from the data collected from their users becoming owners of it and using it for their own benefit. In this way, these users don't have control over the data and, therefore, do not have the capability to decide for what purposes their information will be used.

This Final Degree Project aims to focus on the distributed social computing model as a solution to this problem, focusing on users empowerment of their own data. To deal with this model, we will develop a mobile application focused on carrying out sports activities in smart cities, such as Malaga.

The mobile application offers users a personal profile stored only on their mobile phone. Users can also check Malaga's sports facilities and their real time influx, and record workouts after spending time in a certain location exercising. In the same way, a user can send friend requests to others, becoming friends and being able to access their profile and all their sport activity recorded. In addition, SportU encourages the fight against sedentary lifestyle since users earn points and compete by being physically active.

Keywords: digital avatar, mobile computing, sedentary lifestyle, sport activity, SportU

Índice

Índice de figuras

Índice de tablas

1 Introducción

En este capítulo se describen las razones que han motivado a llevar a cabo este proyecto, los objetivos que se están persiguiendo a través de este, y se describe la estructura de este documento, correspondiente a la memoria del TFG, al mismo tiempo que se enumeran los entregables de este trabajo.

1.1. Motivación

Hoy por hoy, la información de las personas tiene un valor inimaginable para las empresas, ya que se benefician de sus datos sin recibir nada a cambio, creando perfiles digitales que intentan mejorar la experiencia de los usuarios, ofreciéndoles campañas de anuncios personalizadas, sugerencias en base a distintos criterios, venta o intercambio de datos sensibles con otras grandes empresas, etc.

Por este motivo, existe el modelo Digital Avatars [1], basado en Personas como Servicio, o *People as a Service* (PeaaS), que se beneficia de la extendida presencia de teléfonos inteligentes y sus distintos sensores para recopilar información sobre sus usuarios. Así, se puede inferir y explotar un perfil virtual representante del usuario propietario del dispositivo. Dicho perfil se almacena localmente en el teléfono y se emplea para identificarse ante cualquier otro dispositivo de alrededor o cualquier petición.

Con este planteamiento se pretende aprovechar la información recopilada generando un representante virtual del usuario, sin necesidad de estar almacenado en un servidor centralizado, promoviendo la computación social y móvil distribuida, de modo que se esquive la preponderancia de las grandes empresas y siendo el usuario el único dueño de todos sus datos personales, decidiendo con quién los comparte y con quién no.

Teniendo este modelo en consideración, el siguiente paso consiste en elegir cuál es el problema que va a solucionar la aplicación: el sedentarismo en la sociedad. Estos últimos años se

ha ido observando cómo más personas comienzan a realizar una actividad deportiva, aunque el número de personas sedentarias sigue siendo elevado, debido a largas jornadas delante de la televisión, trabajando frente al ordenador, evadiendo la realización de cualquier tipo de actividad física con regularidad... Por si fuera poco, la pandemia también ha sido responsable de este estilo de vida sedentario, ya que algunas personas han normalizado y se han acostumbrado a salir de casa para únicamente aquello indispensable, ni siquiera salen para ejercitarse.

Por tanto, para motivar a la población malagueña a la realización de actividad física y recibir reconocimiento, se plantea la aplicación SportU, mediante la cual un usuario podrá registrar su actividad deportiva haciendo uso de su teléfono móvil y, además, obtendrá una recompensa por ello, compitiendo y socializando con otros usuarios.

1.2. Objetivos

Para poder incentivar a los usuarios a moverse y ejercitarse en la ciudad de Málaga, se desarrolla SportU, una aplicación móvil para Android donde se establecen unas funcionalidades a cubrir.

En primer lugar, un usuario podrá crear y gestionar un perfil indicando sus datos y sus intereses deportivos. Dicho perfil será almacenado localmente en su teléfono móvil, y será compartido con otro usuario interesado cuando se establezca una relación de amistad entre ambos. En caso de que no exista relación de amistad, solo podrá conocerse la información básica identificativa de un usuario.

Para poder brindar a los usuarios un conjunto fiable de instalaciones deportivas, se hará uso de los datos abiertos del Ayuntamiento de Málaga. En concreto, se obtendrá el nombre, la dirección y las coordenadas de las siguientes instalaciones: aparatos de *workout* [2], pistas de atletismo [3], centros deportivos [4], rutas de senderismo [5] y zonas de musculación al aire libre [6]. Además, se incorporarán algunos gimnasios privados y el Complejo Deportivo Universitario de la UMA. Todas estas instalaciones podrán ser consultadas por los usuarios a través de un listado y un mapa con marcadores.

Gracias a los sensores Bluetooth y GPS del teléfono móvil, los usuarios podrán conocer la afluencia en tiempo real en las instalaciones deportivas. Si se trata de un centro deportivo cerrado o de una actividad en interior, donde la señal GPS puede fallar, el control de la ubicación se realizará a través de Bluetooth, que será el encargado de detectar los *beacons*, dispositivos

explicados en el siguiente capítulo. De manera similar, si se trata de una actividad al aire libre, se hará uso de la señal GPS.

En cuanto a la actividad física, los usuarios podrán registrar cuándo han realizado un entrenamiento y anotar su información relevante, siempre que el teléfono móvil haya detectado que el usuario ha estado en la instalación deportiva indicada.

Por último, para fomentar el espíritu competitivo, se otorgarán incentivos a aquellos que realicen actividad física, en función del tiempo que dediquen. Además, existirá un *ranking* en el que se podrá observar la posición de cada uno de los usuarios. Por otro lado, para fomentar la interacción social, un usuario podrá ver cuánta compatibilidad tiene con otros usuarios en base a sus intereses deportivos.

1.3. Estructura del documento

Esta sección trata la estructura de esta memoria, la cual recoge las diferentes fases que han sido necesarias para completar el proyecto en su totalidad. Así, la memoria está organizada en varios capítulos.

1) **Introducción**: En este capítulo se plantean las razones que motivan a llevar a cabo este proyecto, es decir, el porqué, explicando para ello sus objetivos. Además, se añade la estructura del documento y una explicación de los apéndices.

2) **Tecnologías y herramientas**: Recoge todas las tecnologías usadas para elaborar el proyecto de principio a fin: lenguajes de programación, bases de datos, herramientas de desarrollo, de gestión, de documentación, de diseño y otras adicionales.

3) **Metodología**: Dicho capítulo aclara cuál es la metodología de trabajo, cómo se gestiona y una breve explicación de las iteraciones realizadas.

4) **Análisis**: Aquí se plasman todas las funcionalidades y características de la aplicación a través de los requisitos funcionales, no funcionales y los casos de uso correspondientes.

5) **Modelado y diseño**: Este capítulo contempla la arquitectura, las distintas bases de datos, algunos aspectos de diseño, las maquetas de las interfaces y el diagrama de navegación de la aplicación.

6) **Desarrollo e implementación**: En él se menciona todo el proceso de desarrollo seguido, tras dividir en iteraciones las distintas tareas a realizar. Se explica detalladamente las decisiones de implementación y cómo se han desarrollado ciertas funcionalidades.

7) **Conclusiones y líneas futuras**: Este último capítulo aborda las conclusiones finales obtenidas tras realizar este proyecto y se exponen distintas mejoras a considerar para el futuro.

En cuanto a los apéndices, se incluyen dos: el manual de usuario y el manual de instalación y despliegue.

- **Manual de usuario**: Explica el funcionamiento de la aplicación para que un usuario que no haya tenido contacto previo con esta conozca cómo utilizarla en su totalidad.
- **Manual de instalación y despliegue**: Recoge los pasos a seguir poder instalar la aplicación. Además, refiere a la web para configuración de los *beacons*.

Cabe destacar que no se va a proceder con la entrega de un apéndice referente a Documento General de Requisitos (DGR) tal y como se especificó en el anteproyecto, debido a que el contenido más relevante y fundamental de este, es el que se aborda en el capítulo 4 de esta memoria.

2

Tecnologías y herramientas

2.1. Lenguajes de programación

2.1.1. Java

Figura 1: Logo de Java

Java [7] es uno de los lenguajes de programación orientado a objetos más populares. Se trata de un lenguaje imperativo y multiplataforma ejecutable en multitud de dispositivos, muy utilizado en el desarrollo de aplicaciones web y móviles, sistemas operativos y distintos tipos de software.

En este proyecto, Java ha sido el lenguaje empleado para desarrollar la aplicación móvil para Android, existiendo también la posibilidad de haber utilizado Kotlin, pero ha sido descartado al no poseer experiencia en éste y no disponer de todo el tiempo necesario para su correcto aprendizaje.

2.1.2. Python

Python [8] es un lenguaje interpretado multiparadigma de alto nivel, que soporta programación imperativa, parcialmente la programación orientada a objetos, y en menor medida, la programación funcional.

Este sencillo e intuitivo lenguaje, es muy utilizado en cualquier tipo de proyecto: desarrollo web, aprendizaje automático, ciencia de datos, robótica, entre otros; con multitud de tecnologías y librerías que se llevan bien con Python y facilitan el desarrollo.

Figura 2: Logo de Python

Se ha hecho uso de Python en este proyecto para procesar los datos abier-

tos facilitados por el Ayuntamiento de Málaga, almacenándolos posteriormente en una base de datos en la nube evitando así la realización de múltiples consultas a la web de datos abiertos del Ayuntamiento cada vez que se desee obtener dicha información.

2.2. Tecnologías de bases de datos

2.2.1. Couchbase Lite

Figura 3: Logo de CB Lite

Couchbase Lite [9] es una base de datos no relacional (NoSQL), con estilo de documentos JSON, e integrada para dispositivos móviles.

Posee la capacidad de escribir consultas usando la semántica de SQL, lo cual es una ventaja para usuarios no experimentados con este tipo de bases de datos no relacionales.

Puede ser usada como base de datos embebida en el dispositivo de manera local, o mediante sincronización con un servidor disponiendo de los datos en la nube. Para llevar a cabo este proyecto, se ha optado por la primera opción, para poder respetar el modelo de computación social distribuida al máximo dentro de nuestras posibilidades y así almacenar el representante digital del usuario en su dispositivo.

2.2.2. MongoDB

MongoDB [10] es un gestor de bases de datos no relacionales (NoSQL), que emplea documentos con estructura de BSON para almacenar la información deseada, y dispone de una gran flexibilidad, escalabilidad y sistema de consultas de gran potencia.

Figura 4: Logo de MongoDB

Es un sistema de bases de datos que puede ser utilizado en más de diez lenguajes de programación, empleando distintos adaptadores para ello, permitiendo a los desarrolladores satisfacer todas las necesidades a cualquier escala.

En el caso que nos concierne, MongoDB ha sido utilizado junto a Python para almacenar los datos abiertos que el Ayuntamiento de Málaga proporciona acerca de sus instalaciones deportivas: aparatos de *workout*, pistas de atletismo, centros deportivos, rutas de senderismo y zonas de musculación al aire libre.

2.2.3. Firebase Realtime Database

Figura 5: Logo de Realtime Database

Realtime Database [11] es una base de datos NoSQL alojada en la nube, proporcionada por Firebase. Los datos son almacenados en JSON y, como su nombre bien indica, son sincronizados en tiempo real y se mantienen disponibles en los dispositivos de todos los usuarios cuando la aplicación no dispone de una conexión a Internet activa.

La sincronización de los datos es clave en esta tecnología, evitando así tener que realizar peticiones HTTP constantemente. Realtime Database posee un lenguaje de reglas basado en expresiones, para definir cómo se estructuran los datos y cuándo pueden leerse o escribirse.

Ha sido importante el uso de esta base de datos en el proyecto para permitir la fácil integración con Authentication, de modo que se puede consultar en tiempo real todos los usuarios que se registran en la aplicación.

2.3. Herramientas de desarrollo

2.3.1. Android Studio

Android Studio [12] es un entorno de desarrollo integrado (IDE) ofrecido de manera oficial para llevar a cabo el desarrollo de aplicaciones para Android, basado en IntelliJ IDEA.

Figura 6: Logo de Android Studio

Proporciona multitud de funciones como la compilación flexible basada en Gradle, integración con sistemas de control de versiones como GitHub, compatibilidad e integración con los servicios de Firebase, incluyendo ejemplos de uso y de instalación de dependencias, y la creación de distintos emuladores de dispositivos para probar la aplicación desarrollada en estos.

Debido a la amplia funcionalidad y características que posee este IDE, se ha elegido para llevar a cabo el desarrollo de la aplicación Android.

2.3.2. Visual Studio Code

Figura 7: Logo de Visual Studio Code

Visual Studio Code [13] es un editor de código fuente mundialmente reconocido y utilizado. Destaca por poseer soporte y compatibilidad con la mayoría de los lenguajes existentes en la actualidad.

La existencia de plugins y complementos facilitan la utilización de este editor, ya que permiten al desarrollador realizar sus tareas con mayor rapidez y sencillez.

Esta herramienta se incorpora al proyecto para escribir el código Python encargado de descargar y almacenar los datos ofrecidos por el Ayuntamiento de Málaga. Se hace uso de este editor debido a estar familiarizado con él y poder desarrollar el código sin necesidad de construir proyectos complejos.

2.3.3. Firebase Cloud Messaging

Cloud Messaging [14] de Firebase es un sistema de mensajería multiplataforma que permite enviar mensajes seguros y sin coste alguno.

Figura 8: Logo de Cloud Messaging

Estos mensajes son realmente notificaciones *push*, un tipo de notificaciones que se envían desde un servidor o desde un dispositivo, hacia otros dispositivos que tienen instalada la aplicación.

Con Cloud Messaging se puede notificar a los usuarios de la aplicación de distintas formas: enviando notificaciones *push* a todos sin realizar distinción, enviando notificaciones a cierto usuario empleando para ello un *token* asociado a este que sirve como identificador único, o bien, realizando el envío de notificaciones a aquellos usuarios que estén suscritos a un tema.

Esta solución de mensajería ha sido integrada en la aplicación para realizar el envío de notificaciones *push* entre los distintos dispositivos.

Dichas notificaciones contienen datos en función de la petición que se esté realizando en ese momento, por ejemplo, enviar una solicitud de amistad, pero esto será explicado en capítulos posteriores.

2.3.4. Firebase Authentication

Figura 9: Logo de Authentication

Authentication [15] de Firebase proporciona servicios para identificar a los usuarios dentro de cualquier aplicación. Permite gestionar el registro e inicio de sesión, empleando cualquier proveedor de una multitud ofrecida: número de teléfono, Google, Facebook, Twitter, Yahoo, entre otros.

Al registrarse un usuario en una aplicación integrada con Authentication, se almacena en una lista de usuarios junto a sus datos, asignándole un identificador único.

Este servicio ha sido integrado en la aplicación para gestionar la identificación de los usuarios, siendo Google el proveedor elegido, significando esto que el usuario que desee utilizar la aplicación deberá disponer de una cuenta de Google, o bien, crear una.

2.3.5. Moon Modeler

Moon Modeler [16] es una herramienta que permite modelar bases de datos, tanto SQL como NoSQL: MongoDB, Mongoose, MySQL, MariaDB, entre otros.

Esta herramienta ofrece funcionalidades como dibujar diagramas de modelos de datos rápidamente, realizar ingeniería inversa a partir de bases de datos ya existentes, generar *scripts* para crear esquemas, etcétera.

Figura 10: Logo de Moon Modeler

En el proyecto se ha empleado Moon Modeler para establecer una idea inicial de algunos modelos de datos de la aplicación, como puede ser la estructura del representante virtual del usuario que se almacena localmente en su dispositivo móvil.

2.3.6. MongoDB Compass

MongoDB Compass [17] es una herramienta gráfica e interactiva que permite hacer operaciones con los datos de una base de datos MongoDB: consultas, optimizaciones y análisis.

Figura 11: Logo de Compass

Es una alternativa a gestionar las bases de datos a través de una terminal, pues esta herramienta ofrece una interfaz que facilita el uso y consulta de los datos.

Esta herramienta ha sido empleada para consultar visualmente y de manera sencilla los datos almacenados del Ayuntamiento de Málaga, tras haberlos procesado y almacenado haciendo uso del lenguaje de programación Python comentando anteriormente.

2.4. Herramientas de gestión

2.4.1. Git/GitHub

Git [18] es un sistema de control de versiones distribuido. Es de código abierto y su uso es gratuito, empleado para gestionar y llevar un control de todo tipo de proyectos. Permite realizar un registro de todas las versiones que se obtienen a lo largo del desarrollo, pudiendo restaurar una versión anterior o creando diferentes ramas para realizar otras tareas.

Figura 12: Logos de Git y GitHub

Este sistema de control de versiones se usará junto a GitHub [19], un repositorio en línea para alojar proyectos, que conecta a los desarrolladores dentro de una comunidad en la que se puede compartir código de cualquier lenguaje de programación.

Durante el desarrollo de la aplicación estas herramientas han sido utilizadas para llevar un registro de las tareas completadas, obteniendo diferentes versiones y permitiendo la monitorización de cambios, así como la restauración a un punto anterior en caso de contemplar problemas en la aplicación.

2.4.2. Google Drive

Google Drive [20] es la plataforma por excelencia para alojar todo tipo de archivos en línea de manera segura, pudiendo gestionarlos y compartirlos con cualquier colaborador y con fácil acceso desde cualquier tipo de dispositivo.

Figura 13: Logo de Google Drive

En el proyecto se ha empleado Drive para alojar diferentes archivos: documentos de texto con anotaciones realizadas durante las reuniones con el tutor, documentos borradores de la memoria, imágenes en carpetas para almacenar capturas de pantalla tomadas desde el móvil, diagramas de navegación y de componentes de la aplicación, documentos con la extracción de requisitos, imágenes de las tecnologías empleadas, entre otros.

2.4.3. Google Meet

Google Meet [21] es un servicio de videollamadas seguras disponible para todos los usuarios y accesible desde cualquier dispositivo. Permite organizar reuniones de hasta 500 participantes, compartir pantalla para que el resto de asistentes puedan visualizarla, generar subtítulos en tiempo real para seguir con facilidad las reuniones, así como realizar grabaciones de las llamadas que serán alojadas en Drive, etcétera.

Figura 14: Logo de Google Meet

Las reuniones con el tutor han sido llevabas a cabo a través de esta plataforma, permitiendo hablar y tratar de cualquier aspecto del proyecto de manera virtual, obteniendo así más flexibilidad y disponibilidad en cuanto a horarios para poder reunirse.

2.4.4. Cloudinary

Figura 15: Logo de Cloudinary

Cloudinary [22] es una solución en la nube para gestionar el almacenamiento, carga, manipulación y optimización de imágenes, vídeos y distintos recursos multimedia para sitios web y aplicaciones móviles.

Una de las principales ventajas es su gratuidad frente a otros servicios. Existe un plan de pago pero el plan gratuito puede usarse por tiempo indefinido, siempre que no se excedan los límites de almacenamiento concedidos.

En este proyecto se emplea Cloudinary para alojar las imágenes de perfil que suben los distintos usuarios, así como las imágenes correspondientes a las insignias obtenidas por los usuarios de la aplicación al alcanzar un número determinado de puntos.

2.4.5. Trello

Trello [23] es una herramienta para gestionar el trabajo, individual o en equipo, con la que se pueden diseñar planes, colaborar, organizar el trabajo mediante flujos y realizar un seguimiento de las tareas e hitos que componen a este.

Figura 16: Logo de Trello

A través de tableros, clasificables según criterios a elegir, pueden organizarse las tareas de cualquier proyecto, así como marcar su estado de realización.

Dicha herramienta ha sido empleada en el proyecto para gestionar todas las tareas del ciclo de vida del proyecto, desde la implementación en código de los requisitos así como dejar constancia de cualquier aspecto importante.

2.5. Herramientas de documentación

2.5.1. Google Docs

Figura 17: Logo de Google Docs

Google Docs [24] es un procesador de texto en línea que permite crear, editar, y colaborar en documentos de texto en tiempo real desde cualquier dispositivo.

Ha sido empleado para tomar nota de distintos aspectos, como pueden ser: registrar decisiones tomadas, dudas que se van a plantear al tutor, anotaciones realizadas durante las reuniones con el tutor, borradores de ciertos capítulos de la memoria, requisitos de la aplicación, entre otros.

2.5.2. Overleaf

Overleaf [25] es un editor colaborativo de LaTeX en la nube, empleado para escribir y editar documentos de cualquier índole. Dispone de un motor de renderizado que permite visualizar el documento generado tras escribir el código fuente.

Permite gestionar el documento en secciones, incluir imágenes, tablas y otros elementos, definir nuevos comandos necesarios por el usuario, etc.

Se ha utilizado este editor para generar este documento que corresponde a la memoria del TFG, para así obtener un resultado de calidad, bien formateado y correctamente estructurado.

Figura 18: Logo de Overleaf

2.6. Herramientas de diseño

2.6.1. Marvel

Figura 19: Logo de Marvel

Marvel [26] es una herramienta colaborativa de prototipado. Proporciona una sencilla e intuitiva interfaz para generar prototipos de aplicaciones móviles o web de manera rápida, facilitando al usuario una serie de componentes que son utilizados usualmente en las aplicaciones para así poder incorporarlos en las interfaces que vaya a prototipar.

Esta herramienta ha tomado acción en el proyecto para la realización de las maquetas de las interfaces, obteniendo así un prototipo inicial de la aplicación a través del cual se puede navegar para tener una idea del funcionamiento que tendrá y cómo se vinculan unas vistas con otras.

2.6.2. Photoshop

Photoshop [27] es un editor de fotografías y otros gráficos desarrollado por Adobe. Permite realizar gran cantidad de acciones: crear o editar imágenes, añadir formas, textos y efectos, emplear herramientas como lápices y rellenos, entre otros. Aunque su uso no es sencillo y requiere aprendizaje, es ampliamente utilizado entre diseñadores gráficos porque se obtienen muy buenos resultados.

Figura 20: Logo de Photoshop

Dicho editor ha sido empleado para modificar algunos de los iconos existentes en la aplicación. Tras descargar los iconos deseados de Internet, se han editado en varias ocasiones para obtener un resultado exacto, con colores concretos.

2.7. Herramientas y recursos adicionales

2.7.1. Beacon Accent iBKS-105

Figura 21: Dispositivo Beacon iBKS-105

Un *beacon* es un dispositivo transmisor de señales de radio periódicas o continuas que incluyen información como su identificación o ubicación, haciendo uso de *Bluetooth Low Energy* (BLE) para comunicarse. El *beacon* del que se dispone para llevar a cabo este trabajo es el modelo iBKS 105 [28] de Accent Systems.

Dicho *beacon* es considerado oficialmente el primero en ser compatible con iBeacon y Eddystone al mismo tiempo. Se trata de dos estándares desarrollados por Apple y Google respectivamente, que permiten a las aplicaciones móviles detectar las señales de los *beacons* y reaccionar ante ellas. En este trabajo, se ha utilizado Eddystone.

Este dispositivo se emplea en algunos casos de uso distintos, pero para este proyecto se aprovecha su funcionalidad para llevar a cabo la detección de presencia y posicionamiento en los interiores de los centros deportivos, controlando si el usuario está entrenando y en qué sala se encuentra (en el caso de que existan diversos *beacons* repartidos en el interior de la instalación deportiva).

2.7.2. Eddystone

Eddystone [29], creado por Google, es una especificación de protocolo que define un formato de mensaje BLE para los mensajes que emiten los *beacons*. Describe distintos marcos de transmisión que pueden ser usados individualmente o en conjunto: Eddystone-UID, Eddystone-EID, Eddystone-TLM y Eddystone-URL.

Figura 22: Logo de Eddystone

Dentro de este proyecto, se hará uso únicamente del marco Eddystone-UID, cuyo funcionamiento es transmitir un identificador único del *beacon* con longitud de 16 *bytes*, dedicándose 10 de ellos a un espacio de nombres y los 6 restantes a una instancia concreta. La parte del espacio de nombres se dedica para agrupar un conjunto de *beacons*, mientras que la instancia identifica a cada dispositivo individual dentro del grupo.

En el proyecto se hace uso de este marco UID para asignar identificadores a los *beacons*. De esta manera, cuando la aplicación atienda a las señales Bluetooth, detectará a los *beacons* cercanos que estarán enviando su identificador y, así, tras procesar el identificador recibido, se puede conocer en qué zona concreta se encuentra el usuario.

2.7.3. iBKS Config Tool

Figura 23: Logo de iBKS Config Tool

La aplicación móvil iBKS Config Tool [30] permite escanear y configurar los *beacons* iBKS desarrollados por Accent Systems que se encuentren cerca. Está disponible para Android y para iOS, y permite configurar los servicios iBeacon y Eddystone, además de otras características del *beacon*.

Dicha aplicación se usa en el proyecto para configurar el servicio Eddystone del *beacon*, eligiendo el marco UID y estableciendo su identificador, su potencia de transmisión y calibración en decibelios-milivatio, y su intervalo de emisión de señal en milisegundos.

3
Metodología

Este capítulo recoge cómo se define y cómo se desarrolla el proyecto y establece las bases de actuación a lo largo de todo el proceso, detallando algunos aspectos sobre las decisiones tomadas.

3.1. Metodología ágil

Para la realización de un proyecto de esta índole, se plantea la utilización de una metodología ágil para realizar rápidas adaptaciones a cualquier tipo de cambio que surja durante el desarrollo. Debido a las actualizaciones y cambios de versión a las que puede ser sometido un proyecto realizado para Android (debido a la aparición de nuevas librerías, deprecación de métodos al actualizar el nivel de API, etc), algunos requisitos pueden sufrir variaciones en el desarrollo y deban ser revisados e incluso modificados.

Así, se ha escogido la metodología ágil Scrum [31] para llevar a cabo el proyecto. Como bien es conocido, en esta metodología colaboran varias personas, las cuales tienen asignadas roles diferentes: documentalistas, diseñadores, analistas, desarrolladores y probadores o *testers*. Sin embargo, este proyecto solamente lo conforma una única persona, ocupando así todos los roles mencionados, excepto el de *Product Owner*, cubierto por el tutor de este proyecto.

Esta metodología también destaca por la división del proyecto en diversas iteraciones con una duración aproximada de 15 días. Tras cada iteración se realizan reuniones para comentar los aspectos relevantes, verificar los resultados y plantear la siguiente iteración. Además, se plantean reuniones si son necesarias en cualquier otro momento. En este proyecto, las iteraciones han tenido un poco más de duración, y ha habido pausas entre ellas debido a causas externas.

3.2. Planificación y gestión del proyecto

La herramienta Trello ha ayudado al estudiante a gestionar inicialmente el proyecto y plantear una planificación tentativa, modificada posteriormente tras encontrar inconvenientes en algunas de las fases del desarrollo.

Un tablero como el que se recoge en la siguiente imagen se ha usado para plantear en distintas columnas los recursos a estudiar y consultar, recoger los requisitos a implementar, las tareas a realizar a través de la siguiente organización: el trabajo que hay que hacer (*TO DO*), el trabajo que está en proceso (*DOING*) y el trabajo ya realizado (*DONE*), además de una columna dedicada a los problemas que han surgido.

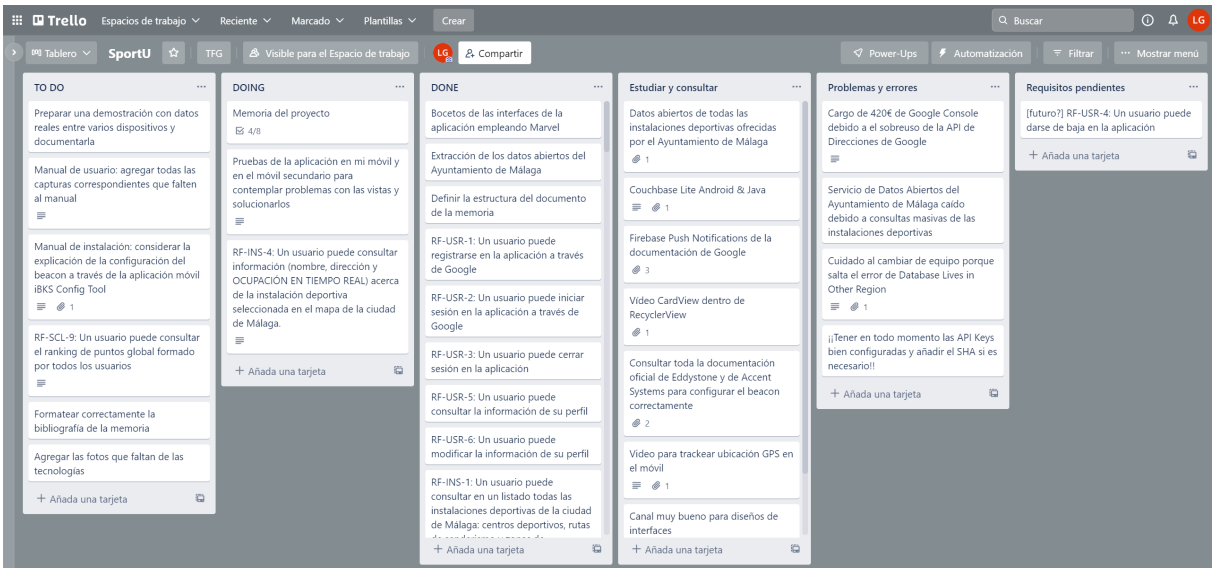


Figura 24: Tablero de Trello con la organización

3.3. Iteraciones realizadas

El proyecto, tal y como Scrum plantea, se ha dividido en diferentes iteraciones o *sprints*, para cubrir correctamente todo el proceso de estudio y desarrollo necesario para obtener un buen resultado.

3.3.1. Iteración 0

Esta iteración puede considerarse una de las más duras de todo el proceso de trabajo, con una duración estimada de 14 días. Es ese momento en el que todo parece ambiguo, no hay

claridad en la mayoría de decisiones y la mayoría del trabajo a realizar parece difícil de abordar. Por esta razón, hay que dedicar bastante tiempo a las siguientes tareas, fundamentales en esta etapa:

- Estudio del desarrollo de aplicaciones móviles con Java para teléfonos inteligentes con sistemas operativos Android.
- Recogida de información y visión general de las distintas técnicas o funcionalidades que se llevan a cabo en el proyecto: notificaciones *push*, uso de sensores GPS y Bluetooth y uso de *beacons*.
- Extracción y clasificación de los requisitos funcionales y no funcionales de la aplicación, y su división en distintos módulos.
- Elaboración de bocetos de las interfaces empleando Marvel para determinar un diseño inicial de la aplicación y obtener su esquema de navegación.
- Generación de la planificación del proyecto a través del tablero de Trello, donde se gestionan y reparten las tareas tal y como se ha visto anteriormente.
- Planteamiento inicial de la base de datos local que será el representante virtual en el teléfono móvil del usuario de la aplicación.

3.3.2. Iteraciones 1-4

Tras superar con éxito la iteración 0, el objetivo es superar de la misma manera las siguientes iteraciones, que se describen brevemente a continuación pero son explicadas en profundidad en el capítulo 6 de este documento.

La iteración 1 comienza con algunas tareas que son consideradas los cimientos de la aplicación. Se incorpora en esta iteración el registro y acceso de usuarios, y la visualización de las instalaciones deportivas a través de un listado. Se estima una duración de 12 días.

En la iteración 2, con una estimación aproximada a la iteración anterior, se tratan aspectos relacionados con la visualización y edición del perfil de los usuarios. Además, se implementa toda la lógica necesaria para la consulta de las instalaciones deportivas en un mapa de la ciudad de Málaga.

Para la iteración 3, en la cual se estiman 15 días, se realizan algunas funcionalidades referentes al registro de actividad deportiva realizada así como su consulta, edición y la obtención de puntos. Por otro lado, se incorpora el listado que contiene a todos los usuarios de la aplicación.

En la ultima iteración, se implementan todas las funcionalidades relativas al ámbito social de la aplicación: envío y confirmación de solicitudes de amistad, gestión de los amigos de un usuario y consulta de sus perfiles, y visualización del *ranking* global. Dicha iteración tiene una duración estimada de 15 días.

4
Análisis

Este capítulo abarca los requisitos de la aplicación propuesta junto a los casos de uso correspondientes, ambos fundamentales para llevar a cabo la planificación, el diseño y el desarrollo del proyecto, pudiendo identificar así las funciones elementales que debe realizar la aplicación.

4.1. Requisitos definidos

Un requisito es una capacidad o condición que se exige para poder resolver un problema o conseguir un objetivo. En base a su naturaleza, se puede hablar de requisito funcional y de requisito no funcional.

Los requisitos funcionales son funciones o servicios que deben ofrecer un sistema, de manera que describan cómo interactúa el usuario con el sistema, y viceversa. Por otro lado, los requisitos no funcionales son restricciones impuestas a las funciones ofrecidas por el sistema.

4.1.1. Requisitos funcionales

Este apartado recoge los requisitos funcionales de la aplicación a desarrollar y los clasifica en distintos grupos.

Requisitos relacionados con la gestión de usuarios

- RF-USR-1: Un usuario puede registrarse en la aplicación a través de Google.
- RF-USR-2: Un usuario puede iniciar sesión en la aplicación a través de Google.
- RF-USR-3: Un usuario puede cerrar sesión en la aplicación.
- RF-USR-4: Un usuario puede consultar la información de su perfil.
- RF-USR-5: Un usuario puede modificar la información de su perfil.

Requisitos relacionados con las instalaciones deportivas

- RF-INS-1: Un usuario puede consultar en un listado todas las instalaciones deportivas de la ciudad de Málaga: centros deportivos, rutas de senderismo y zonas de musculación, ordenadas por cercanía a este.
- RF-INS-2: Un usuario puede visualizar en un mapa de la ciudad de Málaga todas las instalaciones deportivas: centros deportivos, rutas de senderismo y zonas de musculación.
- RF-INS-3: Un usuario puede filtrar en un mapa de la ciudad de Málaga según el tipo de instalación deportiva.
- RF-INS-4: Un usuario puede consultar información (nombre, dirección y ocupación en tiempo real) acerca de la instalación deportiva seleccionada en el mapa de la ciudad de Málaga.

Requisitos relacionados con la actividad deportiva realizada

- RF-ACT-1: Un usuario puede consultar su registro de actividad deportiva realizada.
- RF-ACT-2: Un usuario puede registrar información de una actividad deportiva realizada.
- RF-ACT-3: Un usuario puede editar la información de una actividad deportiva registrada.
- RF-ACT-4: Un usuario puede eliminar una actividad deportiva registrada.
- RF-ACT-5: Un usuario puede consultar los registros de actividad deportiva de un amigo.
- RF-ACT-6: Un usuario puede obtener puntos tras realizar una actividad deportiva.

Requisitos relacionados con la socialización entre usuarios

- RF-SCL-1: Un usuario puede consultar todos los usuarios registrados en la aplicación, ordenados por coincidencia de intereses.
- RF-SCL-2: Un usuario puede consultar todas sus solicitudes de amistad recibidas.
- RF-SCL-3: Un usuario puede consultar todos sus amigos.
- RF-SCL-4: Un usuario puede enviar una solicitud de amistad a otro usuario.

- RF-SCL-5: Un usuario puede aceptar una solicitud de amistad recibida.
- RF-SCL-6: Un usuario puede denegar una solicitud de amistad recibida.
- RF-SCL-7: Un usuario puede consultar el perfil de un amigo.
- RF-SCL-8: Un usuario puede eliminar a un amigo de su lista de amigos.
- RF-SCL-9: Un usuario puede consultar el *ranking* de puntos global formado por todos los usuarios.

4.1.2. Requisitos no funcionales

Este apartado se centra en los requisitos no funcionales de la aplicación a desarrollar mediante los cuales se definen sus características.

- RNF-1: La aplicación se basará en el modelo Digital Avatars, de modo que el usuario disponga de un perfil virtual almacenado en su móvil.
- RNF-2: La aplicación seguirá el modelo de computación social distribuida, evitando utilizar un servidor centralizado que almacene todos los datos y gobierne el funcionamiento de esta.
- RNF-3: El perfil virtual del usuario junto a toda la información recibida de los demás usuarios y la información relativa a las instalaciones deportivas se almacenará localmente en una base de datos no relacional como Couchbase Lite.
- RNF-4: La información de las instalaciones deportivas se descargará de una base de datos no relacional en la nube como MongoDB. Esta descarga se realizará automáticamente al registrarse el usuario por primera vez y será almacenada localmente en la base de datos no relacional Couchbase Lite.
- RNF-5: Los usuarios registrados en la aplicación que mantengan su sesión iniciada en un móvil se almacenarán en una base de datos en tiempo real como Realtime Database de Firebase.
- RNF-6: El registro, inicio y cierre de sesión se gestionará utilizando Authentication de Firebase.

- RNF-7: El envío y recepción de solicitudes entre usuarios se realizará a través de notificaciones push utilizando Cloud Messaging de Firebase.
- RNF-8: La aplicación requerirá el uso de los sensores GPS y Bluetooth del dispositivo móvil en el que se esté utilizando.
- RNF-9: La aplicación requerirá la disponibilidad de conexión a Internet para la descarga de cualquier dato en la nube o la recepción de notificaciones.
- RNF-10: La aplicación escaneará dispositivos Beacon para controlar la ubicación del usuario en interiores.

4.2. Casos de uso

Este apartado recoge una serie de tablas correspondientes a los casos de uso, relacionados con los requisitos funcionales extraídos. Cada caso de uso se compone de un identificador, un título, una descripción, unas pre y post condiciones, un requisito asociado al caso de uso, un escenario principal y uno o varios escenarios alternativos.

Tabla 1: CU01 - Registro de un usuario

Identificador	CU01
Título	Registro de un usuario
Descripción	Un usuario puede registrarse en la aplicación a través de Google
Pre-condición	El usuario ha iniciado SportU
Post-condición	El usuario se registra correctamente
Requisitos	RF-USR-1
Escenario principal	
1. El sistema muestra la pantalla de bienvenida. 2. El usuario selecciona la opción “Acceder con Google” para registrarse. 3. El sistema muestra la pantalla de selección de cuentas de Google mostrando los correos electrónicos disponibles en el teléfono del usuario. 4. El usuario selecciona la cuenta de Google con la que quiere registrarse.	

5. El sistema muestra el formulario con los campos básicos a cumplimentar para el registro: nombre completo, fecha de nacimiento, género y fotografía. 6. El usuario cumplimenta los campos básicos requeridos por el sistema. 7. El usuario selecciona la opción “Siguiente” para continuar con el registro. 8. El sistema comprueba la cumplimentación de todos los campos básicos. 9. El sistema muestra el formulario con el resto de campos a cumplimentar para terminar el registro: descripción e intereses. 10. El usuario cumplimenta el resto de campos requeridos por el sistema. 11. El usuario selecciona la opción “Finalizar” para terminar el registro. 12. El sistema comprueba la cumplimentación del resto de campos. 13. El sistema crea una cuenta para el usuario y lo redirige a la pantalla principal donde se muestra su perfil.
Escenarios alternativos
Campos básicos sin cumplimentar: 8. El sistema detecta que no se han cumplimentado todos los campos básicos. 9. El sistema indica al usuario que debe cumplimentar todos los campos básicos. Retorno al paso 6 del escenario principal.
Resto de campos sin cumplimentar: 12. El sistema detecta que no se han cumplimentado el resto de campos. 13. El sistema indica al usuario que debe cumplimentar el resto de campos. Retorno al paso 10 del escenario principal.

Tabla 2: CU02 - Inicio de sesión de un usuario

Identificador	CU02
Título	Inicio de sesión de un usuario
Descripción	Un usuario puede iniciar sesión en la aplicación a través de Google
Pre-condición	El usuario ha iniciado SportU y se ha registrado previamente
Post-condición	El usuario inicia sesión correctamente
Requisitos	RF-USR-2
Escenario principal	

1. El sistema muestra la pantalla de bienvenida. 2. El usuario selecciona la opción “Acceder con Google” para iniciar sesión. 3. El sistema muestra la pantalla de selección de cuentas de Google mostrando los correos electrónicos disponibles en el teléfono del usuario. 4. El usuario selecciona la cuenta de Google con la que quiere iniciar sesión. 5. El sistema muestra la pantalla principal con el perfil del usuario tras aprobar el inicio de sesión.
Escenarios alternativos
Inicio de sesión fallido: 5. El sistema falla al iniciar la sesión del usuario indicándolo con un mensaje. 6. El sistema redirecciona al usuario a la pantalla de bienvenida.

Tabla 3: CU03 - Cierre de sesión de un usuario

Identificador	CU03
Título	Cierre de sesión de un usuario
Descripción	Un usuario puede cerrar sesión en la aplicación
Pre-condición	El usuario ha iniciado sesión en SportU y se encuentra en cualquier pantalla de la aplicación
Post-condición	El usuario cierra sesión con éxito
Requisitos	RF-USR-3
Escenario principal	
1. El usuario selecciona la opción “Perfil” del menú de navegación. 2. El sistema muestra la sección del perfil del usuario con sus datos. 3. El usuario selecciona la opción “Cerrar sesión”. 4. El sistema redirige al usuario a la pantalla de bienvenida cerrando su sesión.	
Escenarios alternativos	
Cierre de sesión fallido: 4. El sistema falla al cerrar la sesión del usuario indicándolo con un mensaje. 5. El sistema redirecciona al usuario a la pantalla de bienvenida.	

Tabla 4: CU04 - Consulta del perfil de un usuario

Identificador	CU04
Título	Consulta del perfil de un usuario
Descripción	Un usuario puede consultar la información de su perfil
Pre-condición	El usuario ha iniciado sesión en SportU y se encuentra en cualquier pantalla de la aplicación
Post-condición	El usuario ve la información de su perfil
Requisitos	RF-USR-4
Escenario principal	
1. El usuario selecciona la opción “Perfil” del menú de navegación. 2. El sistema muestra la sección del perfil del usuario con sus datos.	
Escenarios alternativos	
Mostrar perfil fallido: 2. El sistema falla al cargar los datos del usuario y se lo indica.	

Tabla 5: CU05 - Edición del perfil de un usuario

Identificador	CU05
Título	Edición del perfil de un usuario
Descripción	Un usuario puede modificar la información de su perfil
Pre-condición	El usuario ha iniciado sesión en SportU y se encuentra en cualquier pantalla de la aplicación
Post-condición	El usuario edita su perfil correctamente
Requisitos	RF-USR-5
Escenario principal	
1. El usuario selecciona la opción “Perfil” del menú de navegación. 2. El sistema muestra la sección del perfil del usuario con sus datos. 3. El usuario selecciona la opción “Editar”. 4. El sistema muestra el formulario de edición del perfil con los campos básicos cumplimentados y disponibles para editar: nombre completo, fecha de nacimiento, género y fotografía.	

5. El usuario modifica los campos básicos deseados. 6. El usuario selecciona la opción “Siguiente” para continuar con la edición de su perfil. 7. El sistema comprueba la cumplimentación de todos los campos básicos. 8. El sistema muestra el formulario de edición del perfil con el resto de campos cumplimentados y disponibles para editar: descripción e intereses. 9. El usuario cumplimenta el resto de campos requeridos por el sistema. 10. El usuario selecciona la opción “Finalizar” para terminar la edición del perfil. 11. El sistema comprueba la cumplimentación del resto de campos. 12. El sistema modifica la información del usuario y lo redirige a la pantalla principal donde se muestra su perfil.
Escenarios alternativos
Campos básicos sin cumplimentar: 7. El sistema detecta que no se han cumplimentado todos los campos básicos. 8. El sistema indica al usuario que debe cumplimentar todos los campos básicos. Retorno al paso 5 del escenario principal.
Resto de campos sin cumplimentar: 11. El sistema detecta que no se han cumplimentado el resto de campos. 12. El sistema indica al usuario que debe cumplimentar el resto de campos. Retorno al paso 9 del escenario principal.

Tabla 6: CU06 - Consulta del listado de las instalaciones deportivas

Identificador	CU06
Título	Consulta del listado de las instalaciones deportivas
Descripción	Un usuario puede consultar en un listado todas las instalaciones deportivas de la ciudad de Málaga: centros deportivos, rutas de senderismo y zonas de musculación, ordenadas por cercanía a este
Pre-condición	El usuario ha iniciado sesión en SportU y se encuentra en cualquier pantalla de la aplicación
Post-condición	El usuario visualiza el listado de instalaciones deportivas
Requisitos	RF-INS-1

Escenario principal
1. El usuario selecciona la opción “Instalaciones” del menú de navegación. 2. El sistema muestra la sección de instalaciones deportivas. 3. El usuario selecciona la opción “Listado” del menú superior. 4. El sistema muestra el listado de instalaciones deportivas de Málaga ordenadas por distancia al usuario.
Escenarios alternativos
Opción mal seleccionada: 3. El usuario selecciona la opción “Mapa” del menú superior. 4. El sistema muestra el mapa de la ciudad de Málaga con las instalaciones deportivas representadas con marcadores de colores en función del tipo de instalación.

Tabla 7: CU07 - Consulta del mapa de las instalaciones deportivas

Identificador	CU07
Título	Consulta del mapa de las instalaciones deportivas
Descripción	Un usuario puede visualizar en un mapa de la ciudad de Málaga todas las instalaciones deportivas: centros deportivos, rutas de senderismo y zonas de musculación
Pre-condición	El usuario ha iniciado sesión en SportU y se encuentra en cualquier pantalla de la aplicación
Post-condición	El usuario visualiza el mapa de instalaciones deportivas
Requisitos	RF-INS-2
Escenario principal	
1. El usuario selecciona la opción “Instalaciones” del menú de navegación. 2. El sistema muestra la sección de instalaciones deportivas. 3. El usuario selecciona la opción “Mapa” del menú superior. 4. El sistema muestra el mapa de la ciudad de Málaga con las instalaciones deportivas representadas con marcadores de colores en función del tipo de instalación.	
Escenarios alternativos	

Opción mal seleccionada:
3. El usuario selecciona la opción “Listado” del menú superior.
4. El sistema muestra el listado de instalaciones deportivas de Málaga ordenadas por distancia al usuario.

Tabla 8: CU08 - Filtrado en el mapa de las instalaciones deportivas

Identificador	CU08
Título	Filtrado en el mapa de las instalaciones deportivas
Descripción	Un usuario puede filtrar en un mapa de la ciudad de Málaga según el tipo de instalación deportiva
Pre-condición	El usuario ha iniciado sesión en SportU y se encuentra en cualquier pantalla de la aplicación
Post-condición	El usuario visualiza el mapa con las instalaciones deportivas filtradas
Requisitos	RF-INS-3
Escenario principal	
1. El usuario selecciona la opción “Instalaciones” del menú de navegación. 2. El sistema muestra la sección de instalaciones deportivas. 3. El usuario selecciona la opción “Mapa” del menú superior. 4. El sistema muestra el mapa de la ciudad de Málaga con las instalaciones deportivas representadas con marcadores de colores en función del tipo de instalación. 5. El usuario selecciona la opción “Polideportivos” en la zona superior del mapa para ocultar los marcadores asociados. 6. El sistema oculta los marcadores de posición de los centros deportivos, manteniendo los marcadores del resto de instalaciones deportivas.	
Escenarios alternativos	
Filtrado de rutas de senderismo: 5. El usuario selecciona la opción “Senderismo” en la zona superior del mapa para ocultar los marcadores asociados.	

6. El sistema oculta los marcadores de posición de las rutas de senderismo, manteniendo los marcadores del resto de instalaciones deportivas.
Filtrado de zonas de musculación: 5. El usuario selecciona la opción “Musculación” en la zona superior del mapa para ocultar los marcadores asociados. 6. El sistema oculta los marcadores de posición de las zonas de musculación, manteniendo los marcadores del resto de instalaciones deportivas.

Tabla 9: CU09 - Consulta de información de una instalación deportiva

Identificador	CU09
Título	Consulta de información de una instalación deportiva
Descripción	Un usuario puede consultar información (nombre, dirección y ocupación en tiempo real) acerca de la instalación deportiva seleccionada en el mapa de la ciudad de Málaga.
Pre-condición	El usuario ha iniciado sesión en SportU y se encuentra en cualquier pantalla de la aplicación
Post-condición	El usuario visualiza el mapa con las instalaciones deportivas filtradas
Requisitos	RF-INS-4
Escenario principal	
1. El usuario selecciona la opción “Instalaciones” del menú de navegación. 2. El sistema muestra la sección de instalaciones deportivas. 3. El usuario selecciona la opción “Mapa” del menú superior. 4. El sistema muestra el mapa de la ciudad de Málaga con las instalaciones deportivas representadas con marcadores de colores en función del tipo de instalación. 5. El usuario selecciona el marcador de una instalación deportiva. 6. El sistema muestra la información de la instalación deportiva: nombre, dirección y ocupación en tiempo real.	
Escenarios alternativos	
No se contemplan escenarios alternativos	

Tabla 10: CU10 - Consulta de toda la actividad deportiva realizada

Identificador	CU10
Título	Consulta de toda la actividad deportiva realizada
Descripción	Un usuario puede consultar su registro de actividad deportiva realizada
Pre-condición	El usuario ha iniciado sesión en SportU y se encuentra en cualquier pantalla de la aplicación
Post-condición	El usuario visualiza el registro de actividad deportiva correctamente
Requisitos	RF-ACT-1
Escenario principal	
1. El usuario selecciona la opción “Actividad” del menú de navegación. 2. El sistema muestra la sección de actividad con todas las actividades deportivas realizadas por el usuario.	
Escenarios alternativos	
No se contemplan escenarios alternativos	

Tabla 11: CU11 - Registro de una actividad deportiva realizada

Identificador	CU11
Título	Registro de una actividad deportiva realizada
Descripción	Un usuario puede registrar información de una actividad deportiva realizada
Pre-condición	El usuario ha iniciado sesión en SportU, se encuentra en cualquier pantalla de la aplicación y ha realizado actividad física
Post-condición	La actividad deportiva se registra correctamente
Requisitos	RF-ACT-2
Escenario principal	
1. El usuario selecciona la opción “Actividad” del menú de navegación. 2. El sistema muestra la sección de actividad con todas las actividades deportivas realizadas por el usuario. 3. El usuario selecciona la opción “Nuevo” para añadir un nuevo entrenamiento.	

4. El sistema sugiere al usuario la instalación deportiva en la que parece haber entrenado tras registrar su ubicación continuamente. 5. El usuario confirma haber entrenado seleccionando la instalación deportiva mostrada. 6. El sistema muestra el formulario de registro de actividad deportiva con campos a cumplimentar: nombre del entrenamiento y descripción. El resto de campos son autocompletados: instalación deportiva, fecha y duración. 7. El usuario cumplimenta los campos requeridos: nombre y descripción, y comprueba los otros. 8. El usuario selecciona la opción "Registrar" para grabar su entrenamiento en su registro de actividad. 9. El sistema registra el entrenamiento con éxito y redirige al usuario a la sección de actividad con todas las actividades deportivas realizadas por el usuario.
Escenarios alternativos
Instalación deportiva incorrecta: 5. El usuario confirma no haber entrenado en la instalación deportiva mostrada. 6. El sistema muestra un listado con instalaciones deportivas permitiendo elegir al usuario dónde ha entrenado. Retorno al paso 7 del escenario principal.
Campos sin cumplimentar 9. El sistema detecta que no se han cumplimentado los campos indicados. Retorno al paso 7 del escenario principal.

Tabla 12: CU12 - Edición de una actividad deportiva registrada

Identificador	CU12
Título	Edición de una actividad deportiva registrada
Descripción	Un usuario puede editar la información de una actividad deportiva registrada
Pre-condición	El usuario ha iniciado sesión en SportU, se encuentra en "Actividad" y ha registrado previamente una actividad física realizada
Post-condición	La actividad deportiva se modifica correctamente

Requisitos	RF-ACT-3
Escenario principal	
1. El sistema muestra la sección de actividad con todas las actividades deportivas realizadas por el usuario. 2. El usuario selecciona la opción “Editar” sobre el entrenamiento correspondiente para editarlo. 3. El sistema muestra el formulario de edición de actividad deportiva con todos los campos cumplimentados y solo algunos disponibles para editar: nombre, descripción e instalación deportiva. 4. El usuario modifica los campos deseados. 5. El usuario selecciona la opción “Registrar” para confirmar la modificación de su entrenamiento. 6. El sistema modifica la información del entrenamiento con éxito y redirige al usuario a la sección de actividad con todas las actividades deportivas realizadas por el usuario.	
Escenarios alternativos	
Campos sin cumplimentar: 6. El sistema detecta que no se han cumplimentado los campos indicados. Retorno al paso 4 del escenario principal.	

Tabla 13: CU13 - Eliminación de una actividad deportiva registrada

Identificador	CU13
Título	Eliminación de una actividad deportiva registrada
Descripción	Un usuario puede eliminar una actividad deportiva registrada
Pre-condición	El usuario ha iniciado sesión en SportU, se encuentra en “Actividad” y ha registrado previamente una actividad física realizada
Post-condición	La actividad deportiva se elimina correctamente
Requisitos	RF-ACT-4
Escenario principal	

1. El sistema muestra la sección de actividad con todas las actividades deportivas realizadas por el usuario. 2. El usuario selecciona la opción “Editar” sobre el entrenamiento correspondiente para eliminarlo. 3. El sistema muestra el formulario de edición de actividad deportiva con todos los campos cumplimentados. 4. El usuario selecciona la opción “Eliminar” para borrar la actividad registrada. 5. El sistema muestra una pantalla de confirmación preguntando si desea eliminar la actividad. 6. El usuario selecciona “Sí” para confirmar el borrado de la actividad registrada. 7. El sistema elimina la actividad con éxito y redirige al usuario a la sección de actividad con todas las actividades deportivas realizadas por el usuario.
Escenarios alternativos
Eliminación fallida: 6. El usuario selecciona “No” para denegar el borrado de la actividad registrada. 7. El sistema mantiene la actividad y redirige al usuario a la sección de actividad con todas las actividades deportivas realizadas por el usuario.

Tabla 14: CU14 - Consulta de toda la actividad deportiva realizada por un amigo

Identificador	CU14
Título	Consulta de toda la actividad deportiva realizada por un amigo
Descripción	Un usuario puede consultar los registros de actividad deportiva de un amigo
Pre-condición	El usuario ha iniciado sesión en SportU y es amigo del usuario a consultar
Post-condición	El usuario visualiza el registro de actividad deportiva de su amigo correctamente
Requisitos	RF-ACT-5
Escenario principal	
1. El usuario selecciona la opción “Social” del menú de navegación. 2. El sistema muestra la sección de interacción social que dispone de un menú superior.	

3. El usuario selecciona la opción “Amigos” del menú superior. 4. El sistema muestra la sección de amigos con las solicitudes de amistad pendientes y un listado con los amigos del usuario. 5. El usuario selecciona el usuario a consultar de entre su lista de amigos. 6. El sistema muestra el perfil del amigo con toda su información, incluyendo sus entrenamientos registrados.
Escenarios alternativos
Selección de “Usuarios”: 3. El usuario selecciona la opción “Usuarios” del menú superior. 4. El sistema muestra la sección de todos los usuarios registrados en la aplicación, distinguiendo entre amigos y no amigos. 5. El usuario selecciona, de entre los usuarios que tiene agregados, uno de ellos para consultar sus entrenamientos. Retorno al paso 6 del escenario principal.
Selección de “Ranking”: 3. El usuario selecciona la opción “Ranking” del menú superior. Redirección al CU24, correspondiente al RF-SCL-9.

Tabla 15: CU15 - Obtención de puntos de una actividad deportiva realizada

Identificador	CU15
Título	Obtención de puntos de una actividad deportiva realizada
Descripción	Un usuario puede obtener puntos tras realizar una actividad deportiva
Pre-condición	El usuario ha iniciado sesión en SportU y ha registrado una actividad deportiva
Post-condición	Los puntos se calculan y se suman al total de puntos del usuario correctamente
Requisitos	RF-ACT-6
Escenario principal	
1. El usuario registra una actividad deportiva realizada tal y como se describe en CU11.	

2. El sistema registra el entrenamiento con éxito y calcula los puntos obtenidos en base a la duración del entrenamiento. 3. El sistema añade los puntos obtenidos de la actividad al total de puntos del usuario y lo redirige a la sección de actividad con todas las actividades deportivas realizadas.
Escenarios alternativos
No se contemplan escenarios alternativos

Tabla 16: CU16 - Consulta de todos los usuarios registrados en la aplicación

Identificador	CU16
Título	Consulta de todos los usuarios registrados en la aplicación
Descripción	Un usuario puede consultar todos los usuarios registrados en la aplicación, ordenados por coincidencia de intereses
Pre-condición	El usuario ha iniciado sesión en SportU
Post-condición	El usuario visualiza el listado de usuarios registrados correctamente
Requisitos	RF-SCL-1
Escenario principal	
1. El usuario selecciona la opción “Social” del menú de navegación. 2. El sistema muestra la sección de interacción social que dispone de un menú superior. 3. El usuario selecciona la opción “Usuarios” del menú superior. 4. El sistema muestra la sección de usuarios registrados en la aplicación (con nombre, lista de intereses, porcentaje de compatibilidad con el usuario interesado en base a intereses, y puntuación) ordenados por dicha compatibilidad de mayor a menor.	
Escenarios alternativos	
Selección de “Amigos”: 3. El usuario selecciona la opción “Amigos” del menú superior. Redirección a CU16 y CU17, correspondientes a RF-SCL-2 y RF-SCL-3 respectivamente.	
Selección de “Ranking”: 3. El usuario selecciona la opción “Ranking” del menú superior. Redirección al CU24, correspondiente al RF-SCL-9.	

Tabla 17: CU17 - Consulta de todas las solicitudes de amistad recibidas

Identificador	CU17
Título	Consulta de todas las solicitudes de amistad recibidas
Descripción	Un usuario puede consultar todas sus solicitudes de amistad recibidas
Pre-condición	El usuario ha iniciado sesión en SportU y tiene solicitudes de amistad
Post-condición	El usuario visualiza el listado de solicitudes de amistad recibidas pendientes por aceptar
Requisitos	RF-SCL-2
Escenario principal	
1. El usuario selecciona la opción “Social” del menú de navegación. 2. El sistema muestra la sección de interacción social que dispone de un menú superior. 3. El usuario selecciona la opción “Amigos” del menú superior. 4. El sistema muestra la sección de amigos con las solicitudes de amistad recibidas pendientes y un listado con los amigos del usuario.	
Escenarios alternativos	
Selección de “Usuarios”: 3. El usuario selecciona la opción “Usuarios” del menú superior. Redirección al CU15, correspondiente a RF-SCL-1.	
Selección de “Ranking”: 3. El usuario selecciona la opción “Ranking” del menú superior. Redirección al CU24, correspondiente al RF-SCL-9.	
Ausencia de solicitudes de amistad: 4. El sistema muestra un mensaje indicando que el usuario no tiene solicitudes de amistad pendientes, junto a su listado de amigos.	

Tabla 18: CU18 - Consulta de todos los amigos

Identificador	CU18
Título	Consulta de todos los amigos
Descripción	Un usuario puede consultar todos sus amigos

Pre-condición	El usuario ha iniciado sesión en SportU y tiene amigos
Post-condición	El usuario visualiza el listado de amigos
Requisitos	RF-SCL-3
Escenario principal	
1. El usuario selecciona la opción “Social” del menú de navegación. 2. El sistema muestra la sección de interacción social que dispone de un menú superior. 3. El usuario selecciona la opción “Amigos” del menú superior. 4. El sistema muestra la sección de amigos con las solicitudes de amistad recibidas pendientes y un listado con los amigos del usuario.	
Escenarios alternativos	
Selección de “Usuarios”: 3. El usuario selecciona la opción “Usuarios” del menú superior. Redirección al CU15, correspondiente a RF-SCL-1.	
Selección de “Ranking”: 3. El usuario selecciona la opción “Ranking” del menú superior. Redirección al CU24, correspondiente al RF-SCL-9.	
Ausencia de amigos: 4. El sistema muestra un mensaje indicando que el usuario no tiene amigos, junto a su listado de solicitudes de amistad recibidas.	

Tabla 19: CU19 - Envío de solicitud de amistad a otro usuario

Identificador	CU19
Título	Envío de solicitud de amistad a otro usuario
Descripción	Un usuario puede enviar una solicitud de amistad a otro usuario
Pre-condición	El usuario ha iniciado sesión en SportU
Post-condición	El usuario envía la solicitud de amistad correctamente
Requisitos	RF-SCL-4
Escenario principal	
1. El usuario selecciona la opción “Social” del menú de navegación.	

2. El sistema muestra la sección de interacción social que dispone de un menú superior. 3. El usuario selecciona la opción “Usuarios” del menú superior. 4. El sistema muestra la sección de usuarios registrados en la aplicación (con nombre, lista de intereses, porcentaje de compatibilidad con el usuario interesado en base a intereses, y puntuación) ordenados por dicha compatibilidad de mayor a menor. 5. El usuario selecciona la opción “Añadir” en el usuario correspondiente. 6. El sistema envía la solicitud de amistad al usuario indicado, recibiendo este una notificación personalizada con dicha petición.
Escenarios alternativos
Imposibilidad de agregar amigos: 5. El usuario tiene agregados a todos los usuarios de la aplicación y no puede enviar solicitud a ninguno.

Tabla 20: CU20 - Aceptación de una solicitud de amistad recibida

Identificador	CU20
Título	Aceptación de una solicitud de amistad recibida
Descripción	Un usuario puede aceptar una solicitud de amistad recibida
Pre-condición	El usuario ha iniciado sesión en SportU y tiene solicitudes de amistad
Post-condición	El usuario tiene un nuevo amigo, el usuario solicitante
Requisitos	RF-SCL-5
Escenario principal	
1. El usuario selecciona la opción “Social” del menú de navegación. 2. El sistema muestra la sección de interacción social que dispone de un menú superior. 3. El usuario selecciona la opción “Amigos” del menú superior. 4. El sistema muestra la sección de amigos con las solicitudes de amistad recibidas pendientes y un listado con los amigos del usuario. 5. El usuario selecciona la opción “Aceptar” en la solicitud de amistad correspondiente. 6. El sistema muestra en la sección de amigos al usuario aceptado y elimina la solicitud del listado de solicitudes recibidas.	

7. El sistema envía una notificación al usuario solicitante indicando que su solicitud de amistad ha sido aceptada.
Escenarios alternativos
Ausencia de solicitudes de amistad: 4. El sistema muestra un mensaje indicando que el usuario no tiene solicitudes de amistad pendientes, junto a su listado de amigos.

Tabla 21: CU21 - Denegación de una solicitud de amistad recibida

Identificador	CU21
Título	Denegación de una solicitud de amistad recibida
Descripción	Un usuario puede denegar una solicitud de amistad recibida
Pre-condición	El usuario ha iniciado sesión en SportU y tiene solicitudes de amistad
Post-condición	El usuario rechaza la solicitud de amistad
Requisitos	RF-SCL-6
Escenario principal	
1. El usuario selecciona la opción “Social” del menú de navegación. 2. El sistema muestra la sección de interacción social que dispone de un menú superior. 3. El usuario selecciona la opción “Amigos” del menú superior. 4. El sistema muestra la sección de amigos con las solicitudes de amistad recibidas pendientes y un listado con los amigos del usuario. 5. El usuario selecciona la opción “Denegar” en la solicitud de amistad correspondiente. 6. El sistema elimina la solicitud del listado de solicitudes recibidas.	
Escenarios alternativos	
Ausencia de solicitudes de amistad: 4. El sistema muestra un mensaje indicando que el usuario no tiene solicitudes de amistad pendientes, junto a su listado de amigos.	

Tabla 22: CU22 - Consulta del perfil de un amigo

Identificador	CU22

Título	Consulta del perfil de un amigo
Descripción	Un usuario puede consultar el perfil de un amigo
Pre-condición	El usuario ha iniciado sesión en SportU y es amigo del usuario a consultar
Post-condición	El usuario visualiza el perfil de su amigo correctamente
Requisitos	RF-SCL-7
Escenario principal	
1. El usuario selecciona la opción “Social” del menú de navegación. 2. El sistema muestra la sección de interacción social que dispone de un menú superior. 3. El usuario selecciona la opción “Amigos” del menú superior. 4. El sistema muestra la sección de amigos con las solicitudes de amistad pendientes y un listado con los amigos del usuario. 5. El usuario selecciona el usuario a consultar de entre su lista de amigos. 6. El sistema muestra el perfil del amigo con toda su información: nombre, descripción, datos personales, intereses, puntuación y entrenamientos.	
Escenarios alternativos	
Ausencia de amigos: 4. El sistema muestra un mensaje indicando que el usuario no tiene amigos, junto a su listado de solicitudes de amistad recibidas.	

Tabla 23: CU23 - Eliminación de un amigo

Identificador	CU23
Título	Eliminación
Descripción	Un usuario puede eliminar a un amigo de su lista de amigos
Pre-condición	El usuario ha iniciado sesión en SportU y es amigo del usuario a consultar
Post-condición	El usuario elimina a su amigo correctamente
Requisitos	RF-SCL-8
Escenario principal	
1. El usuario selecciona la opción “Social” del menú de navegación. 2. El sistema muestra la sección de interacción social que dispone de un menú superior.	

3. El usuario selecciona la opción “Amigos” del menú superior. 4. El sistema muestra la sección de amigos con las solicitudes de amistad pendientes y un listado con los amigos del usuario. 5. El usuario selecciona el usuario a eliminar de entre su lista de amigos. 6. El sistema muestra el perfil del amigo con toda su información: nombre, descripción, datos personales, intereses, puntuación y entrenamientos. 7. El usuario selecciona la opción “Eliminar” para borrar al amigo. 8. El sistema muestra una pantalla de confirmación preguntando si desea eliminar al amigo. 9. El usuario selecciona “Sí” para confirmar la eliminación del amigo. 10. El sistema elimina al amigo con éxito y redirige al usuario a la sección de amigos con las solicitudes de amistad pendientes y un listado con los amigos del usuario.
Escenarios alternativos
Eliminación fallida: 9. El usuario selecciona “No” para denegar la eliminación de su amigo. 10. El sistema mantiene al amigo y redirige al usuario a la sección de amigos con las solicitudes de amistad pendientes y un listado con los amigos del usuario.

Tabla 24: CU24 - Consulta del ranking global de puntos

Identificador	CU24
Título	Consulta del ranking global de puntos
Descripción	Un usuario puede consultar el *ranking* de puntos global formado por todos los usuarios
Pre-condición	El usuario ha iniciado sesión en SportU
Post-condición	El usuario visualiza el ranking correctamente
Requisitos	RF-SCL-9
Escenario principal	
1. El usuario selecciona la opción “Social” del menú de navegación. 2. El sistema muestra la sección de interacción social que dispone de un menú superior. 3. El usuario selecciona la opción “Ranking” del menú superior.	

4. El sistema muestra el ranking global de puntos conformado por todos los usuarios, incluyendo un listado con la puntuación de cada uno.
Escenarios alternativos
Selección de “Usuarios”: 3. El usuario selecciona la opción “Usuarios” del menú superior. Redirección al CU15, correspondiente a RF-SCL-1.
Selección de “Amigos”: 3. El usuario selecciona la opción “Amigos” del menú superior. Redirección a CU16 y CU17, correspondientes a RF-SCL-2 y RF-SCL-3 respectivamente.

5

Modelado y diseño

Este capítulo aborda la arquitectura del proyecto y diagrama de clases de la aplicación, las distintas bases de datos que se han utilizado para llevar a cabo la correcta implementación de la aplicación, así como diversos aspectos y decisiones de diseño, las maquetas de las interfaces y el diagrama de navegación de la aplicación.

5.1. Arquitectura del proyecto

Esta sección explica la arquitectura del proyecto y justifica algunas de las decisiones tomadas para llevarlo a cabo. Para ello, se ha creado un diagrama básico explicativo para ubicar al lector, véase Figura 25.

Para realizar este trabajo, es necesario un teléfono móvil que disponga de conexión a Internet, Bluetooth y GPS. Además, es necesario un proyecto de Firebase en el que se habiliten los servicios de Authentication, Cloud Messaging y Realtime Database.

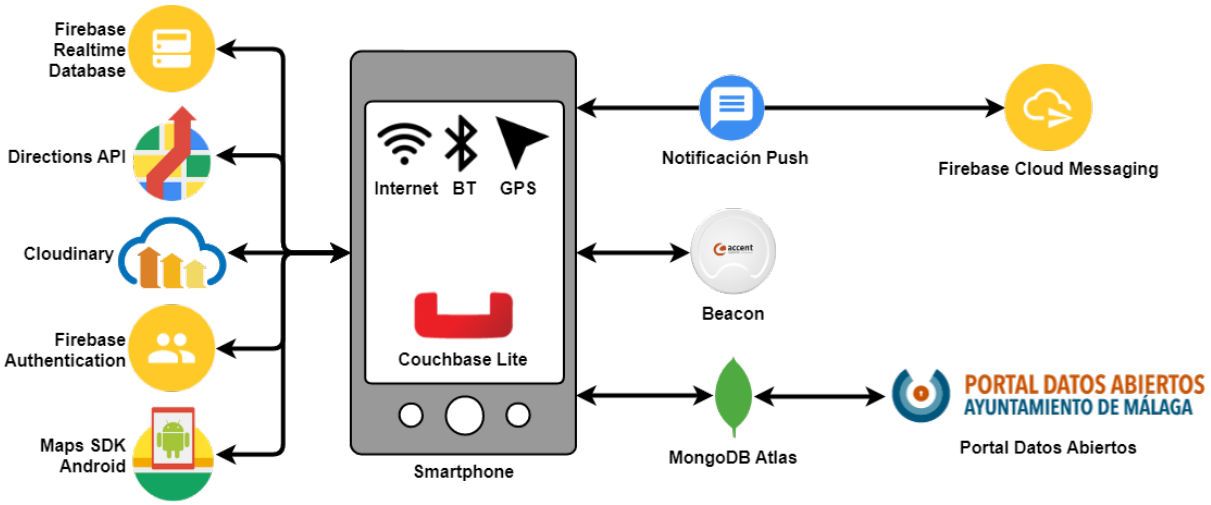


Figura 25: Diagrama de la arquitectura básica del proyecto

También, se requiere de una cuenta en Cloudinary para almacenar las fotografías de los usuarios, y debe disponer de *beacons* previamente configurados.

Como se explica en el capítulo 2, un *beacon* es un dispositivo transmisor de señales de radio periódicas o continuas que incluyen información como su identificación o ubicación, haciendo uso de *Bluetooth Low Energy* (BLE) para comunicarse. En el proyecto, se aprovecha su funcionalidad para llevar a cabo la detección de presencia y posicionamiento en los interiores de los centros deportivos, controlando si el usuario está entrenando y en qué sala se encuentra (en el caso de que existan diversos *beacons* repartidos en el interior de la instalación deportiva).

Bases de datos

Aunque este proyecto se base en el modelo de computación móvil distribuida sin necesidad de un servidor central que almacene los datos, ha sido necesario emplear almacenamiento en la nube.

Por un lado, podría resultar coherente utilizar solamente dos bases de datos, una para almacenar los datos del usuario de manera local en su teléfono móvil, y otra para almacenar en la nube aquellos datos que sean necesarios. Sin embargo, se han empleado tres sistemas de bases de datos y se va a justificar su utilización a continuación.

En primer lugar, se emplea **Couchbase Lite** como base de datos local en el teléfono del usuario en la que se almacena su representante virtual y también se guarda la información de todas las instalaciones deportivas, que es descargada una única vez de la base de datos en la nube de MongoDB Atlas para no tener que consultar esos datos continuamente.

Como se acaba de comentar, la base de datos de **MongoDB Atlas** almacena los datos correspondientes a las instalaciones deportivas de Málaga. Estos datos se obtienen en crudo, sin procesar, del Portal de Datos Abiertos del Ayuntamiento de Málaga [32]. Sin embargo, a través de un código que se ha desarrollado en Python, se han limpiado estos datos y filtrado para obtener únicamente la información que interesa y, acto seguido, dichos datos procesados han sido almacenados en una base de datos MongoDB en la nube. Este trabajo fue llevado a cabo en la iteración 0 del proyecto.

Por último, se hace uso de la base de datos **Realtime Database de Firebase** que, como su nombre indica, sincroniza los datos en tiempo real. Esta base de datos se ha empleado para almacenar los usuarios que se registran en la aplicación en tiempo real, así como para guardar

algunos de sus datos para mostrarlos al consultar el listado de todos los usuarios registrados en la aplicación. Podría haberse utilizado la base de datos MongoDB para almacenar estos datos pero se ha preferido aprovechar el potencial que Firebase ofrece a la hora de integrar los datos de usuarios autenticados con su base de datos en tiempo real. Es por esta razón por la que se emplean dos bases de datos en la nube.

Servicios de Firebase

Para desarrollar el proyecto se han hecho uso de tres servicios de Firebase muy útiles. En resumen, han sido empleados para identificar a los usuarios dentro de la aplicación, almacenar sus datos en tiempo real y enviar notificaciones a los teléfonos móviles de los usuarios.

- **Authentication** ha permitido identificar a los usuarios empleando Google como proveedor, de manera que los usuarios pueden registrarse o iniciar sesión en el sistema haciendo uso de una cuenta de Google.
- **Realtime Database**, como ya se ha comentando en la sección anterior, se ha empleado para almacenar a los usuarios en tiempo real, con la información necesaria.
- **Cloud Messaging** ha sido el responsable del envío y recepción de notificaciones *push* entre los dispositivos móviles de los distintos usuarios de la aplicación, mediante las cuales se han enviado solicitudes de amistad así como el envío de datos para sincronizarlos con la base de datos local de los usuarios.

APIs de Google

De entre la gran variedad de servicios y APIs que ofrece Google, se han empleado dos: el SDK de Maps para Android y la API de Direcciones.

Con el **SDK de Maps para Android**, se ha podido agregar el mapa a la aplicación desarrollada, ofreciendo información adicional sobre las ubicaciones y facilitando la interacción del usuario al agregar marcadores y rutas a este.

La **API de Direcciones** es un servicio que utiliza solicitudes HTTP para mostrar instrucciones de cómo llegar de una ubicación a otra, en formato JSON o XML. Este servicio se ha empleado para trazar las rutas de senderismo y para obtener la distancia exacta por carretera entre la ubicación del usuario y las instalaciones deportivas.

5.2. Modelo de datos

Esta sección sirve para explicar el modelo de datos, esto es, la estructura de las tres bases de datos empleadas en este proyecto.

MongoDB

La base de datos en la nube de MongoDB recopila la información de los datos obtenidos del Portal de Datos Abiertos del Ayuntamiento de Málaga tras haber sido procesados por el código Python desarrollado. Se han distinguido cinco colecciones: *athletics*, *bodybuildingequipment*, *hiking*, *sportscentre* y *workoutequipment*.

Todas están completas por documentos con el mismo formato, aunque se ha querido distinguir las colecciones para agruparlos por tipo de instalación deportiva y así sirve de ayuda de manera visual. Un ejemplo de documento se observa en la siguiente figura:

Figura 26: Estructura empleada en la base de datos de MongoDB

De cada instalación deportiva se está guardando la siguiente información:

- `_id`: Identificador propio del documento autogenerado por MongoDB.
- `id`: Identificador de la instalación deportiva asignado por el Portal de Datos Abiertos.
- `name`: Nombre de la instalación deportiva.
- `address`: Dirección de la instalación deportiva.
- `type`: Tipo de la instalación deportiva.
- `latitude`: Coordenada latitud de la instalación deportiva.

- `longitude`: Coordenada longitud de la instalación deportiva.
- `url`: Enlace de la instalación deportiva.

Couchbase Lite

La base de datos guardada localmente en el teléfono del usuario (Couchbase Lite), almacena las instalaciones deportivas así como el representante virtual o avatar digital del usuario, tal y como se explica anteriormente.

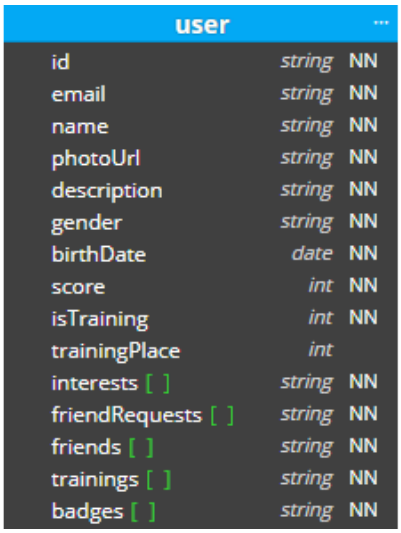


Figura 27: Estructura empleada en la base de datos de Couchbase Lite

Se prescinde de explicar el documento correspondiente a una instalación deportiva ya que posee la misma estructura que la mostrada en el apartado anterior, a diferencia de que no contiene el identificador autogenerado por MongoDB, ya que no es de interés para la base de datos local.

Por tanto, la estructura del documento que almacena al representante virtual del usuario está compuesta de los siguientes campos:

- `id`: Identificador único del usuario asignado por Firebase Authentication.
- `email`: Dirección de correo electrónico con la que el usuario se ha registrado.
- `name`: Nombre completo del usuario.
- `photoUrl`: Enlace de la fotografía de perfil del usuario.

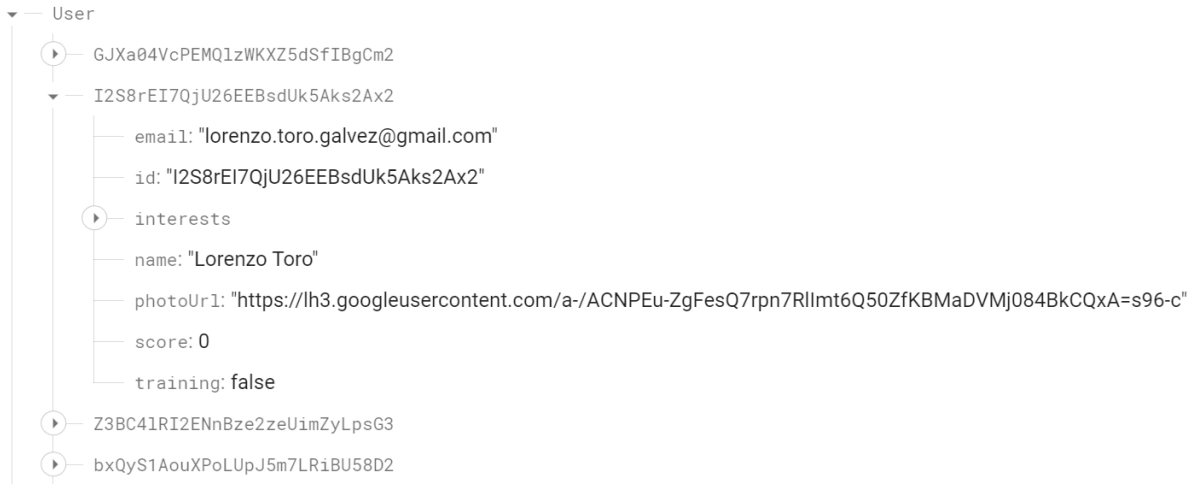
- `description`: Descripción del usuario.
- `gender`: Género del usuario.
- `birthDate`: Fecha de nacimiento del usuario.
- `score`: Puntuación total del usuario.
- `isTraining`: Indica si el usuario está entrenando o no.
- `trainingPlace`: Identificador del lugar donde está entrenando el usuario.
- `interests[]`: Lista de intereses del usuario.
- `friendRequests[]`: Lista de solicitudes de amistad recibidas pendientes por aceptar o denegar. Cada solicitud de amistad recibida contenida en este *array* se representa mediante un JSON.
- `friends[]`: Lista de amigos del usuario. Cada amigo contenido en este *array* se representa mediante un JSON.
- `trainings[]`: Lista de entrenamientos del usuario. Cada entrenamiento contenido en este *array* se representa mediante un JSON.
- `badges[]`: Lista de insignias del usuario. Cada insignia contenida en este *array* se representa mediante un JSON.

Firebase Realtime Database

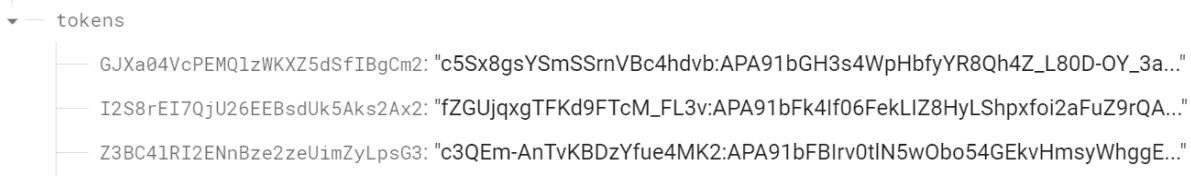
En esta base de datos se recopila cierta información de todos los usuarios de la aplicación en tiempo real, sincronizándose así con todos los dispositivos. Además, se utiliza un diccionario en el que la clave es el identificador único del usuario, y el valor es un *token*, que es el identificador del dispositivo en el que el usuario está autenticado. Dicho *token* es único y será necesario conocerlo para poder enviar notificaciones *push* a los usuarios de la aplicación, distinguiéndose cada uno por su token único e identificativo.

Un usuario registrado en la aplicación y estando autenticado en algún dispositivo y, por tanto, perteneciente a la colección de usuarios aquí registrados, posee la siguiente estructura:

- `id`: Identificador único del usuario asignado por Firebase Authentication.
- `email`: Dirección de correo electrónico con la que el usuario se ha registrado.
- `name`: Nombre completo del usuario.
- `photoUrl`: Enlace de la fotografía de perfil del usuario.
- `score`: Puntuación total del usuario.
- `training`: Indica si el usuario está entrenando o no.
- `trainingPlace`: Identificador del lugar donde está entrenando el usuario.
- `interests`: Lista de intereses del usuario.



(a) Estructura de un usuario



(b) Diccionario de tokens formado por id:token

Figura 28: Estructura empleada en la base de datos Realtime Database

5.3. Modelo de objetos

En este apartado se expone el diagrama de clases (Figura 29) correspondiente al modelo de objetos. Dichos objetos son los manejados dentro de la aplicación y provenientes de la base de datos local del usuario.

Cada uno de los atributos de estos modelos corresponde a los campos del documento que representa al avatar digital del usuario.

Sin embargo, en el diagrama no se incluyen algunos métodos como pueden ser los *getters* y *setters*, para no producir una sobrecarga visual en la imagen, además de no incluir algunos constructores que darían como resultado una imagen muy alargada horizontalmente.

Las clases que se representan son las siguientes:

- **User**: Es el corazón de la aplicación. Representa a un usuario y contiene todos los campos mencionados en la base de datos local del apartado anterior. En cuanto a operaciones, contiene el constructor vacío y algunos constructores por parámetros, además de contener cuatro métodos más: obtención del usuario de la base de datos local, registro del avatar en local, actualización del avatar en local, y actualización del usuario en la nube.

- **SportFacility**: Representa una instalación deportiva y contiene los mismos campos mencionados en el apartado anterior. Posee una operación que opera con la base de datos local para obtener todas las instalaciones deportivas en función de su tipo.

- **Training**: Representa un entrenamiento realizado por un usuario. Sus campos son: nombre, descripción, fecha de realización del entrenamiento, duración, puntuación obtenida e instalación deportiva en la que se ha llevado a cabo.

- **Badge**: Hace referencia a una insignia que puede ser obtenida por un usuario. Se caracteriza por la puntuación requerida para conseguirla, el icono que la representa de manera visual, y una breve descripción.

- **Friend**: Un amigo se representa a través de esta clase. Está formado por un usuario que es considerado amigo y por la última fecha de actualización de sus datos.

- **FriendRequest**: Se refiere a una solicitud de amistad recibida. Se compone del usuario que envía la petición y de la fecha en la que se ha recibido.

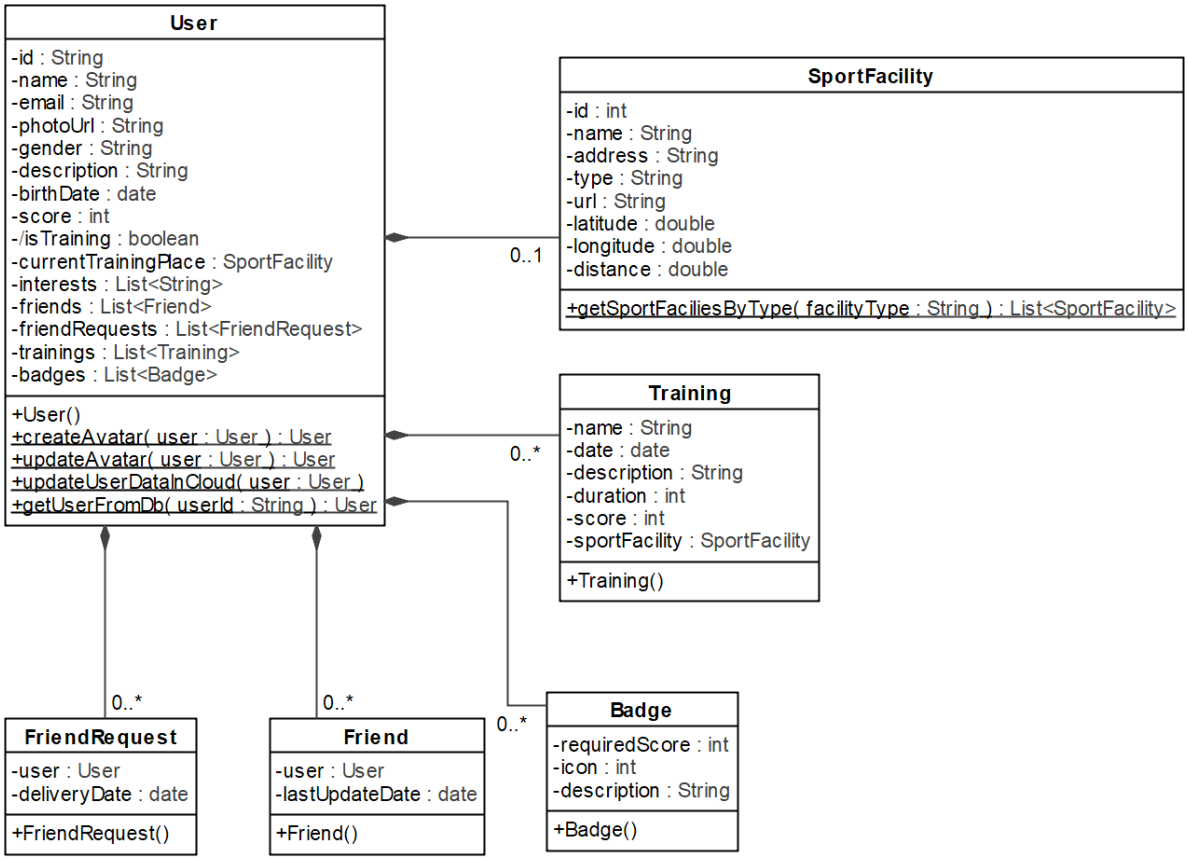


Figura 29: Diagrama de clases de los modelos usados

5.4. Proceso de diseño

A veces, el sentido de la vista es el más poderoso para determinar si el usuario está contento con lo que ve. Por esta razón, este apartado explica de manera breve las decisiones de diseño, en cuanto a colores, iconos y logotipo; además de incluir maquetas de las interfaces y un diagrama de navegación, que permiten construir un prototipo casi definitivo que muestra el funcionamiento del sistema a desarrollar.

5.4.1. Línea de diseño

Durante el desarrollo del proyecto, en todo momento se ha buscado un diseño sencillo y fino, que facilite al usuario la utilización de la aplicación, de modo que pueda comprenderla en su totalidad, y no encuentre iconos o figuras que le resulten ambiguos o difíciles de entender. Aunque no haya sido fácil decidirse por los colores, el resultado parece ser satisfactorio.

Además, se busca que no exista un contraste visual demasiado amplio, para permitir que el texto pueda distinguirse del fondo en todo momento. Se emplea para ello un color suave de fondo y un color fuerte de texto.

En cuanto a los colores empleados en la aplicación, destacan:

- **Azul primario (`5482FC`)**: El color que más resalta en la aplicación. Se emplea en los menús y en la mayoría de botones, así como en algunos elementos textuales. Es un color que destaca, resulta agradable a la vista y permite diferenciarse entre los demás elementos.

- **Azul suave (`EAEFF9`)**: Es el color suave del fondo de todas las interfaces. Realmente es como si fuera blanco, pero basado en la escala del azul, para no salirse de la dinámica azulada.

- **Azul secundario (`132551`)**: Un color poco usado en la aplicación pero necesario en algunos elementos textuales y no textuales, para distinguirlos de otros y no producir una sobrecarga visual para el usuario.

En cuanto al icono principal o logotipo de la aplicación, se planteó un icono entendible que recogiese varios conceptos relacionados con el deporte al mismo tiempo, que fuese sencillo y con poco detalle. Tras navegar mucho por Internet, proponer distintas opciones, editar algunas otras, y crear iconos desde cero, el icono ganador apareció.

Así, se muestra el color azul primario de fondo, y se incluyen unos iconos referentes a una mancuerna, una zapatilla y una camiseta deportiva. Dicho icono fue extraído de Flaticon [33], al igual que otros de los iconos y figuras empleados en la aplicación.

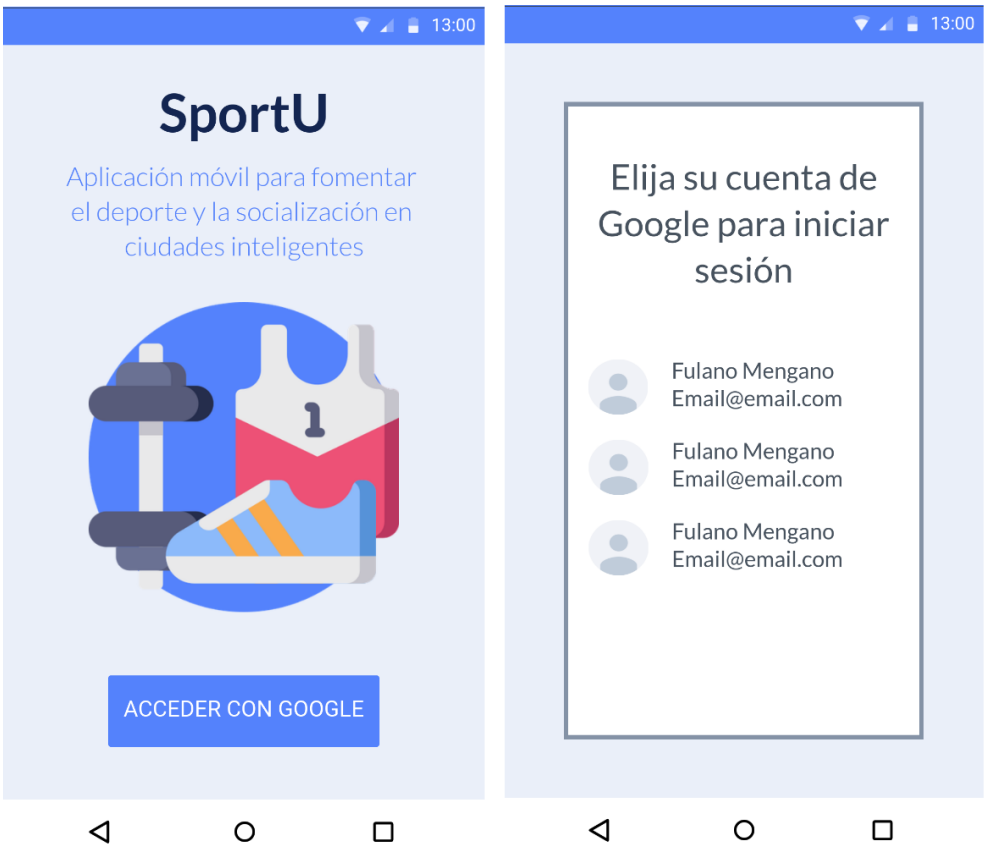
Figura 30: Icono de la aplicación

5.4.2. Prototipado

Este paso es fundamental en el desarrollo de cualquier aplicación debido a que plantea una visión general de cómo se va a representar la aplicación, cómo va a interactuar el usuario con ella y los elementos y componentes que va a necesitar. A través de algunos de los bocetos mostrados a continuación, se ha dado vida a la aplicación.

Cabe destacar que son bocetos, puede que no coincidan con el resultado final o que se tomen decisiones que impliquen cambios en el futuro de la aplicación, viéndose afectada también alguna de las interfaces de esta.

En la Figura 31 se ven las pantallas de inicio de sesión de la aplicación, en las que el usuario debe seleccionar su cuenta de Google con la que desea iniciar sesión o registrarse en el sistema.



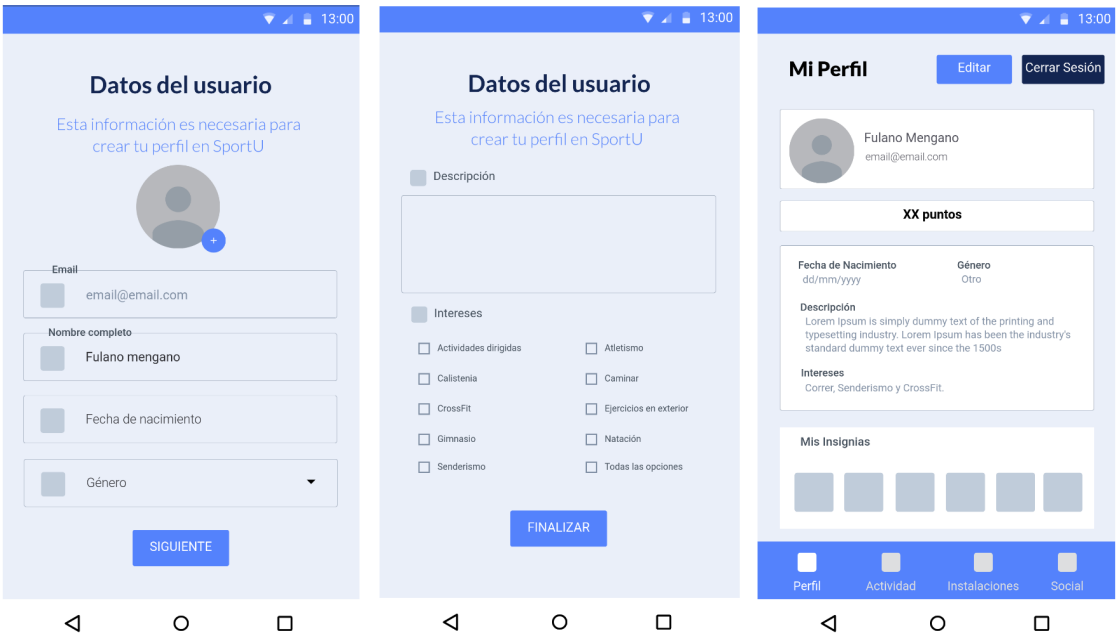
(a) Acceso inicial (b) Selección de cuenta

Figura 31: Pantallas correspondientes al acceso a la aplicación

Acto seguido, pueden ocurrir dos escenarios. El primero consiste en que el usuario no disponga de una cuenta en el sistema, por lo cual es redirigido a las pantallas de registro,

mostradas en las imágenes (a) y (b) de la Figura 32, en las cuales debe cumplimentar los campos con su nombre, fecha de nacimiento, género, descripción e intereses en cuanto a actividades deportivas.

El segundo escenario que puede darse en este caso es que el usuario ya disponga de una cuenta en el sistema, por tanto se inicia la sesión y es redirigido a la pantalla de su perfil, mostrada en la captura (c) de la Figura 32.



(a) Registro parte 1 (b) Registro parte 2 (c) Perfil del usuario

Figura 32: Pantallas correspondientes al registro y perfil del usuario

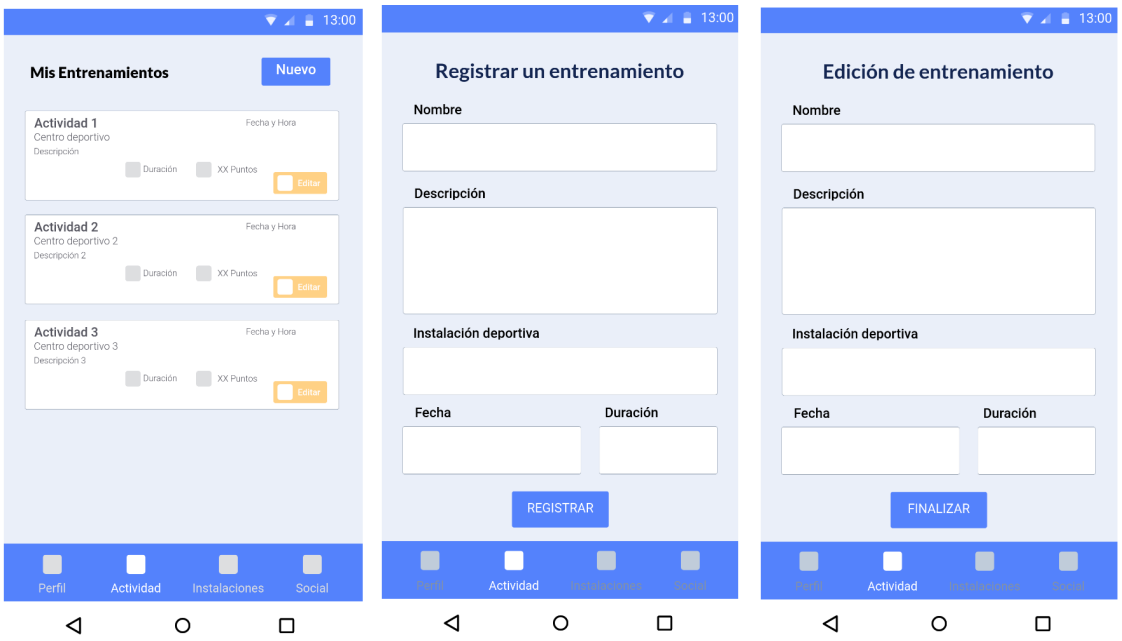
Un menú de navegación inferior formado por cuatro opciones permite al usuario moverse entre las distintas pantallas de la aplicación: “Perfil”, “Actividad”, “Instalaciones” y “Social”.

Al iniciar sesión o al seleccionar la primera opción “Perfil”, se le muestra al usuario la interfaz de su perfil con sus datos personales, su puntuación y sus insignias, así como las opciones de editar sus datos o cerrar la sesión.

Las interfaces de edición de perfil son idénticas a las mostradas en las pantallas (a) y (b) de la Figura 32 y el perfil es mostrado de la misma manera que en la pantalla (c) de la figura mencionada.

Cuando el usuario selecciona la opción “Actividad” del menú de navegación, se muestra una lista con sus entrenamientos registrados. Cada entrenamiento contiene información como

el nombre, la instalación deportiva en la que ha sido realizado, un breve texto describiendo el entrenamiento o las sensaciones del usuario, la duración, los puntos obtenidos, y la fecha y hora de realización. Además, permite al usuario editar alguno de los entrenamientos registrados así como introducir uno nuevo. Los bocetos correspondientes se muestran en la Figura 33.



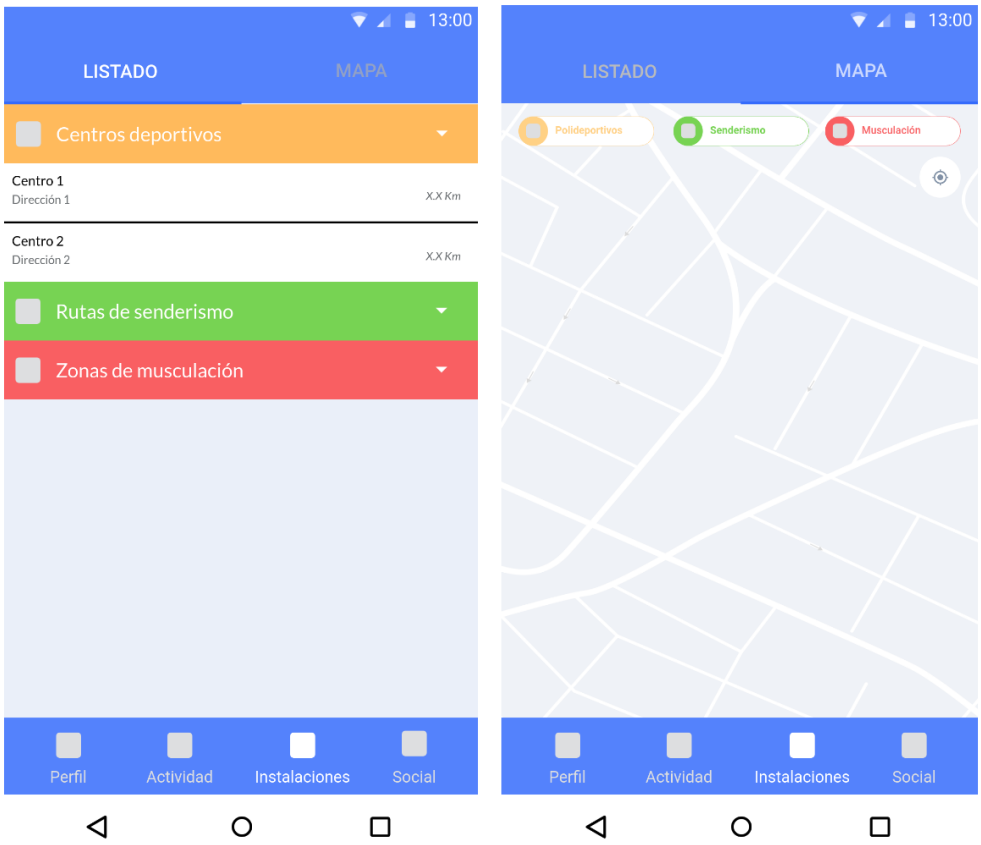
(a) Entrenamientos (b) Registrar entrenamiento (c) Editar entrenamiento

Figura 33: Pantallas correspondientes a los entrenamientos, su registro y edición

Al seleccionar la opción “Instalaciones” del menú inferior de navegación, el usuario es redirigido a una pantalla que dispone de un menú superior de navegación con dos opciones disponibles: “Listado” y “Mapa”.

Por defecto, la primera pantalla mostrada será la correspondiente al listado de las instalaciones deportivas. Dichas instalaciones se mostrarán con su nombre, dirección y distancia al usuario, ordenadas en función de este último atributo, de menor a mayor.

Por otro lado, al seleccionar la opción “Mapa”, se mostrará un mapa centrado en la ciudad de Málaga con multitud de marcadores indicando los centros deportivos, las rutas de senderismo y las zonas de musculación, ofreciendo también unos filtros para ocultar o mostrar dichas instalaciones. Pueden observarse las maquetas de estas pantallas en la Figura 34.



(a) Listado de instalaciones (b) Mapa con las instalaciones

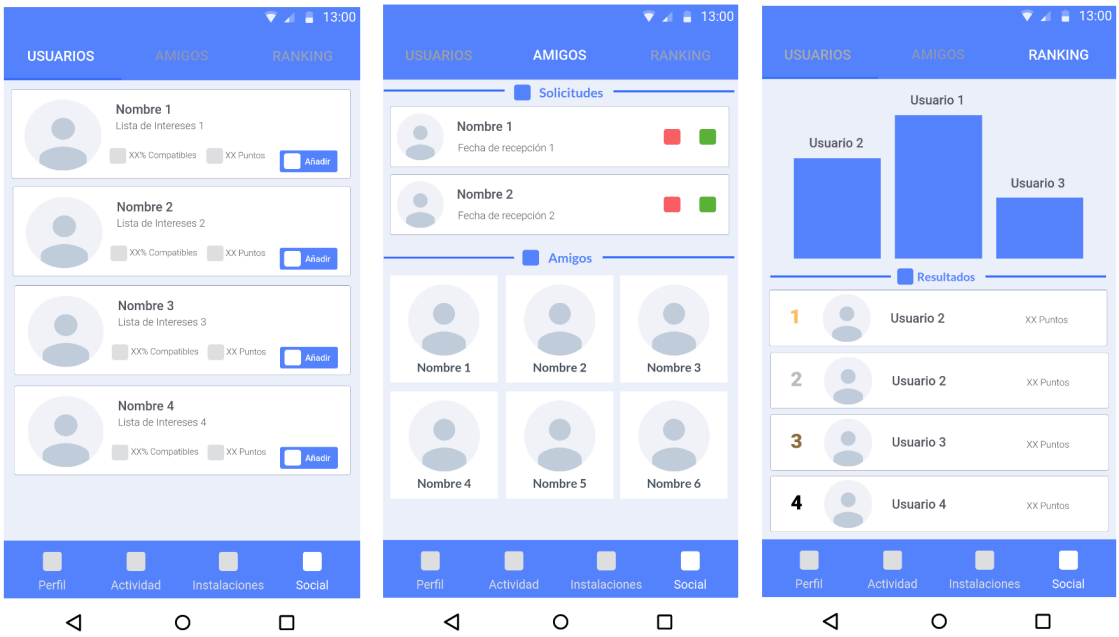
Figura 34: Pantallas correspondientes a las instalaciones deportivas

Por último, si el usuario selecciona la opción “Social” del menú de navegación inferior, se mostrará una pantalla que dispone de un menú superior con tres opciones: “Usuarios”, “Amigos” y “Ranking”, estando seleccionada por defecto la primera opción.

Así, la primera opción de “Usuarios”, observada en la imagen (a) de la Figura 56 mostrará un listado con todos los usuarios registrados en la aplicación, con nombre, intereses, porcentaje de compatibilidad y puntuación, existiendo la posibilidad de enviar una solicitud de amistad a cualquiera de los usuarios. Dicho listado será ordenado en base al porcentaje de compatibilidad, que se calcula comparando los intereses del usuario con los intereses de otro usuario.

Por otra parte, al seleccionar la opción “Amigos”, se mostrarán todos los amigos del usuario así como las solicitudes de amistad recibidas pendientes, permitiendo aceptar o denegar las solicitudes recibidas así como poder acceder al perfil de los amigos para consultar su información y sus entrenamientos.

Por último, se muestra el *ranking* con los usuarios ordenados en base a su puntuación global en la aplicación. Estas dos últimas secciones se plasman en las pantallas (b) y (c) de la Figura 56.



(a) Todos los usuarios (b) Amigos del usuario (c) Ranking

Figura 35: Pantallas correspondientes a la parte social de la aplicación

5.4.3. Diagrama de navegación prototipado

En esta breve sección se muestra un diagrama de navegación de la aplicación obtenido a través de los prototipos diseñados.

Este diagrama es un gráfico estructurado que representa el sistema y sus interfaces y cómo se puede navegar a través de este. Es un recurso muy útil para obtener una visión general de la aplicación y generar un prototipo que sirve de ayuda para el sistema final.

Cabe destacar que se han omitido las navegaciones en los distintos menús, solamente se muestran una vez, para así no sobrecargar con demasiadas líneas la imagen. Además, dicho diagrama no es definitivo, al estar construido en base a las maquetas realizadas, por lo que puede experimentar cualquier tipo de modificación.

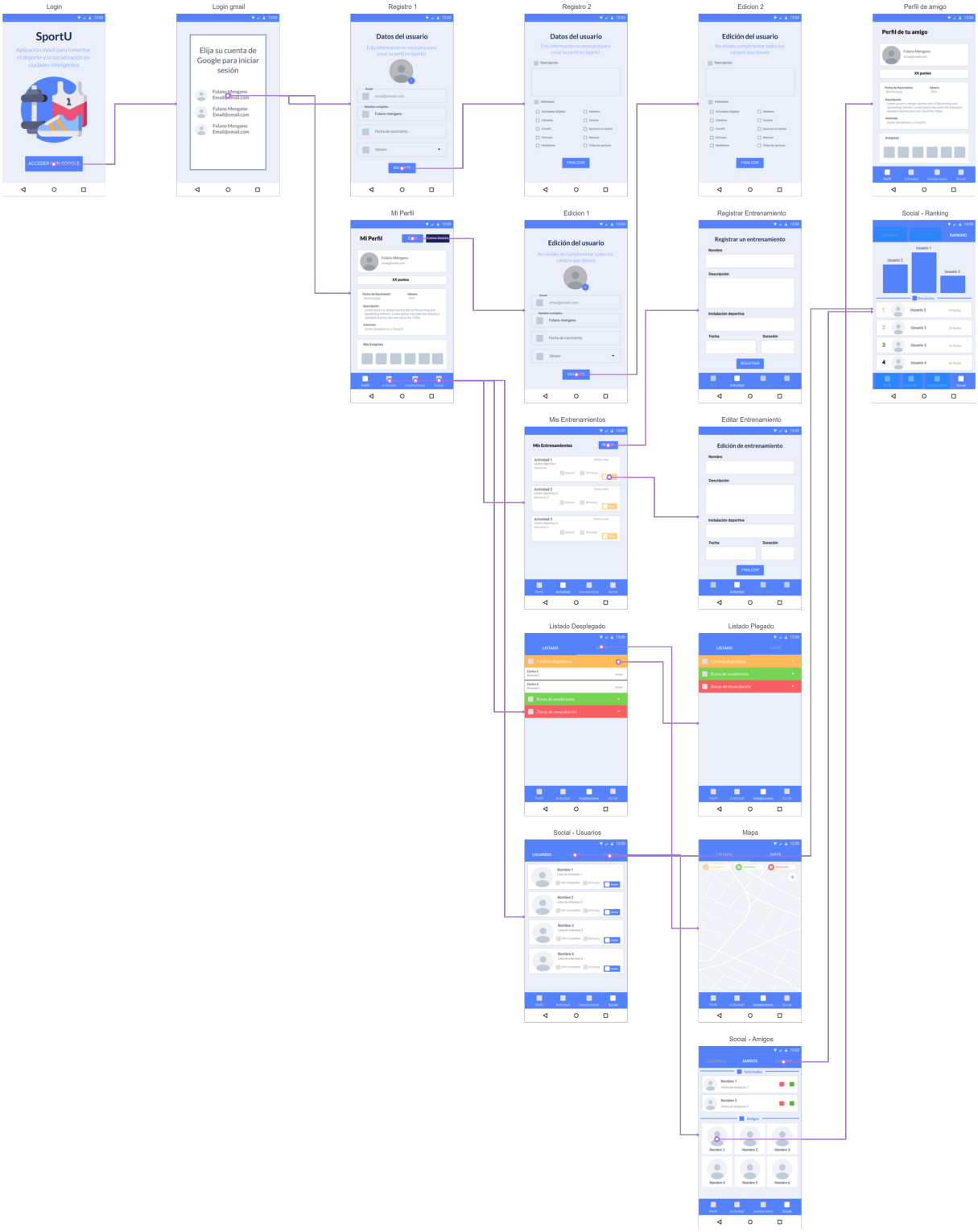


Figura 36: Diagrama de navegación en base a las maquetas

6

Desarrollo e implementación

En este capítulo se comentarán las librerías o dependencias empleadas en la aplicación y se analizará el trabajo realizado en cada una de las iteraciones que forman parte del desarrollo del proyecto así como las decisiones que se han tomado y los problemas que han podido encontrarse.

6.1. Dependencias utilizadas

En ocasiones, con las limitaciones existentes en el lenguaje, no se puede implementar todo lo que un desarrollador pueda imaginar y, para ello, otros desarrolladores se encargan de crear librerías o dependencias que facilitan el trabajo de los demás. Es por ello que, esta sección, recoge un listado de las dependencias que se han empleado en la aplicación y una breve explicación acerca de su uso.

- `com.google.android.gms:play-services-maps`: Necesario para visualizar los mapas, es el SDK de Maps para Android.

- `com.google.android.gms:play-services-location`: Es requerido para hacer uso de los servicios de geolocalización.

- `com.google.android.gms:play-services-auth`: Dependencia utilizada para hacer uso de los servicios de autenticación de Google.

- `com.couchbase.lite:couchbase-lite-android`: Necesaria para hacer uso de Couchbase Lite en el dispositivo móvil del usuario.

- `com.google.firebase:firebase-bom`: Permite administrar todas las versiones de las bibliotecas de Firebase con solo especificar la versión de BoM.
- `com.google.firebase:firebase-analytics`: Proporciona reportes ilimitados de hasta 500 eventos diferentes, como estadísticas de usuarios o uso de la aplicación.
- `com.google.firebase:firebase-auth`: Dependencia para integrar Firebase Authentication con un sistema personalizado.
- `com.google.firebase:firebase-database`: Esta librería es necesaria para trabajar con la base de datos en tiempo real.
- `com.google.firebase:firebase-messaging`: Se emplea para gestionar toda la mensajería cloud, esto es, las notificaciones *push*.
- `de.hdodenhof:circleimageview`: Librería empleada para construir elementos de tipo ImageView con forma circular, para las fotos de perfil e insignias.
- `com.squareup.picasso:picasso`: Dependencia utilizada para descargar y cachear las imágenes alojadas en Cloudinary en tan solo una línea de código.
- `com.github.dhaval2404:imagepicker`: Se emplea para seleccionar una imagen de la galería o tomar una fotografía empleando la cámara, para poner fotografía de perfil del usuario.
- `com.cloudinary:cloudinary-android`: Esta dependencia permite trabajar con el SDK de Cloudinary, permitiendo subir contenido multimedia a dicha nube.
- `com.google.code.gson:gson`: Librería Java que se emplea para convertir objetos Java en su representación JSON correspondiente, y viceversa.
- `com.android.volley:volley`: Biblioteca HTTP que facilita el uso de redes en aplicaciones, empleada para enviar las notificaciones *push*.
- `org.altbeacon:android-beacon-library`: Se trata de una biblioteca de Android que proporciona una API para interactuar con *beacons*.

6.2. Iteraciones realizadas

Como se ha mencionado en el capítulo 3, este proyecto sigue una metodología Scrum en la que el estudiante ocupa todos los roles excepto uno. Así, en este apartado se explica cómo se ha desarrollado cada iteración adjuntando los requisitos implementados, y qué problemas se han encontrado.

6.2.1. Iteración 0

En esta iteración inicial hubo que tomar muchas decisiones que han sido clave para la ejecución del proyecto, ya que en ese momento no existe una visión general de lo que se busca, sino que se plantea una situación difícil. Sin embargo, gracias a la ejecución de distintas tareas, se pudo abordar con éxito.

En primer lugar, el objetivo principal de la iteración es documentarse, investigar y aprender acerca de las tecnologías y recursos que se van a utilizar así como del desarrollo de aplicaciones para teléfonos con sistema operativo Android. A pesar de haber aprendido desarrollo Android en el último curso del grado, los conocimientos adquiridos no llegan a cubrir las funcionalidades que se implementan en este proyecto. El estudiante partía con la ventaja de que el lenguaje de programación en el que se iba a desarrollar la aplicación es Java, debido a que cuenta con cierta experiencia.

En cuanto a las tecnologías y recursos mencionados en el párrafo anterior, en concreto se trata de notificaciones *push*, el uso de sensores GPS y Bluetooth y la configuración y utilización de los *beacons*.

Para aprender a utilizar las notificaciones *push*, primero se hizo uso de la plataforma de Firebase Cloud Messaging, la cual permite enviar notificaciones desde una interfaz gráfica a los dispositivos deseados que usen la aplicación. Un paso previo a la utilización de dicho servicio era la creación de un proyecto Firebase asociado a la cuenta de Google del estudiante. La primera toma de contacto con dicho servicio fue exitosa, aunque las notificaciones que se enviaban llegaban a los teléfonos móviles con un retraso de 4 a 7 minutos, sin motivos aparentes. Más adelante, se aprendió a enviar notificaciones push entre dispositivos, a nivel de código, sin necesidad de usar la plataforma.

En cuanto al uso de los sensores GPS y Bluetooth, se extrajo documentación y algunos

proyectos ofrecidos por Google para investigar en la utilización de dichos sensores, además de visualizar vídeos explicativos.

Referente a los *beacons*, los dispositivos que permiten identificar y localizar al usuario en interiores, se empleó la documentación oficial para obtener información, así como los manuales de usuario y configuración ofrecidos. Del mismo modo, se disponía de acceso a un proyecto realizado anteriormente con *beacons* por otro estudiante, así que podía consultarse el código para tener una aproximación de lo que se buscaba.

A la vez que se iba llevando a cabo todo este proceso de investigación, también se produjo la extracción y clasificación, de manera provisional, de los requisitos funcionales y no funcionales de la aplicación, lo que permitió establecer una pequeña planificación en Trello en base a las funcionalidades a implementar.

Para obtener una idea de lo que sería la aplicación, también se comenzaron a desarrollar gran cantidad de bocetos de las interfaces haciendo uso de la herramienta de prototipado Marvel.

La última tarea que se llevó a cabo en esta primera iteración fue la construcción inicial y provisional de la base de datos local que sería el representante virtual o avatar digital del usuario de la aplicación, almacenado en su teléfono móvil, como ya se ha explicado anteriormente.

6.2.2. Iteración 1

La iteración 1 tenía el objetivo de llevar a cabo algunas tareas fundamentales en este proyecto, como son el registro y acceso de usuarios, y la visualización de las instalaciones deportivas a través de un listado. Así, aunque puedan parecer pocos, los requisitos considerados para esta iteración fueron:

- RF-USR-1: Un usuario puede registrarse en la aplicación a través de Google.
- RF-USR-2: Un usuario puede iniciar sesión en la aplicación a través de Google.
- RF-USR-3: Un usuario puede cerrar sesión en la aplicación.
- RF-INS-1: Un usuario puede consultar en un listado todas las instalaciones deportivas de la ciudad de Málaga: centros deportivos, rutas de senderismo y zonas de musculación, ordenadas por cercanía a este.

Autenticación en la aplicación

Para llevar a cabo los tres primeros, se hizo uso de Firebase Authentication junto a su documentación oficial y dos vídeos explicativos para entender el funcionamiento de este servicio. De esta manera, se pudo gestionar la autenticación de los usuarios fácilmente.

Una ventaja de Authentication es que te permite utilizar una gran cantidad de proveedores de acceso simultáneamente, entre los cuales se encuentran: acceso con correo electrónico y contraseña, con número de teléfono, a través de Google, Facebook, Twitter, GitHub, Microsoft, y varios más. En el caso de SportU, únicamente se ha permitido el registro en la aplicación a través del proveedor de Google.

Identificador	Proveedores	Fecha de creación ↓	Fecha de acceso	UID de usuario
carlostorogggg@gmail.com	G	16 sept 2022	24 sept 2022	bxQyS1AouXPoLUpJ5m7LRiBU58...
silvianavarrommm@gmail....	G	16 sept 2022	16 sept 2022	eRkev0ERS6TLyWm1CHpHl21tND...
lorenzo.toro.galvez@gmail....	G	16 sept 2022	24 sept 2022	I2S8rEI7QjU26EEBsdUk5Aks2Ax2

Figura 37: Usuarios registrados en la aplicación

Así, gracias a dicho servicio, cuando un usuario accede empleando su cuenta de Google, se añade una fila con su información a una tabla de usuarios registrados: identificador, proveedor mediante el cual accede, fecha de creación, fecha de acceso y el UID de usuario, que es el identificador único de usuario a través del cual se mantendrán a los usuarios distinguidos uno de otro. Esta tabla puede observarse en la Figura 37.

Tras seleccionar la dirección de correo con la que el usuario quiere registrarse en la aplicación, es redirigido al formulario de creación de su perfil, basado en una secuencia de dos actividades Android, a través de las cuales el usuario selecciona una foto para su perfil, introduce sus datos personales y selecciona sus intereses deportivos.

Para poder mostrar una imagen circular se hace uso de *CircleImageView*, librería mencionada anteriormente, y para poder tomar una fotografía o seleccionar de la galería se emplea la dependencia *ImagePicker*.

Extracción, procesamiento y listado de los Datos Abiertos

Por otro lado, el requisito que también se implementa en esta iteración es el de obtener los datos de las instalaciones deportivas de Málaga y mostrarlos en un listado dentro de la aplicación.

Primero, fue necesario obtener los recursos JSON del Ayuntamiento de Málaga que contenían toda la información de las instalaciones deportivas deseadas. Se querían obtener datos de los 18 centros deportivos ofrecidos, 7 aparatos de *workout*, 2 pistas de atletismo, 77 puntos que conforman distintas de senderismo y 172 zonas de musculación.

Obtener todos estos datos abiertos del Portal de Datos de Abiertos no fue tarea fácil ya que el servidor del Ayuntamiento de Málaga denegaba constantemente el acceso tras realizar una gran cantidad de peticiones en un corto periodo de tiempo, mostrando una ventana de color rojo con un mensaje indicando "No tiene permiso de acceso: El servidor le ha denegado el acceso al recurso solicitado".

Sin embargo, eran muchos datos para ser mostrados en un mapa en una pantalla de un teléfono móvil ya que iba a producirse una sobrecarga visual excesiva. Es por ello que, aprovechando el script generado en Python que descarga los datos, los limpia y los almacena en la base de datos MongoDB, también se creó un pequeño método que reducía de manera justa las 172 zonas de musculación a su cuarta parte, obteniendo así 43, un número más razonable a la hora de mostrar todas las instalaciones en el mapa. En la Figura 38 pueden observarse algunos fragmentos de código, para verlo en su totalidad se redirige al lector al fichero `openData.py`.

Una vez estos datos fueron procesados y almacenados en la base de datos MongoDB, quedaba el paso de almacenarlos en la base de datos local del usuario y listarlos en una interfaz. Para ello, desde el código Java de la aplicación Android, debía establecerse una conexión con la base de datos MongoDB y se implementaron dos métodos que guardaban los datos en distintos documentos de manera local (mostrado uno de ellos en la Figura 39).

Así, la primera vez que el usuario instala la aplicación, se descargan los datos y no es necesario ninguna vez más que vuelvan a ser descargados, a no ser que el Ayuntamiento los actualice o inserte nuevos, pero esta funcionalidad puede contemplarse a futuro.

```python
def create_dict_from_data(url, data):
    dict = {
        'id': data['properties']['ID'],
        'type': switch_sport_facility_type.get(url),
        'latitude': data['geometry']['coordinates'][1],
        'longitude': data['geometry']['coordinates'][0],
        'url': data['properties']['URL'],
        'address': string.capwords(data['properties']['DIRECCION']),
        'name': string.capwords(data['properties']['NOMBRE'])
    }
    return dict
```

(a) Creación de una instalación deportiva a partir del recurso online

```python
def get_bodybuilding_equipment_by_ascending_distance(list):
    middle_point = calculate_middle_point(list)
    equipment_dict = {}
    for dic in list:
        coords = (dic['latitude'], dic['longitude'])
        distance = geodesic(coords, middle_point).km
        equipment_dict.update({distance: dic})
    equipment_dict = dict(sorted(equipment_dict.items()))
    equipment_list = []
    counter = 0
    for equip in equipment_dict.values():
        if counter % 4 == 0:
            equipment_list.append(equip)
        counter += 1
    return equipment_list
```

(b) Reducción de instalaciones de zonas de musculación

Figura 38: Pantallas correspondientes al acceso a la aplicación

Al tener los datos ya organizados y guardados en local, solo faltaba mostrarlos de manera visual. Para llevar a cabo este proceso, tuvieron que crearse la mayoría de las actividades, los fragmentos y las interfaces que darían vida a la aplicación, para partir de una estructura bien asentada.

De este modo, los datos de las instalaciones deportivas estarían visibles en la sección "Instalaciones", por lo que hizo falta un fragmento para esta sección, que a su vez da la posibilidad de navegar entre dos fragmentos distintos: "Listado" y "Mapa". Por tanto, para listar los datos

en esta sección (y del mismo modo se procedió en el resto de secciones posteriores), se utilizó el componente *RecyclerView* junto a *Adapter*, de modo que facilita que se muestren los datos de manera eficiente, proporcionándoselos y definiendo previamente el aspecto de cada elemento.

```
private void getDataFromMongoDatabase(String collection) {
        MongoCollection<Document> mongoCollection = mongoDatabase.getCollection(collection);
        RealmResultTask<MongoCursor<Document>> findSportFacility = mongoCollection.find().iterator();
        findSportFacility.getAsync(sportFacility -> {
            if (sportFacility.isSuccess()) {
                MongoCursor<Document> results = sportFacility.get();
                while (results.hasNext()) {
                    try {
                        MutableDocument doc = new MutableDocument();
                        Document res = results.next();
                        doc.setInt("id", res.getInteger("id"));
                        doc.setString("type", res.getString("type"));
                        doc.setDouble("latitude", res.getDouble("latitude"));
                        doc.setDouble("longitude", res.getDouble("longitude"));
                        doc.setString("name", res.getString("name"));
                        doc.setString("url", res.getString("url"));
                        doc.setString("address", res.getString("address"));
                        dbManager.getDatabase().save(doc);
                    } catch (CouchbaseLiteException e) {
                        e.printStackTrace();
                    }
                }
            }
        });
    }
```

Figura 39: Obtención y guardado de instalación deportiva en base de datos local

6.2.3. Iteración 2

En esta nueva iteración, se tratan aspectos relacionados con la visualización y edición del perfil de los usuarios, y se implementa la consulta de las instalaciones deportivas en el mapa de la ciudad de Málaga. Los requisitos formales que definen estas funcionalidades son:

- RF-USR-4: Un usuario puede consultar la información de su perfil.
- RF-USR-5: Un usuario puede modificar la información de su perfil.
- RF-INS-2: Un usuario puede visualizar en un mapa de la ciudad de Málaga todas las instalaciones deportivas: centros deportivos, rutas de senderismo y zonas de musculación.
- RF-INS-3: Un usuario puede filtrar en un mapa de la ciudad de Málaga según el tipo de instalación deportiva.

- RF-INS-4: Un usuario puede consultar información (nombre, dirección y ocupación en tiempo real) acerca de la instalación deportiva seleccionada en el mapa de la ciudad de Málaga.

Consulta y edición del perfil

Esta tarea es sencilla y no requirió mucho tiempo. Para la edición del perfil solamente hizo falta reutilizar el código y las interfaces creadas para la creación del perfil, ya que la base es la misma.

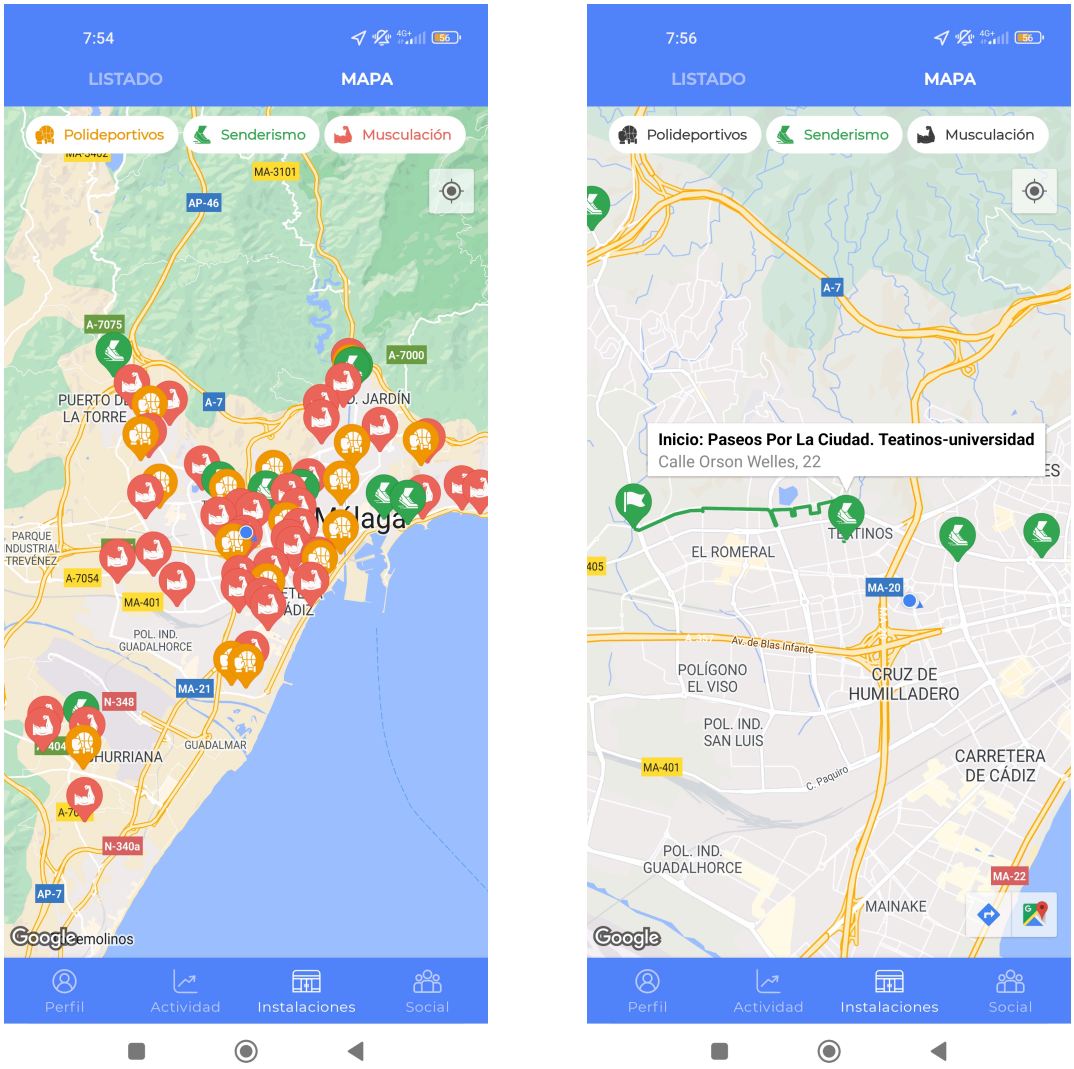
Sin embargo, para mostrar los datos del perfil, sí fue algo más complejo. Se emplearon para ello varios componentes *CardView*, de modo que cada uno se utilizase para mostrar bloques de información: por un lado la fotografía, el email y nombre del usuario, seguido de su puntuación. Por otro lado, una agrupación de su fecha de nacimiento, género, descripción e intereses deportivos. Por último, una pequeña franja con las insignias obtenidas en base a los puntos logrados.

Mapa con las instalaciones deportivas de Málaga

Para conseguir resultados como los mostrados en la Figura 40, hubo que investigar bastante y dedicar muchas horas.

En primer lugar, hubo que implementar el SDK de Maps para Android, así como habilitar los servicios de ubicación, requerir los permisos al usuario para acceder a su ubicación, e incrustar el fragmento del mapa en la vista correspondiente. A la finalización de estas tareas, ya se disponía de un mapa de Málaga centrado en la ubicación del usuario, pero faltaba mostrar los marcadores de todas las instalaciones deportivas y decidir cómo implementar las rutas, porque se disponía de 77 puntos separados, sin orden alguno, todos ellos conformando 10 rutas, por lo que había que distribuir y ordenar los puntos de alguna manera.

Para implementar los marcadores de las instalaciones deportivas, se hacía uso de los datos recogidos en el listado de la iteración anterior, a diferencia de que sufrieron una ligera modificación. Antes existían cinco tipos distintos de instalaciones deportivas, pero para poder mostrar los filtros en el mapa sin sobrecargar mucho la interfaz, se reagruparon formando tres tipos de instalaciones. La reagrupación consistió en añadir las pistas de atletismo junto a los centros deportivos, formando así la agrupación “Polideportivos”, y las máquinas de



(a) Todos los marcadores

(b) Ruta de senderismo desplegada

Figura 40: Pantallas correspondientes al mapa de Málaga

musculación al aire libre con los aparatos de *workout*, formando así la agrupación "Zonas de musculación". Una vez hecho esto, ya se disponían de las instalaciones organizadas en tres tipos, por lo cual se podían insertar tres componentes denominados *Chip* en la parte superior del mapa para permitir mostrar u ocultar en función del tipo de instalación.

A la hora de distribuir los puntos de las rutas de senderismo, es una tarea que debería haberse realizado en el script de Python previamente, pero no se contempló hasta encontrarse con dicha situación, por lo que las rutas son procesadas y reagrupadas en el código Java, obteniendo 10 rutas a partir de los 77 puntos que el Ayuntamiento de Málaga tenía registrados.

Tras estas reagrupaciones comentadas de todas las instalaciones deportivas, resultan un total de 80 aproximadamente.

Para poder construir las rutas de manera que se pintasen en el mapa siguiendo un camino real y no un simple polígono en línea recta, se hizo uso de la API de Direcciones de Google. Gracias a este recurso, puede obtenerse un camino desde un punto a otro, así como las indicaciones para llegar. Incluso se le puede establecer un parámetro con una lista de puntos por los cuales el camino debe pasar. Así, indicándole los puntos que el Ayuntamiento proporciona y estableciendo el inicio y fin de la ruta (véase Figura 41, pudo crearse y pintarse de manera correcta las distintas rutas de senderismo.

Para realizar una petición a la API de Direcciones, se establece el formato de los datos obtenidos (JSON), las coordenadas de origen y destino, el modo de desplazamiento (andando, conduciendo o en bicicleta), y de manera opcional se le facilitan los puntos o *waypoints* que debe atravesar la ruta.

```
private String getUrl(LatLng origin, LatLng destination, String directionMode, ArrayList<LatLng> coordinatesList) {
        String strOrigin = "origin=" + origin.latitude + "," + origin.longitude;
        String strDestination = "destination=" + destination.latitude + "," + destination.longitude;
        String mode = "mode=" + directionMode;
        String output = "json";
        String parameters;

        if (coordinatesList.size() > 2) {
            coordinatesList.remove(0);
            coordinatesList.remove(coordinatesList.size() - 1);
            String waypoints = "waypoints=" + constructAllWaypointsParameter(coordinatesList);
            parameters = strOrigin + "&" + strDestination + "&" + waypoints + "&" + mode;
        } else {
            parameters = strOrigin + "&" + strDestination + "&" + mode;
        }

        String url = "https://maps.googleapis.com/maps/api/directions/" + output + "?" + parameters + "&key=" + getString(R.string.google_maps_key);

        return url;
    }
```

Figura 41: Método que construye la URL para realizar la petición a la API de Direcciones

Aprovechando la potencia de la API de Direcciones, se calcularon las distancias desde la ubicación del usuario hasta cada una de las instalaciones deportivas, información que se muestra en el listado de instalaciones. Para esto, se tenía que calcular, para cada instalación deportiva de las 80 totales, las indicaciones necesarias para desplazarse desde el punto actual hacia la instalación deportiva. Una vez hecho esto, se obtenía la distancia exacta en kilómetros desde un punto a otro, habiendo indicado que el modo de desplazamiento es conduciendo, por lo que obtiene los kilómetros en base a el camino preciso seguido, y no en una simple línea recta trazada en un mapa.

Un factor a considerar era cada cuánto realizar estas peticiones, esto es, cuándo recalcular las distancias en base a cuánto se haya desplazado el usuario desde su última posición, ya que la API de Direcciones de Google lleva un coste asociado, pero se disponía de un crédito gratuito de 300 euros. Inicialmente, al desarrollar el código, no se estableció restricción, es decir, se realizaban 80 peticiones a cada instante, sin considerar si la ubicación del usuario había cambiado o no.

No se le dio más importancia a este detalle y días después es cuando se estableció ese valor para indicar cuándo recalcular las distancias, por ejemplo, al desplazarse el usuario 50 metros respecto de su última ubicación.

Sin embargo, tal y como se puede observar en la Figura 42, se habían realizado un total de 86.000 peticiones, las cuales habían supuesto un coste de 458 euros (véase Figura 43), pero Google había aplicado ciertos descuentos por el volumen de uso, y aún se disponía de una cantidad pequeña de saldo gratuito.

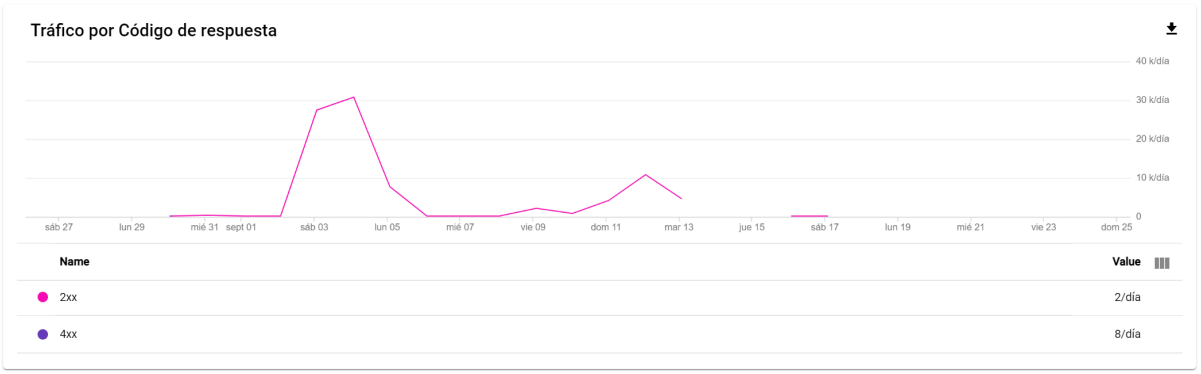


Figura 42: Gráfico con las solicitudes realizadas a la API de Direcciones

Tras percatarse de este hecho, se procedió a calcular las distancias sin hacer uso de la API de Direcciones, para no gastar el saldo gratuito y perder el proyecto en la consola de Google Cloud. Entonces, las distancias pasaron a calcularse en línea recta en un mapa, por lo cual no son exactas, pero sirven para establecer una ordenación en cuanto a cercanía al usuario.

De este modo, también se ahorró el uso de 80 tareas a las cuales había que esperar para obtener así el resultado de las distancias. La implementación pasó de ser grandes bloques de código a dos simples líneas en las que se emplea el método `distanceTo()` de la clase `Location`.

Figura 43: Gráfico con el coste de las solicitudes realizadas a la API de Direcciones

6.2.4. Iteración 3

Esta iteración se dedica a implementar las funcionalidades referentes al registro de actividad deportiva realizada así como su consulta, edición y la obtención de puntos. Por otro lado, se incorpora el listado que contiene a todos los usuarios de la aplicación. Los requisitos asociados a dichas funciones son los siguientes:

- RF-ACT-1: Un usuario puede consultar su registro de actividad deportiva realizada.
- RF-ACT-2: Un usuario puede registrar información de una actividad deportiva realizada.
- RF-ACT-3: Un usuario puede editar la información de una actividad deportiva registrada.
- RF-ACT-4: Un usuario puede eliminar una actividad deportiva registrada.
- RF-ACT-6: Un usuario puede obtener puntos tras realizar una actividad deportiva.
- RF-SCL-1: Un usuario puede consultar todos los usuarios registrados en la aplicación, ordenados por coincidencia de intereses.

Gestión de la actividad deportiva

Para implementar las funcionalidades de registro, edición, borrado y consulta de actividad deportiva, se tomó cierto tiempo pero no resultó complicado.

En primer lugar, se diseñaron todas las interfaces correspondientes, de manera que fuesen sencillas de cumplimentar o de usar. Para poder listar los entrenamientos realizados, basta con seleccionar la sección "Actividad", de modo que aparece el registro de toda la actividad deportiva realizada por el usuario.

Si se desea registrar un nuevo entrenamiento, debe hacerse desde esa misma vista, pero con la particular condición de que el sistema solo permite agregar un entrenamiento realizado si detecta que el usuario ha estado más de 5 minutos en la instalación deportiva correspondiente. En otro caso, el usuario estaría realizando trampas, o bien, podría darse el caso de que hubiese entrenado con la localización desactivada, pero es un factor que reside en la humildad del usuario.

Por consiguiente, para registrar el entrenamiento, se estableció un método en *background* que comprueba si el usuario se encuentra en una instalación deportiva y cuánto tiempo lleva. Además, se implementó que una vez se registrase el entrenamiento, los puntos obtenidos se autocalculan en base a la duración de este.

Para localizar al usuario se emplean dos alternativas: por GPS y mediante *beacons*. En el caso de tratarse del interior de un centro deportivo, como por ejemplo, una sala de musculación, de ciclo, o de alguna actividad dirigida, se decidió que el *beacon* fuese el responsable de controlar la ubicación del usuario. En el caso de tratarse de una instalación deportiva externa, al aire libre, es la señal GPS del teléfono del usuario la responsable de localizar a este.

La configuración del *beacon* también se trata en esta iteración. Partiendo del manual de usuario y de la documentación oficial, y haciendo uso de la aplicación iBKS Config Tool, se asignó un identificador al *beacon* para poder localizarlo a través de Bluetooth. Acto seguido, se desarrolló el código Java correspondiente para localizar este dispositivo de manera continua, haciendo uso de la librería mencionada anteriormente, Android Beacon Library.

En cuanto a la edición del entrenamiento, se decidió permitir solamente la edición de los campos nombre, descripción e instalación deportiva, en caso de que el sistema haya detectado otro lugar de entrenamiento por error. La duración, así como la fecha y hora, y los puntos obtenidos, no recibirían ninguna modificación al no estar permitido en las reglas de la aplicación. Si el usuario decide borrar el entrenamiento, la decisión tomada establecía que los puntos obtenidos en ese entrenamiento no se borrarían de su cómputo total de puntos.

Listado de todos los usuarios registrados

Para obtener todos los usuarios de la aplicación registrados en tiempo real, fue necesario el uso de la base de datos Firebase Realtime Database. Se decidió que, cuando un usuario se registrase en la aplicación, se almacenase su documento JSON representativo en dicha base de datos, solamente los campos necesarios. Si en algún momento se producía alguna modificación, dicho documento en la nube sería actualizado.

```java
private void storeUserDataInCloud(String userId, String name, List<String> interests, String photoUrl) {
        User cloud = new User(userId, name, interests, photoUrl);
        cloud.setEmail(user.getEmail());
        cloud.setScore(0);
        cloud.setTraining(false);

        FirebaseDatabase.getInstance().getReference("User").child(userId).setValue(cloud);
    }
```

(a) Almacenamiento del usuario en la base de datos Realtime Database

```java
private void retrieveAndStoreToken() {
        FirebaseMessaging.getInstance().getToken().
                addOnCompleteListener(task -> {
                    if (task.isSuccessful()) {
                        String token = task.getResult();
                        String userId = FirebaseAuth.getInstance().getCurrentUser().getUid();

                        FirebaseDatabase.getInstance()
                                .getReference("tokens")
                                .child(userId)
                                .setValue(token);
                    }
                });
    }
```

(b) Almacenamiento del token del dispositivo en la base de datos Realtime Database

Figura 44: Métodos para almacenar usuarios y sus dispositivos en la nube

Además de almacenarse el documento del usuario, surgió también la necesidad de almacenar el *token* asociado al dispositivo en el que el usuario iniciase su sesión. Dicho *token* es un identificador único del teléfono móvil en el que el usuario está usando la aplicación. Es necesario para poder enviar notificaciones *push* a dicho dispositivo. Por consiguiente, se implementó que cuando el usuario iniciase sesión en el dispositivo, se almacenase su *token* (junto al UID del usuario proporcionado por Authentication) en la base de datos en tiempo real para así disponer de un listado de usuarios activos a los que poder enviar las notificaciones, ya fuesen

peticiones de amistad o solicitudes de actualización de datos.

Una vez se tenían todos los usuarios almacenados en esta base de datos, ya podían ser listados dentro de la aplicación. Por ello, en la sección "Usuarios" del contexto social de la aplicación, se implementó un *RecyclerView* junto a un *Adapter* como se ha comentado previamente, de modo que cada usuario era representando mediante un componente *CardView* alargado horizontalmente, mostrando los datos representativos de este: su nombre, sus intereses deportivos, el porcentaje de compatibilidad con el propio usuario en base a los intereses, su puntuación total y la posibilidad de enviarle una solicitud de amistad en el caso de que aun no fuesen amigos.

Para realizar la ordenación de este listado de usuarios, se hizo en base al porcentaje de compatibilidad entre el usuario interesado y el usuario objetivo. El cálculo de este porcentaje se implementó de una manera simple, si bien la aplicación ofrece 9 intereses deportivos, el porcentaje se pudo obtener a través del número de coincidencias de intereses entre ambos usuarios.

6.2.5. Iteración 4

En esta última iteración, se implementan todas las funcionalidades de envío y confirmación de solicitudes de amistad, gestión de los amigos de un usuario y consulta de sus perfiles, y visualización del *ranking* global. Los requisitos asociados a dichas funcionalidades son:

- RF-ACT-5: Un usuario puede consultar los registros de actividad deportiva de un amigo.
- RF-SCL-2: Un usuario puede consultar todas sus solicitudes de amistad recibidas.
- RF-SCL-3: Un usuario puede consultar todos sus amigos.
- RF-SCL-4: Un usuario puede enviar una solicitud de amistad a otro usuario.
- RF-SCL-5: Un usuario puede aceptar una solicitud de amistad recibida.
- RF-SCL-6: Un usuario puede denegar una solicitud de amistad recibida.
- RF-SCL-7: Un usuario puede consultar el perfil de un amigo.
- RF-SCL-8: Un usuario puede eliminar a un amigo de su lista de amigos.

- RF-SCL-9: Un usuario puede consultar el *ranking* de puntos global formado por todos los usuarios.

Gestión de las solicitudes de amistad

Para la implementación del envío, aceptación y denegación de las solicitudes de amistad entre usuarios, se hizo uso de las notificaciones *push*, empleando para ello los servicios de Firebase Cloud Messaging.

Un mensaje enviado con Firebase Cloud Messaging puede ser de distintos tipos: mensaje de notificación, mensaje de datos, o mensaje de notificación con carga útil de datos opcional. En este caso, se utilizó el último tipo de mensaje, la notificación con carga de datos.

La estructura de este tipo de mensajes es la siguiente:

```
{
  "message":{
    "token":"bk3RNwTe3H0:CI2k_HHwgIpoDKCIZvvDMExUdFQ3P1...",
    "notification":{
      "title":"Portugal vs. Denmark",
      "body":"great match!"
    },
    "data" : {
      "Nick" : "Mario",
      "Room" : "PortugalVSDenmark"
    }
  }
}
```

Figura 45: Estructura de un mensaje que incluye notificación y datos

Así, en un mensaje se distinguen tres elementos principales:

- El campo `token`: Es el código identificativo del dispositivo que va a recibir la notificación. En el caso de esta aplicación, ya se ha comentado que este *token* se almacena en la base de datos Realtime Database cuando un usuario inicia sesión.

- El campo `notification`: Es un diccionario correspondiente a la parte que es visible por el usuario. En él, se especifican valores como el título de la notificación, el cuerpo, el icono, etc.

- El campo `data`: Es un diccionario correspondiente a la parte de datos que se envían por detrás, no visibles aparentemente por el usuario, pero sí gestionados por la aplicación. En este campo se puede especificar cualquier cosa, pero en el caso de SportU se empleó para enviar los datos del usuario que está creando la notificación, ya sea para enviar una solicitud de amistad o una actualización de su perfil.

Habiendo explicado la estructura del mensaje, se muestra un fragmento de código a través del cual se está creando la estructura del mensaje que se envía en la aplicación:

```
JSONObject mainObject = new JSONObject();
mainObject.put("to", userFCMToken);

JSONObject notiObject = new JSONObject();
notiObject.put("title", title);
notiObject.put("body", body);
notiObject.put("icon", R.drawable.ic_sport);
mainObject.put("notification", notiObject);

JSONObject dataObject = new JSONObject();
dataObject.put("Usuario", new JSONObject(userJson));
mainObject.put("data", dataObject);
```

Figura 46: Estructura de un mensaje enviado en SportU

Para la gestión de la recepción de las notificaciones, se creó un servicio en el proyecto de Android, llamado `MyFirebaseMessagingService`, encargado de tratar la notificación recibida a través de un método principal denominado `onMessageReceived`. En dicho método, se comprueba si en el mensaje se están recibiendo datos, y en ese caso, se procesan esos datos en función de la necesidad a satisfacer.

A la hora de recibir una notificación, pueden contemplarse dos escenarios: la aplicación se encuentra en primer plano o *foreground*, o bien, la aplicación se encuentra en segundo plano o *background*.

El primer caso significa que el usuario tiene la aplicación abierta y está viendo sus contenidos, es decir, está activa en primer plano. En este caso, independientemente del tipo de mensaje que se esté recibiendo, ya sea solamente notificación, solamente datos o ambos juntos, el mensaje es gestionado por completo en el método `onMessageReceived`.

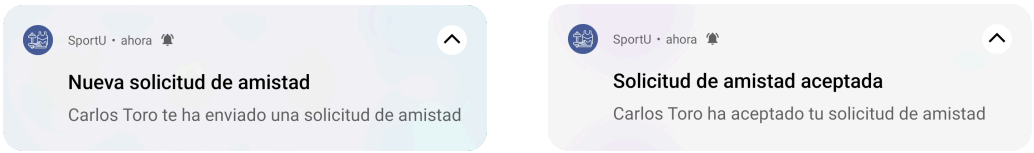
Por otro lado, si la aplicación se encuentra en segundo plano, se distinguen tres casos:

Estado de la app	Notificación	Datos	Ambos
Primer plano	`onMessageReceived`	`onMessageReceived`	`onMessageReceived`
Segundo plano	Bandeja del sistema	`onMessageReceived`	Notificación: bandeja del sistema Datos: en adicionales del intent

Figura 47: Casuística en la recepción de notificaciones

- Si solamente se recibe una notificación sin datos, se gestiona en la bandeja del sistema del dispositivo.
- Si solamente se reciben datos, se gestiona en el método `onMessageReceived`.
- Si llega un mensaje conteniendo notificación y datos, como es el caso de SportU, la notificación se gestiona en la bandeja del sistema y los datos se envían como argumentos a la actividad encargada de procesar esos datos.

Una vez entendida esta lógica, se pudieron implementar las notificaciones para el envío de solicitudes de amistad, y su correspondiente aceptación o denegación de manera exitosa. De este modo, el resultado de las notificaciones pueden observarse en la siguiente figura.



(a) Recepción (b) Confirmación

Figura 48: Notificaciones al recibir y al aceptar una solicitud de amistad

Consulta de amigos y solicitudes de amistad

El siguiente paso, fue implementar la sección de "Amigos" en la que se muestran las solicitudes de amistad pendientes por responder, así como el listado de amigos del usuario. En cierto modo, se subdividió la vista en dos partes, una para las solicitudes, y otra para los amigos.

Las solicitudes se muestran con componentes *CardView* estirados horizontalmente, con dos posibles opciones: una para aceptar la solicitud y otra para denegarla; y los amigos se muestran con componentes *CardView* pero cuadrados, en filas de dos. Para ello, una vez más

se ha hecho uso de *RecyclerView* junto a *Adapter*. Al pulsar sobre alguno de los amigos, el usuario es redirigido a su perfil pudiendo ver su información y sus entrenamientos.

De este modo, si un usuario acepta una solicitud pendiente, se elimina de ese listado, agregándose al listado de amigos y enviando una notificación al usuario solicitante para confirmarle que la amistad se ha establecido. En otro caso, si el usuario deniega la solicitud de amistad, se elimina de ese listado pero no notifica al usuario solicitante.

Al acceder al perfil de un amigo, se decidió implementar una acción que consistía en pedirle los datos de nuevo a ese usuario, es decir, recibir una actualización de su perfil. Esto es debido a que, cuando dos usuarios establecen un vínculo de amistad, ambos obtienen la información del otro hasta ese momento, almacenándose en local, sin usar datos en la nube. Por ello, si dos usuarios establecieron amistad hace dos días, a día de hoy es posible que los datos hayan cambiado y sea necesario pedirlos de nuevo al otro usuario.

Ranking

Por último, se implementó el *ranking* de la aplicación. No es más que una sección más de la parte social de SportU, que consiste en mostrar de manera ordenada en orden descendiente, la clasificación de todos los usuarios en base a los puntos, con el objetivo de promover la competitividad y alentar a la realización de actividad física con más frecuencia.

Al alcanzar un número determinado de puntos, un usuario obtiene una insignia, a modo de incentivo. Un aspecto que quizá se pueda implementar a futuro sea la mejora de estos incentivos, como recompensas físicas o diferentes premios en la aplicación.

7

Conclusiones y líneas futuras

Este último capítulo aborda las conclusiones finales obtenidas tras realizar este proyecto y se exponen distintas mejoras a considerar para el futuro.

7.1. Conclusiones

De manera general, puede destacarse que este proyecto ha cumplido con todos los objetivos establecidos en su inicio. Todos los comienzos son difíciles y, como ya se ha comentado, un proyecto de este ámbito resultaba muy aterrador al no saber por dónde empezar en cuanto a desarrollo.

Al haberse tratado aspectos como servicios de localización mediante GPS, Bluetooth y *beacons*, el envío y recepción de notificaciones *push*, el hecho de enfrentarse a desarrollar una aplicación de este tipo desde cero en Android contando solamente con experiencia en un proyecto de pequeño calibre, resultaba todo un grandioso reto.

Durante el proyecto, no solamente se ha empleado documentación de Internet así como ayuda de foros y vídeos explicativos, sino que el estudiante ha tenido que aplicar, en ciertos casos, un pensamiento lógico y creativo, y otros conocimientos avanzados para poder implementar alguna funcionalidad. Dicho conocimiento ha sido el adquirido durante los años de carrera, que han servido en su mayoría para asentar y desarrollar con éxito el proyecto, desde el planteamiento inicial de requisitos o desarrollo de maquetas hasta obtener su implementación a nivel de código.

También es digno de resaltar el gran apoyo recibido por el tutor. Ha estado siempre pendiente de la evolución del proyecto, muy implicado semanalmente, proponiendo reuniones en las que se abordaban todo tipo de cuestiones, realizando sugerencias, así como ayudando en el

desarrollo de algunas funcionalidades. En conclusión, este proyecto ha servido para crecer a nivel profesional, plantear y proponer una idea desde cero y llevarla a su completa ejecución aplicando las bases del proceso del desarrollo de software, un aspecto que es esencial para obtener el éxito buscado.

7.2. Líneas futuras

Aunque se han intentado cumplir e implementar todas las funcionalidades, existen propuestas para implementar en la aplicación de cara al futuro. Siempre puede mejorarse el resultado obtenido. Seguidamente se exponen dichas propuestas:

- **Aspectos relativos a la privacidad**: Si bien es cierto que los usuarios disponen de un total control sobre sus datos al tenerlos almacenados de manera local en su dispositivo móvil, también se están compartiendo ciertos datos en la nube y a su vez, con otros usuarios. Una mejora podría ser permitir al usuario decidir sobre qué datos se van a compartir de manera pública y cuáles quiere mantener en privado, no accesibles por nadie. Esto es un aspecto que también se trata en el modelo Digital Avatars.

- **Comunicación a través de mensajería**: Un aspecto que no se ha contemplado en esta aplicación, aunque se trata la socialización de los usuarios, es la implementación de funcionalidades referentes a mensajería, es decir, creación de chats entre usuarios y permitir el envío y recepción de mensajes en tiempo real. Podría haber sido incorporado en esta aplicación pero no es una funcionalidad que se hubiese considerado al inicio del proyecto.

- **Añadir amigos por cercanía al usuario**: Una función curiosa podría ser localizar a los usuarios que se encuentran cerca. Para ello, también debería obtenerse permisos, es decir, es un aspecto ligado a la privacidad que se comenta en el primer punto de este apartado. Un usuario podría decidir si quiere ser localizado por otro o no, es decir, si quiere aparecer en el rango de proximidad de otro usuario.

- **Nuevos incentivos y recompensas**: Actualmente, se otorgan solamente insignias al alcanzar una puntuación. Sin embargo, podría contemplarse la participación del organismo del Ayuntamiento de Málaga para conceder así mejores premios o recompensas,

como podría ser otorgar material de actividad física, pases gratuitos de entrenamiento en distintos centros deportivos, etc.

Referencias

[1] *Dynamically Programmable Virtual Profiles as a Service*. Pérez-Vereda, Alejandro, Murillo, Juan M. y Canal, Carlos, 2019, págs. 1789-1794. DOI: 10.1109/SmartWorld-UIC-ATC-SCALCOM-IOP-SCI.2019.00317.

[2] *Instalaciones deportivas - Aparatos - Conjuntos de datos - Datos abiertos Ayto. Málaga.* URL: https://datosabiertos.malaga.eu/dataset/instalaciones-deportivas-aparatos. (Accedido: 05-09-2022).

[3] *Instalaciones deportivas - Atletismo - Conjuntos de datos - Datos abiertos Ayto. Málaga.* URL: https://datosabiertos.malaga.eu/dataset/instalaciones-deportivas-atletismo. (Accedido: 05-09-2022).

[4] *Instalaciones deportivas - Centros deportivos - Conjuntos de datos - Datos abiertos Ayto. Málaga.* URL: https://datosabiertos.malaga.eu/dataset/instalaciones-deportivas-centros-deportivos. (Accedido: 05-09-2022).

[5] *Instalaciones deportivas - Senderismo - Conjuntos de datos - Datos abiertos Ayto. Málaga.* URL: https://datosabiertos.malaga.eu/dataset/instalaciones-deportivas-senderismo. (Accedido: 05-09-2022).

[6] *Instalaciones deportivas - Zonas de musculación - Conjuntos de datos - Datos abiertos Ayto. Málaga.* URL: https://datosabiertos.malaga.eu/dataset/instalaciones-deportivas-zonas-de-musculacion. (Accedido: 05-09-2022).

[7] *¿Qué es Java? Guía para principiantes de Java | Microsoft Azure.* URL: https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-java-programming-language/. (Accedido: 07-09-2022).

[8] *What is Python? Executive Summary | Python.org.* URL: https://www.python.org/doc/essays/blurb/. (Accedido: 07-09-2022).

[9] *Embedded NoSQL Database for Mobile, Desktop More | Lite.* URL: https://www.couchbase.com/products/lite. (Accedido: 07-09-2022).

[10] *¿Qué Es MongoDB? | MongoDB.* URL: https://www.mongodb.com/es/what-is-mongodb. (Accedido: 07-09-2022).

[11] *Firebase Realtime Database | Firebase.* URL: https://firebase.google.com/docs/database. (Accedido: 07-09-2022).

[12] *Introducción a Android Studio | Desarrolladores de Android | Android Developers.* URL: https://developer.android.com/studio/intro?hl=es. (Accedido: 07-09-2022).

[13] *Visual Studio Code - Code Editing.* URL: https://code.visualstudio.com/. (Accedido: 07-09-2022).

[14] *Firebase Cloud Messaging | Firebase.* URL: https://firebase.google.com/docs/cloud-messaging. (Accedido: 07-09-2022).

[15] *Firebase Authentication | Firebase.* URL: https://firebase.google.com/docs/auth. (Accedido: 07-09-2022).

[16] *Moon Modeler | Data Modeling Tool for Databases.* URL: https://www.datensen.com/. (Accedido: 07-09-2022).

[17] *MongoDB Compass | MongoDB.* URL: https://www.mongodb.com/products/compass. (Accedido: 07-09-2022).

[18] *Git.* URL: https://git-scm.com/. (Accedido: 07-09-2022).

[19] *Features | GitHub.* URL: https://github.com/features. (Accedido: 07-09-2022).

[20] *Plataforma de almacenamiento personal en la nube y uso compartido de archivos - Google.* URL: https://www.google.com/intl/es_es/drive/. (Accedido: 07-09-2022).

[21] *Google Meet: llamadas y videollamadas online (con la tecnología de Duo).* URL: https://apps.google.com/intl/es/intl/es_ALL/meet/. (Accedido: 07-09-2022).

[22] *Image and Video Upload, Storage, Optimization and CDN.* URL: https://cloudinary.com/. (Accedido: 07-09-2022).

[23] *Gestiona los proyectos de tu equipo desde cualquier lugar | Trello.* URL: https://trello.com/es. (Accedido: 07-09-2022).

[24] *Documentos de Google: editor de documentos online | Google Workspace.* URL: https://www.google.es/intl/es/docs/about/. (Accedido: 07-09-2022).

[25] *Overleaf, Editor de LaTeX online.* URL: https://es.overleaf.com/. (Accedido: 07-09-2022).

[26] *Marvel - The design platform for digital products. Get started for free.* URL: https://marvelapp.com/. (Accedido: 07-09-2022).

[27] *Adobe Photoshop Oficial | Software de fotografía y diseño.* URL: https://www.adobe.com/es/products/photoshop.html. (Accedido: 07-09-2022).

[28] *iBKS 105 · Accent Systems.* URL: https://accent-systems.com/es/producto/ibks-105/. (Accedido: 07-09-2022).

[29] *google/eddystone: Specification for Eddystone, an open beacon format from Google.* URL: https://github.com/google/eddystone. (Accedido: 07-09-2022).

[30] *iBKS Config Tool User Manual - Android · Accent Systems.* URL: https://accent-systems.com/support/knowledge/ibks-config-tool-user-manual/. (Accedido: 07-09-2022).

[31] *Scrum: qué es, cómo funciona y por qué es excelente.* URL: https://www.atlassian.com/es/agile/scrum. (Accedido: 12-09-2022).

[32] *Bienvenida - Datos abiertos Ayto. Málaga.* URL: https://datosabiertos.malaga.eu/. (Accedido: 14-09-2022).

[33] *Vector Icons and Stickers - PNG, SVG, EPS, PSD and CSS.* URL: https://www.flaticon.com/. (Accedido: 19-09-2022).

[34] *iBKS Config Tool User Manual – Android.* URL: https://accent-systems.com/support/knowledge/ibks-config-tool-user-manual/. (Accedido: 22-09-2022).

Apéndice A
Manual de usuario

A.1. Introducción

Este manual sirve para guiar al usuario en el uso de la aplicación SportU. Para ello se van a describir las acciones que pueden llevarse a cabo junto a fotografías que representan el estado de la aplicación.

A.2. Guía de uso de la aplicación

En primer lugar, el usuario verá la pantalla de bienvenida a través de la cual puede iniciar sesión o registrarse con su cuenta de Google, seleccionando “Acceder con Google”, y eligiendo alguna de las direcciones de correo registradas en su teléfono.

Figura 49: Pantallas de acceso a la aplicación

A continuación, pueden ocurrir dos escenarios. El primero consiste en que el usuario no disponga de una cuenta en el sistema, por lo cual es redirigido a las pantallas de registro, debiendo cumplimentar los campos con su nombre, fecha de nacimiento, género, descripción e intereses en cuanto a actividades deportivas.

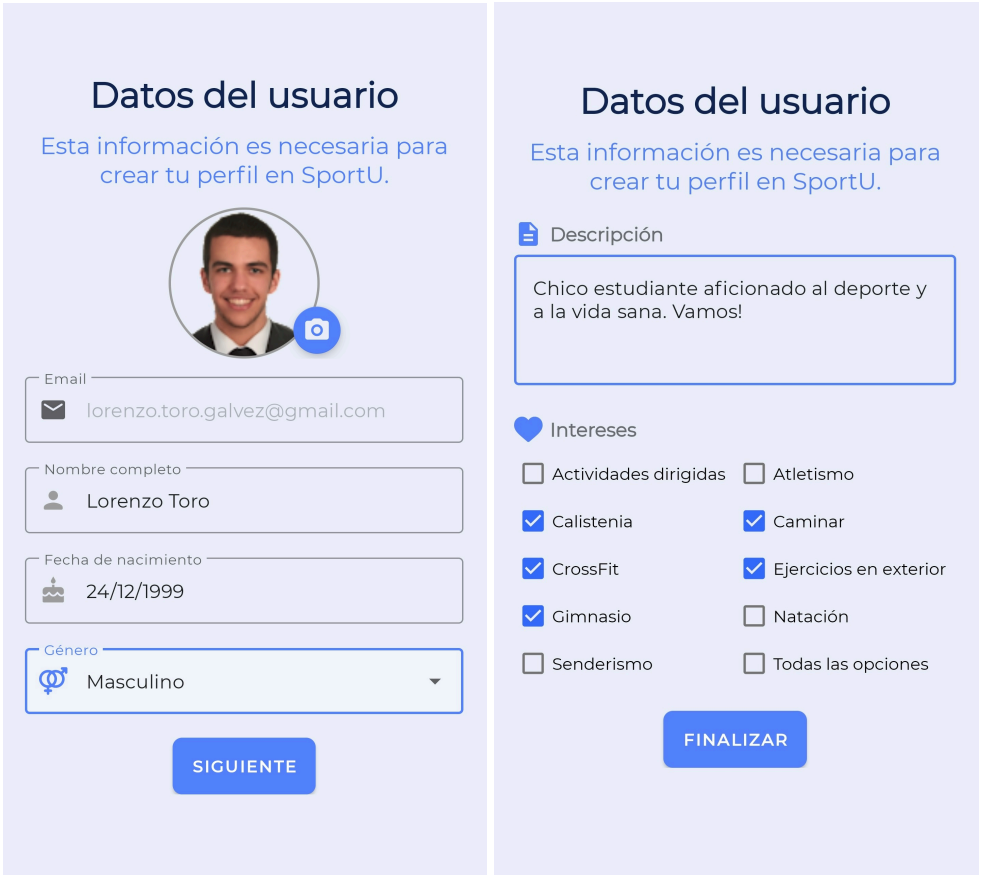


Figura 50: Pantallas de registro en la aplicación

El segundo escenario que puede darse en este caso es que el usuario ya disponga de una cuenta en el sistema, por tanto se inicia la sesión y es redirigido a la pantalla de su perfil, mostrando así todos sus datos personales, su puntuación y sus insignias, así como las opciones de editar sus datos o cerrar la sesión.

Además, se muestra un menú que sirve para desplazarse entre las distintas secciones, esto es, un menú de navegación inferior formado por cuatro opciones permite al usuario moverse entre las distintas pantallas de la aplicación: “Perfil”, “Actividad”, “Instalaciones” y “Social”.

Figura 51: Pantalla del perfil del usuario

Cuando el usuario selecciona la opción “Actividad” del menú de navegación, se muestra una lista con sus entrenamientos registrados.

Cada entrenamiento contiene información como el nombre, la instalación deportiva en la que ha sido realizado, un breve texto describiendo el entrenamiento o las sensaciones del usuario, la duración, los puntos obtenidos, y la fecha y hora de realización.

Además, permite al usuario editar alguno de los entrenamientos registrados así como introducir uno nuevo (véase Figura 52).

Al seleccionar la opción “Instalaciones” del menú inferior de navegación, el usuario es redirigido a una pantalla que dispone de un menú superior de navegación con dos opciones disponibles: “Listado” y “Mapa”.

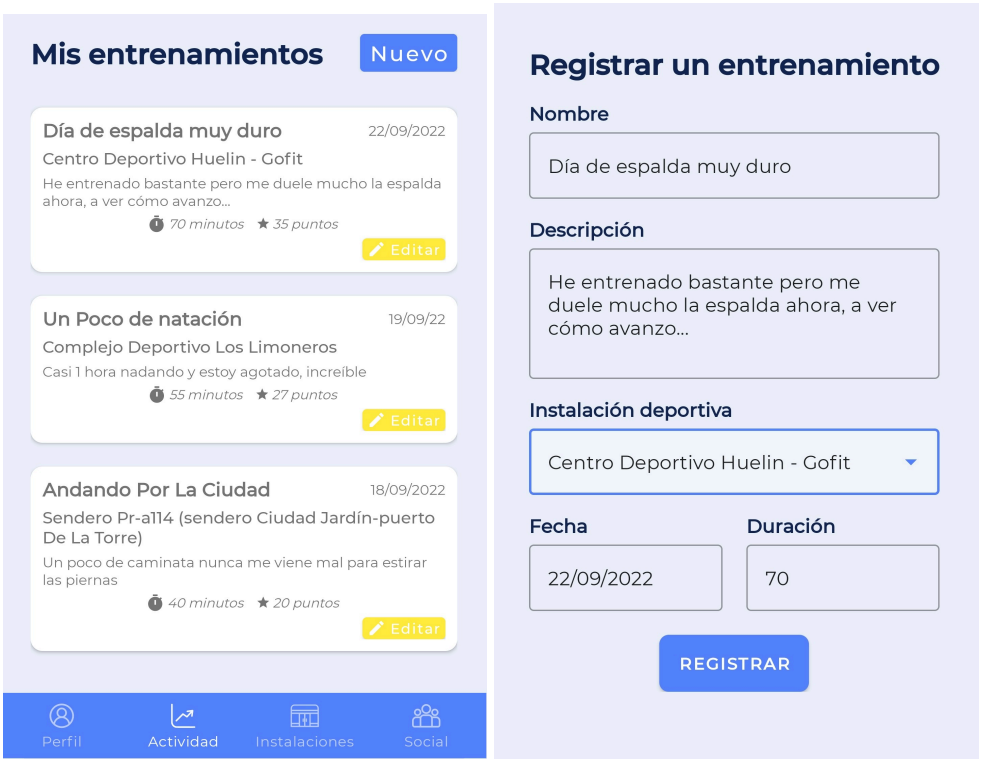


Figura 52: Pantallas de entrenamientos y su registro

Por defecto, la primera pantalla mostrada será la correspondiente al listado de las instalaciones deportivas. Dichas instalaciones se mostrarán con su nombre, dirección y distancia al usuario, ordenadas en función de este último atributo, de menor a mayor.

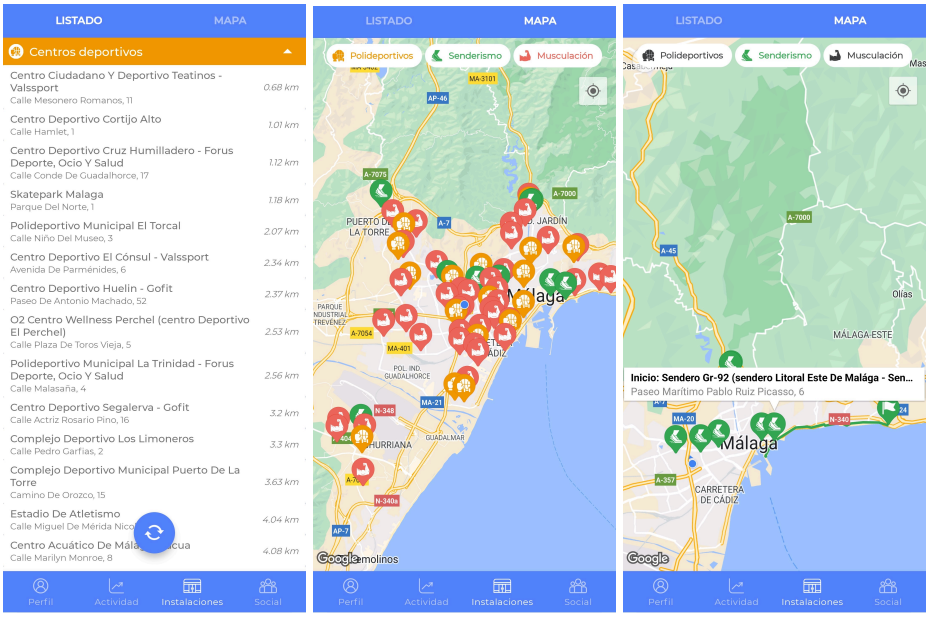


Figura 53: Pantallas de instalaciones deportivas

Por otro lado, al seleccionar la opción “Mapa”, el usuario verá un mapa centrado en la ciudad de Málaga con multitud de marcadores indicando los centros deportivos, las rutas de senderismo y las zonas de musculación, ofreciendo también unos filtros para ocultar o mostrar dichas instalaciones. Pueden observarse estas pantallas en la Figura 53.

Si el usuario selecciona la opción “Social” del menú de navegación inferior, observará una pantalla que dispone de un menú superior con tres opciones: “Usuarios”, “Amigos” y “Ranking”, estando seleccionada por defecto la primera opción.

Figura 54: Pantallas de los usuarios, solicitudes y amigos

Así, la primera opción de “Usuarios”, observada en la imagen (a) de la Figura 56 muestra un listado con todos los usuarios registrados en la aplicación, con nombre, intereses, porcentaje de compatibilidad y puntuación, existiendo la posibilidad de enviar una solicitud de amistad a cualquiera de los usuarios. Dicho listado será ordenado en base al porcentaje de compatibilidad, que se calcula comparando los intereses del usuario con los intereses de otro usuario.

Por otra parte, al seleccionar la opción “Amigos”, se mostrarán todos los amigos del usuario así como las solicitudes de amistad recibidas pendientes, permitiendo aceptar o denegar las solicitudes recibidas así como poder acceder al perfil de los amigos para consultar su información y sus entrenamientos.

Por último, se muestra el *ranking* con los usuarios ordenados en base a su puntuación global en la aplicación.

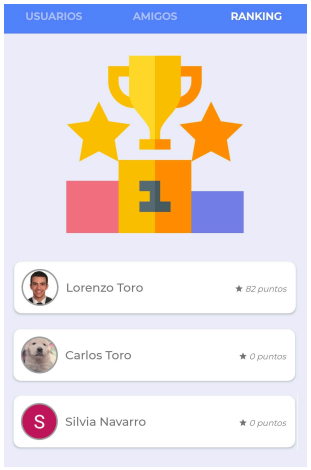
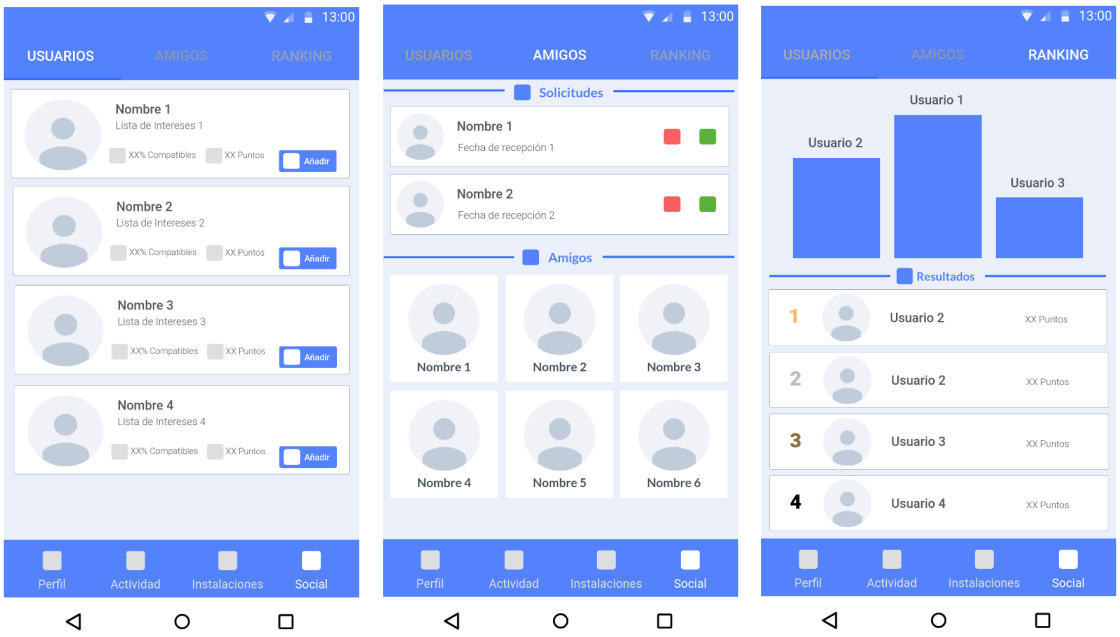


Figura 55: Pantallas del ranking



(a) Todos los usuarios (b) Amigos del usuario (c) Ranking

Figura 56: Pantallas correspondientes a la parte social de la aplicación

Apéndice B
Manual de instalación y despliegue

B.1. Aplicación

Para poder instalar la aplicación en el teléfono del usuario, basta con importar el proyecto y ejecutarla teniendo el teléfono conectado.

B.2. Beacon

Para poder configurar el beacon correctamente, se hace referencia al manual su web oficial [34].

UNIVERSIDAD DE MÁLAGA | uma.es

E.T.S de Ingeniería Informática
Bulevar Louis Pasteur, 35
Campus de Teatinos
29071 Málaga

E.T.S. DE INGENIERÍA INFORMÁTICA