Universidad de Valladolid

Escuela de Ingeniería Informática

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

(Mención en Ingeniería del Software)

OrientaTree

Aplicación móvil para la realización de actividades educativas geolocalizadas en el medio natural

Autor:
Gabriel Rodríguez González

Tutores:
Alejandra Martínez Monés
Juan Alberto Muñoz Cristóbal

Agradecimientos

Muchas gracias a Alejandra y Juan por su ayuda y su dedicación a lo largo de todo este proyecto, a Quico, Nesi y Felipe por esta colaboración que ha resultado tan interesante, y a mi familia, amigos y a Raquel por su apoyo.

Resumen

Las carreras de orientación son un tipo de actividad formativa en el medio natural que pueden beneficiarse del uso de aplicaciones móviles. En este documento se describe el proceso por el que se ha diseñado y desarrollado una aplicación móvil que permite programar actividades de orientación educativa y participar en ellas. El medio natural en el que tienen lugar estas actividades es el Campo Grande de Valladolid.

Las características del proyecto, que requería realizar un diseño de interacción con objetivos muy abiertos, nos llevaron a optar por una metodología de diseño centrado en el usuario, en la que pudimos contar con la participación de miembros de la Facultad de Educación y Trabajo Social. A lo largo del documento se describen las técnicas propias de dicha metodología que han sido empleadas y cómo nos han ayudado a obtener un producto de mayor valor para los usuarios.

Abstract

Orienteering is a type of activity that takes place on natural enviroments and might be complemented with mobile aplications to extend its possibilities. On this document we describe the process of designing and implementing a mobile application that allows users to schedule educational orienteering activities as well as to take part on them. The natural enviroment on which these activities take place is Campo Grande (Valladolid).

The initial requirements of the interaction for this project were very open. That made us choose a user-centered design methodology that enabled us to count with the colaboration of some members from the Education and Social Work Faculty. Throughout this document we will describe the different techniques from that methodology that have been used, and how they have assisted us on getting a much more valuable product for users.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Contexto

La práctica de Actividades Físicas en el Medio Natural (AFMN) está en auge en la actualidad. Dentro del ámbito educativo, este tipo de actividades juega cada vez un papel más destacado, y es más frecuente encontrarlas incluidas en la programación de las materias escolares [9]. En el contexto de las AFMN, las actividades de orientación poseen una posición privilegiada debido su accesibilidad, la facilidad para implementarlas, y a que no hace falta una formación especial para el profesorado [9]. Por otra parte, las actividades de orientación pueden verse apoyadas y enriquecidas en distintos aspectos mediante su integración con las TIC. Este proyecto, que se realiza bajo el contexto de trabajo de fin de grado en Ingeniería Informática, y que está ligado al grupo de investigación GSIC/EMIC (Grupo de Sistemas Inteligentes y Cooperativos / Educación, Medios, Informática y Cultura)[1], ha consistido en el desarrollo de una aplicación para la organización y participación en actividades de orientación educativa. A lo largo de esta introducción se explicarán más detalladamente todos estos conceptos. A continuación se describe brevemente qué es el grupo GSIC/EMIC, en qué áreas desarrolla su actividad, su relación con este proyecto, y cómo colabora con la Facultad de Educación y Trabajo Social.

1.1.1. Colaboración del GSIC/EMIC y la Facultad de Educación y Trabajo Social

El grupo GSIC/EMIC es un grupo transdisciplinar que está compuesto por profesores, investigadores y personal laboral de la Escuela Técnica Superior de Ingenieros de Telecomunicación, la Escuela de Ingeniería Informática y la Facultad de Educación y Trabajo Social de la Universidad de Valladolid. La principal línea de investigación del grupo GSIC/EMIC es la del aprendizaje apoyado por la tecnología. En este ámbito lidera y colabora en distintos proyectos de investigación, donde se involucran también profesionales de otras áreas. Por ejemplo, en lo relativo a este TFG, el grupo lleva colaborando varios años con el departamento de Didáctica de la Expresión Musical, Plástica y Corporal en el apoyo mediante TIC al diseño y puesta en marcha de actividades educativas relacionadas con la orientación. Este proyecto se realiza dentro del marco de esa colaboración. A continuación, se explica brevemente en qué consisten las actividades de orientación y cómo las TIC pueden asistir y enriquecer la realización de estas actividades.

1.1.2. AFMN y orientación

Como se ha mencionado anteriormente, las Actividades Físicas en el Medio Natural (AFMN) están cada vez más presentes en la programación de materias escolares como la Educación Física. En este proyecto nos hemos centrado en las actividades de orientación.

La orientación es un deporte en el que cada participante realiza un recorrido cronometrado, con ayuda de un mapa y una brújula [7]. Los recorridos constan de una salida, una llegada (o meta) y los controles (o balizas). Los controles son los lugares por los que los participantes deben pasar, y están representados en el mapa de cada participante. El mapa que se usa en orientación deportiva es específico de este deporte, e incluye información topográfica, curvas de nivel, vegetación y otros elementos relevantes, con una codificación bien conocida (Figura 1.1). Al comienzo de la carrera, la ubicación de las balizas es desconocida para los participantes, por lo que estos

[1] Sitio web del grupo GSIC/EMIC: http://gsic.uva.es

deben ser capaces de encontrarlas empleando el mapa y la brújula. La modalidad más conocida y practicada de este deporte son las carreras a pie, aunque existen otras modalidades de orientación oficiales como orientación en bicicleta de montaña, orientación de precisión y esquí orientación [7].

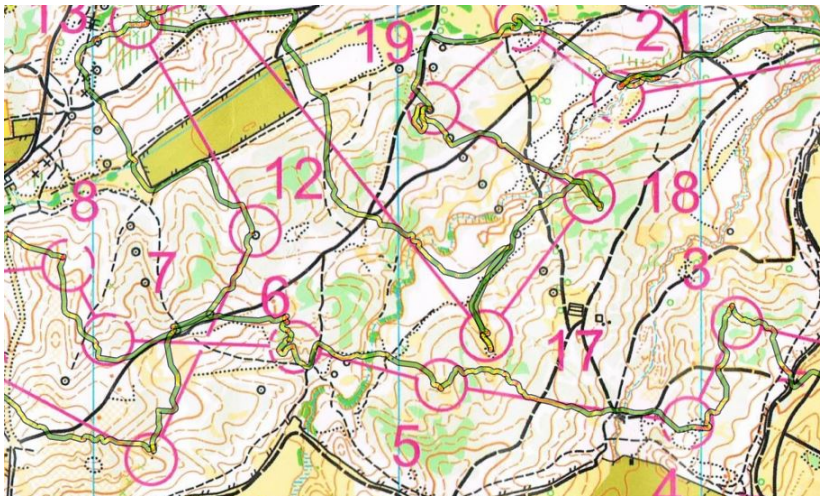

Figura 1.1: Mapa de orientación.

Tradicionalmente, la forma en la que los participantes demostraban que habían pasado por las balizas era utilizando "pinzas perforadoras". Cada participante llevaba consigo una tarjeta de papel en la que iba haciendo marcas con las pinzas según alcanzaba las balizas. Las pinzas de cada baliza marcaban la tarjeta con un patrón único, por lo que los organizadores podían comprobar si el participante había pasado por los distintos controles. Actualmente es más habitual emplear un sistema de control electrónico en el que cada participante lleva consigo un chip que registra su paso por las balizas (en las que también hay un chip).

1.1.2.1. Orientación con fines educativos

En el ámbito educativo, en ocasiones se busca trascender la faceta puramente deportiva de la orientación tradicional, enriqueciendo la actividad con un componente de aprendizaje. Esto da lugar a lo que se conoce como **orientación con fines educativos** [9]. En las actividades de orientación educativa los participantes no se limitan a registrar su paso por las balizas una vez las han encontrado, sino que al llegar a éstas se les plantea una actividad de carácter educativo que pueda enriquecer su experiencia de aprendizaje. Estas actividades pueden ser muy variadas, por ejemplo, responder alguna pregunta relacionada con el entorno en el que se encuentra la baliza, completar un reto o leer cierta información. Para gestionar todo ese contenido educativo, las TIC podrían ser de gran utilidad, dando soporte y permitiendo enriquecer las actividades con ese componente educativo. De hecho, como se verá a continuación, ya existe una gran variedad de aplicaciones móviles que se utilizan para la realización de actividades de orientación.

1.1.2.2. Las aplicaciones móviles en el contexto de la orientación y la orientación con fines educativos

Como se ha mencionado, actualmente existe una gran variedad de aplicaciones móviles que se emplean para la realización de actividades de orientación. Por una parte tenemos las aplicaciones de orientación deportiva. Dentro de este grupo, podemos clasificar a su vez las aplicaciones de la siguiente manera:

- **Realización de carreras/recorridos**. Mediante estas aplicaciones los participantes de una actividad pueden registrar su paso por las distintas balizas. Además, en ellas también se registran los tiempos obtenidos por participantes, con los que se calcula una clasificación en la carrera. Algunos ejemplos de este tipo de aplicaciones son IOrientering[2] y MOBO[3].

[2]IOrientering: http://www.iorienteering.com/
[3]MOBO: http://mobo.osport.ee/

- **Edición de mapas**. Estas aplicaciones ayudan a diseñar mapas de orientación. Algunas de ellas también permiten diseñar y configurar los recorridos. Dentro de este grupo encontramos aplicaciones como Open Orienteering Mapper[4].
- **Juegos de simulación**. Estas aplicaciones se utilizan para practicar y entrenar. Simulan carreras de orientación, planteando a los usuarios diferentes retos gracias a los cuales, éstos pueden mejorar su técnica de interpretación de mapas. La aplicación de escritorio Virtual-O[5] es un ejemplo de este grupo.
- **Análisis de recorrido**. Mediante estas aplicaciones los participantes pueden analizar su recorrido realizado durante una carrera. Entre ellas está la aplicación O'Route Orienteering.
- **Aprendizaje teórico**. Estas aplicaciones ayudan a los usuarios a aprender el contenido teórico de las carreras de orientación. Es decir, con ellas se aprende a interpretar los símbolos propios de la orientación, los mapas y la terminología específica. Algunos ejemplos serían O-Sudoku y O-Symbol.

Además del grupo de las aplicaciones destinadas a la orientación deportiva, están las aplicaciones "generalistas", que sin haber sido concebidas específicamente para ser empleadas en actividades de orientación, se han utilizado para enriquecerlas. En este grupo se encuentran aplicaciones comunes de realidad aumentada o lectores de códigos QR [12]. Por último, están las aplicaciones de "búsqueda del tesoro", en las que los participantes tienen que ser capaces de alcanzar objetos ocultos al llegar a unas coordenadas. Ejemplos de este tipo de aplicaciones serían Geocaching[6] y Munzee[7].

Hasta donde sabemos, no hay aplicaciones móviles destinadas específicamente a la realización de actividades orientación deportiva educativa que combinen el uso de mapas de orientación con contenido educativo. Como veremos en la siguiente sección, esta es la principal motivación para realizar el proyecto.

1.2. Motivación

Este proyecto surgió a partir de una vieja idea en la que habían estado trabajando dos profesores colaboradores del grupo GSIC/EMIC hace veinte años. Aquella idea consistía en crear en el **Campo Grande de Valladolid** un entorno en el que poder realizar actividades de orientación tradicionales, con la peculiaridad de que se usarían como balizas eran elementos del propio entorno. Concretamente, se había realizado la labor de posicionar en un mapa de orientación una serie de árboles del Campo Grande, junto a los cuales el Ayuntamiento había colocado carteles que mostraban cierta información sobre los mismos. Con el tiempo esos carteles se fueron deteriorando, y muchos de ellos desaparecieron. Sin embargo, los árboles siguen allí.

Veinte años después, la intención consistía en retomar el proyecto, pero esta vez, aprovechando las posibilidades que ofrece la tecnología móvil actual para desarrollar una aplicación. En principio, la aplicación estaría ligada al contexto escolar y al aprendizaje formal, especialmente al ámbito de la Educación Física en Primaria. Sin embargo, no se descartaba la idea de hacer que la aplicación fuera extensible a otros contextos (aprendizaje informal, uso turístico, recreativo, etc) y a otros escenarios (no solo el Campo Grande de Valladolid).

En definitiva, la idea era retomar aquel proyecto incorporando las TIC para enriquecer las actividades de orientación con un mayor componente educativo, sin causar un impacto en el medio al no tener que dejar objetos físicos en el entorno, y ofreciendo mayor variedad y versatilidad a la hora de organizar actividades.

1.3. Objetivos

El principal objetivo de este proyecto es:

> *Diseñar y desarrollar una aplicación móvil mediante la que los docentes puedan programar y supervisar actividades de orientación educativa en un espacio físico concreto, y los estudiantes participar en ellas.*

Por otra parte, cabe destacar que el proyecto planteaba retos importantes desde el punto de vista de la usabilidad de las herramientas, debido a que involucra diferentes tipos de usuario, retos tecnológicos complejos, y escenarios de uso novedosos, a lo que se añade el reto de la poca precisión en la geolocalización que ofrecen los teléfonos móviles. Esto dio lugar a dos objetivos secundarios más:

[4]Open Orienteering Mapper: http://www.openorienteering.org/
[5]Virtual-O: https://virtualo.org/
[6]Geocaching: https://www.geocaching.com/play
[7]Munzee: https://www.playmunzee.com/

- *Identificar y aplicar técnicas propias del diseño centrado en el usuario, con el fin de mejorar el diseño de la aplicación.*
- *Estudiar posibles alternativas para hacer frente a los retos derivados de la falta de precisión de los dispositivos actuales de geolocalización.*

En este trabajo de fin de grado, se desarrollará el *front-end* y el *backend* de la aplicación. Como se verá más adelante, la aplicación se comunica con el sistema de *back-end* para obtener los datos de las actividades y el contenido educativo de las balizas. En este proyecto, todos esos datos serán ejemplos representativos de actividades, creados junto con los colaboradores de Educación del proyecto. Es decir, que para configurar los datos existentes o añadir contenido nuevo, habrá que interactuar directamente con el sistema de *back-end*, pues no es objetivo de este proyecto desarrollar una interfaz que permita a un usuario no avanzado configurar nuevos tipos de actividades (aunque sería muy interesante implementarlo en el futuro). Esto es así porque de acuerdo a lo tratado durante las reuniones con los usuarios, se prevé que el rango de variantes y diferentes posibilidades que sería de utilidad incluir resultaría muy amplio y complejo, por lo que requeriría un estudio en profundidad cuya realización no sería asumible tratar de completar en este proyecto.

1.4. Organización del documento

Tras estas secciones de introducción, el resto del documento se estructura de la siguiente manera:

- **Capítulo 2**. En este capítulo tratamos sobre la metodología seguida a lo largo del proyecto. También se habla sobre la planificación, incluyendo un análisis de riesgos. Además, se documenta el seguimiento realizado durante el desarrollo del proyecto, y se hace una estimación del presupuesto a partir de los datos anteriores.
- **Capítulo 3**. En el capítulo 3 se habla sobre los métodos de geoposicionamiento que se suelen utilizar en las aplicaciones móviles. Se discuten sus ventajas e inconvenientes, y se justifica la elección del método escogido para la implementación de esta aplicación.
- **Capítulo 4**. En este capítulo se trata acerca la investigación sobre usuarios realizada al inicio del proyecto. Además, también se incluyen las secciones habituales de la fase de análisis de un proyecto software (análisis de requisitos, dominio y casos de uso).
- **Capítulo 5**. El capítulo 5 trata sobre las decisiones de diseño tomadas para el desarrollo del *front-end* del sistema. Veremos, que gran parte de esas decisiones se fundamentan en las pruebas de usabilidad realizadas con usuarios reales a lo largo del proyecto. También nos detendremos en el diseño del *back-end* de la aplicación, centrándonos especialmente en la estructura de la base de datos NoSQL escogida.
- **Capítulo 6**. En este capítulo se detallan los aspectos más importantes de la implementación de la aplicación, y se explican algunas de las técnicas más interesantes que han sido utilizadas (especialmente el mapa de las actividades).
- **Capítulo 7**. En este capítulo veremos las pruebas realizadas para comprobar el funcionamiento de la aplicación.
- **Capítulo 8**. Por último, en el capítulo 8 se verán las conclusiones obtenidas al final del proyecto, así como las posibles líneas de trabajo futuro.

Capítulo 2

Plan del proyecto

Este capítulo comienza con la descripción de la metodología adoptada para la realización de este trabajo de fin de grado, así como las razones que nos han llevado a elegir dicha metodología. Posteriormente, en la segunda sección del capítulo se describe la planificación establecida al inicio del proyecto. Finalmente, en la última sección del capítulo veremos cómo se han realizado el seguimiento y la monitorización del proyecto.

2.1. Metodología

Como se ha comentado en el capítulo anterior, el objetivo de este proyecto es desarrollar una aplicación móvil mediante la que organizar y participar en actividades de orientación educativa. Se trata de una aplicación novedosa y destinada a usuarios poco expertos. Por lo tanto, es especialmente importante cuidar mucho el diseño de la interacción. Por otra parte, los usuarios de esta aplicación serán docentes del ámbito de la Educación Física y estudiantes de primaria. Como también se ha comentado anteriormente, el entorno colaborador del grupo GSIC/EMIC cuenta con algunos representantes del primer tipo de usuarios de la aplicación, lo cual nos da la oportunidad de involucrar a dichos usuarios en el proceso de diseño. Por otra parte, el desarrollo de la aplicación se ajusta bien a una metodología de diseño incremental. Por todas estas razones, la metodología elegida para la realización de este proyecto ha sido una combinación de diseño centrado en el usuario con desarrollo incremental, tal y como se explica en el resto de esta sección.

2.1.1. Diseño Centrado en el Usuario

La metodología de diseño centrado en el usuario (DCU) consiste en una colección de procesos que tienen como objetivo situar a los usuarios de la aplicación en el centro del diseño y del desarrollo de ésta [22]. Es decir, que el desarrollo del producto software se realiza poniendo especial atención en los **requisitos de los usuarios**, sus **objetivos** y su ***feedback***. Además de esta prioridad por centrar el diseño en las necesidades del usuario, la metodología DCU también persigue **involucrarlos** en el proceso de diseño a través de una serie de técnicas, con el objetivo de obtener productos altamente **usables** para ellos [17]. En resumen, la metodología DCU consta de estos cuatro principios básicos [28]:

- Foco inicial en los usuarios y sus tareas.
- Diseño iterativo.
- Medidas empíricas del comportamiento de los usuarios.

Además, ISO 13407 [18] añade la participación activa de los usuarios y el establecimiento de grupos de trabajo interdisciplinar.

En cada fase de un proyecto que sigue la metodología DCU, es importante ser capaces de entender el contexto en el que se utilizará la aplicación, especificar correctamente los requisitos de usuario, diseñar una solución y evaluarla con respecto a dichos requisitos. Para entender mejor el contexto y los requisitos de usuario, se usan técnicas de **investigación sobre usuarios** (capítulo 4). Por su parte, la elaboración de prototipos de bajo coste puede ser de gran utilidad para diseñar soluciones y presentárselas a los usuarios, con el objetivo de obtener *feedback* desde fases tempranas. Por último, los test de usabilidad pueden servir para evaluar las soluciones que hemos diseñado. Como se verá a lo largo del documento, para la realización de este proyecto se han utilizado todas esas técnicas propias de la metodología DCU.

2.1.2. Desarrollo Incremental

Aunque el DCU se puede integrar con diferentes metodologías de desarrollo, se complementa especialmente bien con las metodologías de **desarrollo ágil** [22] [8]. Esto es debido a la naturaleza iterativa de DCU. Además, al utilizar ambas conjuntamente, se contribuye a reducir algunos de los riesgos del proyecto (sección 2.2.3), así como a maximizar la satisfacción del cliente con el producto final. Por estas razones, para la implementación del software de este proyecto se ha elegido la metodología de **desarrollo incremental** basado en iteraciones. Se trata de una metodología ágil en la que, a diferencia del modelo en cascada, donde no hay iteraciones y todo el sistema se desarrolla de una sola "pasada", el proyecto se divide en incrementos. En cada uno de esos incrementos o iteraciones, se realiza un ciclo completo como el que se ve en la Figura 2.1. De este modo, se va obteniendo cada vez una nueva versión de la aplicación más avanzada, que añade funcionalidad sobre la versión anterior (Figura 2.2). Esa funcionalidad debe aportar algún beneficio concreto a los usuarios, quienes gracias a esta metodología, pueden

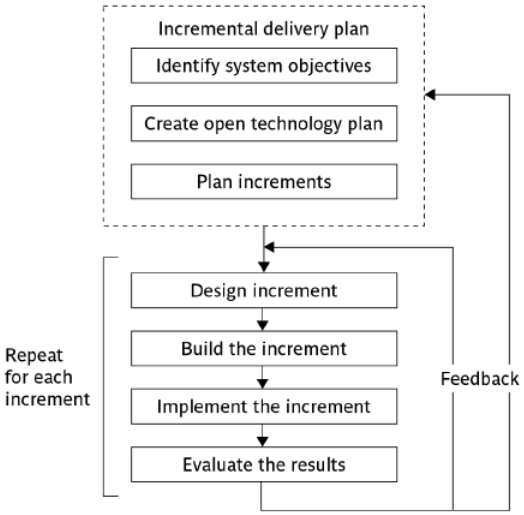

Figura 2.1: Fases del desarrollo incremental.
Imagen obtenida de [15] (cap. 4).

proporcionar al desarrollador *feedback* desde las primeras etapas de desarrollo, a medida que se les vaya entregando nuevas versiones. Finalmente, cuando se hayan completado todas las iteraciones de este proceso, el proyecto estará terminado [15] (cap. 4).

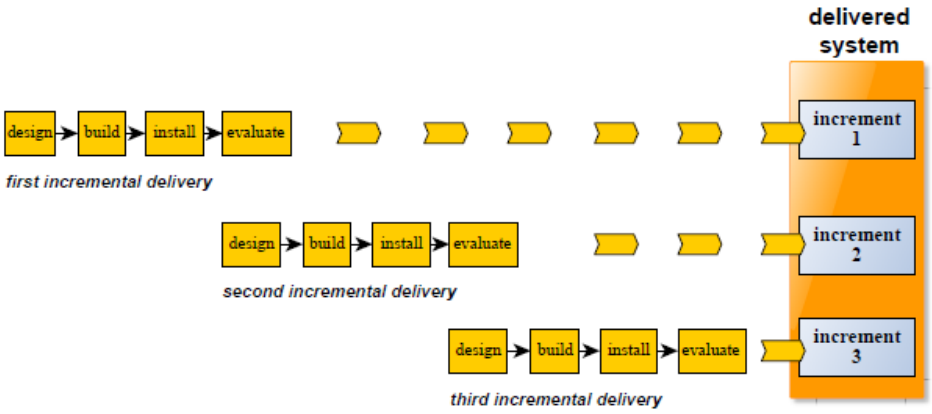

Figura 2.2: Construcción de un sistema a partir de incrementos.
Imagen obtenida de [15] (cap. 4).

Como se ha mencionado anteriormente, esta metodología de desarrollo es especialmente útil en el caso de proyectos como éste, donde se complementa con la metodología DCU. Gracias a la utilización de ambas se hace posible comprender muy bien las necesidades de los usuarios, así como ir comprobando si los productos que desarrollamos las satisfacen adecuadamente. Además, en este proyecto se va a desarrollar una parte de un sistema, por lo que esta aproximación puede ser muy útil para comprender qué funcionalidades conviene priorizar en cuanto a su implementación. Del mismo modo, también puede ayudarnos a reducir el riesgo de implementar funcionalidades que quizá no sean tan importantes para el usuario (*gold-plating*) [15] (cap. 4), como veremos en la subsección dedicada a los riesgos (2.2.3).

2.2. Planificación

Este proyecto consta de algunas particularidades que merece la pena destacar en cuanto a su planificación. Por una parte, siguiendo la metodología DCU, se han reservado las primeras semanas para la realización de una fase inicial en la que llevar a cabo la investigación sobre usuarios.

La otra particularidad es relativa a la forma de adoptar la metodología de desarrollo incremental para la implementación del software. Generalmente, para los proyectos que siguen dicha metodología se recomienda que cada incremento suponga entre un 1 % y un 5 % del total de la funcionalidad del proyecto, y que, preferiblemente, dure menos de un mes [15] (cap. 4). Para este proyecto se establecieron iteraciones de dos semanas. Sin embargo, hubo que planear cada incremento de tal forma que cubriese mayor porcentaje de funcionalidad de lo que habitualmente se recomienda. Esto es así porque si nos hubiéramos ajustado al máximo recomendado (5 % de la funcionalidad), harían falta diez meses para terminar el trabajo. Por esta razón, el proyecto está organizado en cuatro iteraciones, cada una de las cuales debería incrementar en un 25 % la funcionalidad de la aplicación desarrollada. Se deja margen para algunas iteraciones más al final, de una o dos semanas, de tal forma que haya cierta holgura por si alguna tarea se extiende en duración más de lo previsto. En la siguiente subsección, se describe más detalladamente cada una de las iteraciones previstas. Finalmente, algunas de las aproximaciones al método de desarrollo incremental sugieren que lo ideal es que tras la primera iteración tengamos un sistema completo con toda la funcionalidad básica cubierta. Tras las iteraciones posteriores se obtiene como resultado un incremento con una interfaz de usuario cada vez más refinada y detallada [24]. Para este proyecto, no se ha elegido esta aproximación, ya que además de la incertidumbre sobre la interfaz de usuario, también había cierta incertidumbre inicial sobre la funcionalidad y el alcance de la aplicación. Por lo tanto, hubiera sido arriesgado tratar de cubrir toda la funcionalidad básica en una primera versión. Por último, la metodología de diseño centrado en el usuario también desaconseja esta práctica, ya que probablemente resultaría en usuarios descontentos y cambios drásticos que habría que realizar posteriormente.

2.2.1. Fase inicial

Según lo mencionado al inicio de la sección, y siguiendo la metodología DCU, este proyecto contará con una fase inicial en la que se realizará la investigación de usuarios. Durante ese tiempo, se mantendrán reuniones con los usuarios, se elicitarán requisitos, se clarificará el contexto en el que la aplicación será utilizada y se estudiarán los distintos escenarios. Los resultados obtenidos tras esta primera fase pueden consultarse en el capítulo 4.

2.2.2. Iteraciones de desarrollo

Como se ha señalado anteriormente, este proyecto ha sido planeado para constar de cuatro iteraciones. Cada una de esas iteraciones debe producir un incremento en la funcionalidad de la aplicación, y constará de las cuatro fases que pueden observarse en la Figura 2.1:

- Diseño del incremento
- Construcción del incremento
- Implementación del incremento
- Evaluación de los resultados

Al final de cada una de las iteraciones hay una fase de "Evaluación de los resultados" que se materializa en una reunión con los usuarios en la que se se utilizarán técnicas propias de DCU, como test de usabilidad con prototipos de bajo coste y pruebas con las versiones incrementales. Por otra parte, al principio de cada iteración también hay una reunión de seguimiento con los tutores. En esta reunión se tratan los avances realizados en la iteración anterior, se considera el resultado de la evaluación con los usuarios y se revisa la planificación para el incremento

actual. En la Figura 2.3 puede observarse la planificación prevista para las cuatro iteraciones del proyecto. Por último, cabe destacar que las tres últimas iteraciones incluyen una tarea que se desarrolla en paralelo con el resto de tareas durante toda la iteración. Esta tarea consiste en revisar, corregir y refinar la versión del incremento anterior, incorporando el *feedback* y las conclusiones obtenidas tras la evaluación con los usuarios.

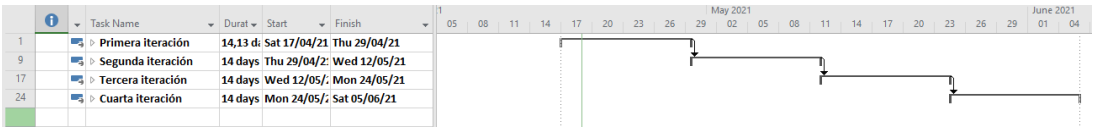

Figura 2.3: Diagrama de Gantt de las cuatro iteraciones del proyecto.

2.2.2.1. Primera iteración (17/04/21 - 29/04/21)

El propósito de la primera iteración es obtener un **prototipo de bajo coste** interactivo que cubra las funcionalidades principales de la aplicación y que permita hacer una prueba de usabilidad con los usuarios.
Respecto a la implementación, al término de esta iteración se prevee tener una primera versión de toda la parte de **autenticación** de usuarios y de programación de actividades por parte de los docentes (ver Figura 2.4).

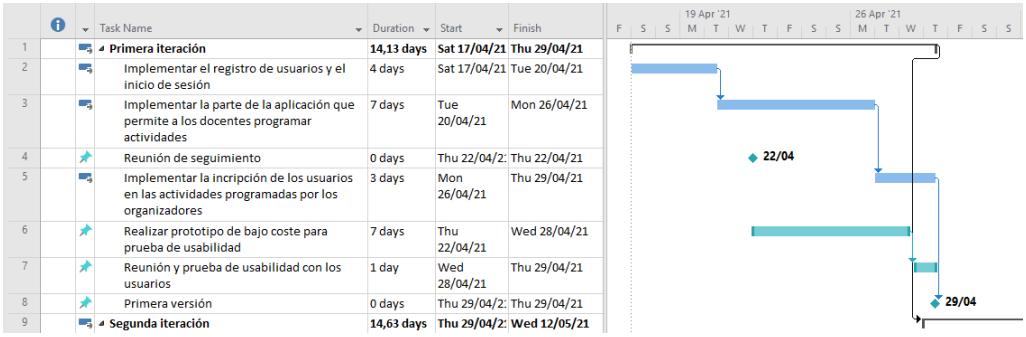

Figura 2.4: Diagrama de Gantt de la primera iteración.

2.2.2.2. Segunda iteración (29/04/21 - 12/05/21)

La segunda iteración tiene como objetivo obtener como incremento una implementación que permita al usuario completar la forma más "básica" de actividad de **orientación**, es decir, sin contenido "educativo". En esta iteración hay una tarea, relativa a la detección del paso por una baliza, que es especialmente delicada, por las dificultades técnicas. El diagrama de Gantt diseñado para esta iteración puede observarse en la Figura 2.5.

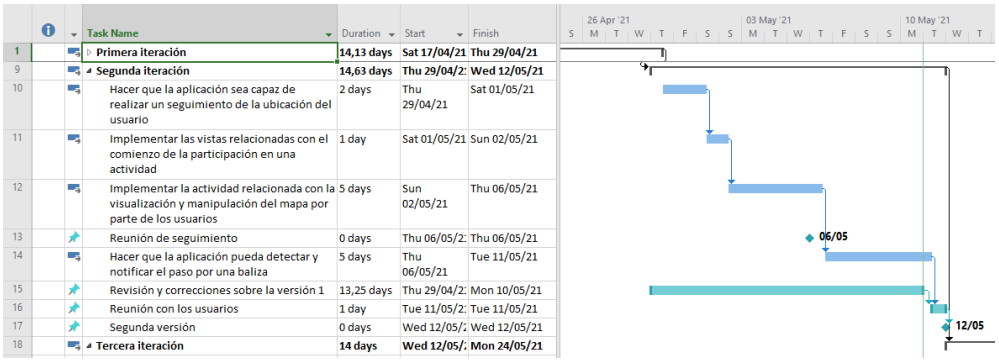

Figura 2.5: Diagrama de Gantt de la segunda iteración.

2.2.2.3. Tercera iteración (12/05/21 - 26/05/21)

La versión de la aplicación resultante de esta iteración debe permitir realizar actividades de orientación pero con el añadido del contenido educativo en las balizas. Con esta versión, los usuarios pueden completar las actividades propuestas al pasar por una baliza, y también pueden compartir los datos de su actividad con el docente (ver Figura 2.6).

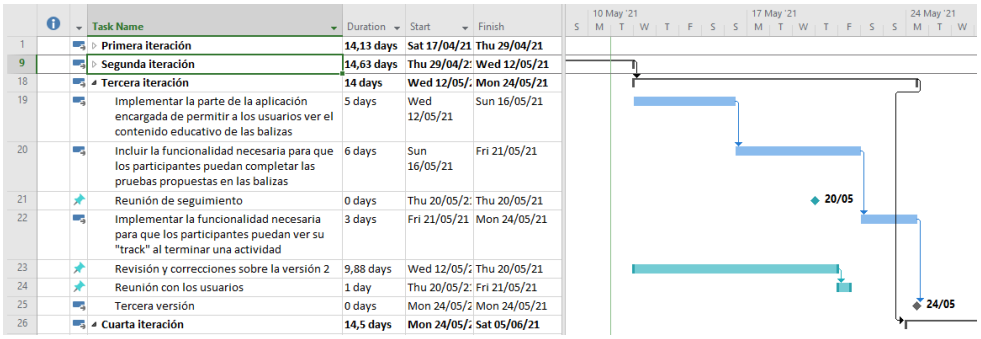

Figura 2.6: Diagrama de Gantt de la tercera iteración.

2.2.2.4. Cuarta iteración (26/05/21 - 05/06/21)

El objetivo para la cuarta iteración es tener toda la funcionalidad de la aplicación implementada. Los principales añadidos que contendrá esta versión sobre su predecesora serán: permitir a los docentes consultar los datos y las respuestas de los participantes de una actividad completada, y permitir a los usuarios ver su perfil y sus *settings*. Las actividades previstas para esta iteración pueden consultarse en la Figura 2.7.

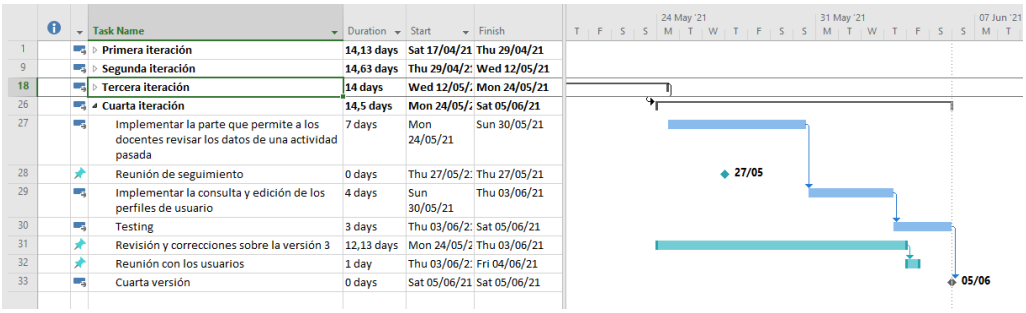

Figura 2.7: Diagrama de Gantt de la cuarta iteración.

2.2.3. Análisis de riesgos

Otra fase importante de la planificación de un proyecto de desarrollo de software es la realización de un análisis de riesgos. Un riesgo hace referencia a un "posible evento o condición que, en caso de llegar a producirse, tendrá un efecto positivo o negativo sobre los objetivos de un proyecto" [15] (cap. 7). Durante la planificación de un proyecto se deben tener en cuenta los riesgos a los que este está expuesto, siguiendo los pasos que se describen en las siguientes subsecciones.

2.2.3.1. Identificación de riesgos

Este paso consiste en identificar los posibles riesgos que pueden aparecer durante el desarrollo de un proyecto. A la hora de identificar los riesgos, existen principalmente dos técnicas que pueden facilitarnos la tarea: las ***checklist*** y las sesiones de ***brainstorming*** [15] (cap. 7).

Las *checklist* recogen la experiencia de proyectos pasados, e información sobre algunos riesgos recurrentes en los proyectos de desarrollo de software. Un ejemplo bien conocido de *checklist* de este tipo es la lista con los diez riesgos más comunes en los proyectos de desarrollo de software, elaborada por B.W. Boehm [4].

Por su parte, las sesiones de *brainstorming* se realizan junto con los *stakeholders* del proyecto con experiencia en proyectos similares. A menudo, estas personas tienen facilidad para identificar posibles riesgos que de otro modo pasaríamos por alto.

Para la identificación de riesgos de este proyecto, se han empleado ambas técnicas, obteniendo los riesgos representados en la Tabla 2.1. Puede comprobarse que varios de estos riesgos han sido directamente extraídos de la lista de Boehm anteriormente mencionada. Sin embargo, otros han sido obtenidos al reflexionar sobre la planificación y tras las reuniones con los profesores.

2.2.3.2. Análisis y priorización de riesgos

Durante esta fase, se analiza de entre los riesgos identificados, cuáles son los más importantes o los que más pueden afectar al desarrollo del proyecto y, por lo tanto, a los que debemos prestar mayor atención y recursos. A la hora de asignar prioridad a los riesgos, una de las técnicas que podemos utilizar es el análisis de **exposición al riesgo**, también conocido como ***Risk Exposure* (*RE*)** [15] (cap. 7). Mediante esta técnica, se calcula para cada riesgo un valor numérico, al que denominamos *Risk Exposure*, y que representa cuán fuerte está nuestro proyecto expuesto a ese riesgo y, por lo tanto, la **prioridad** que debemos concederle. El *RE* se calcula de la siguiente manera:

$$RE = probabilidad \times impacto$$

Las variables *probabilidad* e *impacto* representan la probabilidad de que el riesgo se manifieste y cuánto de "grave" sería en caso de hacerlo, respectivamente. A cada una de esas variables se les asigna un valor entre 0 y 10. Finalmente, cuanto más alto sea el *RE* de un riesgo, mayor prioridad debemos asignarle. Como puede observarse, en la Tabla 2.1 aparecen los riesgos de este proyecto representados de mayor a menor prioridad, según lo calculado mediante la técnica de *Risk Exposure*.

Riesgo	**Probabilidad**	**Impacto**	**Prioridad**
1 Errores al estimar la duración de las tareas	8	7	56
2 Cambios en los requisitos durante el desarrollo	7	7	49
3 Implementación de las funcionalidades equivocadas	7	6	42
4 Implementación de interfaz de usuario equivocada	6	5	30
5 Errores o *bugs* durante el desarrollo	5	5	25
6 *"Gold-plating"*	6	4	24
7 Estancamiento por falta de experiencia en algún punto	4	4	16
8 "Parálisis por análisis"	6	2	12
9 Problemas con la máquina del desarrollador	2	3	6
10 Enfermedad/indisponibilidad del desarrollador	2	2	4

Tabla 2.1: Identificación de los principales riesgos del proyecto

2.2.3.3. Planificación de riesgos

Una vez identificados y analizados los riesgos, el siguiente paso es decidir qué medidas vamos a tomar en caso de que éstos lleguen a manifestarse. A la hora de enfrentarnos a un riesgo hay varios tipos de medidas que podemos adoptar. Las más importantes son las medidas de **reducción** y las medidas de **contingencia**. Las medidas de reducción, son aquellas que se aplican sin esperar a que un riesgo se materialice. De esta forma, son útiles bien para reducir la probabilidad de que un riesgo ocurra, o bien para reducir el impacto negativo que podría llegar a tener en caso de manifestarse. Por su parte, las medidas de contingencia solo se aplican una vez que el riesgo ya se ha materializado, y tienen como objetivo reducir los daños que este pueda causar. En las tablas que siguen a continuación se muestran las medidas de reducción y contingencia previstas para los riesgos más importantes del proyecto.

Riesgo	**1 Errores al estimar la duración de las tareas**
Reducción	Planificación de las tareas dejando margen por si requieren más tiempo del esperado
Contingencia	Replanificación, reajustes en el alcance del proyecto y/o reducción de funcionalidades a implementar

Tabla 2.2: Riesgo 1: medidas de reducción y contingencia

Riesgo	**2 Cambios en los requisitos durante el desarrollo**
Reducción	DCU previene la probabilidad de que esto ocurra, y el desarrollo incremental el impacto que tendría en caso de ocurrir
Contingencia	Contactar con los usuarios para ponernos de acuerdo en los nuevos requisitos

Tabla 2.3: Riesgo 2: medidas de reducción y contingencia

Riesgo	**3 Implementación de las funcionalidades equivocadas**
Reducción	DCU ayuda a prevenir esta situación, al igual que la utilización de prototipos y el desarrollo incremental
Contingencia	Gracias a los incrementos y las reuniones frecuentes, si este riesgo se manifiesta, los errores serán pequeños, por lo que se podrá desechar la funcionalidad equivocada y continuar el proyecto

Tabla 2.4: Riesgo 3: medidas de reducción y contingencia

Riesgo	**4 Implementación de la interfaz de usuario equivocada**
Reducción	DCU ayuda a prevenir esta situación, al igual que la utilización de prototipos y el desarrollo incremental
Contingencia	Realizar nuevos prototipos para mostrarselos a los usuarios y ponernos de acuerdo sobre cómo tiene que se la interfaz

Tabla 2.5: Riesgo 4: medidas de reducción y contingencia

Riesgo	**5 Errores o *bugs* durante el desarrollo**
Reducción	Asegurarse de que cada incremento en la funcionalidad funciona bien antes de continuar con el siguiente incremento
Contingencia	Si no se detecta rápido el error, volver al último incremento que sí funcionaba y volver a empezar desde ahí

Tabla 2.6: Riesgo 5: medidas de reducción y contingencia

Riesgo	**6 "*Gold-plating*"**
Reducción	Preguntar a los usuarios cuando haya alguna duda sobre si lo que se va a implementar es importante/prioritario o no
Contingencia	Reunirse con los usuarios para asegurar que lo que estamos implementando les aporta alguna utilidad tangible

Tabla 2.7: Riesgo 6: medidas de reducción y contingencia

Riesgo	**7 Estancamiento por falta de experiencia en algún punto**
Reducción	Antes de comenzar a implementar, identificar las áreas clave sobre las que se va a necesitar conocimiento, y formarse en ellas
Contingencia	Aprovechar la circunstancia de estar en un grupo multidisciplinar para preguntar a alguien que tenga experiencia en la materia

Tabla 2.8: Riesgo 7: medidas de reducción y contingencia

Riesgo	**8 "Parálisis por análisis"**
Reducción	Emplear técnicas de DCU para entender cuáles son los requisitos clave de la aplicación y empezar a trabajar con los que resulten más sencillos
Contingencia	Buscar funcionalidad o características de la aplicación de las que no haya duda que será necesarias para empezar a implementarlas mientras se siguen diseñando otras soluciones

Tabla 2.9: Riesgo 8: medidas de reducción y contingencia

Riesgo	**9 Problemas en la máquina del desarrollador**
Reducción	Mantener el contenido del proyecto "a salvo" utilizando copias de seguridad y repositorios software remotos.
Contingencia	El grupo GSIC/EMIC podría proporcionar alguna máquina de *back-up* en caso de que fuera necesario

Tabla 2.10: Riesgo 9: medidas de reducción y contingencia

Riesgo	**10 Enfermedad/indisponibilidad del desarrollador**
Reducción	Planificar dejando un margen para que las tareas puedan extenderse en caso de necesidad
Contingencia	Si es necesario, aprovechar el margen previsto para este tipo de problemas que puedan ocurrir

Tabla 2.11: Riesgo 10: medidas de reducción y contingencia

2.3. Seguimiento del proyecto

En esta sección se describe el seguimiento realizado del proyecto, las diferencias respecto a la planificación inicial, los riesgos que se manifestaron, y las consecuencias que tuvieron.

2.3.1. Monitorización de riesgos

Durante el desarrollo de este proyecto se manifestaron algunos de los riesgos descritos en 2.2.3. A continuación veremos cuáles fueron y el impacto que tuvieron:

1. **Errores al estimar la duración de las tareas**. Éste fue el riesgo que se manifestó más a menudo y con mayor intensidad, pues se subestimó en varias ocasiones la complejidad de las tareas. Debido a que esto era previsible, al hacer la planificación ya se había asignado a este riesgo el *RE* más elevado (56). Para paliar sus efectos se aplicaron las medidas previstas, es decir, replanificación y reducción de características o funcionalidades menos importantes. Aún así, los efectos se notaron en la duración total del proyecto.

2. **Errores o *bugs* durante el desarrollo**. Éste riesgo se manifestó principalmente en dos puntos: al implementar el servicio en segundo plano que se encarga de hacer el seguimiento de la ubicación del usuario, y tras hacer una refactorización del código. En el primer caso, al ser una característica nueva (y fundamental para la aplicación) la que se estaba intentando implementar, no quedó más remedio que asumirlo y hacer pruebas hasta solucionar los errores. En el segundo caso, gracias al control de versiones, se pudo aplicar la medida de contingencia de volver a la última versión sin errores. A causa de este riesgo, el proyecto se retrasó en total dos o tres días aproximadamente.

3. **Estancamiento por falta de experiencia en algún punto**. Este riesgo se manifestó en varios puntos del desarrollo, con distinto grado de intensidad. El peor caso ocurrió al implementar la parte de la aplicación encargada de enviar notificaciones al usuario cuando éste alcanza una baliza. Al igual que el riesgo anterior, se estima que este también tuvo un impacto de unos dos o tres días en total.

2.3.2. Comparación respecto a la planificación inicial

A raíz de los riesgos que se manifestaron durante el proyecto que hemos visto en la sección anterior, se produjeron algunas desviaciones temporales respecto a la planificación inicial. En la Figura 2.8 se ha representado con barras azules la planificación inicial por iteraciones, y con barras marrones la misma información pero de acuerdo a las fechas reales de las iteraciones. La barra morada representa las tareas de desarrollo que hubo que realizar al final de las cuatro iteraciones previstas, con las que en un principio no se había contado. En la Figura 2.9 se han incluido además las reuniones mantenidas con los usuarios en las que se les mostró el progreso de la aplicación.

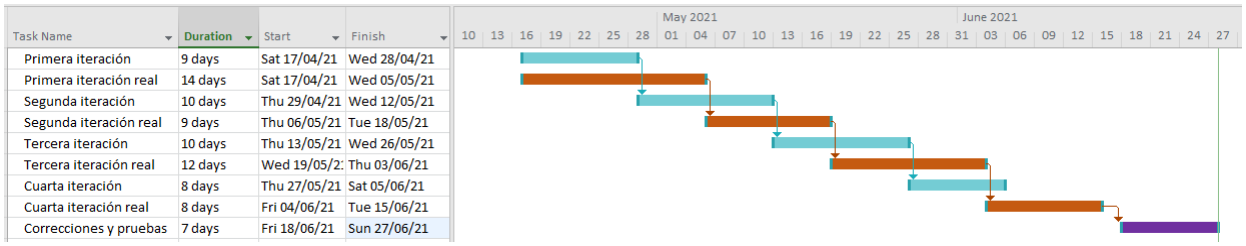

Figura 2.8: Planificación inicial y seguimiento.

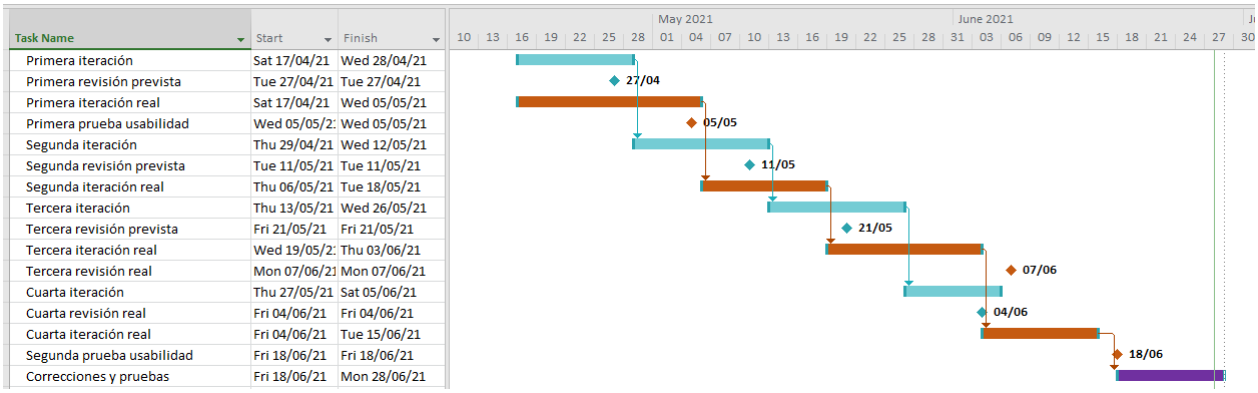

Figura 2.9: Planificación inicial y seguimiento (*millestones* incluidos).

2.4. Presupuesto

Este proyecto se desarrolla bajo el contexto de un trabajo de fin de grado, por lo que no supone costes reales en cuanto a la remuneración del desarrollador. Sin embargo, a continuación se hace una estimación de cuál podría ser el coste si el proyecto se encargase a una empresa de software. Para calcular el presupuesto se han tenido en cuenta los siguientes aspectos:

- **Coste de personal**. El coste que supone un programador *junior* para una empresa ronda los 18 € por hora. La estimación temporal para este proyecto es de unas 480 horas, por lo tanto, el coste de personal serían $18 \times 480 = \mathbf{8640}$ €.
- **Coste de hardware**. El hardware utilizado para desarrollar este proyecto ha consistido en una máquina que costó en el momento de su compra unos 650 €, y tiene una vida útil estimada de 7 años (84 meses). La duración estimada de este proyecto en meses es 3, por lo tanto el coste del hardware se podría calcular así: $3/84 \times 650 = \mathbf{23}$ €.
- **Coste de software**. Todo el software necesario para el desarrollo de este proyecto o bien es gratuito, o se dispone de una licencia gratuita, o se puede utilizar un plan de pruebas gratuito, por lo tanto, el coste es **0** €.

Sumando los tres costes obtenemos un presupuesto aproximado de $\mathbf{8640} + \mathbf{23} + \mathbf{0} = \mathbf{8663}$ €.
Como hemos visto en 2.3, finalmente el proyecto se alargó en unos 20 días, lo que supondría un sobrecoste de personal de $8 \times 20 \times 18 = \mathbf{2880}$ €, que hay que sumar al precio calculado en la planificación. Finalmente, el presupuesto del proyecto sería $8663 + 2880 = \mathbf{11543}$ €.

Capítulo 3

Geoposicionamiento en Aplicaciones Móviles

Uno de los objetivos de este proyecto consistía en analizar y estudiar las diferentes posibilidades existentes a la hora de geoposicionar objetos y usuarios desde las aplicaciones móviles. La intención era encontrar un método que fuese adecuado para el contexto de uso esperado de esta aplicación y que se adaptara a las necesidades y objetivos de los usuarios.

3.1. Métodos para el geoposicionamiento en las aplicaciones móviles

A continuación, se introducen brevemente los diferentes métodos analizados, junto con las ventajas e inconvenientes que cada uno presenta para este proyecto, así como las razones por las que finalmente se ha decidido emplear.

3.1.1. Geoposicionamiento a través de GPS

Hoy en día, prácticamente todos los dispositivos móviles vienen con un sistema GPS integrado de fábrica. El **Sistema de Posicionamiento Global** (**GPS**; en inglés, ***Global Positioning System***), fue desarrollado por el Departamento de Defensa de Estados Unidos, y actualmente es propiedad de la Fuerza Espacial de los Estados Unidos. Este sistema, localiza a un usuario mediante **trilateración**, con los datos de al menos cuatro satélites [14]. La localización mediante GPS tiene un margen de error de entre cinco y diez metros. Hay métodos, como el **GPS diferencial**, que permiten minimizar este error. Sin embargo, esta tecnología requiere unos receptores especiales, con los que los dispositivos móviles no cuentan.

Una de las premisas de las que partíamos al iniciar este proyecto consistía en que era prioritario no provocar un impacto en el medio donde las actividades se desarrollasen. Las balizas no serían objetos físicos desplegados en el lugar por los organizadores de la actividad, sino que debían ser elementos ya existentes en el entorno, cuya posición es conocida. Esto, facilita la organización de las actividades, además de proteger el entorno en el que se desarrollan. Todo ello hizo que el geoposicionamiento mediante GPS fuera la primera opción que consideramos. Sin embargo, las actividades realizadas mediante esta aplicación tienen como escenario el Campo Grande, y las balizas son principalmente árboles que en él se encuentran. Algunos de esos árboles están muy próximos entre sí, por lo que los márgenes de error ofrecidos por el GPS nos parecían demasiado amplios. Esto fue lo que nos llevó a estudiar otras alternativas a la hora de geoposicionar, las cuales se describen a continuación.

3.1.2. Geoposicionamiento con apoyo de otros dispositivos

A continuación se enumeran algunos métodos que se pueden emplear para geoposicionar a un usuario empleando otras tecnologías distintas al GPS. Todos estos métodos, pueden utilizarse tanto en solitario, como junto con la localización por GPS, para complementarla y hacerla más precisa.

- **Balizas electrónicas** (en inglés, ***beacons***). Los *beacons* son dispositivos transmisores que se utilizan para radiar una señal, normalmente ***bluetooth*** de baja energía, a dispositivos móviles que se encuentran cerca de ellos, sin necesidad de establecer una sincronización previa [6]. Utilizar estos dispositivos permitiría **reducir el margen de error propio del GPS**, pero implicaría tener que adquirir los tranmisores y desplegarlos en

el escenario de la actividad. Además, pueden requerir cierto mantenimiento. Como mínimo, hay que recargar su batería en algún momento.

- **NFC (*Near Field Comunication*)**. NFC es una tecnología de comunicación inalámbrica, de corto alcance y alta frecuencia que permite el intercambio de datos entre dispositivos [20]. A efectos de esta aplicación, este método tiene más o menos las mismas ventajas e inconvenientes que los *beacons*. Al igual que éstos, también sería necesario desplegar los dispositivos por el terreno. Para establecer conexión entre dispositivos mediante esta tecnología, hay que estar a una distancia de **centímetros**, por lo que la **certeza de que el usuario ha encontrado la baliza sería máxima**. También como los *beacons*, es probable que su uso sólo fuera útil combinado con el GPS. De otro modo, la aplicación perdería toda noción de dónde está el usuario durante la actividad salvo cuando pasa por una baliza.

- **Código QR (*Quick Response code*)**. Un código QR es un módulo para almacenar información en una matriz de puntos o en un código de barras bidimensional [19]. Los dispositivos móviles pueden escanear estos códigos mediante un lector específico. Se puede generar un código para cada baliza, y dejarlo en algún lugar a la vista, de tal modo que cuando el usuario pase por allí, pueda escanearlo. Entonces la aplicación puede registrar el paso del usuario por la baliza, y desencadenar alguna acción si es necesario. Al igual que los dos métodos anteriormente comentados, sirve para proporcionar a la aplicación una **gran certeza** de que el usuario ha pasado por un lugar en concreto. Respecto a los dos métodos anteriormente referidos, éste tiene la ventaja de ser más barato y requerir menos mantenimiento.

- **Reconocimiento de los elementos del entorno**. Se pueden identificar elementos del entorno mediante el reconocimiento de imágenes o visión artificial. Esta técnica es una disciplina científica que incluye métodos para adquirir, procesar, analizar y comprender las imágenes del mundo real con el fin de producir información numérica o simbólica para que puedan ser tratados por un ordenador [13]. Este método, consistiría en que el usuario obtuviese una imagen de la baliza mediante la cámara de su dispositivo. La aplicación sería capaz de discernir si la foto contiene al elemento del entorno adecuado. La razón de que se haya decidido no incluir este método por el momento, es que la dificultad técnica es muy grande. Los elementos que habría que reconocer son muy variados, y muy posiblemente, cambiantes. A día de hoy, sería difícil encontrar a nuestra disposición una tecnología que permitiese reconocer un árbol (posiblemente rodeado de más árboles) a lo largo de las distintas estaciones del año y desde distintos ángulos, con un grado de certeza aceptable. Sin embargo, a medida que la tecnología avance, puede que éste se convierta en un método muy interesante para incluir en la aplicación en el futuro.

A la hora de incorporar a la aplicación una o varias de estas tecnologías, a parte de las ventajas e inconvenientes mencionados anteriormente, habría que tener en cuenta también otros aspectos. Por ejemplo, es importante considerar la mayor susceptibilidad al bandalismo y al deterioro que presentan estos métodos. Por otra parte, también se debe tener en cuenta que pueden limitar las posibilidades de generalización de la aplicación a otros escenarios, puesto que allí donde se pretenda realizar una actividad, previamente habrá que preparar el terreno. Por último, todos ellos causan un impacto en el entorno.

3.1.3. Método de geoposicionamiento empleado en esta aplicación

En conclusión, atendiendo a la premisa de ofrecer una solución que no necesitara un despliegue previo y que no tuviera un impacto en el entorno, se optó por el **GPS**. Aunque en un principio nos pareció que su precisión no era suficientemente buena, el resto de ventajas que ofrece pesaron más, y nos hicieron decantarnos por este método. Esto implica que al diseñar las actividades, habrá que tomar la precaución de no situar las balizas a una distancia tan pequeña entre sí como para que el GPS pueda arrojar resultados ambiguos. Por otra parte, de las reuniones mantenidas con los usuarios se deduce que la precisión ofrecida por el GPS es suficiente para satisfacer sus necesidades, al menos en el contexto de este trabajo de fin de grado. La razón es que los usuarios reportan utilizar otras aplicaciones de orientación que también emplean el GPS como método de geoposicionamiento y cuya precisión, en general, les parece aceptable, si bien es cierto que un escenario como el del Campo Grande presenta retos adicionales por la complejidad del entorno y la potencial proximidad y similitud de las balizas entre sí.

Por último, cabe destacar que el margen de error característico del GPS puede llevar a la situación de que un usuario se aproxime a una baliza y la aplicación automáticamente considere que la ha encontrado, cuando en realidad, es posible que el usuario no haya identificado el elemento del entorno que tenía que encontrar pese a estar cerca de él. Este problema es abordado con mayor detalle en la siguiente sección.

3.2. Ayuda al usuario para la identificación de objetos físicos usados como balizas

Como se ha comentado en la sección anterior, en esta aplicación el problema de la geoposición tiene dos facetas. Por una parte, está la **certeza** con la que la aplicación puede llegar a saber si un usuario ha encontrado o no una baliza. Por otro lado, está la facilidad con la que un usuario es capaz de **reconocer** una baliza concreta. Es decir, que aún cuando un usuario se aproxime lo suficiente a una baliza como para que la aplicación interprete que la ha encontrado, puede ocurrir que dicho usuario no esté consiguiendo reconocer qué elemento del entorno conforma la baliza. Dependiendo del entorno, puede haber muchos elementos en un radio de diez metros (el margen de error esperado del GPS). Al fin y al cabo, en la orientación clásica, los participantes saben lo que buscan y qué aspecto tiene. Pero esto no tiene por qué ser así en el caso de las balizas que utilizará la aplicación, que formarán parte del entorno.

La primera las dos cuestiones anteriormente mencionadas podría llegar resolverse prácticamente en su totalidad utilizando los métodos de geoposicionamiento apoyados mediante dispositivos físicos que se han detallado en la sección previa. Sin embargo, mientras no se incorporen dichos métodos al sistema, se pueden aprovechar algunas funcionalidades de la aplicación para reducir la incertidumbre, si ese es el objetivo. Por ejemplo, mediante la aplicación se pueden plantear al usuario **retos o preguntas estrechamente relacionados con el entorno de la baliza**, de tal forma que el hecho de que éste sea capaz de responder correctamente, ya constituya por sí mismo una prueba más de que el usuario efectivamente a llegado al lugar que debía.

Para la cuestión de la capacidad de reconocer una baliza próxima por parte de los participantes, la aplicación puede proporcionar al usuario **pistas** (preparadas por los organizadores de la actividad). Estas pistas pueden ser imágenes o descripciones. Además, el uso de un mapa de orientación deportiva, donde se representa la baliza con precisión, ayuda también al usuario a identificar con certeza el elemento del entorno que corresponde a la baliza.

En conclusión, las personas que organicen las actividades y configuren los recorridos, así como el "contenido educativo" de las balizas, podrán "jugar" con estos dos ajustes para conseguir una actividad con las características que deseen.

3.3. Resumen del capítulo

En este capítulo se han analizado las distintas formas que existen de geoposicionar objetos y usuarios en el contexto de las aplicaciones móviles, así como sus ventajas e inconvenientes en el caso particular de este proyecto. Finalmente, el método elegido ha sido el geoposicionamiento mediante GPS. Como hemos visto, este sistema podrá verse complementado mediante la utilización "pistas" y "retos" que ayuden a reducir la incertidumbre causada por el margen de error propio del GPS, en aquellos casos en los que los diseñadores de las actividades lo consideren oportuno. Por último, no se descarta llegar a implementar en el futuro alguno de los métodos de geoposicionamiento apoyado mediante dispositivos discutidos en este capítulo. Esto dependerá en gran medida de si finalmente se opta por generalizar la aplicación a otros escenarios distintos al Campo Grande, o si por el contrario, se elige mejorar sus características para "afinar" su funcionamiento en dicho escenario particular y permitir implementar en él actividades en las que sea necesario un grado de precisión máximo.

Capítulo 4

Análisis

En este capítulo veremos la investigación sobre usuarios propia de la metodología DCU realizada al inicio del proyecto. Además, veremos cómo dicha investigación nos ayudó a comprender mejor el dominio del sistema, las necesidades de los usuarios, los requisitos y los casos de uso, los cuales quedan documentados también en este capítulo.

4.1. Investigación sobre Usuarios

Tal y como vimos en el capítulo 1, uno de los objetivos de este proyecto es *identificar y aplicar técnicas propias del diseño centrado en el usuario, con el fin de mejorar el diseño de la aplicación.* Además, en el capítulo 2 se mencionó que una de las principales características de dicha metodología es que busca *situar a los usuarios en el centro del diseño de la aplicación.* La investigación sobre usuarios consiste en un conjunto de métodos y técnicas que tienen como propósito ayudar a conseguir precisamente eso, centrar el diseño en el usuario [16]. Gracias a la investigación sobre usuarios aspiramos a alcanzar un entendimiento profundo sobre las personas que van a utilizar nuestro sistema, lo cual aumenta las posibilidades de crear un producto que ofrezca una experiencia de uso agradable, facilidad de aprendizaje y buenos niveles de usabilidad.

4.1.1. Proceso de investigación con usuarios

En el capítulo 2 vimos que al inicio del proyecto, antes de comenzar con las iteraciones propias de la metodología de desarrollo incremental, se reservaba una "fase inicial" para llevar a cabo tareas de investigación sobre usuarios. La investigación sobre usuarios es especialmente importante durante el análisis del proyecto, ya que conviene tener un conocimiento profundo sobre nuestros usuarios antes de comenzar con el diseño. A continuación veremos en mayor detalle los usuarios sobre los que hemos centrado la investigación, así como los métodos de recogida de datos empleados.

4.1.1.1. Usuarios objetivo de la investigación

En este proyecto, desde un primer momento pueden distinguirse dos grupos de usuarios bien diferenciados. Por una parte está el grupo de los docentes, quienes utilizan la aplicación para organizar actividades y ver el comportamiento de los estudiantes durante las mismas. El segundo grupo está formado por los estudiantes. Por otra parte, en el capítulo 1 del documento hemos visto que el grupo GSIC/EMIC cuenta con algunos colaboradores de la Facultad de Educación y Trabajo Social. Esta colaboración nos ha brindado la oportunidad de mantener reuniones y entrevistas con usuarios reales representativos del primer grupo (docentes, especialmente del ámbito de la Educación Física). De esta forma la aplicación se ha diseñado poniendo como eje las necesidades del profesorado, que es el que va a diseñar y poner en marcha las actividades educativas. El profesorado también representará las necesidades del alumnado, pero como se discutirá más adelante, una limitación de este proyecto ha sido la ausencia de representantes del alumnado en el proceso de investigación y diseño iterativo.

4.1.1.2. Métodos de recogida de datos

En este apartado veremos los métodos de recogida de datos empleados para la investigación sobre usuarios del proyecto.

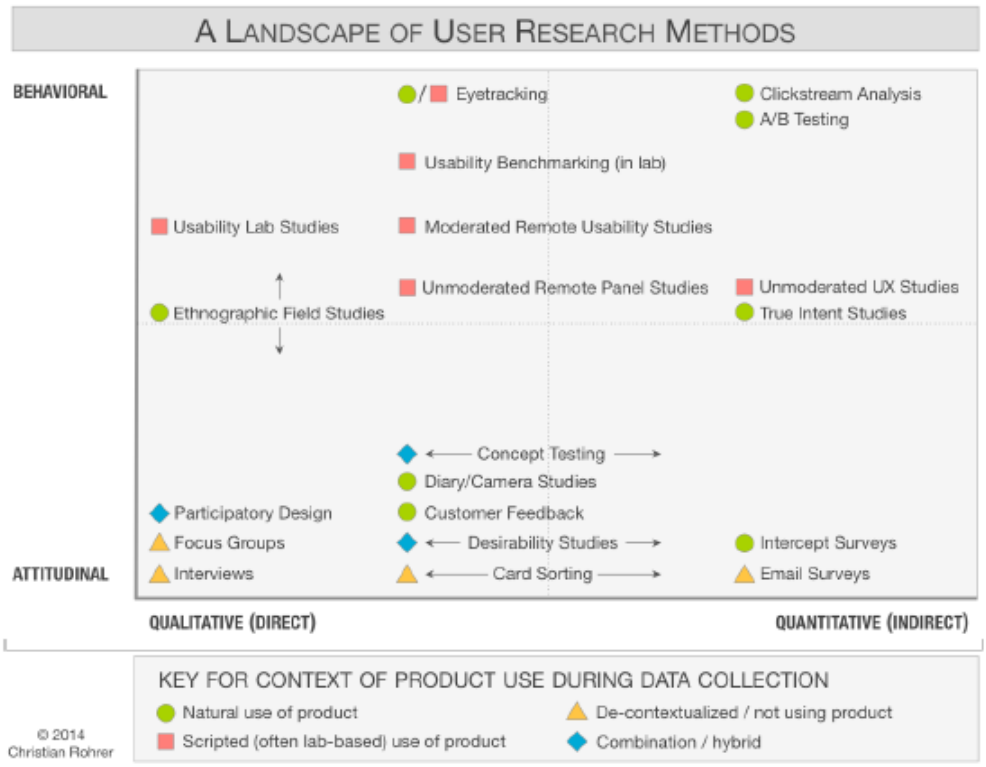

Figura 4.1: Clasificacion de métodos de investigación sobre usuarios
Imagen obtenida de [25].

La Figura 4.1 proporciona una visión general de algunos de los métodos de recogida de datos más comunes, clasificados en dos ejes de acuerdo a su naturaleza *actitudinal* o *del comportamiento*, y a si son *cuantitativos* o *cualitativos*. El primer criterio hace referencia a si un método de recogida de datos se basa en "lo que los usuarios dicen" o en "lo que los usuarios hacen" [25]. Por su parte, el segundo criterio clasifica los distintos métodos atendiendo a si los datos que se recogen son generados a partir de la observación directa (*cualitativos*) o mediante observación indirecta (*cuantitativos*). Habitualmente, los métodos *cuantitativos* implican la utilización de encuestas u otras herramientas analíticas, y nos permiten hacernos una idea de *cuánto* de común es un problema o *cuánto* de importante. En cambio, los métodos *cualitativos* acostumbran a ser más útiles a la hora de entender las causas y las posibles soluciones [25] para un problema. La Tabla 4.1 muestra los métodos que han sido utilizados para la recogida de datos a lo largo de este proyecto.

Fecha	**Método**	**Tipo**	**Fase**
10/03/21	Reunión de *kick-off*	actitudinal y cualitativo	Análisis
19/03/21	*Group interview*	actitudinal y cualitativo	Análisis
24/03/21	*Stakeholders meeting*	actitudinal y cualitativo	Análisis
07/04/21	*Focus group* para *concept testing*	actitudinal y cualitativo	Análisis/Diseño
07/05/21	Test de usabilidad	de comportamiento y cualitativo	Diseño
18/06/21	Test de usabilidad	de comportamiento y cualitativo	Diseño

Tabla 4.1: Métodos de recogida de datos empleados en el proyecto

En la siguiente subsección analizaremos en qué consiste cada método y cómo se ha llevado a la práctica.

4.1.1.3. Desarrollo de la recogida de datos

En este apartado analizaremos cómo se ha realizado la recogida de datos en el proyecto, utilizando los métodos que se enumeran en la Tabla 4.1. Como podrá observarse, en esta sección no se habla en detalle de los *test de usabilidad*. Esto es así porque ese método pertenece a la fase de *Diseño*, y por lo tanto, realizaremos un análisis más exhaustivo del mismo en el capítulo 5. A continuación se describen los métodos pertenecientes a la fase de análisis:

- **Reunión de *kick-off***. Al inicio de un proyecto, la reunión de *kick-off* sirve para juntar a los participantes del mismo, formar equipo y discutir la visión inicial. En esta reunión también se tratan temas como el alcance del proyecto, los objetivos y el plan [30]. En nuestro caso, este encuentro tuvo lugar por videoconferencia el día 10 de marzo de 2021. En él participamos la mayoría de las personas involucradas en el proyecto: tres miembros del grupo GSIC/EMIC (dos de los cuales, tutores de este trabajo de fin de grado), dos colaboradores del grupo GSIC/EMIC pertenecientes a la Facultad de Educación, y el autor de este trabajo.

- **Entrevista (*group interview*)**. Las entrevistas son un método de recogida de datos en el que el investigador realiza preguntas a los usuarios sobre algún tema de interés con el objetivo de obtener cierta información [23]. Para este trabajo, el día 19 de marzo de 2021 se realizó una entrevista grupal por videoconferencia en la que participamos conjuntamente los tres miembros de la Facultad de Educación colaboradores del proyecto, y el autor de este trabajo en el rol de entrevistador. En dicha entrevista el objetivo era comprender los requisitos de la aplicación, así como conocer la forma en que a los usuarios les gustaría interactuar con la tecnología para lograr sus objetivos. La entrevista se condujo de manera *semi-estructurada* [26], es decir, estuvo basada en un guion, pero se dejó libertad para tratar temas emergentes.

- ***Stakeholders meeting***. Un *stakeholders meeting* es una reunión en la que participan todas las personas involucradas en un proyecto. Estas reuniones se pueden organizar con diferentes objetivos. En nuestro caso, la reunión tuvo lugar el día 24 de marzo de 2021 y en ella participamos todas las personas vinculadas a este proyecto. En cierto modo, la reunión fue una continuación del *kick-off*, ya que se trataron los mismos temas con la novedad de que los *stakeholders* de la Facultad de Educación presentaron por escrito un documento en el que describían todas las actividades que les gustaría realizar con la aplicación. De este modo, la reunión nos sirvió para concretar algunas ideas, descartar otras, y plantear otras para posibles proyectos futuros.

- ***Focus group* para validación de conceptos (*concept testing*)**. Un *focus group* en un método de investigación de usuarios en el que los investigadores reunen a un pequeño grupo de personas (típicamente entre seis y nueve) para revisar y discutir cuestiones sobre un diseño en particular [3]. Estos encuentros son de gran utilidad en las primeras fases de un proyecto, y sirven para evaluar la idea, ver las diferentes aproximaciones que se pueden adoptar y cómo de valiosas son las distintas propuestas. En este proyecto se reunió por videoconferencia un *focus group* el día 4 de abril de 2021. En él se presentó un prototipo de baja fidelidad[1] a los tres representantes del grupo de usuarios conformado por los docentes. El objetivo era validar conceptos y comprobar que se habían comprendido bien sus necesidades. La Figura 4.2 muestra el prototipo empleado para esta sesión de recogida de datos. Se trata de un *wireframe*[2] creado con la herramienta Adobe XD[3]. Emplear un wireframe interactivo fue de gran utilidad teniendo en cuenta que la sesión tuvo que ser online debido a las circunstancias provocadas por la pandemia COVID-19. La Figura 4.3 muestra una imagen del desarrollo de la sesión.

4.1.2. Resultados de la investigación sobre usuarios

En esta sección, se describen los resultados obtenidos tras la realización de las técnicas de investigación sobre usuarios anteriormente descritas. En el contexto de la metodología de DCU, es habitual plasmar esos resultados mediante la utilización de *personas* y *escenarios*. A continuación veremos en qué consisten dichas técnicas y qué resultados arrojan sobre la investigación de usuarios en este proyecto.

[1] Un prototipo de baja fidelidad es un prototipo de bajo coste y con poco realismo en el aspecto visual respecto a lo que será producto final. A diferencia de los prototipos de alta fidelidad, destinados las pruebas usabilidad, éstos se utilizan para comprobar que la funcionalidad que se va a desarrollar es correcta, sin prestar demasiada atención a los aspectos visuales o estéticos.

[2] Un wireframe es una representación digital de una página de un producto/aplicación con la que el usuario puede interactuar clickando sobre ella.

[3] Adobe XD: https://www.adobe.com/products/xd.html

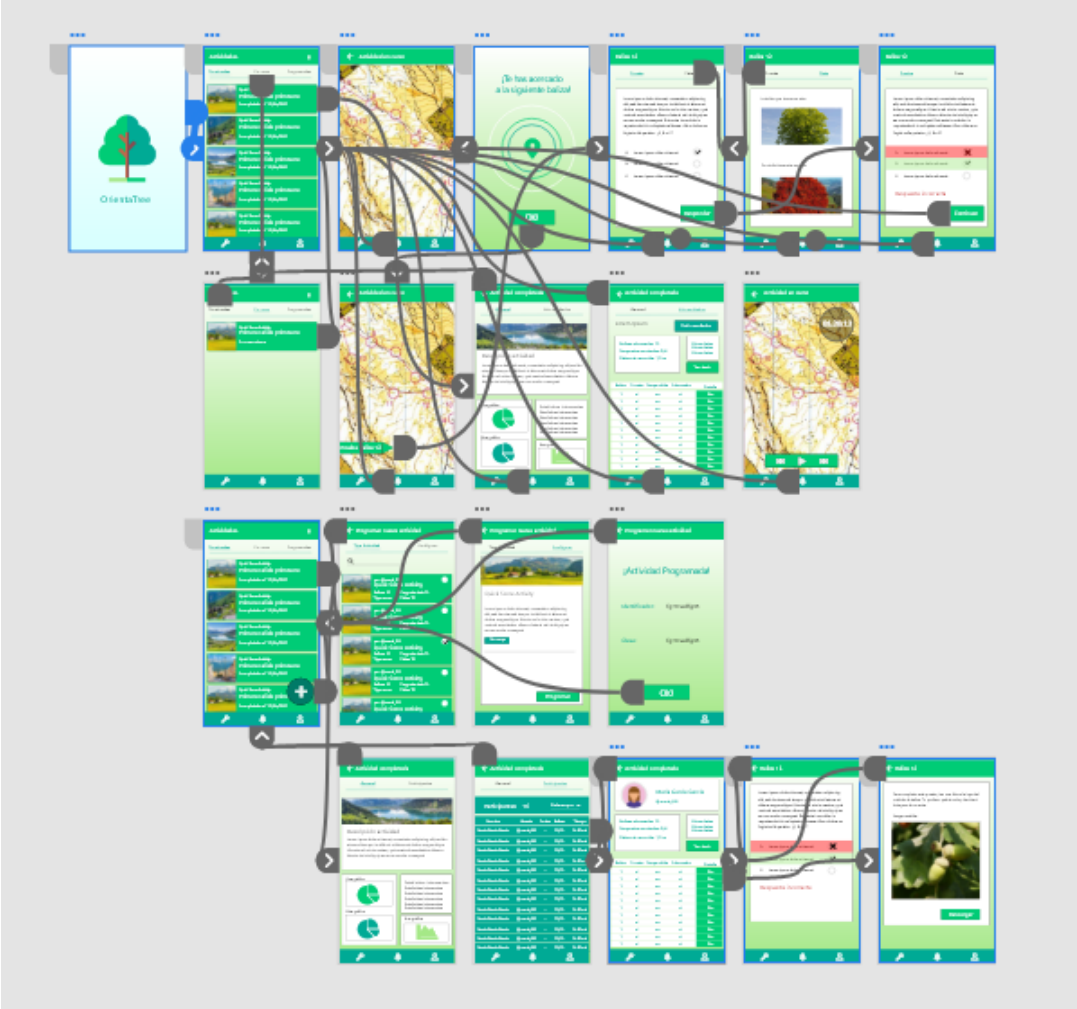

Figura 4.2: Prototipo de baja fidelidad creado con Adobe XD para la validación de conceptos con los usuarios.

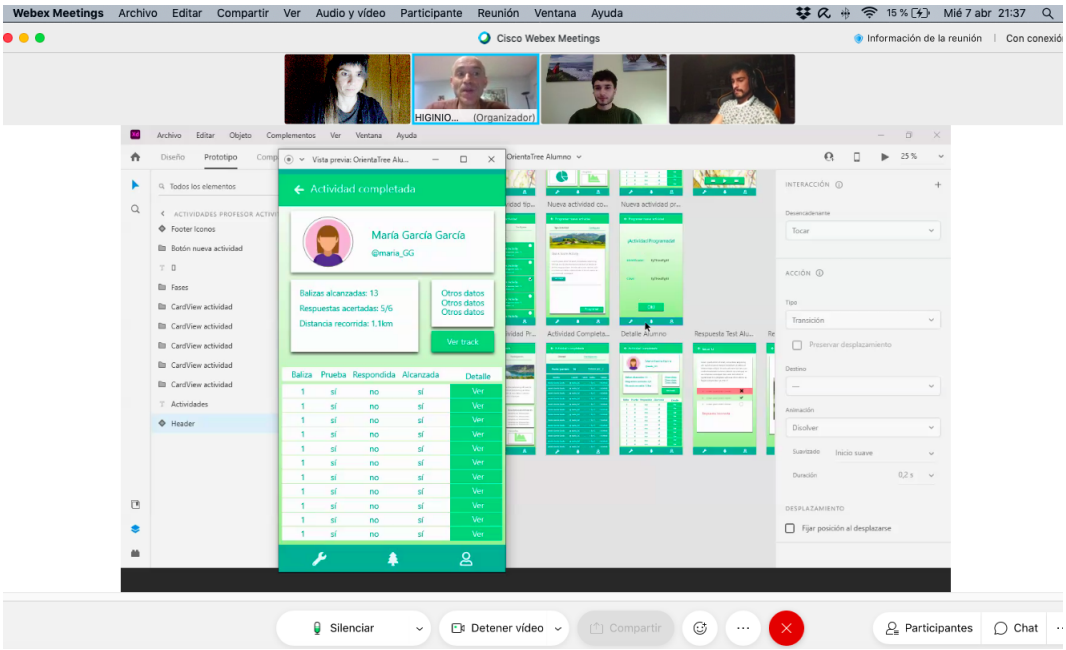

Figura 4.3: Transcurso de la sesión de *Focus group* para la validación de conceptos.

4.1.2.1. Personas

El término *persona* hace referencia a un arquetipo de usuario cuyos objetivos y características lo hacen representativo de un conjunto de usuarios reales más amplio [10]. El procedimiento consiste en escribir descripciones de una o dos páginas sobre usuarios ficticios, incluyendo sus patrones de comportamiento, objetivos y habilidades, así como información contextual y del entorno en el que esa *persona* opera. Habitualmente, en estas descripciones también se incluyen una fotografía y algunos detalles ficticios más, que dotan de mayor realismo a las *personas* y ayudan a los diseñadores a empatizar mejor con los usuarios. Gracias a ello, se reduce el riesgo para los diseñadores de incurrir en lo que se conoce como *diseño autorreferencial* [10]. El *diseño autorreferencial* es lo que ocurre cuando los diseñadores conciben un producto como si ellos mismos fueran los usuarios finales, cuando en realidad, puede que sus objetivos, habilidades y necesidades sean muy distintos a los de dichos usuarios. La Figura 4.4 muestra de manera esquemática las distintas fases del proceso de diseño y cómo *empatizar* con el usuario tiene una importancia especial dentro de ese proceso, para lo cual, la utilización de *personas* es de gran ayuda.

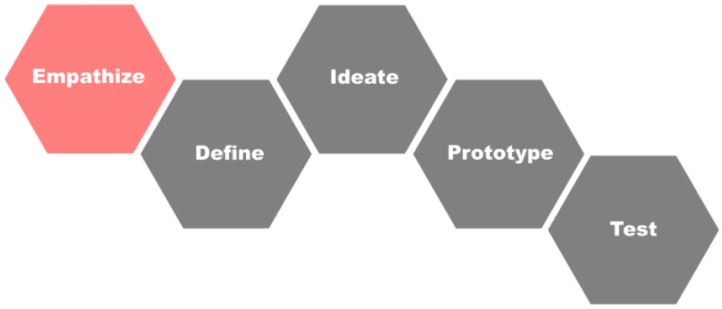

Figura 4.4: *Design thinking process.*
Imagen obtenida de [10].

En este proyecto, la investigación sobre usuarios se ha centrado más en el grupo de usuarios formado por los profesores, ya que es al que se ha tenido acceso. Para representar a este grupo, se ha creado la *persona* de Alfredo Blanco, como puede observarse en la Figura 4.5. A su vez, el grupo de usuarios formado por los estudiantes de primaria se ha caracterizado mediante la persona de Úrsula Buendía, cuya descripción viene detallada en la Figura 4.6.

Cuando en un proyecto se obtienen varias *personas* puede ser difícil y confuso tratar de diseñar teniendo a todas en mente a la vez. En estos casos se recomienda asignarles una prioridad. De este modo se define una *persona primaria*, que será la que más se tenga en cuenta a la hora de diseñar. El resto de *personas* serán *secundarias*. La recomendación es: *"diseña para la primaria y acomoda para las secundarias"* [10]. En este proyecto la *persona* de Alfredo Blanco es la *primaria*, mientras que Úrsula Buendía es la *secundaria*. Finalmente, como puede observarse en las Figuras 4.5 y 4.6, a la hora de crear ambas *personas* nos hemos centrado especialmente en entender los hábitos, las actitudes y las aptitudes que cada uno de los grupos presenta respecto a las TIC, así como sus objetivos y frustraciones. Esto es debido a que ambos grupos podrían tener diferencias significativas en ambos aspectos, y habrá que tenerlo en cuenta a la hora de diseñar.

Figura 4.5: *Persona* 1. Alfredo Blanco, profesor de Educación Física.

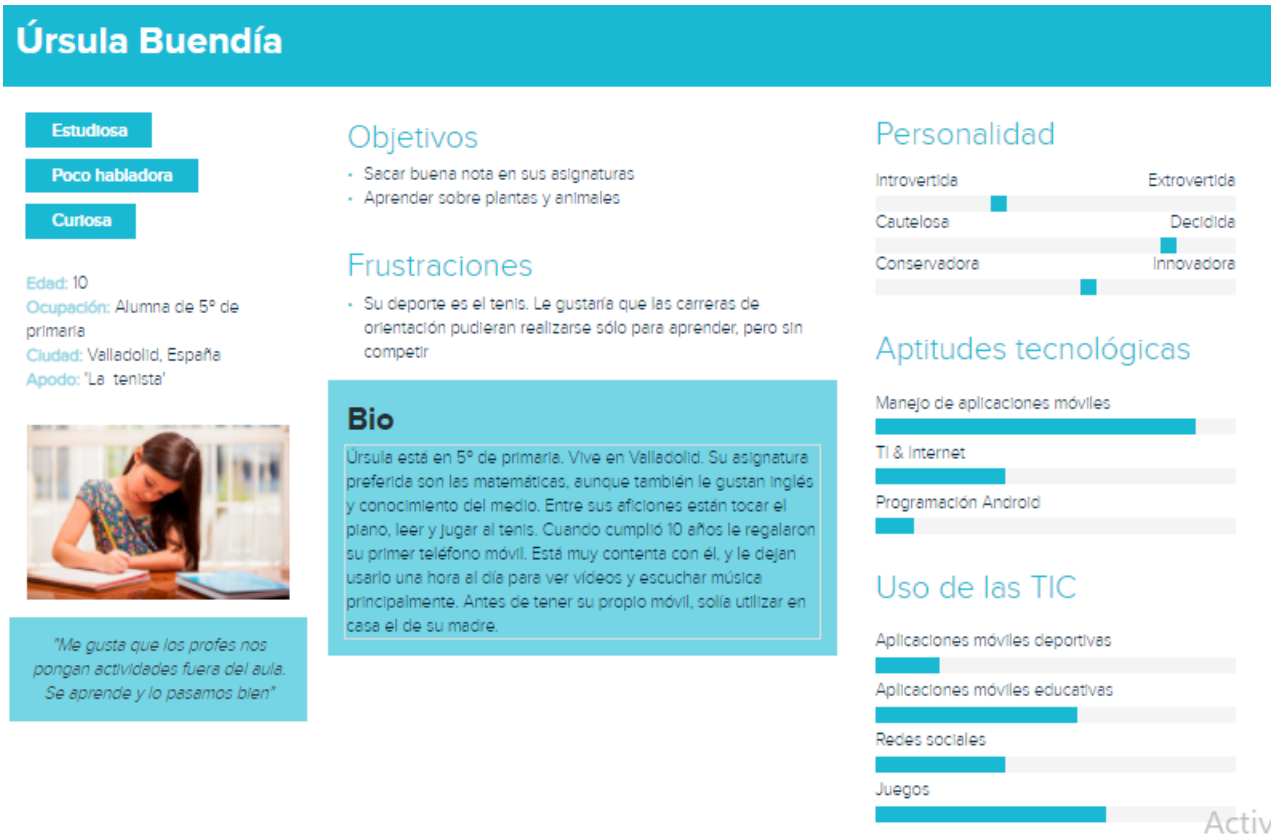

Figura 4.6: *Persona* 2. Úrsula Buendía, alumna de 5º de primaria.

4.1.2.2. Escenarios

Como hemos visto, en la metodología de DCU la utilización de *personas* es de gran ayuda a la hora de identificar cuáles son los usuarios/actores de nuestro sistema, así como sus características y necesidades. Es habitual complementar esta técnica con la utilización de *escenarios*. Un escenario es una situación ficticia que describe cómo una *persona* interactuaría con un producto o aplicación en un contexto determinado para lograr un objetivo [10]. Los escenarios ayudan a los diseñadores a entender cómo interactúan los usuarios con el sistema y cómo tiene que ser el flujo de esta interacción. Además, son de gran utilidad a la hora de recoger los requisitos del sistema. Los escenarios se escriben a alto nivel y vinculados a una *persona*. Describen un caso de uso que se considera probable que ocurra en una interacción real, donde la *persona* utiliza el sistema para lograr un objetivo.

A continuación se describen los escenarios identificados para este proyecto, donde Alfredo y Úrsula son los actores protagonistas.

Escenario 1	
Actor	Úrsula (alumna).
Motivación	Comenzar una actividad de orientación educativa.
Intención	Anotar su hora de salida y visualizar el mapa.
Acción	Accede al mapa de la actividad, anota su hora de salida y comienza con su participación.
Resolución	La aplicación comprueba que Úrsula está en la salida, entonces le permite visualizar el mapa, y empieza a hacer un seguimiento de su ubicación.
Descripción	Úrsula, junto a sus compañeros de clase, va a participar en una actividad de orientación educativa organizada por su profesor Alfredo. Antes de comenzar la actividad, todos se reúnen en el lugar de salida. Allí, Alfredo va dando la salida a los distintos participantes. Previamente, todos se han descargado el mapa de la actividad en la aplicación. Úrsula, cuando llega su turno, solicita a la aplicación comenzar. Entonces la aplicación comprueba que Úrsula se encuentra en el lugar de comienzo, anota la hora de salida y le muestra el mapa de la actividad. Úrsula comienza a buscar las balizas.

Tabla 4.2: Escenario 1

Escenario 2	
Actor	Úrsula (alumna).
Motivación	Completar su paso por una baliza.
Intención	Identificar correctamente la baliza y completar el reto que la aplicación le plantee.
Acción	Solicita a la aplicación que muestre el reto y da respuesta al mismo.
Resolución	Úrsula responde a la pregunta planteada. La aplicación anota el resultado y proporciona un *feedback*. Úrsula continua con la actividad.
Descripción	Úrsula está en medio de una actividad de orientación educativa. Con ayuda del mapa que le muestra la aplicación, cree que se debe de estar acercando a una baliza. De repente, su móvil vibra y emite un sonido, indicando a Úrsula que acaba de alcanzar la segunda baliza de la actividad (el chopo). Entonces Úrsula abre en la aplicación el reto de la baliza. La aplicación le muestra una foto del chopo y una pregunta tipo test relacionada con la forma de las hojas de ese árbol. Gracias a la foto y a la ubicación de la baliza en el mapa de orientación que le muestra la aplicación, Úrsula es capaz de reconocer el árbol. Entonces responde a la pregunta. Acto seguido, la aplicación anota su respuesta y la felicita por haber acertado. Úrsula continua con la actividad.

Tabla 4.3: Escenario 2

Escenario 3	
Actor	Úrsula (alumna).
Motivación	Avanzar en una actividad.
Intención	Averiguar dónde está situada sobre el mapa.
Acción	Solicitar a la aplicación ayuda con la ubicación.
Resolución	La aplicación le muestra a Úrsula dónde está, y Úrsula puede continuar con la actividad.
Descripción	Úrsula aún está en medio de una actividad de orientación educativa propuesta por su profesor Alfredo. A Úrsula se le suelen dar muy bien las carreras de orientación. Sin embargo, en esta ocasión se ha despistado un poco y lleva un rato tratando de entender dónde está situada. Úrsula pide a la aplicación que le muestre dónde está en el mapa de orientación. En este caso, Alfredo había configurado la actividad de tal modo que estuviese permitido solicitar ayuda a la aplicación. Gracias a ello, Úrsula se consigue ubicar y continúa con la actividad.

Tabla 4.4: Escenario 3

Escenario 4	
Actor	Úrsula (alumna).
Motivación	Conocer sus resultados en una actividad.
Intención	Visualizar sus respuestas a las preguntas, así como su recorrido seguido sobre el mapa de orientación (**en adelante lo llamaremos *track***).
Acción	Seleccionar la tarea que ha realizado, y navegar entre los distintos elementos.
Resolución	Úrsula visualiza su *track* y se da cuenta de dónde puede mejorar.
Descripción	Úrsula acaba de terminar la actividad de orientación deportiva en la que estaba participando. Ella y sus compañeros están deseando saber cuáles han sido los resultados. Una vez que todo el mundo ha terminado, Úrsula comprueba la clasificación. Ha quedado entre las cinco primeras. Revisa su *track*. Si no fuera por el tiempo que perdió cuando se despistó y tuvo que solicitar ayuda con la ubicación, probablemente habría quedado primera o segunda.

Tabla 4.5: Escenario 4

Escenario 5	
Actor	Úrsula (alumna).
Motivación	Participar en la última actividad organizada por su profesor Alfredo.
Intención	Encontrar la actividad y unirse a ella.
Acción	Una vez encontrada la actividad, inscribirse.
Resolución	Úrsula queda inscrita en la actividad, tras demostrar que había sido invitada a ella por Alfredo.
Descripción	Úrsula tiene que inscribirse mediante la aplicación a una nueva actividad que ha programado su profesor Alfredo. Alfredo les ha proporcionado unas claves para identificar la actividad y unirse. Úrsula encuentra la actividad, proporciona las credenciales adecuadas y queda inscrita.

Tabla 4.6: Escenario 5

Escenario 6	
Actor	Alfredo (profesor).
Motivación	Gestionar los participantes de una de las actividades que ha organizado.
Intención	Averiguar quiénes se han apuntado a la actividad.
Acción	Encontrar la actividad y visualizar los participantes.
Resolución	Alfredo ve la lista de participantes y comprueba quiénes faltan.
Descripción	Alfredo ha programado una actividad de orientación educativa para sus estudiantes el próximo martes. Antes de esa fecha, quiere comprobar si todos ellos ya se han apuntado. Consulta los datos la actividad en la aplicación y se da cuenta de que su alumna María no está inscrita. Entonces se da cuenta de que probablemente María faltó a clase el día que facilitó las claves para unirse. En la hora siguiente de clase, la avisará para que así pueda incorporarse.

Tabla 4.7: Escenario 6

Escenario 7	
Actor	Alfredo (profesor).
Motivación	Comprobar qué tal les ha ido a sus estudiantes durante la actividad.
Intención	Ver datos, estadísticas y resultados sobre los estudiantes en las distintas pruebas.
Acción	Buscar la última actividad realizada y observar los resultados que la aplicación le muestra.
Resolución	Selecciona la actividad que estaba buscando y se da cuenta de algunos datos interesantes.
Descripción	Alfredo realizó hace unos días una actividad de orientación educativa con sus estudiantes. Ahora quiere hacerse una idea de qué tal les fue a éstos durante la actividad. Busca la más reciente de sus actividades realizadas y observa que su alumna María García fue la más rápida en encontrar todas las balizas. Además le llama la atención el hecho de que varios de sus estudiantes no alcanzaron la baliza junto al haya. También observa que la pregunta de la baliza del tejo, es la que más han fallado sus estudiantes. El próximo día en clase hablarán sobre estas balizas y sus preguntas.

Tabla 4.8: Escenario 7

Escenario 8	
Actor	Alfredo (profesor).
Motivación	Ver el comportamiento de un estudiante en concreto durante una actividad.
Intención	Visualizar los datos del estudiante.
Acción	Seleccionar la actividad, localizar al estudiante entre los participantes y ver sus datos.
Resolución	Alfredo consigue ver la actuación del estudiante durante su actividad.
Descripción	Alfredo realizó ayer una prueba de orientación educativa en el Campo Grande con sus estudiantes. Tras repasar los resultados generales de la prueba, observa que sólo uno de sus estudiantes no consiguió terminar la actividad en el tiempo establecido. Entonces, Alfredo abre los datos del estudiante durante la actividad. Comprueba que el estudiante respondió solo dos preguntas y que ambas respuestas fueron erróneas. A continuación, observa el *track* del estudiante y se da cuenta de que tras llegar a la segunda baliza, estuvo toda la actividad parado al lado del estanque. Alfredo saca la conclusión de que probablemente el estudiante se frustró al fallar las preguntas y perdió el interés en la actividad.

Tabla 4.9: Escenario 8

Escenario 9	
Actor	Alfredo (profesor).
Motivación	Programar una nueva actividad de orientación educativa.
Intención	Elegir el tipo de actividad que quiere realizar, cuándo y dónde.
Acción	Seleccionar el tipo de actividad y configurar algunos parámetros para dejarla programada.
Resolución	La actividad queda programada y Alfredo puede invitar a sus estudiantes a participar.
Descripción	Alfredo va a preparar para sus estudiantes una actividad de orientación educativa. Busca entre los tipos de actividades existentes las que sean en el Campo Grande de Valladolid. Alfredo elige una actividad de tipo "educativa naranja" de 10 balizas. A continuación, selecciona la fecha, la hora y la duración de la actividad. También si la actividad se desarrollará en modo *lineal* (alcanzar balizas con un cierto orden) o *score* (alcanzar balizas en cualquier orden) [7] y si estará permitido o no para los participantes solicitar ayuda con la ubicación. Finalmente, elige un título para hacerla más fácil de identificar. Tras confirmar los datos, la actividad queda programada y se generan un código que Alfredo facilitará a sus estudiantes para que puedan inscribirse en ella.

Tabla 4.10: Escenario 9

4.1.3. Conclusiones de la investigación sobre usuarios

Tras estas primeras labores de investigación sobre usuarios, y después de plasmar los resultados mediante la caracterización de *personas* y la identificación de escenarios, ya podemos formarnos una idea suficientemente clara de quiénes son los usuarios de la aplicación y cómo deben interactuar con ella para lograr sus objetivos. Como se ha mencionado anteriormente, hay dos grupos bien diferenciados: *docentes* y *estudiantes de primaria*. La *persona* de Alfredo encarna el primer grupo de usuarios, al cual hemos designado como *primario* ya que es en el que pondremos más énfasis a la hora de diseñar. Úrsula representa al segundo grupo de usuarios (*secundario*), el cual también se tiene presente en el diseño, pero con menor intensidad. El mayor punto de diferenciación entre ambos grupos es el de sus objetivos, ya que unos utilizan la aplicación para organizar y monitorizar actividades, y los otros para participar en ellas. En el estudio de *personas* también hemos visto que ambos grupos de usuarios son "no expertos", ya que carecen de conocimiento avanzado sobre computación o sistemas operativos. Además, según los propios usuarios

nos han manifestado durante las entrevistas realizadas, conseguir un coste de interacción[4] reducido es crítico para que la aplicación pueda satisfacer sus necesidades. Esto implica que para este proyecto ha sido necesario tener muy en cuenta la usabilidad, como veremos a lo largo del documento.

4.2. Requisitos

4.2.1. Requisitos funcionales

Los requisitos funcionales describen las funcionalidades que incorporará el sistema. A continuación se describen los requisitos funcionales identificados para este proyecto tras el análisis de literatura, las reuniones con los tutores, y la investigación sobre usuarios.:

- **RF-01** El sistema deberá permitir a los usuarios registrarse en la aplicación mediante e-mail y contraseña.
- **RF-02** El sistema deberá permitir a los usuarios programar actividades de orientación, escogiendo de entre un catálogo de actividades existentes.
- **RF-03** Al programar una nueva actividad, el sistema permitirá al usuario organizador de la misma elegir si la actividad será *score* [7] o lineal, y si estará permitido que los participantes soliciten a la aplicación ayuda con la ubicación[5].
- **RF-04** El sistema deberá permitir a los usuarios inscribirse como participantes de una actividad, mediante unas claves proporcionadas por el organizador de la actividad.
- **RF-05** El sistema deberá ser capaz de detectar el paso de los participantes de las actividades por las balizas.
- **RF-06** El sistema deberá ser capaz de determinar si una baliza tiene contenido educativo asociado, y en caso afirmativo, mostrárselo al usuario.
- **RF-07** El sistema deberá ser capaz de notificar a los participantes de una actividad su paso por una baliza.
- **RF-08** El sistema deberá permitir a los participantes de las actividades responder a los retos planteados en las balizas de una actividad.
- **RF-09** El sistema deberá ser capaz de mostrar un *feedback* positivo al usuario en caso de que su respuesta a un reto sea correcta, o una corrección en caso de que la respuesta sea incorrecta.
- **RF-10** El sistema deberá ser capaz de comprobar si un usuario se encuentra en el lugar de salida a la hora de poder iniciar su participación en una actividad, así como de detectar cuándo éste llega a la meta.
- **RF-11** El sistema deberá mostrar a los participantes de las actividades un cronómetro que indique el tiempo que llevan participando.
- **RF-12** Una vez acabada la actividad, el sistema deberá permitir a los participantes revisar sus respuestas dadas a los retos de las balizas, así como su *track* (recorrido seguido indicado sobre el mapa de orientación). El organizador de la actividad también podrá ver estos datos de cada participante.
- **RF-13** Habiendo conexión a internet, el sistema deberá permitir al organizador de una actividad observar en tiempo real las balizas alcanzadas por cada participante, así cómo sus respuestas dadas a los retos asociados a las mismas.
- **RF-14** El sistema deberá permitir al organizador de una actividad conocer los participantes inscritos en la misma.
- **RF-15** El sistema deberá permitir a los participantes de una actividad visualizar el mapa de orientación de esa actividad y manipularlo (*zoom-in, zoom-out* y rotar).

[4]El coste de interacción es la suma de esfuerzos que el usuario debe realizar al interactuar con una aplicación para alcanzar sus objetivos [5]. Algunos ejemplos de esfuerzos que los usuarios tienen que realizar y que debemos minimizar son: leer, *clickar*, tocar, escribir, hacer *scroll*, cambiar el foco te atención, memorizar o esperar, entre otros.

[5]La orientación clásica no contempla que los participantes puedan utilizar dispositivos con los que conocer su ubicación actual (excluyendo el mapa y la brújula). Sin embargo, en las actividades de orientación educativa que se realizarán mediante esta aplicación, el organizador de cada actividad decide si los participantes pueden ver ubicación actual representada sobre el mapa

- **RF-16** El sistema deberá ser capaz de comprobar que un usuario ha comenzado su participación en una actividad como requisito para permitirle ver el mapa de orientación.
- **RF-17** Una vez que una actividad haya terminado, el sistema deberá seguir permitiendo a los participantes ver el mapa de orientación si así lo desean.
- **RF-18** El sistema deberá permitir a un usuario eliminar su perfil.
- **RF-19** El sistema deberá permitir a un usuario editar sus datos de perfil (nombre, apellidos y foto).
- **RF-20** El sistema deberá ser capaz de mostrar a los usuarios una descripción y las normas de aquellas actividades en las que participan.
- **RF-21** El sistema deberá mostrar a los participantes de una actividad un resumen de su participación, en el que se incluyan la hora de inicio, la hora de finalización, el tiempo total y las balizas alcanzadas.
- **RF-22** Mientras la aplicación esté realizando el seguimiento de la ubicación de un participante en una actividad, el sistema deberá ser capaz de mostrar una notificación que permita al usuario saber que la actividad está en curso y se está monitorizando su ubicación.
- **RF-23** Si un participante abandona una actividad en curso (por ejemplo porque cierra sin querer la aplicación), el sistema deberá permitirle retomarla siempre y cuando ésta no haya terminado.
- **RF-24** El sistema deberá ser capaz de continuar con la monitorización de la ubicación del participante de una actividad aunque éste se mueva por las distintas pantallas de la aplicación, minimice la aplicación o bloquee el dispositivo.
- **RF-25** El sistema deberá permitir a un participante de una actividad, durante el transcurso de la misma, conocer cuántas balizas ya ha alcanzado y cuántas le quedan por alcanzar.
- **RF-26** El sistema deberá permitir a los usuarios restablecer su contraseña en caso de que la olviden.
- **RF-27** El sistema deberá ser capaz de verificar que el correo de los usuarios es real.
- **RF-28** El sistema deberá permitir a los organizadores de una actividad eliminar dicha actividad, de tal forma que la información deje de estar disponible también para los participantes de la misma.
- **RF-29** El sistema deberá permitir usar un mapa de orientación para la actividad, sobre el que se indicará la ubicación de las balizas, y la posición del usuario si se solicita y está permitido por el organizador.
- **RF-30** El sistema deberá contar un un catálogo de plantillas de actividad (4.4.1) a partir de las cuales el organizador pueda programar una actividad de ese tipo.

4.2.1.1. Requisitos de información

Los requisitos de información recopilan todos los datos con los que trabaja la aplicación y que soportan información. A continuación se describen los requisitos de información identificados para este proyecto:

- **RI-01** El sistema almacenará la siguiente información sobre los usuarios de la aplicación:
  - Nombre
  - Apellido/s
  - Correo electrónico
  - Contraseña
  - Imagen de usuario
- **RI-02** Sobre la participación de un usuario en una actividad, el sistema almacenará la siguiente información:
  - Hora de comienzo
  - Hora de finalización
  - Balizas alcanzadas
  - Hora de llegada a cada baliza

  - Respuesta proporcionada al reto asociado a cada baliza
  - Historial de ubicaciones durante la actividad (*track*)

- **RI-03** El sistema almacenará la siguiente información sobre las actividades:
  - Título de la actividad
  - Identificador para que los usuarios puedan encontrarla
  - Clave para que los usuarios puedan unirse a ella
  - Hora de comienzo
  - Hora de fin
  - Usuario organizador
  - Participantes
  - Plantilla a partir de la cual se ha programado
  - Si es *score* o lineal
  - Si permite solicitar ayuda con la ubicación o no
- **RI-04** El sistema almacenará la siguiente información sobre las plantillas a partir de las cuales se programan las actividades:
  - Nombre de la plantilla
  - Tipo
  - Mapa
  - Descripción
  - Normas
  - Balizas
- **RI-05** El sistema almacenará la siguiente información sobre las balizas:
  - Nombre
  - Coordenadas geográficas
  - Número relativo dentro de la actividad
  - Imagen

  En caso de que la baliza tenga asociada algún contenido educativo (reto) también almacenará la siguiente información:
  - Texto
  - Pregunta
  - Respuesta correcta
  - Posibles respuestas (solo si la pregunta es tipo test)

4.2.2. Requisitos no funcionales

- **RNF-01** La aplicación se comunicará con un sistema de *back-end*, de donde obtendrá y a donde enviará los datos de las actividades.
- **RNF-02** El sistema de *back-end* deberá permitir la integración con otra aplicación cliente en un futuro, que servirá para crear y configurar nuevas plantillas de actividades. Para ello, la información almacenada en su base de datos debe ser accesible a través de peticiones HTTP, mediante una REST API.
- **RNF-03** El sistema de *back-end* deberá implementar reglas de seguridad que permitan establecer qué usuarios pueden acceder a cada tipo de información en la base de datos.
- **RNF-04** El sistema deberá permitir ser generalizado a otros escenarios que no sean el Campo Grande de Valladolid (es donde se desarrollan todas las actividades diseñadas en este proyecto).

4.2.2.1. Requisitos de usabilidad

Como veremos a continuación, la especificación de los requisitos de usabilidad se ha realizado utilizando medidas cualitativas heurísticas establecidas de acuerdo con los usuarios, tales como "poco tiempo" o "la mayor parte de las veces". Estas medidas se concretarán a la hora de realizar las pruebas de usabilidad teniendo en cuenta de las características del prototipo prototipo o de la versión utilizada en cada una de ellas.

- **RNF-05** Un usuario no experto debe ser capaz de programar con éxito una actividad por sí mismo/a.
- **RNF-06** Un usuario que ya conoce la aplicación debe ser capaz de programar una actividad en poco tiempo.
- **RNF-07** Un usuario no experto debe ser capaz de encontrar una actividad e inscribirse en ella exitosamente por sí mismo/a.
- **RNF-08** Los usuarios deberán ser capaces de encontrar actividades e inscribirse en ellas en poco tiempo.
- **RNF-9** Los participantes de una actividad deben ser capaces de reconocer los elementos del entorno que actúan como baliza la mayor parte de las veces.
- **RNF-10** Los usuarios deben ser capaces de encontrar por sí mismos la pantalla de la aplicación que les permite dar respuesta a los retos de las balizas durante las actividades.

4.3. Casos de uso

Los casos de uso representan el conjunto de interacciones que un usuario puede realizar con el sistema. Como hemos visto, esta aplicación permitirá tanto organizar actividades como participar en ellas. Sin embargo, los usuarios no se registran como "organizadores" o "participantes", sino que hay un único tipo de usuario. Es decir, que en unas ocasiones un usuario asumirá el rol de "organizador" y en otras el de "participante". Esto es así para facilitar una eventual generalización de la aplicación a otros contextos más allá de las aulas y el ámbito puramente escolar.

Por motivos de claridad, los casos de uso aparecen divididos en tres Figuras (4.7, 4.8, 4.9). En la primera de ellas, aparecen los casos de uso que son independientes del rol que el usuario tenga en cada momento. La segunda muestra los casos de uso relativos a un usuario en el rol de "organizador". Por último, en la Figura 4.9 aparecen los casos de uso pertenecientes a un usuario en el rol de "participante".

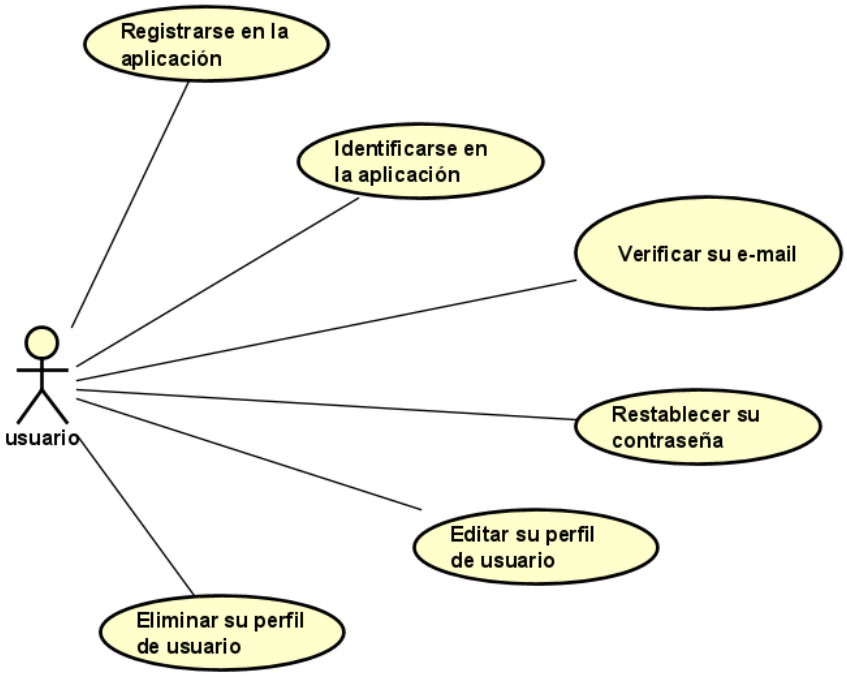

Figura 4.7: Casos de uso independientes del rol del usuario.

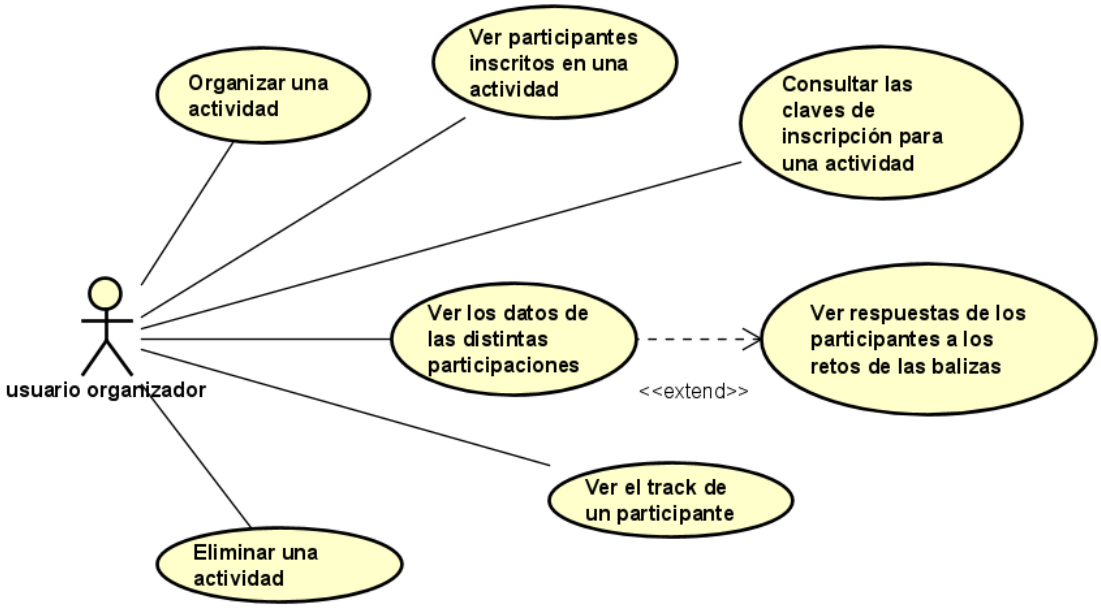

Figura 4.8: Casos de uso pertenecientes a un usuario en el rol de "organizador".

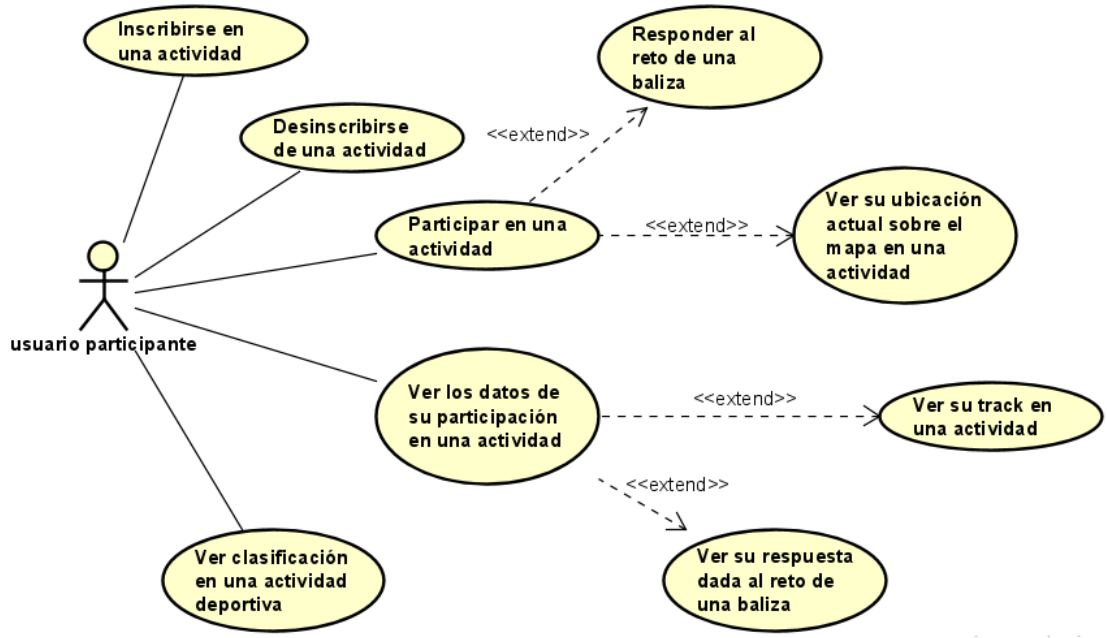

Figura 4.9: Casos de uso pertenecientes a un usuario en el rol de "participante".

4.4. Modelo del dominio

En esta sección se describe el modelo del dominio de la aplicación. En él se representan las entidades conceptuales utilizadas y las relaciones existentes entre ellas. La Figura 4.10 es un diagrama de clases en el que se representa el modelo del dominio de este proyecto.

4.4.1. Relación de clases

En este apartado se describen brevemente las clases representadas en el modelo del dominio de la Figura 4.10, con el objetivo de establecer una correspondencia entre dichas clases y las entidades del mundo "real" que

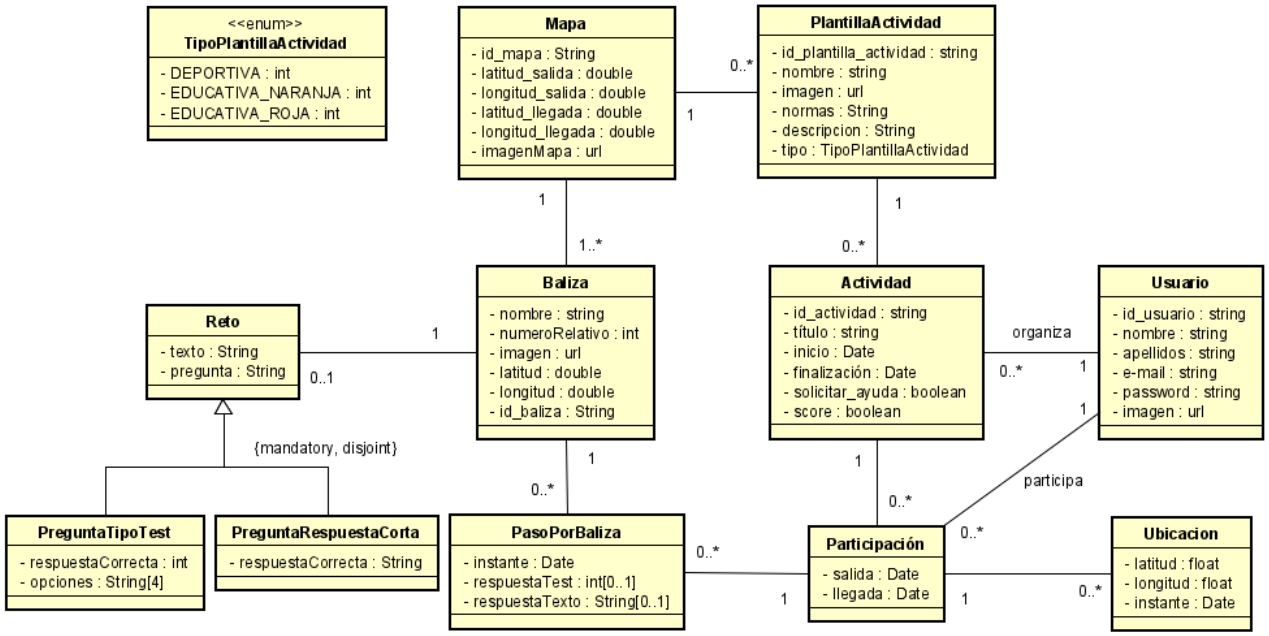

Figura 4.10: Modelo del dominio.

conceptualizan:

- **PlantillaActividad**. Los organizadores de las actividades no crean dichas actividades desde cero mediante la aplicación, sino que seleccionan un tipo de actividad ya existente de un "catálogo". Lo que el usuario selecciona es una *PlantillaActividad*, a la que nos referimos en adelante como (*plantilla*). Ésta encapsula la información general de la actividad, es decir, datos como dónde se desarrolla, qué tipo de actividad es, sus normas o su descripción. Además, las plantillas están asociadas a un *mapa*. Por último, las *plantillas* son "reutilizables", y se pueden programar muchas actividades a partir de cada una de ellas.

- **Actividad**. Las *actividades* son "instancias" de plantillas. Están asociadas a un usuario como *organizador*. Éste, al crearlas, selecciona una fecha de comienzo y una fecha de fin. También tiene la opción de decidir si la actividad se realizará según la modalidad de *score* o si estará permitido que los participantes soliciten asistencia con la ubicación. Como vemos, las actividades también tienen asociada una lista de *participantes*.

- **Mapa**. Los *mapas* contienen la información sobre el recorrido que los participantes tienen que realizar durante la actividad. A cada mapa le corresponde una imagen geoposicionada en la que aparece representado dicho recorrido. Los mapas también contienen la información sobre la localización de la salida y la meta de la actividad. Además, los mapas tienen asociado un conjunto de balizas.

- **Baliza**. La clase *baliza* del modelo del dominio contiene toda la información relativa a las balizas (puntos por los que tienen que pasar los participantes) de la actividad, es decir, datos como su ubicación y el número relativo de la baliza dentro del recorrido de la actividad. Las balizas están asociadas a un mapa, y pueden estar asociadas o no a un *reto*, dependiendo del tipo de la plantilla.

- **Reto**. Un *reto* está asociado a una baliza, y representa el contenido educativo de ésta. Las balizas de las actividades creadas a partir de una plantilla de tipo ***DEPORTIVA*** no tienen asociado ningún reto, ya que los participantes se limitan a pasar por ellas. En cambio, si la plantilla de la actividad es de tipo ***NARANJA***, cada baliza tendrá asociado un reto de la subclase *PreguntaRespuestaCorta*. Si la plantilla es ***ROJA***, entonces a cada baliza le corresponderá un reto de la subclase *PreguntaTipoTest*. Como puede observarse en la Figura 4.10, la relación de ambas subclases con la superclase *Reto* es de tipo *mandatory* y *disjoint*. Esto quiere decir que cualquier instancia de un reto, tiene que pertenecer obligatoriamente a una y solo una de las subclases. El alcance de este proyecto no contemplaba la implementación de más subclases descendientes de *Reto*, sin embargo, esto sería de valor para los usuarios, ya que les gustaría poder realizar más tipos de actividades educativas. Por lo tanto, esto podría ser una futura mejora interesante para proyectos futuros.

- **PreguntaTipoTest**. Esta clase representa el tipo de reto correspondiente a las balizas de las actividades programadas a partir de una plantilla *ROJA*. Contienen las posibles respuestas que se mostrarán al usuario, así como cuál es la respuesta correcta (para poder corregir la elección del usuario y/o proporcionarle un *feedback*).
- **PreguntaRespuestaCorta**. Representa el tipo de reto correspondiente a las balizas de las actividades programadas a partir de una plantilla *NARANJA*. Contienen la información sobre cuál es la respuesta correcta.

- **Usuario** Esta clase representa la información que la aplicación guarda sobre los usuarios. Como vemos en la Figura 4.10, en la clase Usuario aparecen los datos de perfil. Además, vemos que cada usuario puede estar asociado a muchas actividades, a veces como organizador y otras como participante (en este caso el usuario se relaciona con la clase intermedia *participación*).

- **Participación** Una *participación* contiene la información relativa a la actuación de un usuario durante una actividad en la que participa. En ella se anotarán la hora de salida y la hora de finalización. Además, una participación estará asociada a un conjunto de *ubicaciones* y de *pasos por balizas*.

- **PasoPorBaliza**. El *paso por una baliza* representa una baliza alcanzada por un usuario durante una actividad. En él se anotará la hora a la que se produjo ese paso, así como la respuesta dada por el usuario al reto asociado la baliza (en caso de que hubiera algún reto).

- **Ubicación** Durante el transcurso de las actividades, la aplicación realiza un seguimiento de la ubicación de los participantes. De este modo, se forma un registro de ubicaciones gracias al cual se puede reconstruir el *track* del usuario durante la actividad. Una ubicación consta simplemente de un instante de tiempo y unas coordenadas.

4.5. Modelo de interacción

En esta sección se describen los principales casos de uso de la aplicación mediante la utilización de diagramas de secuencia.

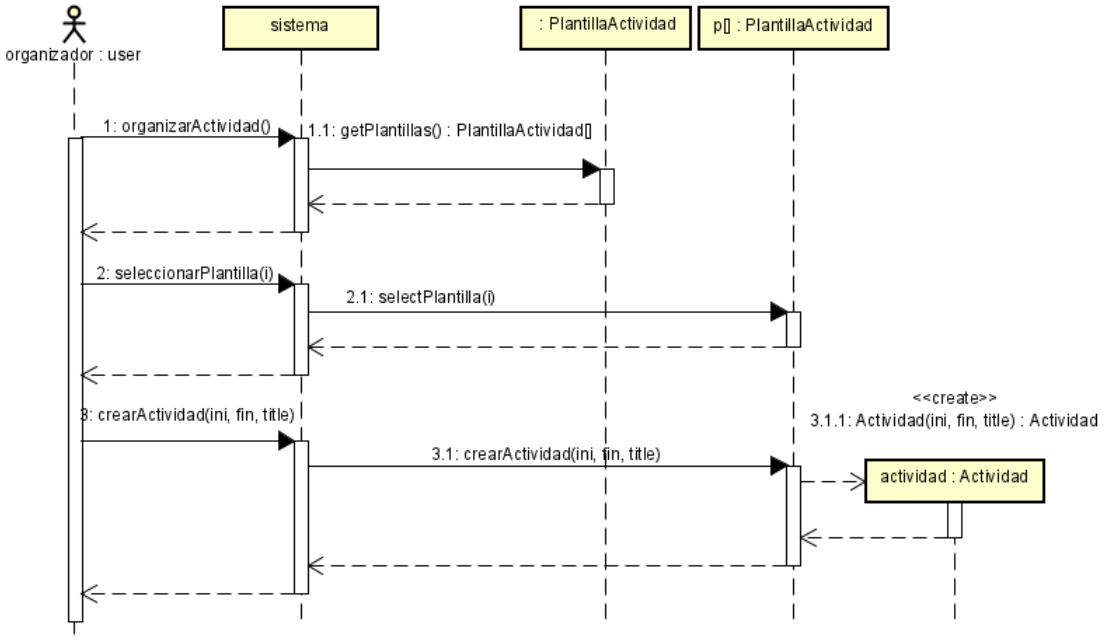

Figura 4.11: Diagrama de secuencia del CU "Organizar una actividad".

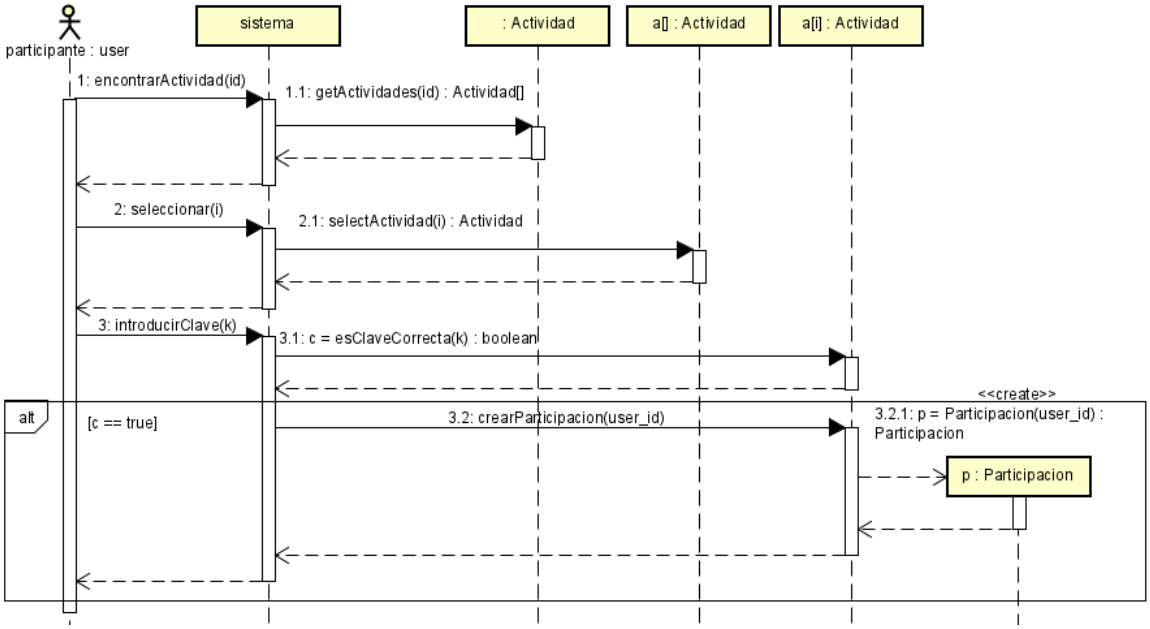

Figura 4.12: Diagrama de secuencia del CU "Inscribirse en una actividad".

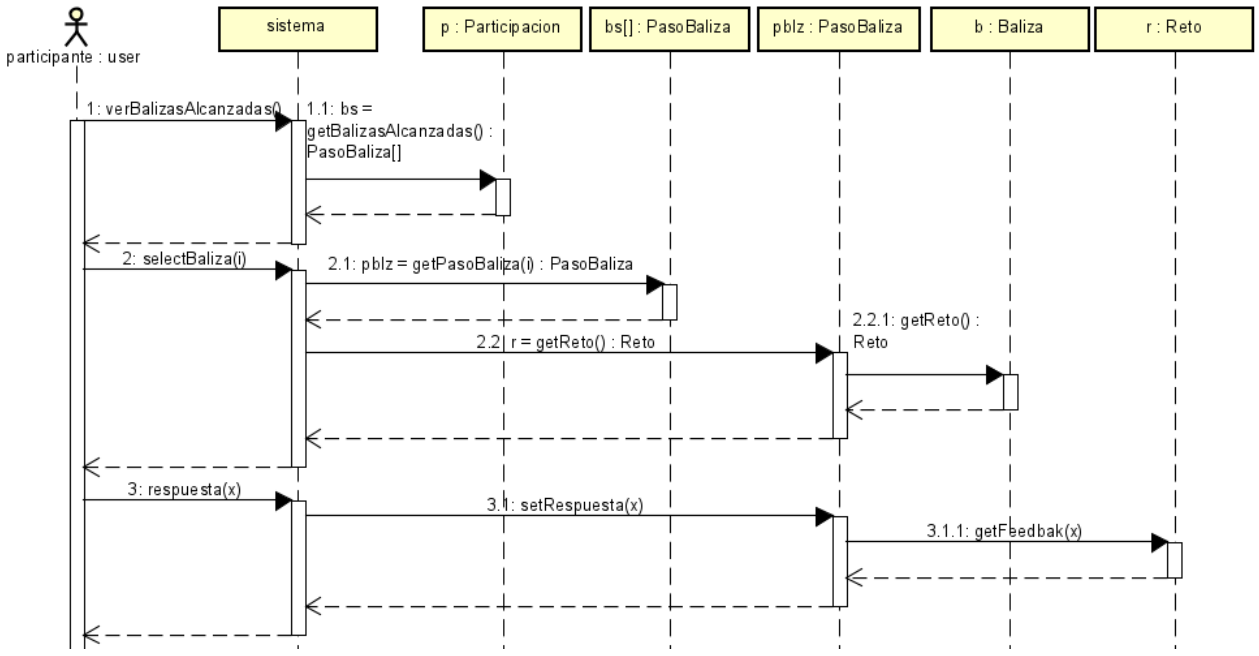

Figura 4.13: Diagrama de secuencia del CU "Responder al reto de una baliza".

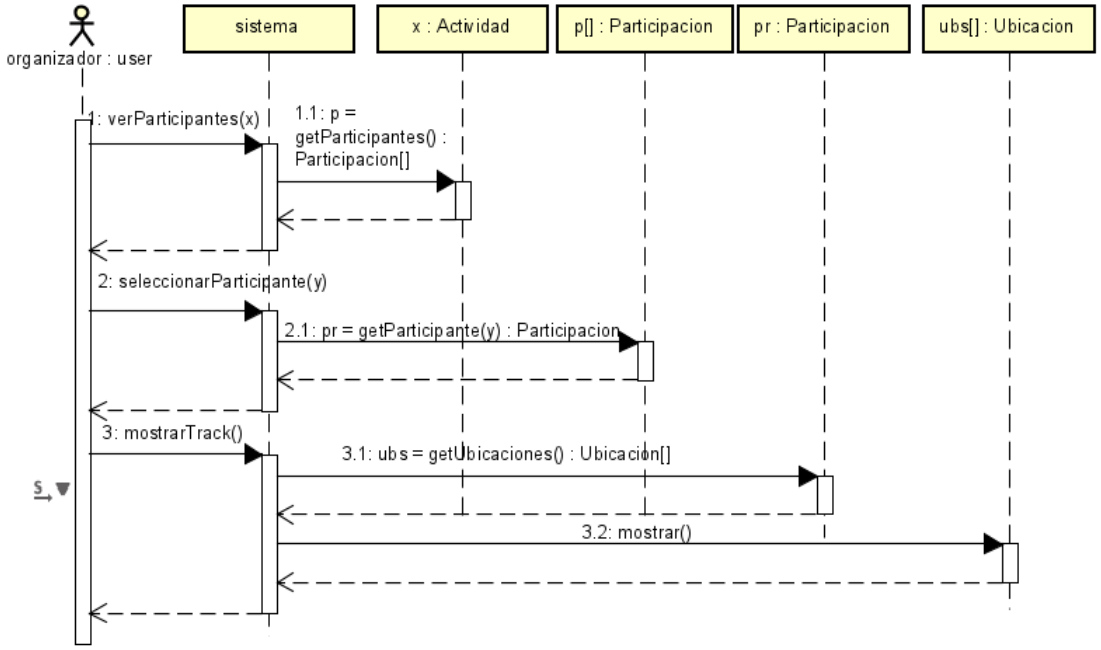

Figura 4.14: Diagrama de secuencia del CU "Ver el *track* de un participante".

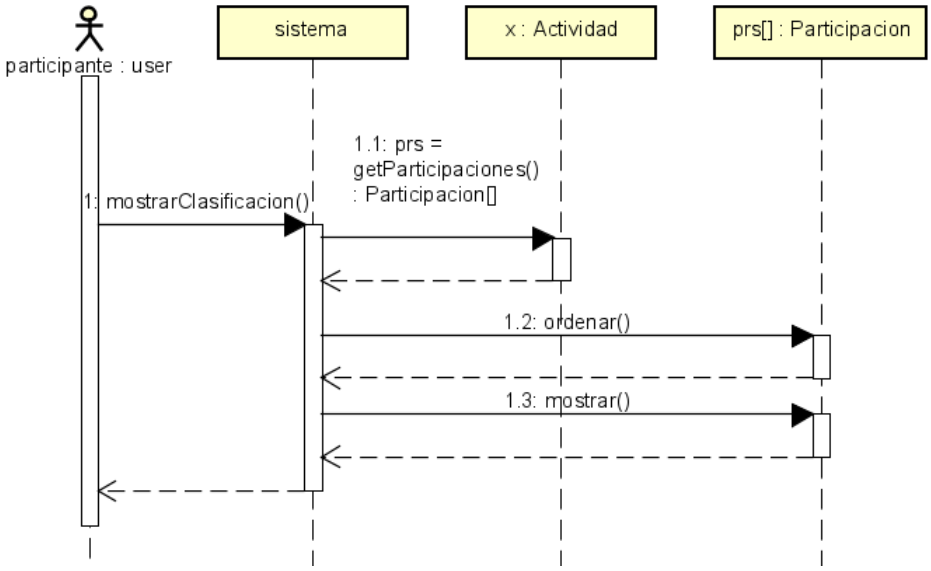

Figura 4.15: Diagrama de secuencia del CU "Ver clasificación de una actividad".

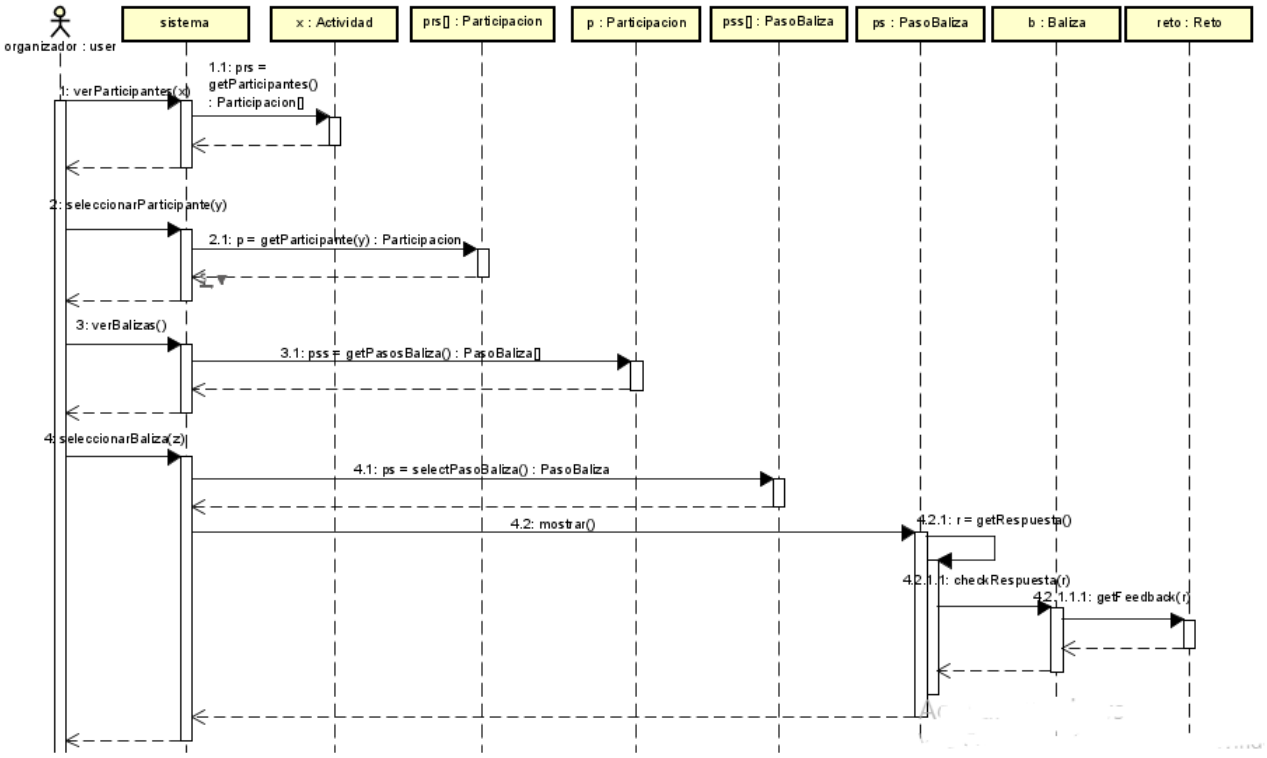

Figura 4.16: Diagrama de secuencia del CU “Ver respuestas de los participantes a los retos”.

Capítulo 5

Diseño

Siguiendo la metodología descrita en el capítulo 2, el diseño ha seguido un modelo iterativo, con pruebas de usabilidad que involucran a participantes en los momentos críticos [28]. Además, se han tenido en cuenta principios generales de usabilidad, y se han considerado guías proporcionadas por las plataformas para facilitarla. En este capítulo veremos las decisiones de diseño tomadas basándonos en todo lo anterior, así como la arquitectura del sistema y el diseño de la base de datos.

5.1. Guías de diseño aplicadas

En esta sección se describen las guías generales de diseño en las que nos hemos basado en este proyecto. Veremos que, por un lado, están basadas en las heurísticas de Nielsen [21], muy conocidas en el mundo del diseño de sistemas interactivos. Por otro lado, también se han seguido las guías de Material Design[1]. Finalmente, en esta sección veremos cómo se han combinado las guías de Material con las heurísticas de Nielsen, de tal forma que la aplicación de las primeras ha servido como medio para ajustarse a las segundas.

5.1.1. Descripción de las heurísticas de Nielsen

Con el objetivo de alcanzar buenos niveles de usabilidad, a la hora de realizar el diseño nos hemos basado en la lista de Los 10 Principios Generales para el Diseño de la Interacción, de Jakob Nielsen [21], los cuales se describen a continuación:

1. **Visibilidad del estado del sistema.** El sistema debe comunicar a los usuarios su estado actual. Esto les permite conocer el resultado y las consecuencias de sus acciones, de tal modo que la interacción resulta predecible.

2. **Elementos del sistema relacionados con elementos del mundo real.** El diseño debe hablar "el lenguaje del usuario", y no una jerga interna. Se recomienda seguir convenciones y hacer que la información aparezca en un orden lógico y natural.

3. **Los usuarios tienen el control y libertad sobre la interacción.** Es especialmente importante que sea posible deshacer acciones realizadas por error sin que el usuario quede atascado en la interacción.

4. **Consistencia y estándares.** Se debe mantener la consistencia interna (dentro de un mismo producto o familia de productos) y externa (convenciones propias de la industria). Esto facilita el aprendizaje de los usuarios, ya que los elementos resultan familiares y la interacción predecible.

5. **Prevención de errores.** En aquellas situaciones propensas a errores, se recomienda anticiparse a ellos para evitar que ocurran. Esto puede agilizar notablemente la interacción.

6. **Reconocer en lugar de recordar.** Se debe reducir al mínimo la cantidad de información que el usuario tiene que recordar. Para ello se recomienda mantener los elementos de la interfaz y las acciones visibles siempre que se pueda.

[1] Material Design: https://material.io/

7. **Flexibilidad y eficiencia de uso.** Se recomienda incluir "atajos" para las acciones habituales. Además, conviene permitir a los usuarios configurar o personalizar dichos "atajos".

8. **Estética y diseño minimalista.** Las interfaces no deben incluir contenido irrelevante, ya que puede disminuir la visibilidad relativa de los elementos que sí son importantes.

9. **Ayudar a los usuarios a reconocer los errores, entender su causa y recuperarse de ellos.** Se deben mostrar mensajes de error claros y entendibles. Además se recomienda acompañarlos de alguna sugerencia sobre cómo solucionarlos.

10. **Ayuda y documentación.** Aunque la mejor situación es aquella en la que el usuario no necesita ninguna explicación adicional para completar sus tareas, se recomienda incluir documentación que permita entender cómo realizarlas en caso de que fuera necesario.

5.1.2. Material Design

Material Design es un sistema de diseño creado por Google para ayudar a equipos a construir experiencias de usuario de alta calidad en Android, IOS, Flutter y web. Pone a nuestra disposición una serie de componentes[2] con los que construir nuestra interfaz de usuario. Además, también ofrece una serie *guidelines* con recomendaciones para que podamos sacar el máximo partido a esos componentes, y para que usemos los temas, los colores, la tipografía y otros elementos, de manera efectiva. Basar el diseño de una aplicación en Material Design puede aumentar significativamente las probabilidades de obtener un producto con buenos niveles de usabilidad, ya que facilita cumplir con los principios enumerados en la lista de Nielsen. En cierto modo, Material nos evita tener que "reinventar" todo cada vez para ajustarnos a las recomendaciones anteriores. A continuación se muestran algunas capturas de pantalla de de la aplicación en las que se puede observar cómo Material Design nos ha ayudado a la hora de seguir los principios de la lista de Nielsen. El primer ejemplo lo encontramos en la Figura 5.1, que muestra la pantalla que ve el usuario al abrir la aplicación. La interfaz de esa pantalla se ha construido empleando los siguientes componentes y *guidelines* de Material:

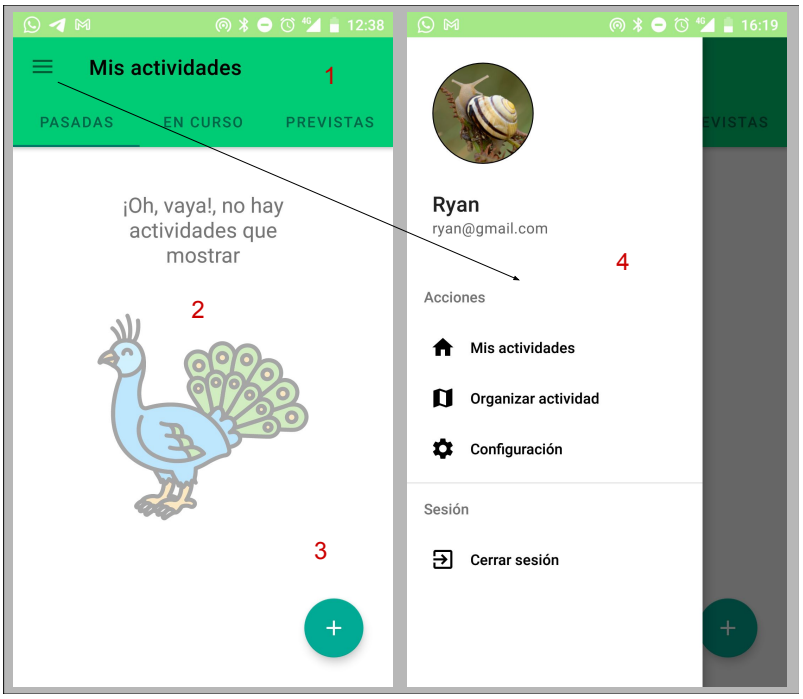

Figura 5.1: Ejemplo de utilización de componentes y *guidelines* de Material en el diseño de la aplicación. 1: *App Bar* 2: *Empty State* (icono hecho por Flat Icons y obtenido de www.flaticon.com) 3: *Floating Action Button* 4: *Navigation Drawer*.

[2]Un componente Material hace referencia a un bloque interactivo para la construcción de interfaces de usuario

1. *App Bar*. El *App Bar* tiene el color primario del tema de la aplicación (verde claro), como recomienda Material. Sobre él se indica en letras negras en qué pantalla nos encontramos. Las tres etiquetas en la parte inferior del *App Bar* son un *Tab*. Se recomienda usar este componente para alternar entre contenido similar (en este caso actividades clasificadas según su estado temporal). Por último, en la esquina superior izquierda del *App Bar* vemos un icono de menú. Como puede observarse, estamos aplicando el principio 4: *consistencia y estándares*.

2. *Empty State*. Los *Empty States* son útiles a la hora de comunicar que no hay elementos para mostrar en una pantalla (aplicación del principio 1: *hacer visible el estado del sistema*). En estos casos, Material recomienda utilizar una imagen con un tono neutro o humorístico, y un mensaje claro que explique al usuario lo que ocurre. El aspecto de ambos no debe ser "accionable", sino que se debe apreciar que su finalidad es informativa (principio 3: *prevención de errores*).

3. *Floating Action Button*. Se recomienda usarlo para iniciar la acción principal de esa pantalla (en este caso, unirse a una nueva actividad). Para el relleno del botón se recomienda emplear el color secundario del tema elegido, es decir, verde oscuro. También se puede apreciar que sobre dicho botón, a diferencia de sobre otros elementos, el icono es blanco. Esto es así porque el color negro no alcanza los criterios de legibilidad sobre un tono tan oscuro.

4. *Navigation Drawer*. Al pulsar el icono de menú del *App Bar* aparece este componente. Material recomienda utilizarlo para acceder a partes de la aplicación que guarden poca relación entre sí. Tanto este componente como el anterior, constituyen otro ejemplo de aplicación del principio 4: *consistencia y estándares*.

Como puede apreciarse en las capturas de la Figura 5.1, también se aplica el principio 8: *estética y diseño minimalista*, ya que todos los elementos que constituyen la interfaz son relevantes y aportan algún valor a la interacción del usuario. A continuación, en la Figura 5.2 tenemos otro ejemplo de aplicación de los principios de Nielsen mediante la utilización de componentes y *guidelines* de Material. En la imagen se muestra la secuencia de pasos necesarios para programar una nueva actividad:

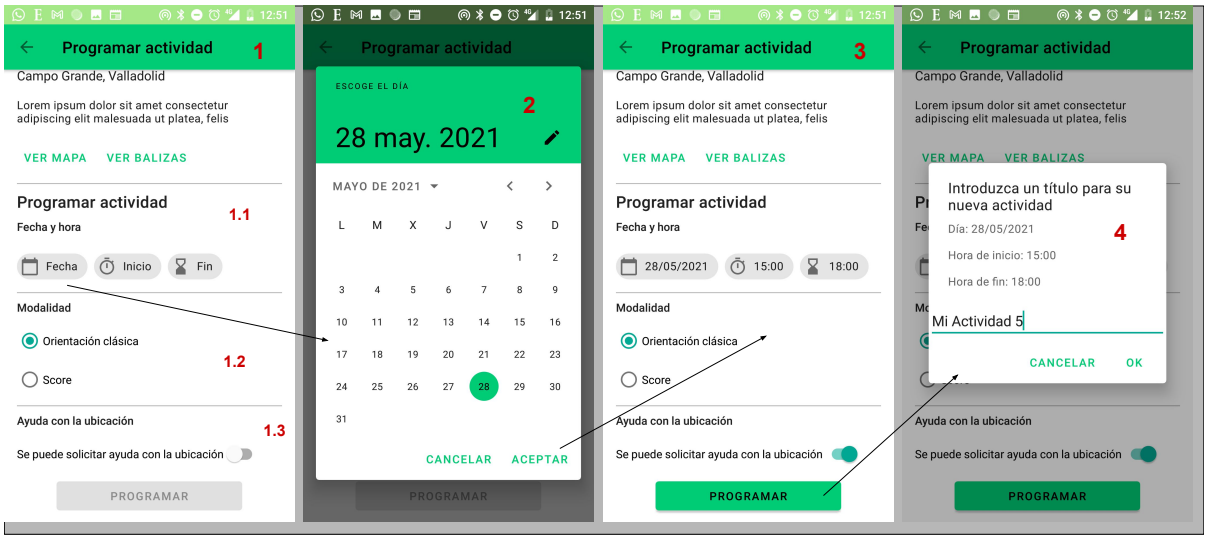

Figura 5.2: Secuencia de acciones necesarias para programar una actividad.

1. En la primera pantalla podemos apreciar la utilización de los componentes *Chips* (1.1), *Radio Buttons* (1.2) y *Switch* (1.3). Vemos que los *Chips* vienen acompañados de iconos que ayuda a los usuarios a comprender las acciones que van a desencadenar (principio 2: *elementos del sistema relacionados con elementos del mundo real*). La utilización del *Switch* para representar la activación o desactivación de un servicio también constituye una implementación del principio 2, al ser una clara metáfora de los interruptores del "mundo real".

2. En la segunda pantalla aparece el *Date Picker* de Material. Este componente está optimizado para que el usuario pueda elegir una fecha con el menor coste de interacción posible (principio 7: *Flexibilidad y eficiencia en el uso*). Además, también relaciona elementos del sistema con elementos del mundo real (principio 2), ya que utiliza la abstracción de un calendario, que es el objeto del mundo real con el que las personas elegimos fechas de manera natural.

3. La tercera pantalla es igual que la primera con la diferencia de que el usuario ya ha configurado la actividad. Como puede observarse, los *Chips* muestran el día y las horas elegidas (principio 6: *reconocer el lugar de recordar*), por lo que el usuario aún puede cerciorarse de que los parámetros son correctos antes de continuar.

4. En la última pantalla, para terminar el proceso, se pide al usuario que introduzca un título con el que identificar su actividad. En esta ocasión el componente de Material utilizado es un *Dialog*. Vemos que de nuevo se recuerda al usuario la fecha para la que ha programado su actividad (principio 6), y que también se le permite cancelar la acción en caso de que no esté de acuerdo con los datos introducidos en el paso anterior, tal y como sugiere el principio 3: *Los usuarios tienen el control y libertad sobre la interacción.*

En definitiva, en los ejemplos anteriores hemos visto cómo Material puede facilitarnos mucho la tarea de construir una interfaz consistente, con elementos bien diferenciados, y que proporcione una experiencia de uso agradable para el usuario. En la Tabla 5.1 se representa de manera resumida cómo se ha utilizado Material para seguir las heurísticas de Nielsen. En la siguiente subsección trataremos en mayor detalle sobre cómo ha sido el proceso de diseño.

Heurística	**Componentes, recomendaciones y guías Material**
Visibilidad del estado del sistema	**1.** Elección de colores de la interfaz con **Material Color Tool**. **2.** Seguimiento de las guías de tipografía de Material. **3.** Utilización de *Empty States*. **4.** *Progress Bar* para indicar que el sistema está ejecutando alguna acción.
Elementos del sistema relacionados con elementos del mundo real	**1.** Utilización de componentes Material que sirvan como "metáfora" de elementos del "mundo real", como los *Switches*.
Los usuarios tienen el control y libertad sobre la interacción	**1.** Utilización de *Material Dialogs* que permitan a los usuarios cancelar las interacciones.
Consistencia y estándares	**1.** Utilización de componentes Material (los usuarios están ya habituados a ellos, pues hay muchas otras aplicaciones que las utilizan). **2.** Seguimiento de las guías Material en cuanto al *theme* (tema) de una aplicación (indicaciones respecto al color, la forma y la jerarquía de los componentes). Esto ayuda a mejorar la consistencia interna del producto.
Prevención de errores	**1**. Utilización de *Material Input Layouts*, que permiten acompañar las áreas de texto con varios tipos de indicaciones y elementos (*helper text*, mensajes de error, iconos, contador de caracteres, etc). Gracias a esto se puede prevenir que el usuario introduzca información con contenido o formato incorrecto.
Reconocer en lugar de recordar	**1.** Utilización de *Material Dialogs* durante las interacciones que recuerden al usuario los datos clave de dichas interacciones. **2.** Los componentes, la tipografía y las reglas de jerarquía también ayudan a mantener la información clara y accesible en todo momento.
Flexibilidad y eficiencia de uso	**1.** Los componentes de Material tales como los *Date Picker*, los *Sliders* o los *Radio Buttons* están muy probados y optimizados de tal forma que maximizan la eficiencia de uso por parte de los usuarios.
Estética y diseño minimalista	**1.** Seguimiento de las guías de tipografía, forma y jerarquía de lo componentes. **2.** Utilización de *Dividers* para separar la información y presentarla de manera clara.
Ayudar a los usuarios a reconocer los errores	**1.** Utilización de componentes como los *Material Input Layouts* que permiten incorporar mensajes de error. **2.** Utilización de *Material Dialogs* y *SnackBars* para mostrar cada tipo de error de manera efectiva dentro de la interfaz de acuerdo a su importancia. **3.** Utilización de *Empty States*.
Ayuda y documentación	**1.** Utilización efectiva de iconos siguiendo las guías de Material de tal forma que los elementos resulten autodescriptivos para los usuarios.

Tabla 5.1: Aplicación de las heurísticas de Nielsen basada en Material Desgin

5.2. Proceso de diseño iterativo

Dentro del proceso de diseño iterativo propio de la metodología DCU, se considera muy recomendable probar la usabilidad de un producto antes de su lanzamiento. Esto permitirá detectar errores en el diseño en etapas tempranas, reduciendo el coste de corregirlos. A la hora de probar la usabilidad de un producto, la técnica más importante es el *test de usabilidad*. Los test de usabilidad, según vimos en la Figura 4.1, son un método de recogida de datos *cualitativo* y *del comportamiento*. Durante la fase de diseño de este proyecto se han realizado dos pruebas de usabilidad con usuarios, utilizando prototipos de la aplicación. En las siguientes subsecciones veremos en qué consistieron esas pruebas.

5.2.1. Primer test de usabilidad

Las primeras fases del diseño fueron dedicadas a crear la parte de la aplicación que permite tanto organizar nuevas actividades como inscribirse en ellas. La intención al realizar este primer test de usabilidad era poner a prueba ese diseño.

5.2.1.1. Prototipo de alta fidelidad

Para construir un prototipo de estas características, lo más habitual es o bien programarlo mediante código creando una versión de la aplicación con funcionalidad reducida, o bien diseñar un prototipo digital mediante alguna herramienta específica, como Adobe XD. En esta prueba se han utilizado ambas técnicas. La parte de la prueba relacionada con el registro de usuarios y la programación de las actividades se hizo utilizando una versión real de la aplicación, correspondiente al producto obtenido tras la segunda iteración de desarrollo (ver capítulo 2). Para el resto de tareas se diseñó un prototipo digital con Adobe XD. En la Figura 5.3 se muestran algunas pantallas de dicho prototipo.

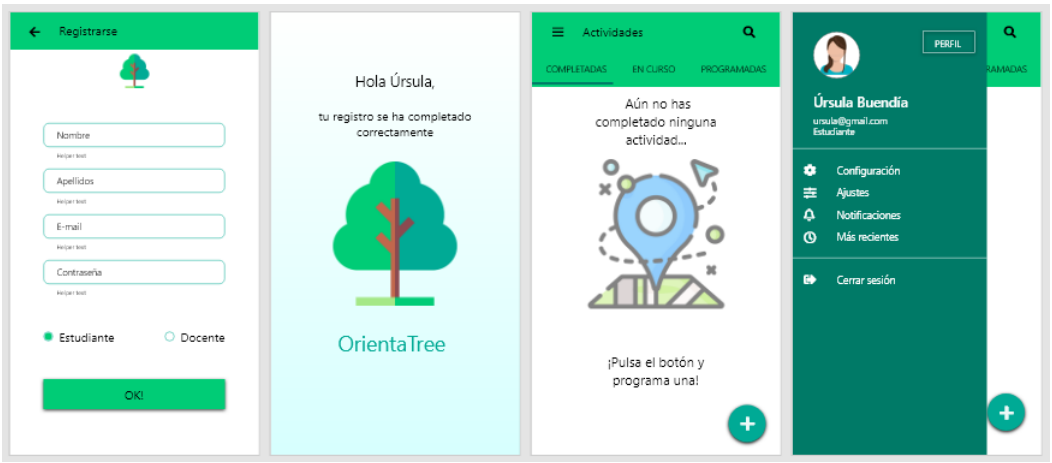

Figura 5.3: Varias pantallas del prototipo de alta fidelidad creado con Adobe XD.
(El icono principal de la aplicación está hecho por Freepik y ha sido obtenido de www.flaticon.com)

5.2.1.2. Planificación de la prueba

A la hora de planificar una prueba de usabilidad, no hay una estructura fija que sea necesario seguir. Cuando el número de personas que conforman un equipo de trabajo es reducido, puede ser útil seguir la estructura propuesta por David Travis [27]. Para este proyecto se ha tomado como referencia dicha estructura, obteniendo la siguiente planificación:

- **Objetivos**. Esta prueba de usabilidad tuvo como objetivos *comprobar que el coste de interacción de la aplicación era reducido* y que el manejo de la misma resultaba de *fácil y rápido aprendizaje para los usuarios*. Para medir ambos parámetros se utilizaron las siguientes métricas cuantitativas:
  - *Tasa de éxito*
  - *Tasa de error*

Como métricas cualitativas se consideraron las siguientes:

- *Apreciaciones subjetivas del/la participante*
- *Observaciones advertidas por el facilitador durante la prueba*

- **Tareas propuestas a los participantes**. Las tareas que se propuso a los usuarios completar fueron las siguientes:

  1. **Registrarse en la aplicación**
  2. **Organizar una actividad de tipo "educativa naranja"**[3]
  3. **Organizar una actividad de tipo "deportiva"**[3]
  4. **Ver participantes inscritos en una de mis actividades programadas**
  5. **Unirse como participante a una actividad**
  6. **Revisar las respuestas de una alumna durante una actividad terminada**
  7. **Dar la salida a un participante durante una actividad en curso**

  Más adelante veremos los resultados para cada una de las tareas propuestas.

- **Elección de participantes**. A la hora de seleccionar a los participantes de una prueba, es habitual buscar perfiles que se adapten a las *personas* creadas durante la investigación sobre usuarios en la fase de análisis. Para esta prueba, hemos tenido acceso a tres representantes de nuestro grupo primario de usuarios: un profesor y una profesora de la Facultad de Educación, y también un estudiante de esa misma Facultad.

- **Elección de una metodología**. En este caso, el test de usabilidad consistió en una **prueba de laboratorio**, es decir, que se realizó de manera presencial y con un *facilitador* que se encargaba de explicar a los participantes lo que tienen que hacer y de observar y tomar notas *in situ* sobre el desarrollo de la prueba.

5.2.1.3. Desarrollo de la prueba

La prueba se realizó en el laboratorio del grupo GSIC/EMIC. Los participantes se sometieron al test de uno en uno, asumiendo el autor de este documento el rol de *facilitador*. Las tareas anteriormente descritas fueron propuestas a los participantes en forma de escenarios, es decir, contextualizando la situación para que los participantes se sintieran cómodos y entendieran bien el objetivo de la prueba. Finalmente, tras completar la prueba con cada uno de los participantes se hizo una sesión de ***debriefing*** (*interrogatorio*), con todos ellos a la vez. Durante el *debriefing* se repasaron aquellos momentos de la prueba en los que los participantes tuvieron problemas para completar una tarea, así como aquellos que les resultaron confusos por algún motivo. Esto permitió obtener una visión de los posibles problemas desde su perspectiva. También se comentaron aquellos aspectos que les habían agradado especialmente, y se discutieron distintas propuestas para mejorar la aplicación.

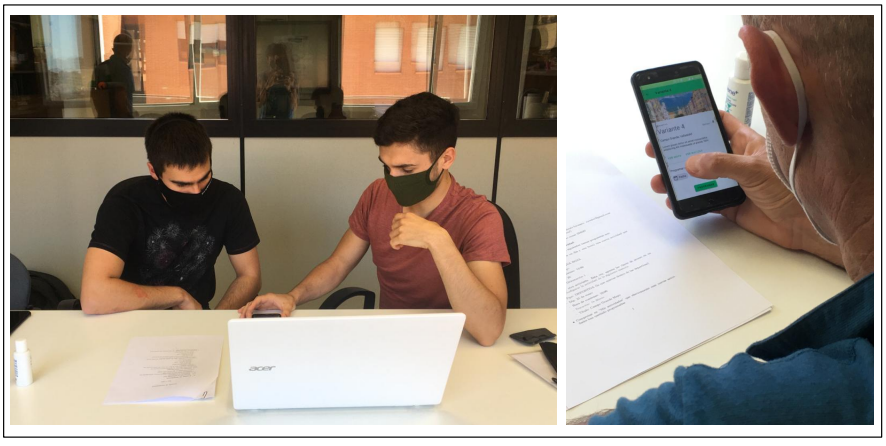

Figura 5.4: Desarrollo de la prueba de usabilidad.

[2]"educativa naranja" y "deportiva" son dos de los tipos de actividades que ofrece la aplicación (ver capítulo 4)

5.2.1.4. Resultados y conclusiones

Los resultados obtenidos tras la realización de la prueba de usabilidad aparecen resumidos en la Tabla 5.2.

Resltados del test de usabilidad				
Participante 1	**Participante 2**	**Participante 3**	**Tasa de éxito**	**Observaciones**
Tarea 1: registrarse en la aplicación				
completada	completada	completada	100 %	sin problemas
Tarea 2: programar actividad "educativa naranja"				
completada (con "pasos en falso")	completada parcialmente (aprox. 75 %)	completada (con "pasos en falso")	(aprox. 90 %)	**1.** Cierta confusión al elegir el tipo de actividad adecuado **2.** Dificultades con el *Time Picker* de Material
Tarea 3: programar actividad "deportiva"				
completada	completada	completada	100 %	**3.** Completada mucho más rápido que la tarea anterior debido al aprendizaje
Tarea 4: revisar participantes inscritos en una actividad programada				
completada	completada	completada	100 %	sin problemas
Tarea 5: apuntarse como participante a una actividad				
completada (con "pasos en falso")	completada (con "pasos en falso")	completada	100 %	**4.** Dificultad para encontrar la pantalla adecuada **5.** Elevado coste de interacción
Tarea 6: ver respuestas dadas por una alumna determinada				
completada	completada	completada	100 %	sin problemas
Tarea 7: dar la salida a un participante				
completada	ayuda para completar	completada	66 %	**6.** Confusión con las etiquetas de un *Tab*

Tabla 5.2: Resumen de los resultados de la prueba de usabilidad

Vemos que la tabla muestra para cada tarea propuesta el grado en que cada participante pudo completarla con éxito, junto con las observaciones advertidas por el *facilitador* durante la realización de la misma. A propósito de esas observaciones, en la Figura 5.5 aparecen ilustrados los problemas que dieron lugar a las **observaciones 1 y 2**. Todos los participantes tuvieron problemas a la hora de reconocer la actividad del tipo "educativa naranja", lo cual hizo evidente que la forma de representar el tipo de la actividad usando una tipografía naranja no era una buena idea. En la versión final, este error se corrigió haciendo explícito el tipo de la actividad. Respecto a la **observación 2**, al principio a los participantes les resultó un poco confuso elegir la hora mediante el *Time Picker* de Material. Sin embargo, la siguiente tarea puso de manifiesto que este problema sólo se producía la primera vez que los participantes intentaban programar una actividad (observación 3), por lo que finalmente se optó por mantener ese elemento de la interfaz sin cambios, ya que es probable que cualquier otra alternativa hubiera tenido el mismo problema. Durante el *debriefing* se repasaron los problemas que surgieron a la hora de encontrar la pantalla adecuada para realizar la tarea 5. Se concluyó que estos problemas también se debían a que era la primera vez que los usuarios tenían contacto con la aplicación, y que no había que darle mayor importancia. La **observación 5** vino motivada por el hecho de que las credenciales que hay que introducir para inscribirse en una nueva actividad eran demasiado largas (8 caracteres). En las siguientes versiones de la aplicación, esta longitud se redujo a 4 caracteres.

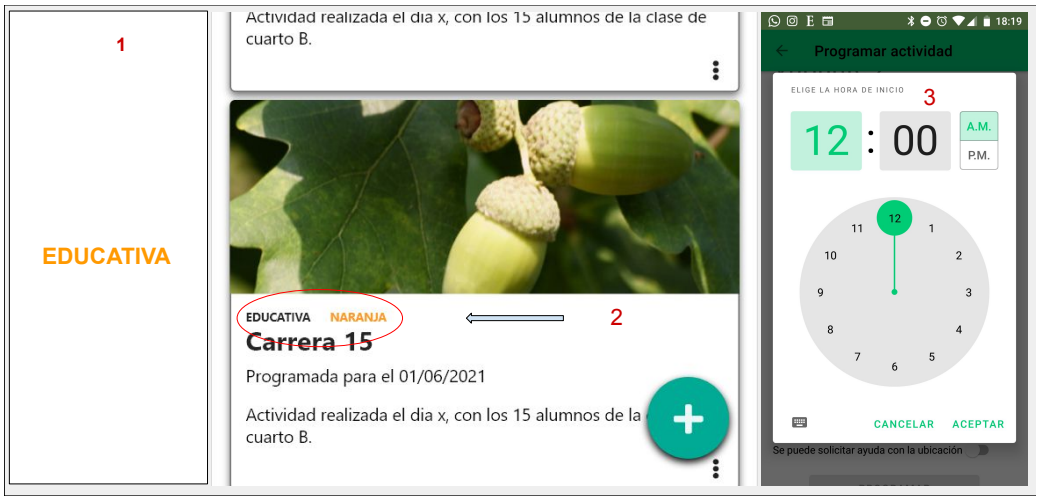

Figura 5.5: Elementos confusos de la interfaz.
1: Forma confusa de representar el tipo de la actividad. **2:** Corrección sobre la forma de mostrar el tipo de la actividad. **3:** *Time Picker* de Material.

Finalmente, en la tarea 7 es donde se detectó el problema más importante (**observación 6**). Como vemos, supuso que uno de los participantes no pudiera completar la tarea sin la intervención del *facilitador*. La Figura 5.6 muestra el problema. Los usuarios tenían dificultades a la hora de encontrar la pantalla desde la que dar la salida a los participantes de una actividad. La razón es que hay dos pantallas muy parecidas, con *Tabs* idénticos y con las mismas etiquetas, pero con comportamiento diferente. Esto es un "anti-patrón" que se debe evitar. En la versión final de la aplicación se combinan ambas pantallas, de tal forma que no hace falta utilizar el *Tab*.

Respecto a las impresiones manifestadas por los participantes al término de la prueba, todos ellos coincidieron en que les gustaba cómo estábamos planteando la interfaz de la aplicación. Les parecía fácil de usar y agradable estéticamente. Desde su punto de vista, el mayor problema que tenía la aplicación era la dificultad para unirse a una nueva actividad, para lo cual sugirieron la posibilidad de utilizar credenciales más cortas, como se ha mencionado anteriormente.

En conclusión, esta prueba sirvió para detectar algunos problemas con la interfaz de usuario en etapas tempranas. Gracias a ello fue bastante sencilla su reparación. También sirvió para confirmar que, en general, íbamos bien encaminados con el diseño de la interfaz. Por último (y aunque no era el objetivo inicial de la prueba), también se trataron algunos temas emergentes que surgieron. Lo más destacable es que se decidió que los usuarios de la aplicación no se registrarían con el rol de "estudiante" (participa en actividades) o "docente" (organiza actividades), sino que cualquier usuario podría ser participante en unas actividades y organizador en otras. Esto haría más flexible la aplicación y sería más fácil hacerla extensible a otros contextos fuera del ámbito educativo.

5.2.2. Segundo test de usabilidad

La prueba descrita en la sección anterior sirvió para probar, mediante un prototipo interactivo, la usabilidad de la parte del sistema que engloba las tareas de programación de actividades, inscripción y visualización de resultados de las mismas. Sin embargo, tras esa prueba, aún quedaba por revisar la usabilidad de la parte del sistema que da soporte a la realización y el transcurso de las actividades de orientación en sí. Para ello se realizó una segunda prueba, de la cual hablaremos en esta sección.

5.2.2.1. Prototipo de la aplicación

Para la realización de esta prueba se utilizó un prototipo *codificado*, es decir, una versión de la aplicación que cubría toda la funcionalidad básica para poder realizar el test, y que los participantes instalaron en sus dispositivos.

5.2.2.2. Planificación de la prueba

Dada la parte del sistema que se iba a probar en esta ocasión, la prueba había que realizarla *in situ* en el escenario real en el que tienen lugar las actividades, es decir, el Campo Grande de Valladolid. Al igual que en la

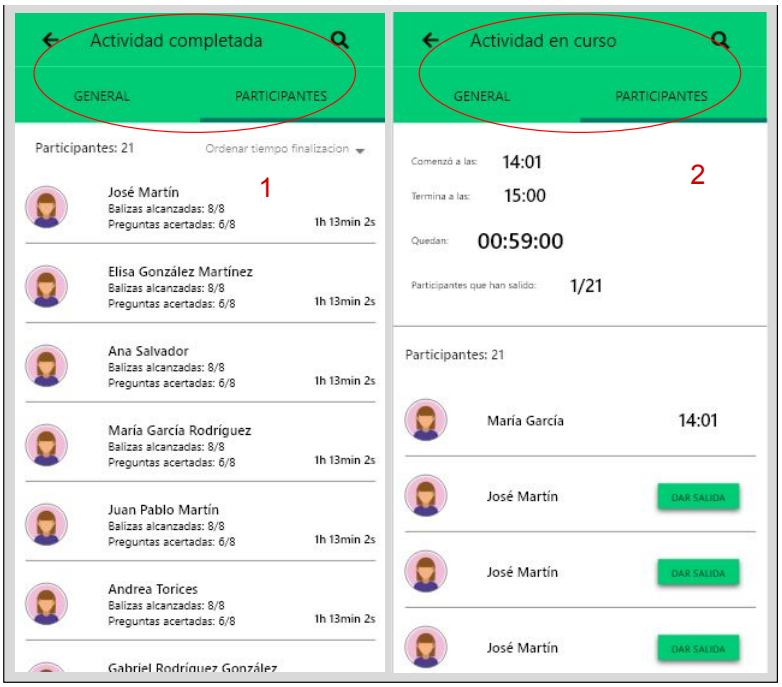

Figura 5.6: Etiquetas confusas.
1: En una actividad completada, la etiqueta "Participantes" mostraba la lista de participantes **2:** En una actividad en curso, la etiqueta mostraba información general sobre la actividad y la lista con los participantes para darles la salida.

primera prueba de usabilidad, para planificar ésta nos hemos basado en las recomendaciones de David Travis [27], obteniendo la siguiente estructura:

- **Objetivos**. La prueba fue concebida con un enfoque fundamentalmente *formativo*, es decir, la intención era recopilar datos que nos ayudasen a mejorar el diseño de la aplicación antes de su lanzamiento [29] (cap 3). No obstante, también había un componente *sumativo*, por el cual tratamos de medir el grado en que ciertos parámetros habían sido alcanzados. En este caso, el enfoque *sumativo* iba dirigido a medir cómo de bien funcionaba el aspecto de la geoposición, y si satisfacía o no las necesidades y expectativas de los usuarios. En esta prueba se utilizaron métricas de *rendimiento* y de *satisfacción* [29] (cap 3). Entre las primeras están:
  - **Distancia (metros)**. Distancia a la que la aplicación detecta el paso por una baliza.
  - **Retardo (segundos)**. Diferencia temporal con la que la aplicación notifica a todos los miembros de un grupo de participantes que han alcanzado una baliza, el paso por dicha baliza (idealmente debería ser 0 si llegan a la vez).
  - **Calibrado del mapa**. Precisión con la que la imagen del mapa de la actividad ha sido georreferenciada.

  Por su parte las métricas de *satisfacción* tienen que ver con lo que los usuarios piensan y sienten sobre su experiencia. Hablaremos sobre dichas valoraciones más adelante, al analizar los resultados y las conclusiones. Finalmente, esta prueba también tenía como objetivo identificar aspectos emergentes del diseño de la aplicación.
- **Tareas**. Para esta prueba de usabilidad se asignaron distintas tareas a los participantes en función del rol que éstos desempeñasen en cada momento (organizador/participante). La prueba se estructuró en tres actividades, en cada una de las cuáles, un usuario hacía de organizador/a y el resto de participantes. Las tareas del usuario en el rol de organizador/a fueron las siguientes:
  - Programar una actividad para la fecha y hora establecidas.
  - Monitorizar el comportamiento de los participantes durante el transcurso de la actividad.
  - Visualizar las respuestas dadas por los participantes, así como el *track* seguido por éstos durante la actividad.

Por su parte, las tareas asignadas a los usuarios en el rol de participante fueron:

- Realizar la actividad buscando las balizas y dando respuesta a los retos planteados en ellas.
- Utilizar de vez en cuando la funcionalidad que permite ver dónde uno se encuentra situado sobre el mapa, para probar qué tal calibrada está la imagen.
- Visualizar sus respuestas y su *track* al terminar la actividad.

- **Elección de participantes**. Los participantes de esta prueba fueron los mismos que en la prueba de usabilidad anterior, es decir, los dos profesores y el estudiante de la Facultad de Educación que se encuentran ligados a este proyecto. Además, debido a la complejidad técnica y de gestión de esta prueba (ya que se desarrollaba en un parque de grandes dimensiones), mis tutores de este trabajo de fin de grado me asistieron en el rol de observadores, a la vez que también hacían de participantes.

- **Elección de una metodología**. Esta prueba se realizó al aire libre, en el Campo Grande de Valladolid, pues esa era la única forma de poder probar una aplicación de estas características que permite realizar actividades físicas en el medio natural.

5.2.2.3. Desarrollo de la prueba

Durante la realización de esta prueba se completaron tres actividades de distinto tipo (deportiva, naranja y roja). En cada una de ellas, uno de los usuarios asumió el rol de organizador/a, y el resto hicieron de participantes. Mientras tanto, uno de los tutores de este trabajo hacía de observador del organizador y otro de observador de los participantes (además de ser participante también). Para facilitar la toma de datos por parte de los observadores, y para que todos pudiéramos compartir ideas, en lugar de completar cada participante la actividad por su lado (que sería lo normal), íbamos todos en grupo. En la Figura 5.7 se han recogido algunas fotos tomadas durante el transcurso de la prueba.

Figura 5.7: Varias fotos tomadas el día de la segunda prueba de usabilidad, en el Campo Grande.

5.2.2.4. Resultados y conclusiones

A partir de las observaciones realizadas durante la prueba, así como el *feedback* proporcionado por los usuarios, podemos concluir que, en general, los resultados de la prueba fueron positivos, y que los usuarios manifestaron estar satisfechos con la aplicación. A continuación se describen aquellos aspectos de la aplicación para los que la prueba arrojó unos mejores resultados:

- **Geoposicionamiento**. Uno de los objetivos de la prueba, era comprobar qué tal se comportaba la aplicación en cuanto a la precisión con la que es capaz de obtener la ubicación de los usuarios. Además, también había que comprobar si esa precisión era la adecuada de cara a la usabilidad, ya que tanto un exceso como un defecto en la misma podía ser problemático. Hay que tener en cuenta que en un entorno como el del Campo Grande, puede haber zonas por las que está prohibido caminar. A menudo, los árboles que representan una baliza están en medio de una de esas zonas. La aplicación no debería forzar al usuario a tener que acercarse tanto como para verse obligado a pisar una de esas zonas prohibidas. En ese sentido, la aplicación se comportó adecuadamente, ya que detectaba el paso por las balizas a una distancia conveniente (radio de 20 metros aproximadamente). Además, la ubicación se obtenía con una precisión similar para todos participantes, de tal forma que al llegar en grupo a una baliza, todos recibían la notificación dentro de un lapso de tiempo de unos 3 segundos. Los usuarios manifestaron estar muy satisfechos con el comportamiento de la aplicación en ese aspecto, por lo que no hubo que realizar más ajustes en cuanto a la precisión a la hora de geolocalizar.

- **Facilidad a la hora de reconocer las balizas**. En el capítulo 3 vimos que uno de los retos al comenzar el proyecto era hallar la forma de conseguir que los usuarios reconocieran el elemento del entorno que constituye una baliza cuando lo tuvieran delante. Por esa razón, en esta prueba también queríamos comprobar cuál era la tasa de éxito a la hora de reconocer las balizas por parte de los usuarios. En todos los casos, los participantes fueron capaces de reconocerlas gracias a las indicaciones del mapa de orientación y las fotografías que se muestran. Los usuarios se mostraron satisfechos con este aspecto de la aplicación, por lo que todo parece indicar que este método de asistencia al reconocimiento de las balizas es adecuado.

- **Manipulación del mapa de la actividad**. En las actividades de orientación tradicionales se utiliza un mapa de orientación impreso en papel. Los mapas en papel tienen muchas características deseables en cuanto a la facilidad y flexibilidad que ofrecen a la hora de ser manipulados. De igual modo, para esta aplicación necesitábamos que el mapa fuera una imagen que permitiera ser rotada, aumentada y alejada con la mayor facilidad posible. Además, tenía que ser una imagen georreferenciada, de tal modo que se pudiera representar la ubicación del usuario sobre ella. Para conseguir esto se utilizó la funcionalidad de *overlay* (capa superpuesta) ofrecida por el SDK de Google Maps para Android[4] (ver 6.5). Se observó que los usuarios manipulaban el mapa con facilidad, encontrando de forma natural la manera de hacer rotaciones y zoom.

- **Visualización del *track* realizado por los participantes**. Los los participantes de una actividad, al término de la misma, pueden ver sobre el mapa de orientación su trayectoria seguida (*track*). El organizador también puede ver esta información para cada participante. Esta funcionalidad se cubrió mostrando a los usuarios el mapa de la actividad y proporcionándoles un *Slider* mediante el que podían controlar el avance del *track* a lo largo del tiempo de la actividad (Figura 5.10). El *track* se mostraba como una línea curva que unía una serie de ubicaciones consecutivas conocidas. La funcionalidad gustó mucho a los usuarios, ya que según nos hicieron saber, ofrece amplias posibilidades en el ámbito educativo. También se mostraron satisfechos con la forma en que podían visualizarlo y manipularlo mediante la aplicación. Sin embargo, les gustaría poder verlo también representado en su totalidad como una imagen fija. Esta característica fue introducida más adelante (segunda imagen de la Figura 5.10).

A partir de la prueba, también se obtuvieron varios aspectos en los que la aplicación podría mejorar. Algunos de ellos se han podido corregir en el marco de este trabajo de fin de grado. Sin embargo, otros quedan recogidos y clasificados para trabajo futuro. A la hora de decidir cuáles de estos aspectos se corregirían y cuáles quedarían para trabajo futuro, se le asignó a cada uno una prioridad considerando la relevancia (o el valor) de la implementación y el tiempo que supondría realizarla, siguiendo el criterio que se muestra en la Figura 5.8. De acuerdo a ese criterio, en las Tablas 5.3, 5.4, 5.5, 5.6 y 5.7 aparecen priorizadas todos las tareas que surgieron a partir de la prueba.

[4]Sitio web de Google Maps para desarrolladores: https://developers.google.com/maps/documentation

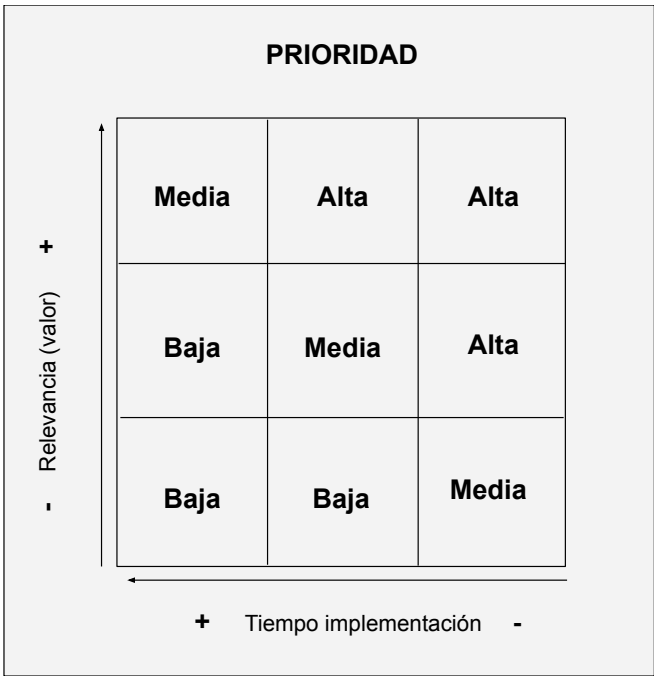

Figura 5.8: Criterio para asignar prioridad a las implementaciones.

Problemas de usabilidad detectados	**Prioridad**
Cuando un participante alcanza una baliza, no se le indica con la suficiente claridad que a continuación tiene que resolver el reto asociado a la baliza (Figura 5.11).	Alta
El organizador podría ver más claramente quién ha terminado la actividad y quién no, si esta información estuviera respaldada por un código de colores (ejemplo: participante que ha llegado a la meta aparece en color verde).	Media
Las notificaciones de paso por baliza deberían interrumpir lo que el usuario esté haciendo para asegurarse de que éste se da cuenta del evento.	Media
El término "en curso" para designar a las actividades que tienen lugar en el momento presente puede ser confuso, ya que parece implicar que el usuario está participando activamente en la actividad, cuando esto no tiene por qué ser así (por ejemplo, podría estar esperando a que le den la salida).	Baja
El término "organizar" para designar la acción por la que el organizador fija una actividad para un día y una hora puede ser confuso, pues parece implicar que el organizador puede crearla y configurar dicha actividad. Quizá sería más apropiado el término "programar", pero habría que estudiar que opción es mejor.	Baja
Al mostrar las balizas alcanzadas, éstas aparecen ordenadas de forma que la alcanzada más recientemente se muestra arriba, mientras que hay que hacer *scroll* para ver las que fueron alcanzadas anteriormente. En ocasiones resulta confuso.	Baja
Sería útil que el organizador pudiera ver información general sobre la actividad de un participante (especialmente las balizas alcanzadas) sin tener que *clicar* sobre él. Esto mejoraría notablemente el coste de interacción (Figuras 5.12 y 5.13).	Media
Sería conveniente proporcionar al organizador una pantalla desde la que pueda monitorizar datos generales sobre transcurso de la actividad. Ahora mismo, solo puede ver el mapa e información específica de los participantes si *clica* sobre ellos (Figura 5.9).	Alta

Tabla 5.3: Problemas de usabilidad detectados durante la segunda prueba de usabilidad.

***Bugs* detectados durante la prueba**	
Bug	**Prioridad**
Si un participante cierra por error la pantalla en la que se ve el mapa de la actividad, no puede volver a abrirla sin reiniciar la aplicación.	Alta
Si un usuario quiere cerrar su sesión durante una actividad y hacer *log-in* con otra cuenta (lo cual no debería ser muy común), la aplicación falla.	Alta

Tabla 5.4: *Bugs* detectados en la aplicación durante la realización de la prueba de usabilidad.

Posibles puntos de mejora	**Prioridad**
Mejorar el calibrado de las imágenes de los mapas, pues el *track* aparece desplazado sobre ellas.	Media
Mejorar la calidad con la que se ven los mapas de las actividades. Actualmente hay cierta pérdida de definición respecto a la imagen original.	Media

Tabla 5.5: Posibles puntos en los que la aplicación podría mejorar a raíz de las observaciones de los usuarios durante la prueba de usabilidad.

Modificaciones en la lógica de las actividades	**Prioridad**
La aplicación debería proporcionar a los participantes de una actividad la posibilidad de abandonar ésta, de tal forma que el organizador pueda saber al instante que esa persona ya no está participando.	Alta
Mientras un participante está en medio de una actividad, no debería poder abandonar las pantallas relativas a dicha actividad.	Media
Convendría permitir a los participantes realizar varios intentos a la hora de dar respuesta a un reto. El número de intentos permitidos podría fijarse en dos para los retos tipo test, y tres para los retos de respuesta corta. También se podría dejar al organizador configurar este parámetro.	Media
No se debería permitir al participante abandonar la pantalla del reto de una baliza hasta que responda al mismo.	Media

Tabla 5.6: Modificaciones en la lógica y las "reglas" de las actividades.

Funcionalidad adicional	**Prioridad**
El *feedback* que se proporciona a los participantes (5.12) debería ser más visual, intuitivo y "motivador" (considerar que normalmente los participantes serán niños/as).	Media
Sería de gran utilidad que el organizador pudiera ver en tiempo real (sobre el mapa) dónde se encuentra cada participante durante la actividad.	Media
Para poder comparar los *tracks* de distintos participantes sería interesante permitir mostrar varios a la vez sobre un mismo mapa.	Media
Se debería permitir a los usuarios registrarse en la aplicación mediante su cuenta de **educacyl**.	Media
Sería conveniente que la aplicación abriese automáticamente la pantalla del reto de una baliza cuando detecta que un participante la ha alcanzado.	Media
Se debería permitir al organizador de la actividad enviar avisos a los participantes durante el transcurso de la misma. Especialmente útil sería que hubiera un mensaje específico para solicitar a todos los participantes que regresen al punto de encuentro.	Media

Tabla 5.7: Funcionalidad que sería interesante implementar en un futuro.

Tras analizar los resultados y priorizar las tareas descritas en las tablas anteriores, se corrigieron todas aquellas cuestiones cuya prioridad era "Alta", que eran las que entraban dentro de la planificación y recursos del proyecto. A continuación veremos en qué se materializaron esas correcciones, y las decisiones de diseño tomadas más importantes. En primer lugar, en la Figura 5.9 podemos observar por un lado cómo era la pantalla de monitorización de la actividad en la versión de la aplicación con la que se realizó la prueba, y a continuación, la modificación que se introdujo a partir de los resultados obtenidos tras la misma. Como podemos apreciar, se ha incluido una pantalla adicional, en la que el organizador puede ver el mapa de la actividad. Durante la prueba de usabilidad, también descubrimos que la aplicación facilitaría el trabajo de los usuarios si permitiera visualizar el *track* completo seguido por un participante en una sola imagen. En la versión de la aplicación utilizada durante la prueba, los usuarios sólo podían ver a progresión a lo largo del tiempo del recorrido seguido, pero no lo podían representar entero. En la Figura 5.10 vemos las modificaciones introducidas.

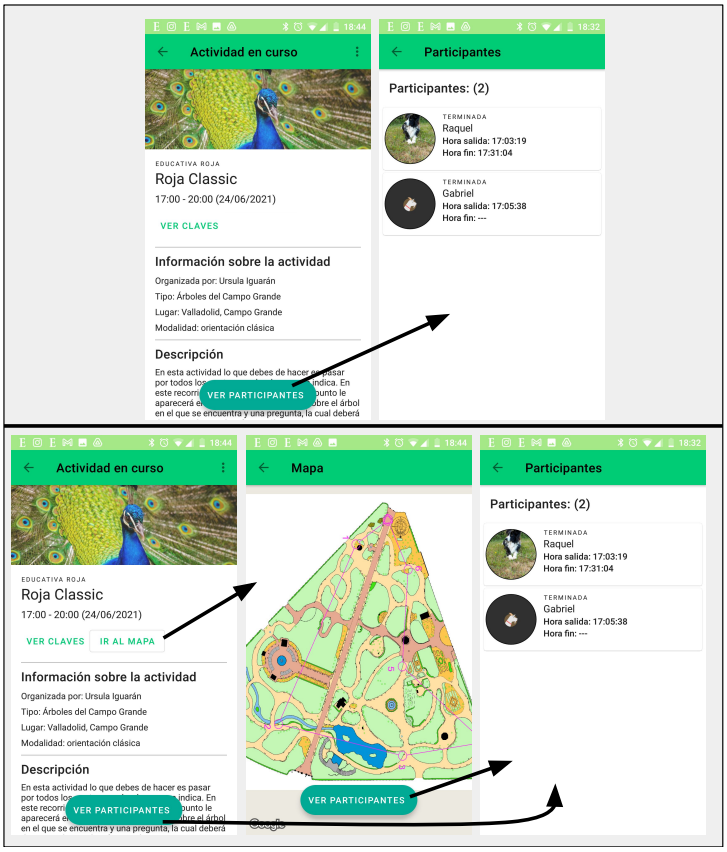

Figura 5.9: Pantalla del organizador.
La imagen superior muestra la pantalla de monitorización de la actividad que veía el organizador. En la imagen inferior vemos que, tras la prueba de usabilidad, se ha añadido una nueva pantalla (centro).

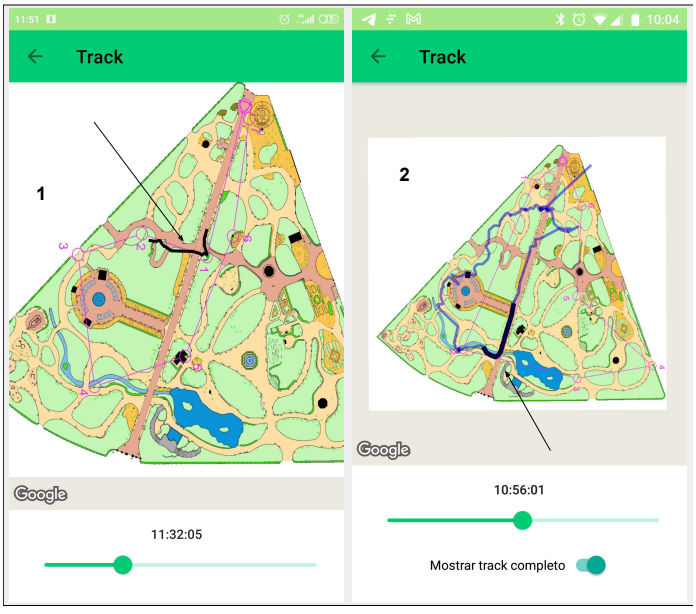

Figura 5.10: Problema con los *tracks*.
1: Versión de la aplicación empleada en la segunda prueba de usabilidad. **2:** Versión posterior de la aplicación en la que el usuario puede ver el *track* completo.

Otra característica que fue introducida tras la realización de la prueba, fue la de indicar a los participantes,

mediante un componente *Badge* de Material, el número de balizas alcanzadas para las que aún no han respondido al reto (Figura 5.11). Esta modificación tenía el objetivo de ayudar a los participantes a saber cómo actuar una vez encontrada una baliza.

Figura 5.11: Confusión con el botón de las balizas.
1: Versión empleada en la prueba de usabilidad (al principio los usuarios no sabían dónde pulsar para responder al reto de una baliza). **2:** Modificación introducida a raíz de la prueba. El *Badge* indica el número de balizas cuyo reto está pendiente de responder.

Por otra parte, en la Figura 5.12 se ilustran varios puntos en los que la usabilidad de la aplicación también podría mejorar. En primer lugar, durante la prueba se detectó que para los organizadores sería útil ver más información sobre los participantes desde la pantalla "Participantes" (especialmente el número de balizas alcanzadas, respondidas, acertadas y falladas). En esa misma pantalla, también sería de utilidad representar a los participantes con distinto color dependiendo de si aún no han comenzado la actividad, si están participando ahora, o si ya han terminado, del mismo modo que se hace para representar si el reto de una baliza ha sido respondido y si la respuesta ha sido correcta o no (pantalla "Balizas alcanzadas"). La tercera pantalla de la Figura 5.12 muestra el reto de una baliza junto con el *feedback* que se proporciona al participante. Según pudimos conocer gracias a la prueba, ese *feedback* tiene un tono demasiado sobrio, y debería ser más efusivo y motivador para niños.
Finalmente, para solucionar la cuestión de que el organizador pueda, de un solo vistazo, hacerse una mejor idea general de toda la actividad realizada por un participante, se ha pensado que se podría utilizar un *dashboard* inspirado en el que emplea Moodle[5] para representar la totalidad de las preguntas respondidas y falladas por un usuario (Figura 5.13), pero adaptándolo a una interfaz móvil. Sin embargo, esta modificación, al igual que gran parte de las recogidas en las tablas mostradas previamente en esta sección, quedaría pendiente para futuras mejoras de la aplicación, ya que su implementación demandaría mucho más tiempo del disponible en un TFG.

[5]Sitio web de Moodle: https://moodle.org/

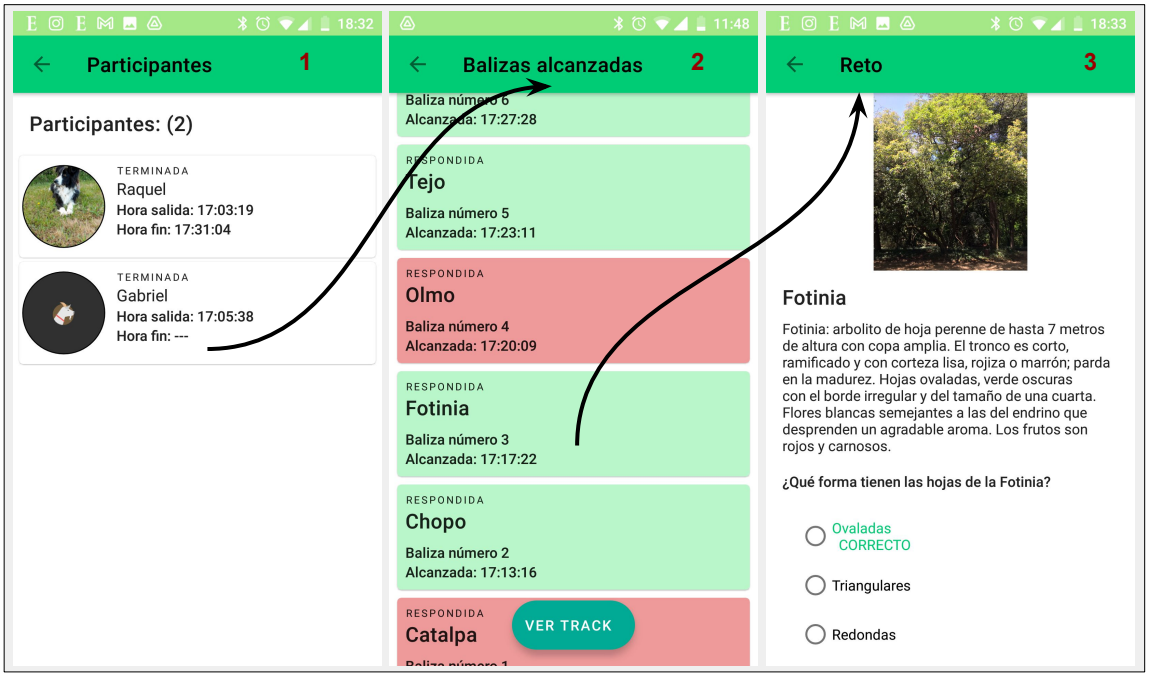

Figura 5.12: Problema de usabilidad.
1: Listado de participantes desde el que el organizador puede *clickar* y ver la actividad de cada uno. **2:** Balizas alcanzadas por cada participante. **3:** Reto en una baliza.

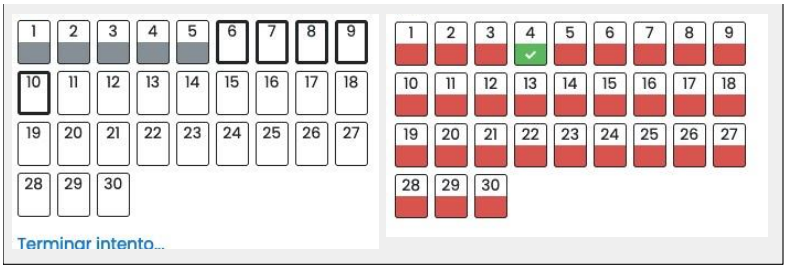

Figura 5.13: *Dashboard* de Moodle.
Este *dashboard* podría inspirar la funcionalidad de que un organizador tenga consciencia del recorrido de un participante de un solo vistazo.

5.3. Arquitectura del sistema

Esta aplicación se ha diseñado siguiendo el paradigma cliente–servidor. Como vemos en la Figura 5.14, la aplicación cliente o *front-end* (aplicación Android) abarca las capas de *Presentación* y *Lógica*, mientras que el *back-end* (Firebase) es el servidor, que se encarga de la recuperación y manipulación de los *Datos*. Siguiendo este esquema, la aplicación cliente gestiona la lógica de las actividades. Los datos que se generan durante dichas actividades se almacenan en el *back-end* de Firebase, de tal forma que se sincronizan y están actualizados para todos los usuarios. Del mismo modo, la aplicación cliente obtiene los datos de las actividades y el contenido de las balizas mediante consultas que realiza hacia el *back-end*. Por último, en el *back-end* también se realiza toda la gestión de usuarios y almacena su información.

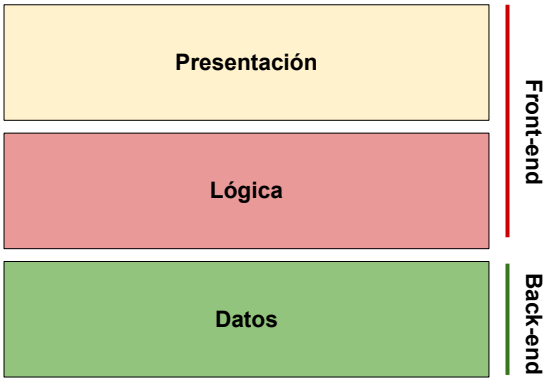

Figura 5.14: Capas de la aplicación y distribución entre *front-end* y *back-end*.

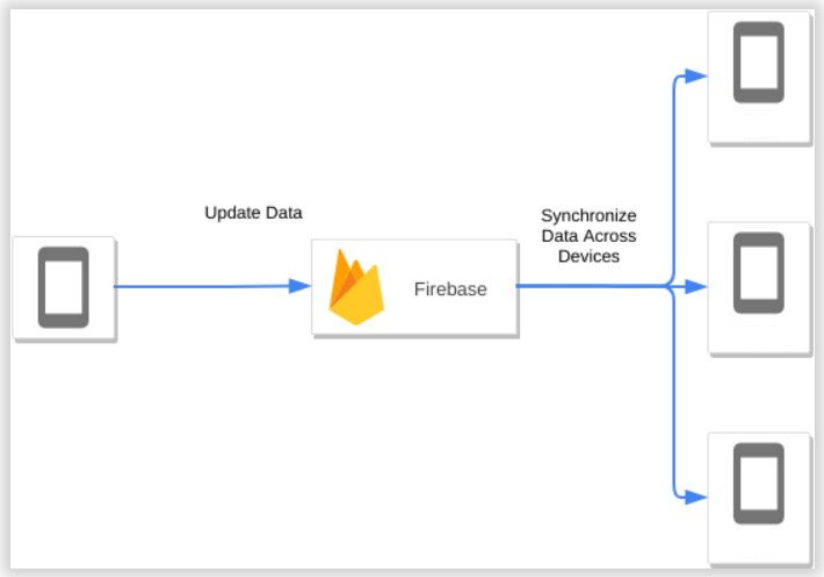

Figura 5.15: Sincronización de datos entre dispositivos mediante Firebase.
Imagen obtenida de [31].

5.4. Back-end

Como se ha mencionado en la sección anterior, el *back-end* de este sistema se encarga de gestionar y almacenar los datos de la aplicación, y de servirlos a los clientes cuando éstos lo solicitan. Para su implementación se ha elegido **Firebase**. Firebase es una plataforma de Google para el desarrollo de aplicaciones. Ofrece herramientas para cubrir gran parte de los servicios de *back-end* que habitualmente las aplicaciones requieren, tales como autenticación de usuarios, base de datos o almacenamiento de archivos, entre otros. Todos estos servicios están en la nube, alojados en componentes de *back-end* que Google se encarga de gestionar y mantener. Las aplicaciones cliente que utilizan los servicios de Firebase se comunican directamente con esos componentes. Esto lo hacen simplemente mediante la invocación *end-points*. De este modo, los desarrolladores no tienen que preocuparse de programar el *back-end*, pues de eso ya se encarga Firebase. Así, es habitual incluir Firebase dentro de los productos categorizados como "***back-end as a service***". Finalmente, a diferencia de Google Cloud, Firebase va destinado principalmente a desarrolladores de *front-end*, a quienes les facilita poder centrarse en crear la parte de cliente de la aplicación, en lugar de tener que preocuparse de desarrollar y configurar también el *back-end*. Dicha característica hizo que Firebase se adaptase muy bien a las necesidades de este proyecto. Además, como vimos al tratar sobre los requisitos no funcionales (4.2.2), partíamos de la premisa de que la información del *back-end* debía ser accesible como REST

[5]Sitio web de Firebase: https://firebase.google.com/

API (característica que también ofrece Firebase). Los servicios de Firebase que se han utilizado en la aplicación son los siguientes:

- **Firebase Authentication** para la autenticación de usuarios.
- **Cloud Firestore** como base de datos.
- **Cloud Storage** para el almacenamiento de archivos.
- **Cloud Functions** para ejecutar código de *back-end*.

5.4.1. Base de datos Cloud Firestore

Para la base de datos de este proyecto se ha utilizado **Cloud Firestore**, de Firebase. Cloud Firestore es una base de datos NoSQL en la nube. El modelo de datos de Cloud Firestore está estructurado en *colecciones* y *documentos*. Las colecciones son contenedores que almacenan un conjunto de documentos. Los *documentos* contienen campos con soporte para diferentes tipos de datos (valores numéricos, *strings*, *timestamps*, etc) en los cuales se almacena la información. Los documentos conforman la unidad de almacenamiento en Cloud Firestore. Finalmente, se pueden crear *subcolecciones* bajo un documento, de tal forma que los datos se estructuran de una manera jerárquica. En la Figura 5.16 vemos un ejemplo de cómo se organizan los documentos en colecciones. En esa misma Figura podemos apreciar que, en esencia, los documentos son registros ligeros que contienen campos mediante los que "mapean" valores, por lo que guardan gran relación con los objetos JSON.

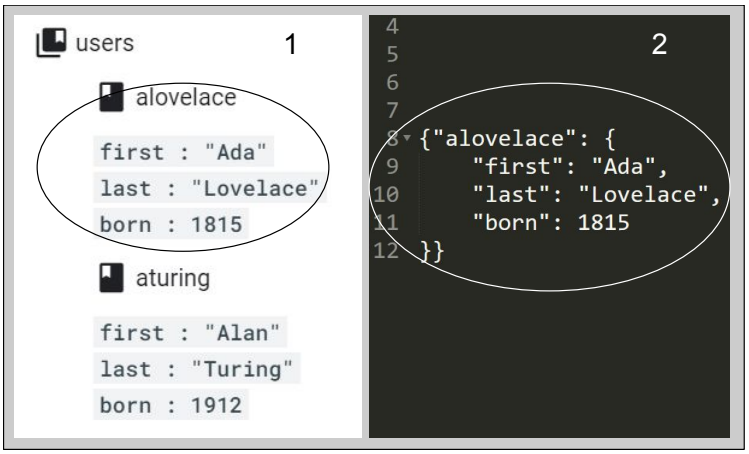

Figura 5.16: Colecciones y documentos.
1: Imagen obtenida de la documentación de Cloud Firestore [11]. **2:** Representación del documento ("alovelace") en forma de objeto JSON.

5.4.1.1. Estructura y documentos

En la Figura 5.17 podemos observar cómo se han organizado los datos en este proyecto, utilizando colecciones y documentos de Cloud Firestore. A continuación se describe también la estructura de los documentos de Cloud Firestore utilizados en esta aplicación, con la terminología y tipos de datos propios de esta base de datos.

- **Plantilla**
  - **id**: string
  - **balizas**: array
  - **color**: string
  - **descripción**: string
  - **lugar**: string
  - **mapa**: string
  - **normas**: string

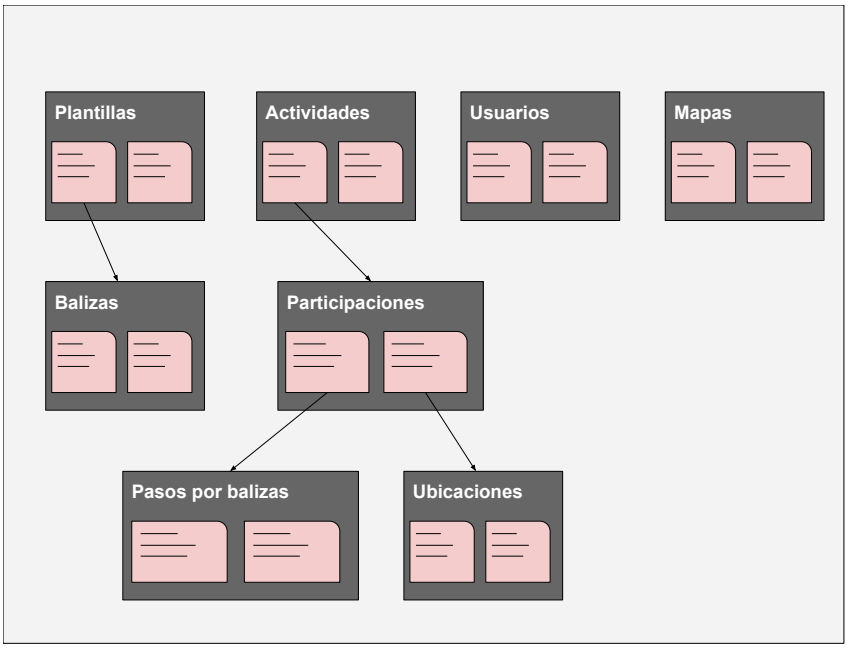

Figura 5.17: Estructura de la base de datos organizada en colecciones y documentos.

- **tipo**: string
- **password**: string
- **salidaLat**: number
- **salidaLng**: number
- **metaLat**: number
- **metaLng**: number

- **Baliza**
  - **id**: string
  - **ubicacion**: geopoint
  - **nombre**: string
  - **numero**: number
  - **opcionesRespuesta**: array
  - **pregunta**: string
  - **respuestaTest**: number
  - **texto**: string
  - **tespuestaTexto**: string
- **Actividad**
  - **id**: string
  - **horaInicio**: timestamp
  - **horaFin**: timestamp
  - **key**: string
  - **visibleID**: string
  - **participants**: array
  - **score**: boolean
  - **ayudaUbicacion**: boolean

  - **plantilla**: string
  - **organizador**: string
  - **título**: string

- **Participación**
  - **horaComienzo**: timestamp
  - **horaFinalizacion**: timestamp
  - **participante**: string
  - **estado**: string

- **PasoPorBaliza**
  - **respuestaCorrecta**: boolean
  - **respondida**: boolean
  - **baliza**: string
  - **instante**: timestamp
  - **respuestaEscrita**: string
  - **respuestaTest**: number

- **Ubicación**
  - **ubicación**: geopoint
  - **instante**: timestamp

- **Usuario**
  - **id**: string
  - **nombre**: string
  - **apellidos**: string
  - **tieneFoto**: boolean

- **Mapa**
  - **id**: string
  - **zoomInicial**: number
  - **zoomMaximo**: number
  - **zoomMinimo**: number
  - **centradoEn**: geopoint
  - **esquinasMapa**: array
  - **esquinasOverlay**: array

5.4.1.2. Justificación del diseño de la base de datos

A la hora de diseñar la base de datos de la aplicación en Cloud Firestore, se han aplicado los principios recogidos en la documentación oficial [11]. En esta sección se justifican las decisiones tomadas al de diseñar la base de datos. En primer lugar, merece la pena destacar algunos detalles sobre el funcionamiento de Cloud Firestore que nos ayuden a entender los *trade-offs* que ha habido que considerar:

- **A la hora de diseñar, se deben considerar las reglas de seguridad**. Dependiendo del diseño que hagamos de la base de datos, será más o menos complicado implementar las reglas de seguridad de la misma.
- No se puede hacer consultas sobre campos específicos de un documento. **Se lee el documento entero**. Esto implica que si nuestros documentos son muy grandes y abarcan mucha información, estaremos constantemente leyendo datos que no necesitamos, lo cual puede penalizar el rendimiento. Tampoco se pueden escribir reglas de seguridad que afecten solo a ciertos campos de un documento.

- **El tamaño de los documentos tiene límites**. Por ello, si pensamos guardar dentro de un documento una colección de datos (lo cual puede ayudar a minimizar el número de lecturas), hay que tener en cuenta que la solución no será escalable en caso de que esa colección vaya a crecer mucho (miles de registros).

- Si bien para la realización de este trabajo de fin de grado se ha utilizado un plan de prueba gratuito, hay que tener en cuenta que Cloud Firestore es un servicio de pago, **siendo gratuita hasta un número determinado de operaciones diarias, que es suficiente para un uso no masivo de la aplicación.** Por lo tanto, conviene minimizar dicho número.

- **El tamaño de las colecciones no penaliza el rendimiento**. Una consulta sobre una colección seguirá siendo eficiente aunque esa colección tenga una enorme cantidad de documentos.

- **En ocasiones, para optimizar el número de consultas, se pueden duplicar algunos datos**. Sin embargo, habrá que garantizar la consistencia utilizando Cloud Functions. Por ello, solo se recomienda utilizar esta técnica si la operación va a ser muy frecuente en nuestra aplicación.

Teniendo estas consideraciones en mente, veamos por qué se han tomado las distintas decisiones de diseño:

- Las *Participaciones* dependen de una actividad. Sin embargo, podrían haberlo hecho de un usuario en su lugar. El problema hubiera sido que el organizador de una actividad, para hacer la monitorización de la misma y ver actualizaciones en tiempo real, necesita "escuchar" los cambios ocurridos en las *Participaciones* de su actividad. Si las *Participaciones* fuesen una subcolección bajo un documento usuario, el organizador tendría que obtener la lista de participantes y para cada uno de ellos, mediante una consulta, encontrar la participación correspondiente, lo cual sería muy poco eficiente.

- Las *Actividades* podrían haber dependido de la plantilla a la que pertenecen. Pero en ese caso, un usuario que quiere obtener las actividades en las que es participante, tendría que buscar para cada plantilla todas sus actividades, y ver si está apuntado a alguna. Lo mismo hubiera ocurrido si las *Actividades* dependieran del usuario que las ha organizado. Por esta razón, se decidió situar las *Actividades* en el primer nivel de la base de datos.

- Los documentos de *Paso por baliza* incluyen (entre otros datos) la información de lo que sería la respuesta al reto de la baliza. Esto es así porque por una parte, a penas aumenta el tamaño de los documentos de *Paso por baliza*, mientras que sí puede ahorrar lecturas. Por otra parte, es frecuente acceder a ambos datos a la vez.

- Los *Mapas* podrían haberse "embebido" dentro de las *Plantillas*. El único problema es que hubiera dificultado la reutilización de los mapas. En un futuro, es posible que esta aplicación se complemente con otra que sirva para crear y configurar actividades. Se ha decidido dejar ambos objetos desacoplados, para no limitar las posibilidades de dicha aplicación. Siguiendo esa lógica se podría haber aplicado lo mismo para las *Balizas*. Sin embargo, el hecho de que dependan de la *Plantilla* nos facilitó implementar la lógica de las carreras, ya que a cada baliza le podíamos poner un atributo que representase su orden relativo dentro de la actividad. No sería complicado modificar esto en caso de que la futura aplicación lo requiriese.

- Los documentos de las *Plantillas* y las *Actividades* poseen sendos atributos que representan sus *Balizas* y sus *Participantes*, respectivamente. Esto es así porque resulta útil para reducir el número de consultas realizadas (ejemplo de utilización de la técnica de duplicar algunos datos).

5.4.1.3. Reglas de seguridad

Debido a la naturaleza poco segura de las aplicaciones cliente, no se debe dejar a éstas la responsabilidad de "decidir" *quién* puede acceder a *qué* datos de la aplicación. Por ello, Firebase permite establecer **reglas de seguridad**. Gracias a esto, cada vez que llegue una petición a la base de datos, Firebase se encargará de autorizarla o no, de acuerdo a las reglas de seguridad establecidas. En este proyecto, tras hacer el estudio de cómo deberían ser estas reglas de seguridad, se ha llegado a la creación de cuatro niveles distintos de acceso a los datos (representados en la Figura 5.18):

- **Nivel A**: cualquier usuario autenticado y cuyo e-mail haya sido verificado puede leer estos datos. No se permiten más operaciones (esto habrá que modificarlo si se llega a crear la aplicación destinada a la creación de nuevas plantillas de actividades).

- **Nivel B**: cualquier usuario autenticado y cuyo e-mail haya sido verificado, puede leer estos datos y realizar escrituras.
- **Nivel C**: cualquier usuario autenticado puede crear y leer. Solo un usuario autenticado y con su e-mail verificado puede editar y borrar un documento. Además, ese documento tiene que ser el suyo propio.
- **Nivel D**: cualquier usuario autenticado y cuyo e-mail esté verificado puede realizar lecturas. Para realizar actualizaciones, borrados y crear nuevos documentos, se deben cumplir las condiciones anteriores y además la *Participación* debe pertenecer al propio usuario.

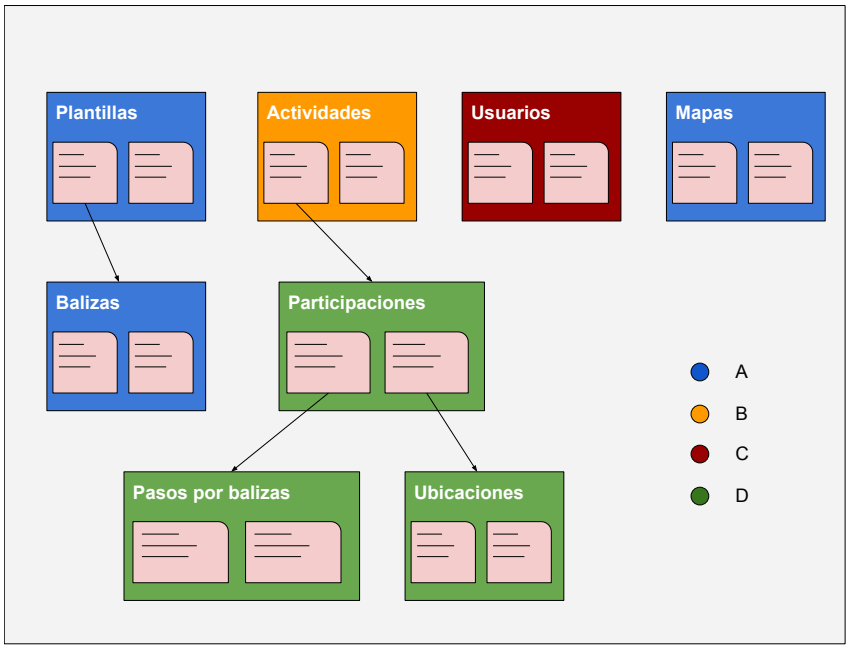

Figura 5.18: Niveles de seguridad en la base de datos de la aplicación.
Nivel A (azul), **Nivel B** (naranja), **Nivel C** (granate), **Nivel D** (verde).

5.4.2. Firebase Authentication

Otro de los servicios de Firebase que se ha utilizado en esta aplicación es Firebase Authentication, para la autenticación de usuarios. Firebase Authentication permite gestionar los usuarios de una aplicación de manera sencilla, ofreciendo varios métodos de autenticación. Como vimos en el análisis de requisitos, esta aplicación debía permitir a los usuarios registrarse mediante e-mail y contraseña, por lo que éste ha sido el método elegido. Firebase Authentication también ha resultado de gran utilidad a la hora de implementar la verificación del correo de los usuarios, así como la posibilidad para éstos de recuperar su contraseña en caso de olvido, ya que basta con codificar unas sencillas instrucciones para que Firebase se encargue de enviar los correos de verificación y recuperación.

5.4.3. Firebase Storage

Firebase Storage es un servicio de almacenamiento especialmente diseñado para almacenar imágenes, audio y vídeo. En esta aplicación, se ha utilizado este servicio de Firebase para almacenar las imágenes de las plantillas, de las balizas, de los mapas y las fotos de perfil de los usuarios.

5.4.4. Cloud Functions

Las Cloud Functions de Firebase son código de *back-end* que se puede ejecutar en base a distintos tipos de eventos o desencadenadas por peticiones HTTP. En nuestro caso, las Cloud Functions son útiles para realizar tareas de asegurar la consistencia dentro de la base de datos, ya que como se mencionó en la sección 5.4.1.2, en ocasiones se han duplicado algunos datos por razones de eficiencia. Cabe destacar que para poder utilizar las Cloud Functions en nuestro proyecto, es necesario tener un plan de pago de Firebase (*plan blaze*). Dicho plan ofrece

hasta un número máximo de ejecuciones diarias de Cloud Functions de manera gratuita. Sin embargo, una vez rebasada esa cuota, se empieza a cobrar (y es imposible establecer un límite a partir del cual se detenga el servicio automáticamente). Por ese motivo, para utilizar este servicio, cuando la aplicación está en producción, conviene que haya una persona encargada de monitorizar frecuentemente el uso de los servicios de Firebase. En este TFG no se ha contratado dicho plan, y las Cloud Functions no están en funcionamiento en el momento de entregar el proyecto. Lo que sí se ha hecho es, utilizando el emulador local de Firebase (6.4), desarrollar y probar las Cloud Functions que serían necesarias, por si en un futuro se quieren poner en producción. A continuación se describen esas funciones, de las cuales, actualmente se encarga la aplicación cliente, pero que sería mejor práctica ejecutarlas en el servidor:

- Al crear una nueva *Participación* (un usuario se ha unido a una actividad), actualizar automáticamente el array "participantes" de la *Actividad*, para mantener ambos datos sincronizados. Aplicar lo mismo para cuando se elimine una *Participación*.

- Cuando se elimine un usuario del servicio de autenticación (Firebase Authentication) porque éste ha decidido borrar su cuenta, eliminar también automáticamente el documento correspondiente en Cloud Firestore.

5.5. Front-end

Como hemos visto en 5.3, el *front-end* de este sistema es una aplicación Android, responsable de las capas de *Presentación* y *Lógica* del sistema. Siguiendo este esquema, para obtener los datos de las actividades, los usuarios, las balizas, y el resto de elementos, la aplicación cliente se comunica con el *back-end* que actúa como base de datos remota. Durante el transcurso de las actividades, la lógica de las mismas se ejecuta en la aplicación cliente de cada participante, y los datos que se van generando se almacenan en el *back-end*.

5.5.1. Estructura del *front-end*

A continuación se representa mediante las Figuras 5.19, 5.20, 5.21 y 5.22 la estructura que se ha seguido en el diseño de la aplicación cliente.

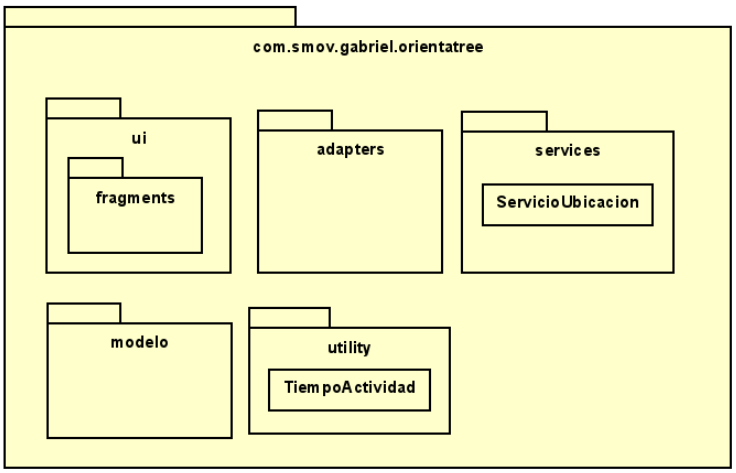

Figura 5.19: Diagrama de paquetes de la aplicación cliente.

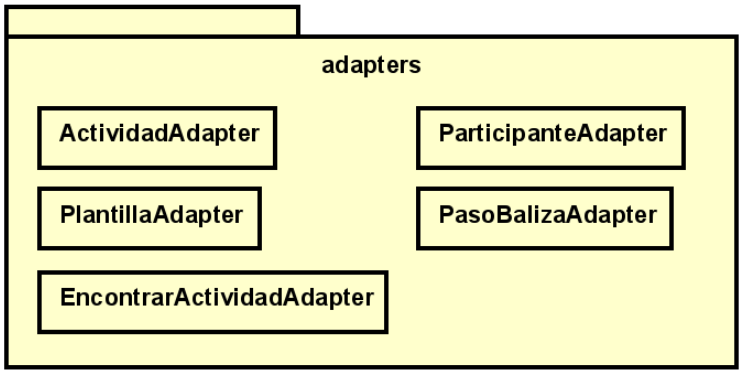

Figura 5.20: Detalle del paquete **adapters**.

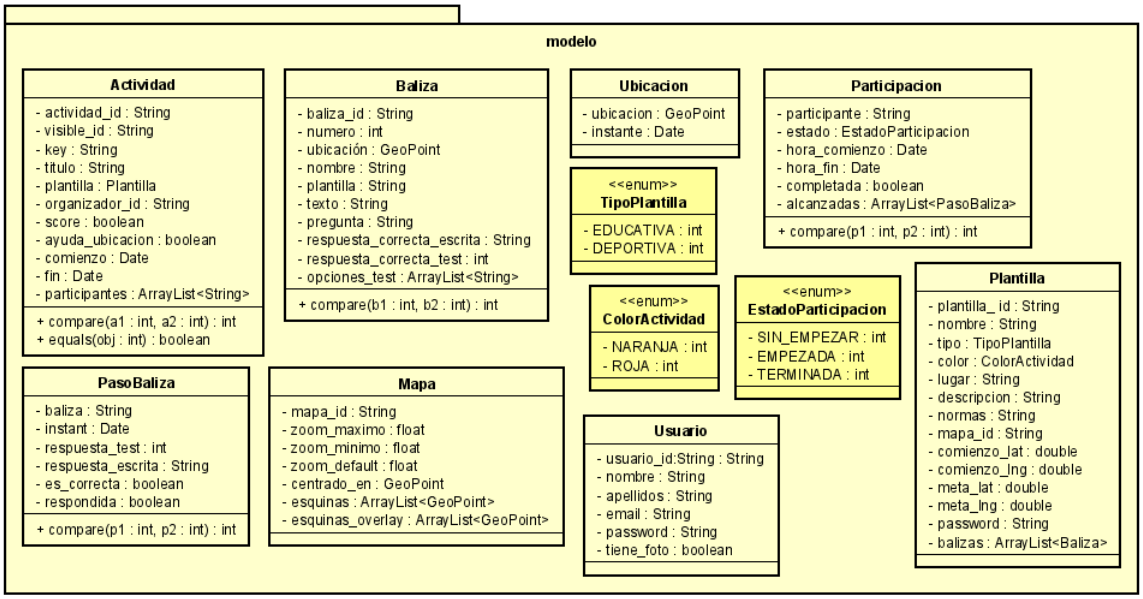

Figura 5.21: Detalle del paquete **modelo**. Se han omitido las operaciones *getter* y *setter* de las clases, así como los constructores.

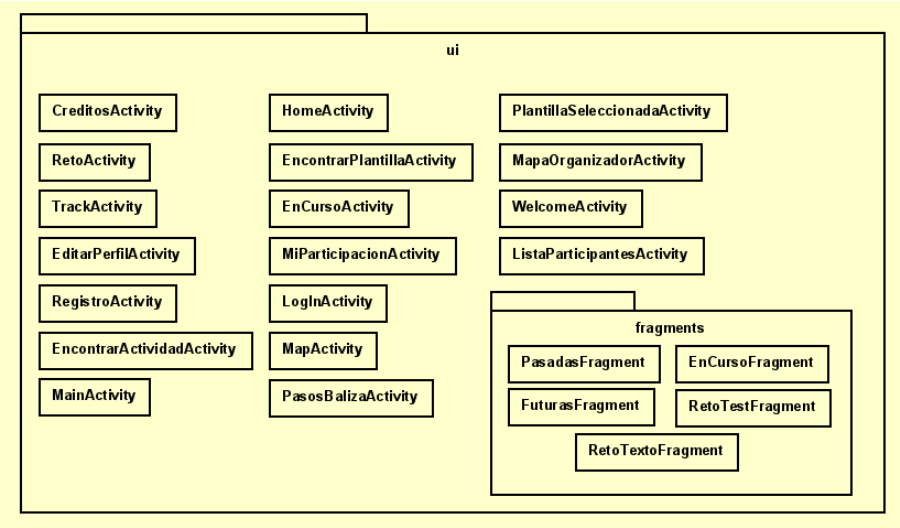

Figura 5.22: Detalle del paquete **ui**.

5.5.2. Servicio en primer plano para controlar la lógica de las actividades

Como se ha mencionado al inicio de la sección 5.3, la lógica de las actividades reside en la aplicación cliente. Mientras un usuario toma parte en una actividad, la aplicación tiene que iniciar una tarea mediante la cual solicite constantemente actualizaciones sobre la ubicación del dispositivo. También tiene que ir almacenando ese historial de ubicaciones en el *back-end*, de tal modo que luego sea posible reconstruir el *track*. Además, tiene que ser capaz de determinar cuál es la siguiente baliza que el usuario debe alcanzar, y debe realizar una consulta hacia el *back-end* tanto para obtener el contenido educativo de la baliza alcanzada, como para actualizar la respuesta dada por el usuario. Para este tipo de tareas, que se ejecutan sin necesidad de que el usuario interactúe directamente con ellas, Android ofrece un componente de aplicación llamado ***Service***. En este caso, el componente elegido para realizar estas tareas es un ***foreground service*** (servicio en primer plano), que se caracteriza porque hace saber al usuario que se está ejecutando mediante una notificación que permanece fija en la barra de estado. En la Figura 5.23 podemos observar cuál es el ciclo de vida de un componente *Servicio*.

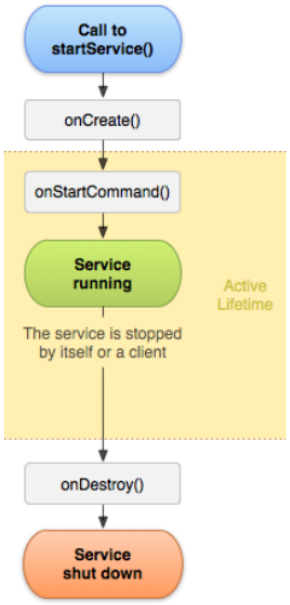

Figura 5.23: Ciclo de vida de un componente *Servicio*.
Imagen obtenida de [1].

En el caso de nuestra aplicación, cada vez que el usuario decide comenzar o retomar una actividad, se llama al

método **startService** y el Servicio se pone en marcha. Mientras el servicio se ejecuta, el usuario puede navegar por las distintas pantallas de la aplicación, bloquear su dispositivo o minimizar la aplicación. El Servicio se detendrá a sí mismo si el usuario completa la actividad o si se le acaba el tiempo. También se detendrá si el usuario decide abandonar o si cierra la aplicación. En este último caso, aún podrá retomar la actividad.

Capítulo 6

Desarrollo

En este capítulo se describen las herramientas utilizadas para el desarrollo de la aplicación y se explica cómo se han implementado algunas de las características más interesantes del proyecto.

6.1. Lenguajes de programación y entornos de desarrollo

La aplicación Android se ha desarrollado utilizando el lenguaje **Java**. Ocasionalmente se ha utilizado también **JavaScript** para el desarrollo de Cloud Functions. Las reglas de seguridad de la base de datos se han escrito utilizando el lenguaje propio de Firebase, el cual está basado en CEL[1].

6.1.1. Android Studio

Para el desarrollo de la aplicación Android, se ha utilizado Android Studio[2], que es el entorno de desarrollo integrado oficial para la plataforma Android. Algunas de las ventajas más importantes que ofrece son:

- Compilación flexible basada en Gradle.
- Integración con GitHub.
- Compatibilidad integrada con Google Cloud Platform[3] y Firebase.
- Emulador de dispositivos Android.

6.1.2. Visual Studio Code

Visual Studio Code[4] ha sido el IDE utilizado durante el desarrollo de este proyecto para la creación del código ejecutado en el servidor (Cloud Functions). La razón de utilizar este IDE, es que viene por defecto con soporte para JavaScript y Node.js.

6.2. Glide

Para la descarga de imágenes desde la aplicación Android, se ha utilizado Glide[5]. Glide es una *librería* que ofrece una API muy sencilla de usar, mediante la que se puede encontrar, decodificar y mostrar imágenes de manera eficiente. Está especialmente indicada para aquellas aplicaciones que permiten al usuario hacer *scroll* para ir descargando más imágenes.

[1] CEL (Common Expression Language): https://github.com/google/cel-spec
[2] Android Studio: https://developer.android.com/studio
[3] Google Cloud Platform: https://cloud.google.com/
[4] Visual Studio Code: https://code.visualstudio.com/
[5] Glide: https://bumptech.github.io/glide/

6.3. Control de versiones: GitHub

Para realizar el control de versiones de la aplicación y para alojar el código fuente se ha utilizado GitHub[6]. La metodología de trabajo seguida es la conocida como *Gitflow Workflow*. Esta metodología consiste en la utilización de *feature branches* para ir añadiendo nuevas características junto con dos *branches* especiales: *main* (o *master*) y *develop*. La rama *main* contiene el histórico oficial de todas las *releases* de la aplicación, mientras que *develop* sirve como *branch* de integración de nuevas características [2].

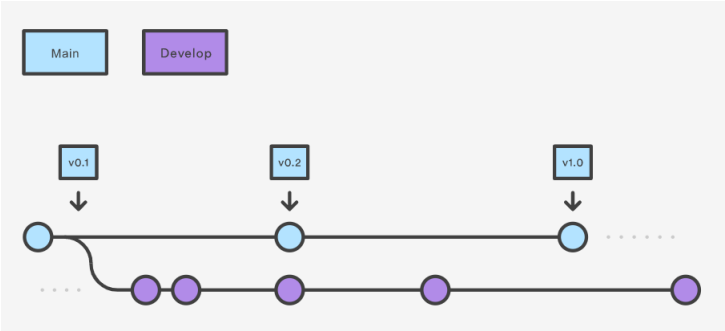

Figura 6.1: Representación de la metodología *Gitflow Workflow*.
Imagen obtenida de [2].

6.4. Firebase Emulator Suite

Firebase Emulator Suite es un conjunto de herramientas para desarrolladores que permiten hacer pruebas en local sobre la mayoría de servicios que ofrece Firebase, simulando el comportamiento de éstos. En este proyecto se ha utilizado el Emulator Suite local para probar las Cloud Functions. Gracias a ello, se puede simular una base de datos Cloud Firestore y probar el comportamiento de nuestras funciones ejecutadas sobre dicha base de datos, en lugar de sobre nuestra base de datos de producción.

6.5. Mapa de las actividades y Google Maps SDK

A priori, uno de los puntos que podrían suponer mayor dificultad técnica en cuanto a su desarrollo era encontrar la forma adecuada de implementar la visualización y el manejo de los mapas de las actividades. Lo ideal, sería disponer de una imagen georreferenciada que los usuarios pudieran manipular fácilmente, y que permitiera representar puntos geográficos y formas sobre ella. Para resolver este problema se utilizó una característica que ofrece Google Maps SDK: *ground overlay*. Esta técnica consiste en superponer una imagen sobre un mapa, como muestran las Figuras 6.2 y 6.3. Una vez tenemos la imagen situada en el lugar adecuado, tan solo tenemos que ocultar el mapa que hay "por debajo", de tal modo que de la sensación de que no existe y que sólo está la imagen.

[6]GitHub: https://github.com/

Figura 6.2: Imagen de un mapa de orientación superpuesta sobre el Campo Grande de Valladolid.

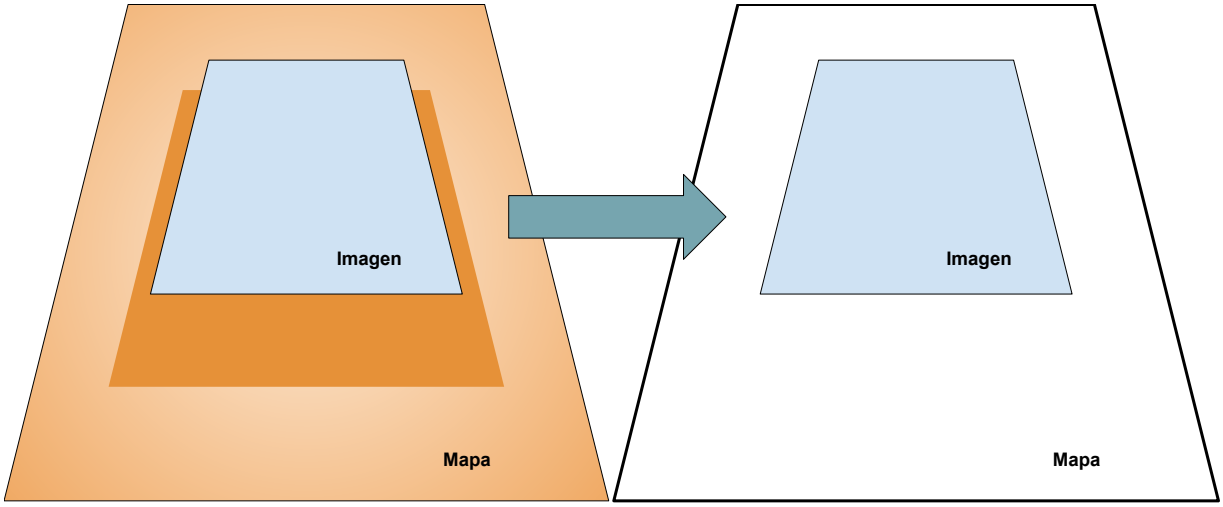

Figura 6.3: Una vez la imagen está situada en el lugar adecuado, ocultamos el mapa que hay "debajo".

Respecto a las balizas, en la Figura 6.4 vemos cómo aparecen representadas sobre el mapa. Cabe destacar que forman parte de la imagen superpuesta al mapa, por lo que las representaciones son completamente estáticas. Esto es así porque el usuario no tiene que manipularlas ni realizar ninguna operación con ellas. La aplicación tan solo debe conocer sus coordenadas, de tal modo que para cada nueva ubicación del usuario, podamos calcular la distancia a la siguiente baliza y detectar si ha sido alcanzada.

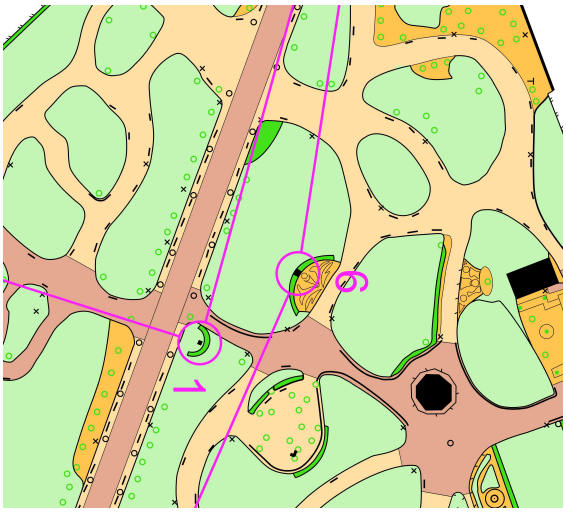

Figura 6.4: Detalle de la representación de balizas sobre el mapa.

Finalmente, Google Maps SDK también permite dibujar formas sobre un mapa. Esta característica ha sido utilizada para representar el *track* de los usuarios durante las carreras. Concretamente, se han empleado las *Polylines*, que consisten en una serie de puntos unidos mediante lineas. De este modo, para dibujar el *track* tan solo hay que crear un *Polyline* con todos los puntos que se quieran mostrar, y estos aparecerán sobre el mapa unidos entre sí. Para representar la progresión del participante como una sucesión de puntos que los usuarios puedan controlar (ver Figura 5.10), basta con *mapear* cada ubicación que forma parte del *Polyline* con un punto de la barra *Slider*.

Capítulo 7

Pruebas

En este capítulo veremos las pruebas que se han realizado para comprobar el correcto funcionamiento de la aplicación. El capítulo se ha dividido en tres secciones. En primer lugar están las pruebas de las reglas de seguridad de la base de datos, a continuación se muestran las pruebas de la aplicación como tal, y por último, las pruebas realizadas para comprobar el funcionamiento de las Cloud Functions.

7.1. Reglas de seguridad de la base de datos

Para probar las reglas de seguridad de la base de datos, que determinan *quién* puede acceder a *qué* datos de la aplicación, se ha utilizado la técnica de **particiones en clases de equivalencia**. Dicha técnica nos ayuda a optimizar los casos de prueba. Para realizar este proceso se ha utilizado la herramienta que proporciona la consola de Firebase a tal efecto (Figura 7.1).

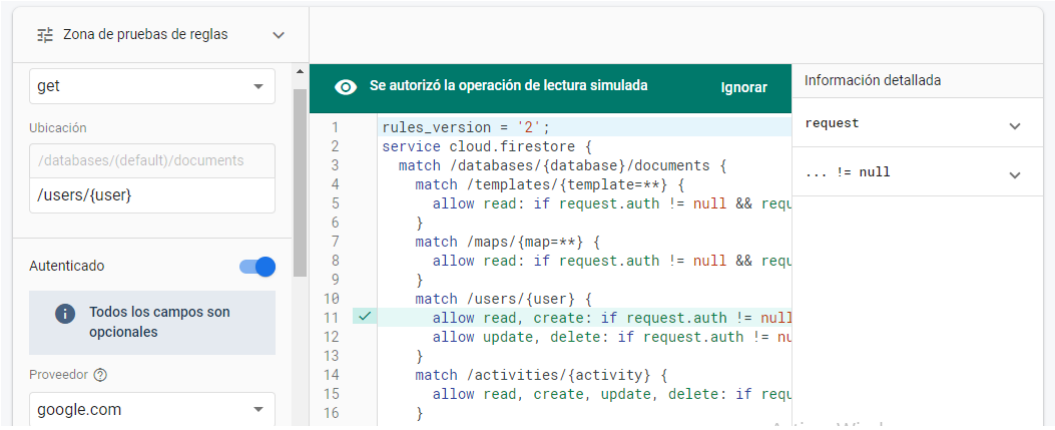

Figura 7.1: Herramienta integrada de Firebase para probar las reglas de seguridad.

Cabe destacar que, como se verá a continuación, los casos de prueba se han dividido en subsecciones atendiendo a las distintas categorías de acceso dentro de la base de datos que definimos en 5.4.1.3 (“A”, “B”, “C” y “D”).

7.1.1. Categoría “A”

Dentro de la categoría “A” se encontraban las siguientes colecciones: Plantillas, Balizas y Mapas. Sobre los documentos de esta categoría solo se pueden hacer operaciones de lectura.

Lectura categoría "A"		
Condición	**Clases válidas**	**Clases no válidas**
Usuario autenticado	Usuario está autenticado (1)	Usuario no autenticado (1.1)
Usuario verificado	Usuario está verificado (2)	Usuario no verificado (2.1)

Tabla 7.1: Clases de equivalencia para la categoría "A".

Lectura categoría "A"			
Caso	**Clases cubiertas**	**Esperado**	**Obtenido**
Verificado y autenticado	(1) y (2)	Autorizado	Autorizado
No autenticado	(1.1)	Denegado	Denegado
No verificado	(2.1)	Denegado	Denegado

Tabla 7.2: Casos de prueba para la categoría "A".

7.1.2. Categoría "B"

La categoría "B" la conforma la colección de las Actividades.

Lectura/escritura categoría "B"		
Condición	**Clases válidas**	**Clases no válidas**
Usuario autenticado	Usuario está autenticado (1)	Usuario no autenticado (1.1)
Usuario verificado	Usuario está verificado (2)	Usuario no verificado (2.1)

Tabla 7.3: Clases de equivalencia para la categoría "B".

Lectura/escritura categoría "B"			
Caso	**Clases cubiertas**	**Esperado**	**Obtenido**
Autenticado y verificado	(1) y (2)	Autorizado	Autorizado
No autenticado	(1.1)	Denegado	Denegado
No verificado	(2.1)	Denegado	Denegado

Tabla 7.4: Casos de prueba para la categoría "B".

7.1.3. Categoría "C"

En la categoría "C" de la base de datos está la colección Usuarios.

Lectura/creación categoría "C"		
Condición	**Clases válidas**	**Clases no válidas**
Usuario autenticado	Usuario está autenticado (1)	Usuario no autenticado (1.1)

Tabla 7.5: Clases de equivalencia para operaciones de lectura/creación en la categoría "C".

Lectura/creación categoría "C"			
Caso	**Clases cubiertas**	**Esperado**	**Obtenido**
Autenticado	(1)	Autorizado	Autorizado
No autenticado	(1.1)	Denegado	Denegado

Tabla 7.6: Casos de prueba para operaciones de lectura/creación en la categoría "C".

Borrado/Actualización categoría "C"		
Condición	**Clases válidas**	**Clases no válidas**
Usuario autenticado	Usuario está autenticado (1)	Usuario no autenticado (1.1)
Usuario verificado	Usuario está verificado (2)	Usuario no verificado (2.1)
ID usuario == ID doc	ID usuario == ID doc (3)	ID usuario != ID doc (3.1)

Tabla 7.7: Clases de equivalencia para operaciones de borrado/actualización en la categoría "C".

Borrado/actualización categoría "C"			
Caso	**Clases cubiertas**	**Esperado**	**Obtenido**
Autenticado, verificado e ID == ID doc	(1), (2), (3)	Autorizado	Autorizado
No autenticado	(1.1)	Denegado	Denegado
No verificado	(2.1)	Denegado	Denegado
ID usuario != ID doc	(3.1)	Denegado	Denegado

Tabla 7.8: Casos de prueba para operaciones de borrado/actualización de la categoría "C".

7.1.4. Categoría "D"

La categoría "D" comprende las siguientes colecciones: Participaciones, Ubicaciones, PasosPorBaliza.

Lectura categoría "D"		
Condición	**Clases válidas**	**Clases no válidas**
Usuario autenticado	Usuario está autenticado (1)	Usuario no autenticado (1.1)
Usuario verificado	Usuario está verificado (2)	Usuario no verificado (2.1)

Tabla 7.9: Clases de equivalencia para operaciones de lectura en la categoría "D".

Lectura categoría "D"			
Caso	**Clases cubiertas**	**Esperado**	**Obtenido**
Autenticado y verificado	(1) y (2)	Autorizado	Autorizado
No autenticado	(1.1)	Denegado	Denegado
No verificado	(2.1)	Denegado	Denegado

Tabla 7.10: Casos de prueba para operaciones de lectura en la categoría "D".

Escritura categoría "D"		
Condición	**Clases válidas**	**Clases no válidas**
Usuario autenticado	Usuario está autenticado (1)	Usuario no autenticado (1.1)
Usuario verificado	Usuario está verificado (2)	Usuario no verificado (2.1)
ID usuario == ID doc	ID usuario == ID doc (3)	ID usuario != ID doc (3.1)

Tabla 7.11: Clases de equivalencia para operaciones de escritura en la categoría "D".

Escritura categoría "D"			
Caso	**Clases cubiertas**	**Esperado**	**Obtenido**
Autenticado, verificado e ID == ID doc	(1), (2), (3)	Autorizado	Autorizado
No autenticado	(1.1)	Denegado	Denegado
No verificado	(2.1)	Denegado	Denegado
ID usuario != ID doc	(3.1)	Denegado	Denegado

Tabla 7.12: Casos de prueba para operaciones de escritura en la categoría "D".

7.2. Pruebas de la aplicación

En esta sección veremos las pruebas realizadas con la aplicación para comprobar el funcionamiento de la misma.

7.2.1. Gestión de usuarios

CP01	**Registrarse**
Descripción	El usuario se registra en la aplicación.
Acción	El usuario rellena los campos de texto con su información y pulsa el botón de **registrarse**.
Resultado esperado	Se crea un nuevo usuario en Firebase Authentication. Se crea un nuevo documento en Cloud Firestore con los datos del usuario. El usuario recibe un correo mediante el que poder verificar su e-mail.
Resultado obtenido	Correcto.

Tabla 7.13: CP01. Registrarse.

CP01-1	**Registrarse (el e-mail no es válido)**
Descripción	El usuario trata de registrarse en la aplicación introduciendo un e-mail cuyo formato no es el adecuado.
Acción	El usuario rellena correctamente todos los campos de texto del registro, excepto el del e-mail, donde introduce una cadena de caracteres incorrecta (por ejemplo, sin un símbolo @) y pulsa **registrarse**.
Resultado esperado	Se informa al usuario de que el e-mail introducido no tiene el formato adecuado.
Resultado obtenido	Correcto.

Tabla 7.14: CP01-1. Registrarse (el e-mail no es válido).

CP02	**Iniciar sesión**
Descripción	El usuario inicia sesión.
Acción	El usuario completa los campos de texto con su e-mail y contraseña, y pulsa el botón **iniciar sesión**.
Resultado esperado	La aplicación identifica al usuario y le muestra la pantalla principal, desde la que puede acceder al resto de pantallas.
Resultado obtenido	Correcto.

Tabla 7.15: CP02. Iniciar sesión.

CP02-1	**Inicio de sesión (e-mail incorrecto)**
Descripción	El usuario trata de iniciar sesión pero escribe mal su e-mail.
Acción	El usuario escribe su e-mail de forma incorrecta y su contraseña de forma correcta. Pulsa el botón de **iniciar sesión**.
Resultado esperado	La aplicación informa al usuario de que o bien el e-mail o bien la contraseña que ha introducido es incorrecto.
Resultado obtenido	Correcto.

Tabla 7.16: CP02-1. Iniciar sesión (e-mail incorrecto).

CP02-2	**Inicio de sesión (contraseña incorrecta)**
Descripción	El usuario trata de iniciar sesión escribiendo una contraseña incorrecta.
Acción	El usuario completa los campos de texto con su e-mail y una contraseña incorrecta, y pulsa **iniciar sesión**.
Resultado esperado	La aplicación informa al usuario de que o bien el e-mail o bien la contraseña que ha introducido es incorrecto.
Resultado obtenido	Correcto.

Tabla 7.17: CP02-2. Iniciar sesión (contraseña incorrecta).

CP02-3	**Inicio de sesión (e-mail no verificado)**
Descripción	El usuario trata de iniciar sesión sin haber verificado su e-mail.
Acción	El usuario completa los campos de texto con su e-mail y contraseña, y pulsa el botón **iniciar sesión**.
Resultado esperado	La aplicación informa al usuario de que para poder acceder, primero tiene que verificar su correo. Le ofrece la posibilidad de que se le reenvíe un correo de verificación.
Resultado obtenido	Correcto.

Tabla 7.18: CP02-3. Iniciar sesión (e-mail no verificado).

CP03	**Restablecer contraseña**
Descripción	El usuario establece una nueva contraseña para su cuenta.
Acción	Desde la pantalla de inicio de sesión, el usuario pulsa **restablecer contraseña**.
Resultado esperado	La aplicación envía al usuario un correo desde el que puede modificar su contraseña. El usuario modifica su contraseña siguiendo las instrucciones de ese correo e inicia sesión exitosamente.
Resultado obtenido	Correcto.

Tabla 7.19: CP03. Restablecer contraseña.

CP04	**Cerrar sesión**
Descripción	El usuario cierra su sesión.
Acción	El usuario pulsa el botón **cerrar sesión**.
Resultado esperado	Se cierra la sesión del usuario y se redirige a éste a la página de inicio de sesión. No se le permite acceder a otras pantallas hasta que no vuelva a identificarse.
Resultado obtenido	Correcto.

Tabla 7.20: CP04. Cerrar sesión.

CP05	**Cerrar sesión en medio de una actividad**
Descripción	El usuario cierra su sesión mientras se encuentra participando en una actividad.
Acción	Mientras el usuario está participando en una actividad, éste pulsa el botón de **cerrar sesión**.
Resultado esperado	El usuario queda *desloggeado* y se le redirige a la pantalla de inicio de sesión. El servicio que durante la actividad se encargaba de rastrear su ubicación se detiene automáticamente.
Resultado obtenido	Correcto.

Tabla 7.21: CP05. Cerrar sesión en medio de una actividad.

CP06	**Eliminar cuenta**
Descripción	El usuario elimina su cuenta.
Acción	Desde la pantalla de "configuración de perfil", el usuario pulsa **eliminar cuenta**.
Resultado esperado	La cuenta del usuario se elimina de Firebase Authentication. El documento de Cloud Firestore con los datos del usuario también se elimina. Se redirige al usuario a la página de inicio de sesión.
Resultado obtenido	Correcto.

Tabla 7.22: CP06. Eliminar cuenta.

7.2.2. Programación de actividades

CP07	**Programar actividad**
Descripción	El usuario programa una actividad.
Acción	Desde la pantalla de una plantilla de actividad, el usuario completa la información de la actividad que quiere programar y confirma que quiere continuar.
Resultado esperado	La actividad queda programada. Se crea un nuevo documento en Cloud Firestore que contiene los datos de la actividad. Se informa al organizador de cuáles son las claves para poder inscribirse en ella.
Resultado obtenido	Correcto.

Tabla 7.23: CP07. Programar actividad.

CP07-1	**Programar actividad (fecha anterior a la actual)**
Descripción	El usuario trata de programar una actividad para una fecha pasada.
Acción	El usuario selecciona mediante el *Date Picker* y los *Time Pickers* una fecha anterior a la actual.
Resultado esperado	La aplicación informa al usuario del error y no crea la actividad. Le permite elegir de nuevo la fecha sin perder el resto de sus preferencias.
Resultado obtenido	Correcto.

Tabla 7.24: CP07-1. Programar actividad (fecha anterior a la actual).

CP08	**Acceder a una plantilla**
Descripción	El usuario accede a la pantalla de una plantilla (desde la cual el posible programar una actividad).
Acción	Buscar la plantilla a partir de la cual se quiere programar una nueva actividad y pulsar sobre ella para acceder. La aplicación solicita una contraseña que se introduce correctamente.
Resultado esperado	La aplicación dirige al usuario a la pantalla de la plantilla seleccionada.
Resultado obtenido	Correcto.

Tabla 7.25: CP08. Acceder a una plantilla.

CP08-1	**Acceder a una plantilla (contraseña incorrecta)**
Descripción	El usuario accede a la pantalla de una plantilla (desde la cual el posible programar una actividad).
Acción	Buscar la plantilla a partir de la cual se quiere programar una nueva actividad y pulsar sobre ella para acceder. La aplicación solicita una contraseña que se introduce de manera incorrecta.
Resultado esperado	La aplicación informa del error y no permite ver la pantalla de la plantilla.
Resultado obtenido	Correcto.

Tabla 7.26: CP08-1. Acceder a una plantilla (contraseña incorrecta).

7.2.3. Búsqueda de actividades

CP09	**Encontrar una actividad**
Descripción	El usuario busca una actividad a partir de su identificador para poder unirse a ella.
Acción	En la pantalla de unirse a una nueva actividad, introducir en la barra de búsqueda el identificador de la actividad.
Resultado esperado	La aplicación encuentra la actividad y se la muestra al usuario.
Resultado obtenido	Correcto.

Tabla 7.27: CP09. Encontrar una actividad.

CP09-1	**Encontrar una actividad (*colisión*)**
Descripción	El usuario busca una actividad a partir de su identificador para poder unirse a ella (hay varias actividades con el mismo identificador).
Acción	En la pantalla de unirse a una nueva actividad, introducir en la barra de búsqueda el identificador de la actividad.
Resultado esperado	La aplicación muestra al usuario todas las actividades cuyo identificador es igual al que ha introducido.
Resultado obtenido	Correcto.

Tabla 7.28: CP09-1. Encontrar una actividad (*colisión*).

CP10	**Inscribirse en una actividad**
Descripción	El usuario, tras haber localizado una actividad a partir de su identificador, se inscribe en ella.
Acción	El usuario pulsa sobre el botón **inscribirse** de la actividad e introduce la clave correcta.
Resultado esperado	El usuario queda inscrito en la actividad. En Cloud Firestore se crea un nuevo documento *Participación*. En el documento de la actividad se actualiza el array de los participantes inscritos. Se informa al usuario de que la inscripción se ha completado.
Resultado obtenido	Correcto.

Tabla 7.29: CP10. Inscribirse en una actividad.

CP10-1	**Inscribirse en una actividad (clave incorrecta)**
Descripción	El usuario, tras haber localizado una actividad a partir de su identificador, trata de inscribirse en ella pero introduce una clave incorrecta.
Acción	El usuario pulsa sobre el botón **inscribirse** de la actividad e introduce una clave incorrecta.
Resultado esperado	La inscripción no se realiza, se informa al usuario del error, y se le permite volver a intentarlo.
Resultado obtenido	Correcto.

Tabla 7.30: CP10-1. Inscribirse en una actividad (clave incorrecta).

7.2.4. Participación en actividades

CP11	**Comprobar ubicación del usuario al inicio**
Descripción	Cuando un participante se dispone a comenzar una actividad, la aplicación debe comprobar que se encuentra en un lugar cercano a la salida.
Acción	Estando físicamente ubicado dentro de un radio de 150m respecto a la posición de salida, desde la pantalla de actividad *en curso*, pulsar sobre el botón **comenzar**.
Resultado esperado	La aplicación reconoce que el usuario está en un lugar cercano a la posición de salida, por lo que actualiza en Cloud Firestore la hora de salida y el estado de la *Participación*.
Resultado obtenido	Casi siempre correcto. En ocasiones, el usuario está en la salida pero la aplicación no lo detecta bien. En esos casos se puede abrir una aplicación que rastree la ubicación, como Google Maps. Tras unos instantes, la precisión de las ubicaciones se va afinando. Tras hacer ésto se vuelve a intentar comenzar la actividad y ya funciona.

Tabla 7.31: CP11. Comprobar ubicación del usuario al inicio.

CP13	**Detección de paso por baliza (modo lineal)**
Descripción	Un usuario participante de una actividad se acerca a la baliza que tiene que alcanzar y la aplicación lo detecta.
Acción	Durante el transcurso de una actividad, aproximarse a la próxima baliza que haya que alcanzar.
Resultado esperado	La aplicación envía una notificación informando de que se ha alcanzado la baliza cuando el usuario se encuentra a 20m aproximadamente. En Cloud Firestore se genera un documento de PasoBaliza.
Resultado obtenido	La gran mayoría de veces es correcto. Cuando no lo es, basta con esperar unos pocos segundos al lado de la baliza.

Tabla 7.32: CP13. Detección de paso por una baliza (modo lineal).

CP13-1	**Detección de paso por baliza (modo lineal y la baliza no es la adecuada)**
Descripción	Un participante se acerca a una baliza, pero no es la que tiene que alcanzar.
Acción	Durante el transcurso de una actividad, aproximarse a una baliza que no sea la siguiente que hay que alcanzar.
Resultado esperado	La aplicación no realiza ninguna acción.
Resultado obtenido	Correcto.

Tabla 7.33: CP13-1. Detección de paso por una baliza (modo lineal y la baliza no es la adecuada).

CP14	**Detección de paso por cualquier baliza (modo *score*)**
Descripción	Un participante se acerca a una baliza (aún no alcanzada) en una actividad de tipo *score*.
Acción	Durante el transcurso de una actividad, aproximarse a una baliza no alcanzada aún.
Resultado esperado	La aplicación lo detecta cuando el usuario se encuentra a unos 20m y envía una notificación informando. En Cloud Firestore se genera un documento de PasoBaliza.
Resultado obtenido	La gran mayoría de veces es correcto. Cuando no lo es, basta con esperar unos pocos segundos al lado de la baliza.

Tabla 7.34: CP13. Detección de paso por cualquier baliza (modo *score*).

CP15	**Paso por baliza ya alcanzada**
Descripción	Un participante se acerca a una baliza que ya había alcanzado en esa misma actividad (*score* o lineal).
Acción	Durante el transcurso de una actividad, aproximarse a una baliza que ya haya sido alcanzada.
Resultado esperado	La aplicación no realiza ninguna acción.
Resultado obtenido	Correcto.

Tabla 7.35: CP15. Detección de paso por cualquier baliza (modo *score*).

CP16	**Paso por meta tras alcanzar todas las balizas**
Descripción	Un participante se aproxima a la meta después de haber alcanzado todas las balizas de la actividad.
Acción	Durante el transcurso de una actividad, aproximarse a la meta una vez alcanzadas todas las balizas.
Resultado esperado	La aplicación informa de que se ha llegado a la meta, anota el instante de finalización y cambia el estado de la Participación.
Resultado obtenido	Correcto.

Tabla 7.36: CP16. Paso por meta tras alcanzar todas las balizas.

CP16-1	**Paso por meta sin haber alcanzado todas las balizas**
Descripción	Un participante se aproxima a la meta durante la actividad sin haber alcanzado aún notas las balizas.
Acción	Durante el transcurso de una actividad, aproximarse a la meta sin haber alcanzado aún todas las balizas.
Resultado esperado	La aplicación no realiza ninguna acción.
Resultado obtenido	Correcto.

Tabla 7.37: CP16-1. Paso por meta sin haber alcanzado todas las balizas.

CP17	**Corrección de pregunta tipo test (respuesta correcta)**
Descripción	Un participante responde adecuadamente a una pregunta de tipo test.
Acción	Durante el transcurso de una actividad, el participante pulsa sobre la baliza cuyo reto va a responder, y responde de manera correcta.
Resultado esperado	La aplicación detecta que la respuesta es correcta e informa al usuario. Actualiza en Cloud Firestore el documento del *PasoBaliza*. No permite al usuario volver a responder.
Resultado obtenido	Correcto.

Tabla 7.38: CP17. Corrección de pregunta tipo test (respuesta correcta).

CP18	**Correción pregunta tipo test (respuesta equivocada)**
Descripción	Un participante responde a una pregunta de tipo test de manera incorrecta.
Acción	Durante el transcurso de una actividad, el participante pulsa sobre la baliza cuyo reto va a responder, y responde de manera incorrecta.
Resultado esperado	La aplicación detecta que la respuesta no es incorrecta. Informa al usuario y le muestra cuál es la respuesta correcta. Se actualiza el documento de *PasoBaliza* en Cloud Firestore. No permite volver a responder al usuario.
Resultado obtenido	Correcto.

Tabla 7.39: CP18. Corrección de pregunta tipo test (respuesta equivocada).

CP19	**Corrección pregunta de respuesta corta (respuesta correcta)**
Descripción	Un participante responde a una pregunta de respuesta corta de manera correcta.
Acción	Durante el transcurso de una actividad, el participante pulsa sobre la baliza cuyo reto va a responder, y responde de manera correcta.
Resultado esperado	La aplicación detecta que la respuesta es correcta e informa al usuario. Actualiza el documento de *PasoBaliza* en Cloud Firestore y no permite al usuario que vuelva a responder.
Resultado obtenido	Correcto.

Tabla 7.40: CP19. Corrección pregunta de respuesta corta (respuesta correcta).

CP20	**Corrección pregunta de respuesta corta (respuesta incorrecta)**
Descripción	Un participante responde a una pregunta de respuesta corta de manera incorrecta.
Acción	Durante el transcurso de una actividad, el participante pulsa sobre la baliza cuyo reto va a responder, y responde de manera incorrecta.
Resultado esperado	La aplicación detecta que la respuesta es incorrecta e informa al usuario. Actualiza el documento de *PasoBaliza* en Cloud Firestore y no permite al usuario que vuelva a responder.
Resultado obtenido	Correcto.

Tabla 7.41: CP20. Corrección de pregunta de respuesta corta (respuesta incorrecta).

CP21	**Abandono de una actividad**
Descripción	Un usuario abandona una actividad en curso en la que estaba participando.
Acción	Durante el transcurso de una actividad, desde la pantalla de *actividad en curso*, abrir el menú de *overflow*, pulsar en **abandonar actividad** y **confirmar**.
Resultado esperado	Se detiene el servicio en primer plano que realiza el seguimiento de la ubicación del usuario. Se informa al usuario de que a abandonado la actividad. Se actualiza en Cloud Firestore el documento de su *Participación* y se pone en modo "Terminada".
Resultado obtenido	Correcto.

Tabla 7.42: CP21. Abandono de una actividad.

7.3. Pruebas de Cloud Functions

Finalmente, en esta sección mostramos las pruebas realizadas para comprobar el funcionamiento de las Cloud Functions que habría que utilizar al poner la aplicación en producción. Para llevar a cabo estas pruebas se ha utilizado el emulador local de Firebase.

7.3.1. Sincronización de datos duplicados

En las Figuras 7.2, 7.3 y 7.4 vemos cómo las funciones se desencadenan automáticamente cuando creamos o eliminamos un documento *Participación* (un usuario se inscribe o se da de baja de una actividad).

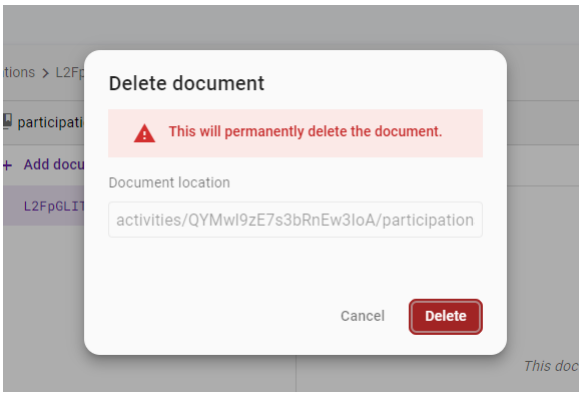

Figura 7.2: Borrado de una *Participacion* en el emulador local de Cloud Firestore.

```
i  functions: Beginning execution of "us-central1-sanitizeParticipationsOnDelete"
i  functions: Finished "us-central1-sanitizeParticipationsOnDelete" in ~1s
```

Figura 7.3: Mensaje de ejecución de la Cloud Function de borrado en el CLI de Firebase.

```
i  functions: Beginning execution of "us-central1-sanitizeParticipationsOnCreate"
i  functions: Finished "us-central1-sanitizeParticipationsOnCreate" in ~1s
```

Figura 7.4: Mensaje de ejecución de la Cloud Function de añadir *Participacion* en el CLI de Firebase.

7.3.2. "Limpiar" la base de datos

En las siguientes Figuras vemos cómo la función encargada de eliminar el documento con los datos de un usuario se desencadena automáticamente al eliminar a ese usuario del servicio de autenticación (Firebase Authentication).

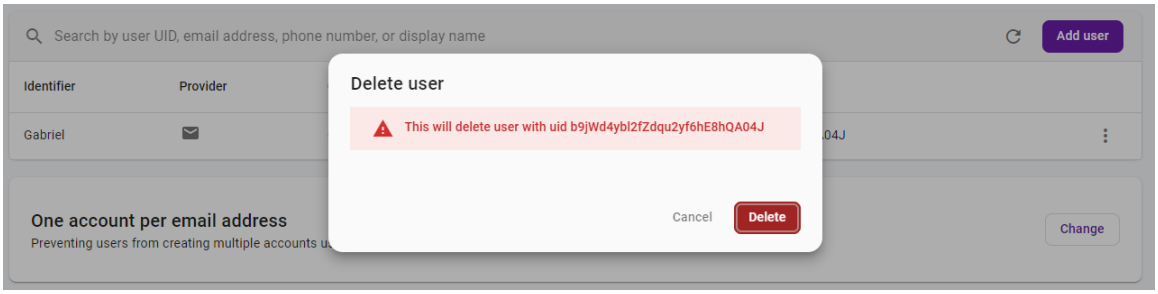

Figura 7.5: Eliminar usuario desde el emulador local de Firebase Authenticacion.

```
i  functions: Beginning execution of "us-central1-deleteUserDoc"
>  Documento eliminado
i  functions: Finished "us-central1-deleteUserDoc" in ~1s
```

Figura 7.6: Mensaje de ejecución de la Cloud Function de eliminar el documento *Usuario* en el CLI de Firebase.

Capítulo 8

Conclusiones y trabajo futuro

8.1. Conclusiones

Durante el desarrollo de este trabajo de fin de grado se ha conseguido desarrollar una aplicación móvil que permite a los usuarios programar actividades de orientación educativa en el Campo Grande de Valladolid, así como participar en ellas y revisar los resultados. Recordando los objetivos iniciales, podemos decir que éstos han sido conseguidos, y para cada uno de ellos se han alcanzado los siguientes hitos:

- *Diseñar y desarrollar una aplicación móvil mediante la que los docentes puedan programar y supervisar actividades de orientación educativa en un espacio físico concreto, y los estudiantes participar en ellas.*
  - Se ha desarrollado una aplicación móvil que cumple con los requisitos estipulados al inicio del proyecto.
  - En colaboración con un alumno y varios profesores de la Facultad de Educación, se han diseñado actividades de orientación educativa que han quedado alojadas en un *back-end* de Firebase.
- *Identificar y aplicar técnicas propias del diseño centrado en el usuario, con el fin de mejorar el diseño de la aplicación.*
  - Se han aplicado distintos métodos de investigación sobre usuarios a lo largo del proyecto: entrevista, *stakeholders meeting* y *focus group*, gracias a los cuáles nos hemos podido hacer una mejor idea de la forma en la que los usuarios utilizan la aplicación.
  - Se han realizado dos pruebas de usabilidad con usuarios, a partir de las cuáles se ha extraído conocimiento valioso para realizar mejoras en la aplicación.
  - Se han aplicado principios, heurísticas y guías de usabilidad en el desarrollo de la aplicación.
- *Estudiar posibles alternativas para hacer frente a los retos derivados de la falta de precisión de los dispositivos actuales de geolocalización.*
  - Se ha investigado sobre las distintas opciones disponibles a la hora de realizar una geolocalización precisa, considerando las ventajas e inconvenientes de cada método.
  - Se ha aplicado exitosamente una forma de geolocalizar a los participantes de las actividades en la aplicación.

Como mayor contribución, creemos que hemos aportado una primera propuesta de aplicación basada en el estudio de las necesidades y objetivos de los usuarios. Se ha alcanzado un entendimiento con los usuarios que ha dado lugar a una herramienta operativa que puede seguir siendo mejorada y ampliada en trabajo futuro. Las limitaciones del trabajo son muchas, debido al límite de tiempo. Quizá la más importante viene por el hecho de que no hemos trabajado con los estudiantes a la hora de hacer el diseño centrado en el usuario.

8.2. Trabajo futuro

A continuación se describen algunos de los aspectos en los que el sistema desarrollado podría incorporar nuevas funcionalidades o mejorar las ya existentes:

- Desarrollar aplicación que permita la creación y configuración de nuevos tipos de actividades ("plantillas").
- Incorporar las mejoras y las correcciones de prioridad "Media" y "Baja" expuestas tras la segunda prueba de usabilidad (ver 5.2.2.4).
- Incluir la opción de realizar actividades en modo "sin conexión". En este caso habría que deshabilitar ciertas funcionalidades, como las actualizaciones en tiempo real que recibe el organizador sobre la actividad de los participantes.

8.3. Valoración personal

Desde el punto de vista del aprendizaje, este proyecto ha sido muy enriquecedor. Una de mis principales motivaciones a la hora de realizar este TFG era aprender sobre usabilidad. En ese sentido, este proyecto ha supuesto la oportunidad de diseñar desde cero la interacción de los usuarios con una aplicación. Gracias a ello, he podido poner en práctica principios conocidos de usabilidad, como las heurísticas de Nielsen. También he aprendido mucho sobre el diseño de interfaces de usuario, gracias a haber trabajado siguiendo las guías de Material Design. En el aspecto metodológico, el proyecto ha sido de gran utilidad para practicar técnicas de diseño centrado en el usuario. El hecho de haber realizado iteraciones con usuarios para hacer evolucionar el diseño ha sido un aprendizaje muy valioso. El TFG también me ha servido para profundizar en conceptos sobre Android, ya que, previamente, casi no tenía experiencia en el desarrollo de aplicaciones móviles. Por último, este proyecto me ha servido para aprender sobre análisis y diseño de bases de datos NoSQL, como Cloud Firestore.

Bibliografía

[1] Android Platform. *Services overview*. Computer Software. Android. URL: https://developer.android.com/guide/components/services#Types-of-services (visitado 07-07-2021).

[2] Atlassian Team. *Gitflow Workflow*. URL: https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow (visitado 27-06-2021).

[3] Nick Babich. *Why Focus Groups Benefit the Web Design Process*. 30 de jul. de 2020. URL: https://xd.adobe.com/ideas/process/user-research/why-focus-groups-benefit-web-design-process/ (visitado 07-06-2021).

[4] Barry Boehm. «Software risk management: principles and practices». En: *IEEE software* 8 (1 ene. de 1991), págs. 32-41.

[5] Raluca Budiu. *Interaction cost: Definition*. 31 de ago. de 2013. URL: https://www.nngroup.com/articles/interaction-cost-definition/ (visitado 28-04-2021).

[6] Carlos Cabello. *9 usos reales para comprender qué son los "beacons"*. 7 de jul. de 2016. URL: https://www.nobbot.com/redes/9-usos-reales-comprender-los-beacons/ (visitado 28-04-2021).

[7] Juan Manuel Casado Mora. *Deporte de Orientación*. 1.ª ed. Club de Orientación Veleta. 2009. URL: http://www.criptanavertical.com/ORIENTACION/I%5C%20CARRERA%5C%20ORIENTACION%5C%20SAN%5C%20GREGORIO/Manual%5C%20Orientacion.pdf.

[8] Claire Drumond. *Agile Project Management*. URL: https://www.atlassian.com/agile/project-management (visitado 25-05-2021).

[9] Juan Carlos Escaravajal Rodríguez y Antonio Extremera. «Las Aplicaciones Tecnológicas en el Deporte de Orientación y en Educación Física». En: *Habilidad Motriz: revista de ciencias de la actividad física y del deporte* 53 (oct. de 2019), págs. 28-40.

[10] Patrick Faller. *Putting Personas to Work in UX Design: What Are They and Why They're Important?* 17 de dic. de 2019. URL: https://xd.adobe.com/ideas/process/user-research/putting-personas-to-work-in-ux-design/ (visitado 30-05-2021).

[11] Firebase Team. *Cloud Firestore*. Computer Software. Firebase. URL: https://firebase.google.com/docs/firestore (visitado 17-06-2021).

[12] Vanesa Gallego Lema y col. «La orientación en el medio natural: aprendizaje ubicuo mediante el uso de tecnología». En: 2.23 (2017), págs. 755-770.

[13] Ana González Marcos y col. *Técnicas y Algoritmos Básicos de Visión Artificial*. Universidad de La Rioja, 2006.

[14] GPS.gov. *El sistema de posicionamiento global*. URL: https://www.gps.gov/systems/gps/spanish.php (visitado 28-04-2021).

[15] Bob Hughes y Mike Cotterell. *Software Project Management*. 5.ª ed. McGraw-Hill Education, 2009.

[16] Ditte Hvas Mortensen. *User Research: What It Is and Why You Should Do It?* Ene. de 2021. URL: https://www.interaction-design.org/literature/article/user-research-what-it-is-and-why-you-should-do-it (visitado 07-06-2021).

[17] Interaction Design Foundation. *User Centered Design*. URL: https://www.interaction-design.org/literature/topics/user-centered-design (visitado 20-05-2021).

[18] ISO. *Human-centered design processes for interactive systems*. en. Standard ISO 13407:1999. Jun. de 1999. URL: https://www.iso.org/standard/21197.html.

[19] ISO/IEC. *Information technology – Automatic identification and data capture techniques – QR Code 2005 bar code symbology specification*. en. Standard ISO/IEC 18004:2006. 2006. URL: https://www.iso.org/standard/43655.html.

[20] NFC Forum. *NFC Technology*. URL: https://nfc-forum.org/what-is-nfc/about-the-technology/ (visitado 28-04-2021).

[21] Jakob Nielsen. *10 Usability Heuristics for User Interface Design*. URL: https://www.nngroup.com/articles/ten-usability-heuristics/ (visitado 05-06-2021).

[22] Ekaterina Novoseltseva. *User-Centered Design: An Introduction*. URL: https://usabilitygeek.com/user-centered-design-introduction/ (visitado 07-06-2021).

[23] Kara Pernice. *User Interviews: How, When and Why to Conduct Them*. 7 de oct. de 2018. URL: https://www.nngroup.com/articles/user-interviews/ (visitado 07-06-2021).

[24] Joe Ritmeyer. *Incremental UX*. 29 de sep. de 2015. URL: https://uxdesign.cc/incremental-ux-62aa1283b105 (visitado 22-04-2021).

[25] Christian Rohrer. *When to Use Which User-Experience Research Methods*. 12 de oct. de 2014. URL: https://www.nngroup.com/articles/which-ux-research-methods/ (visitado 03-06-2021).

[26] Debbie Stone y col. *User Interface Design and Evaluation*. Morgan Kaufmann, 2005. Cap. 2.

[27] David Travis. *The 1-page usability test plan*. 2 de sep. de 2013. URL: https://www.userfocus.co.uk/articles/usability_test_plan_dashboard.html (visitado 07-06-2021).

[28] David Travis. *The Fable of the User Centered Designer*. 2010. URL: https://www.userfocus.co.uk/fable/ (visitado 30-05-2021).

[29] Tom Tullis y Bill Albert. *Measuring the User Experience*. 2.ª ed. Morgan Kaufmann, 2013.

[30] Usability.gov. *Kick-Off meeting*. URL: https://www.usability.gov/how-to-and-tools/methods/kick-off-meeting.html (visitado 06-06-2021).

[31] Visual Paradigm Team. *Firebase*. Computer Software. Visual Paradigm. URL: https://online.visual-paradigm.com/diagrams/templates/google-cloud-platform-diagram/firebase/# (visitado 03-07-2021).

Apéndice A

Manual de usuario

A continuación se explica el funcionamiento de la aplicación desde el punto de vista de los usuarios. Mediante capturas de pantalla, se mostrarán las principales características de la aplicación y el flujo normal de acciones que los usuarios deben tomar para completar las tareas principales.

A.1. Inicio de sesión y registro

En esta aplicación los usuarios se identifican mediante e-mail y contraseña. Para que un usuario pueda utilizar la aplicación, no basta con que se registre, sino que también tendrá que verificar su e-mail. Para ello, se le enviará un correo con un *link* desde el que puede verificar su dirección, como el que se muestra en la Figura A.2. Análogamente, también podrá recuperar su contraseña en caso de olvido mediante un mecanismo similar, pulsando en **restablecer contraseña** (Figura A.1).

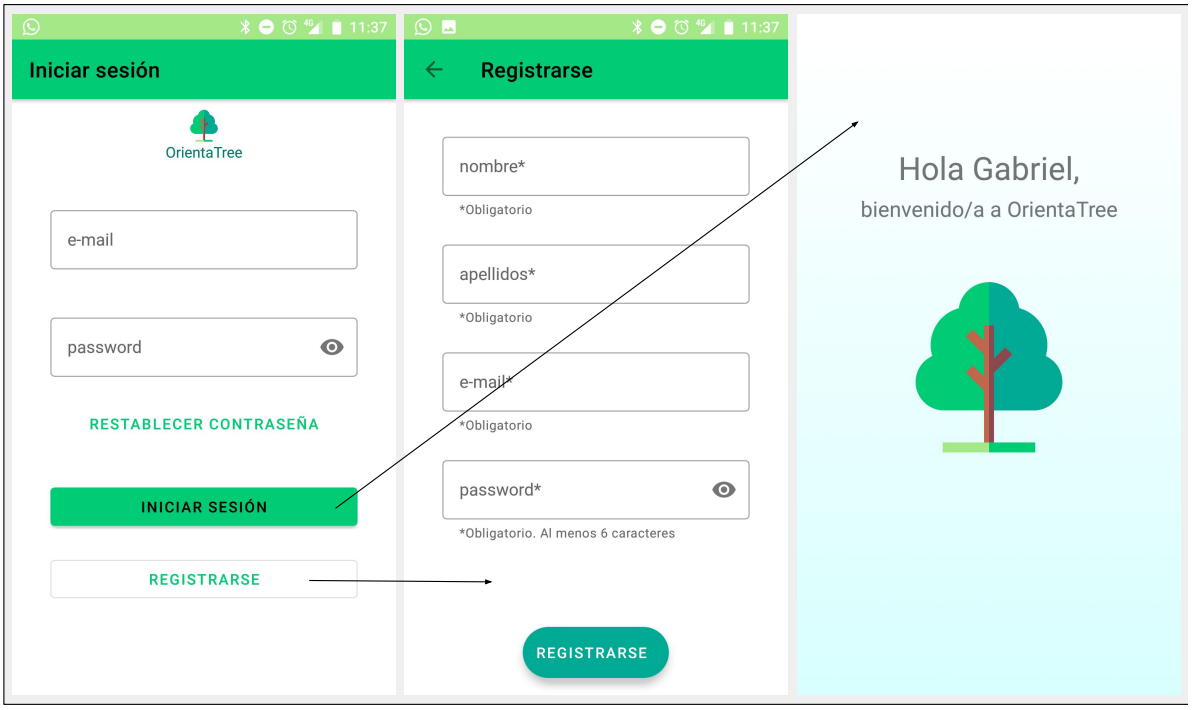

Figura A.1: Pantallas de *login*, registro y bienvenida.

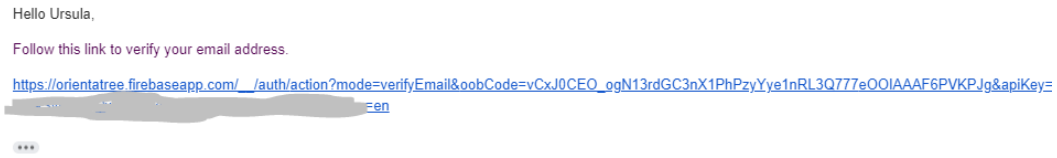

Figura A.2: Correo de verificación.

A.2. Pantalla de inicio

A continuación se muestra la pantalla de **Mis actividades** (Figura A.3), que actúa como pantalla de inicio, al ser lo que se muestra al usuario nada más iniciar sesión. En ella el usuario puede observar las actividades en las que toma parte (ya sea como organizador o como participante). En la imagen vemos que esta pantalla cuenta con un menú desde el que se abre un *Navigation Drawer*, el cual nos permite *navegar* al resto de pantallas principales de la aplicación.

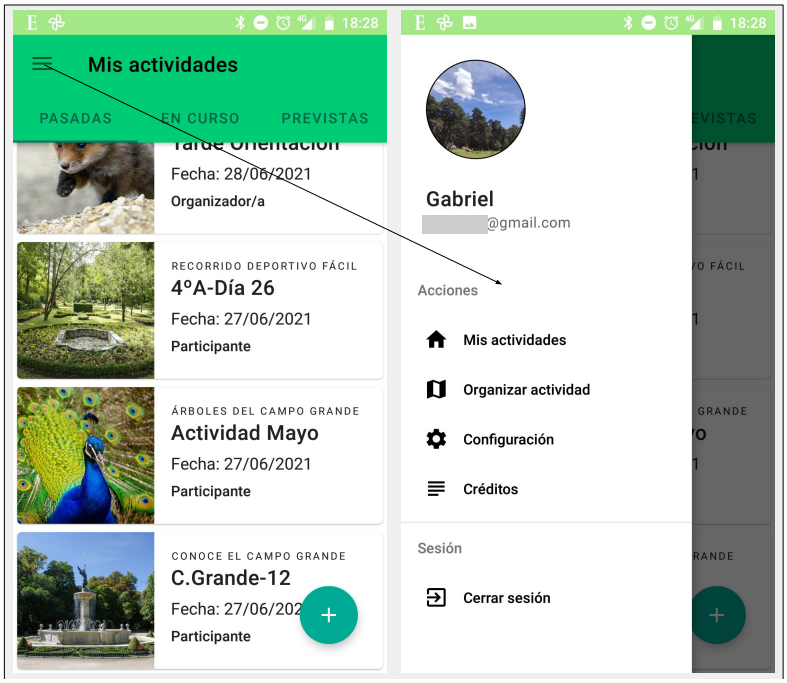

Figura A.3: Pantalla de inicio y *Navigation Drawer*.

A.3. Configuración de perfil

En la Figura A.4 vemos cómo podemos utilizar el *Navigation Drawer* para ir desde la pantalla de inicio hasta la pantalla de **Configuración de perfil**. En esta pantalla, un usuario puede modificar sus datos personales así como su foto de perfil. Por último, en la esquina superior derecha dispone de un *overflow* menú. Ese menú, al desplegarse, muestra las acciones relacionadas con la gestión de la cuenta de usuario, tales como **cerrar sesión** o **eliminar cuenta**.

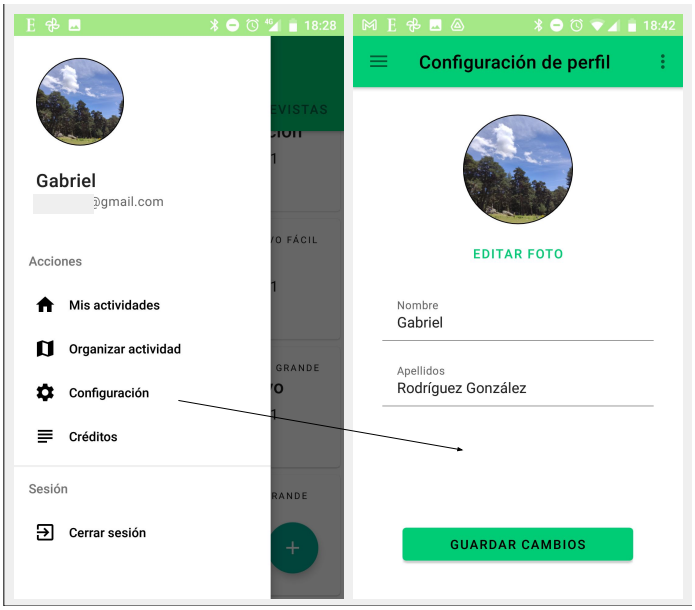

Figura A.4: Configuración de perfil (accesible desde *Navigation Drawer*).

A.4. Organizar actividad

Desde el *Navigation Drawer*, también podemos acceder al área de organización de nuevas actividades. En la Figura A.5 vemos cómo desde la pantalla **Organizar actividad** podemos seleccionar una plantilla a partir de la cual programar la nueva actividad. Las plantillas están protegidas mediante una contraseña, para asegurar que los participantes no puedan acceder a su información y ver el mapa de las actividades antes de comenzar su participación. Desde esta pantalla nos dirigimos a **Programar actividad**, dónde encontramos la información relacionada con la plantilla. Además, desde esta pantalla también podemos programar una nueva actividad a partir de la plantilla correspondiente, como se muestra en las Figuras A.6 y A.7.

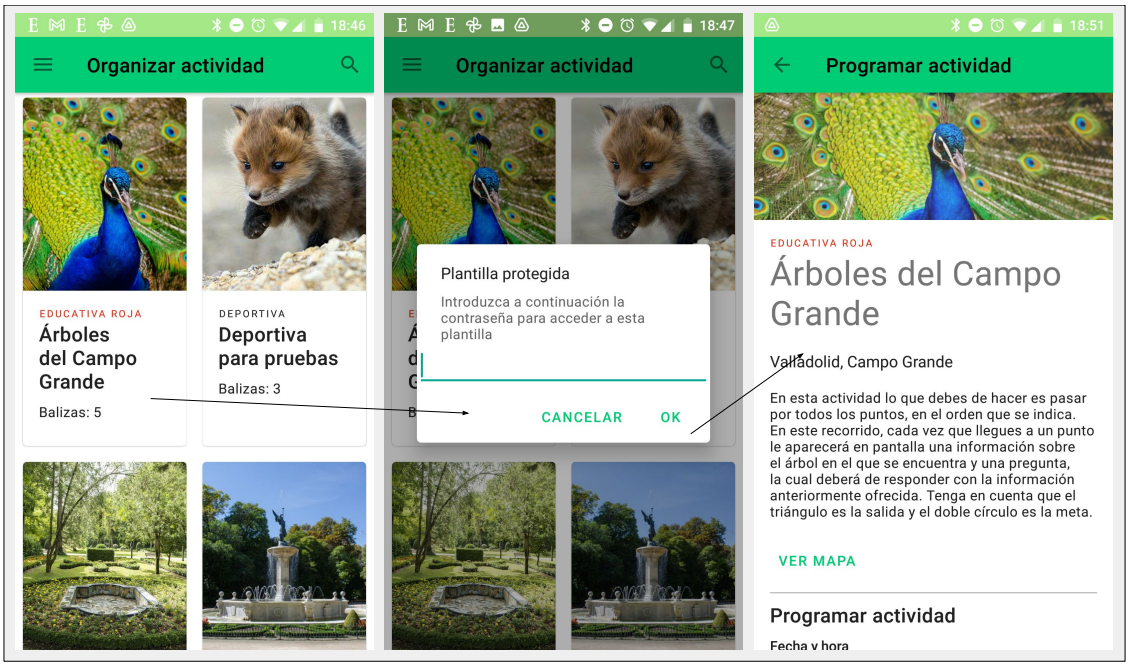

Figura A.5: Página de selección de plantilla de actividad (accesible desde *Navigation Drawer*) y plantilla seleccionada.

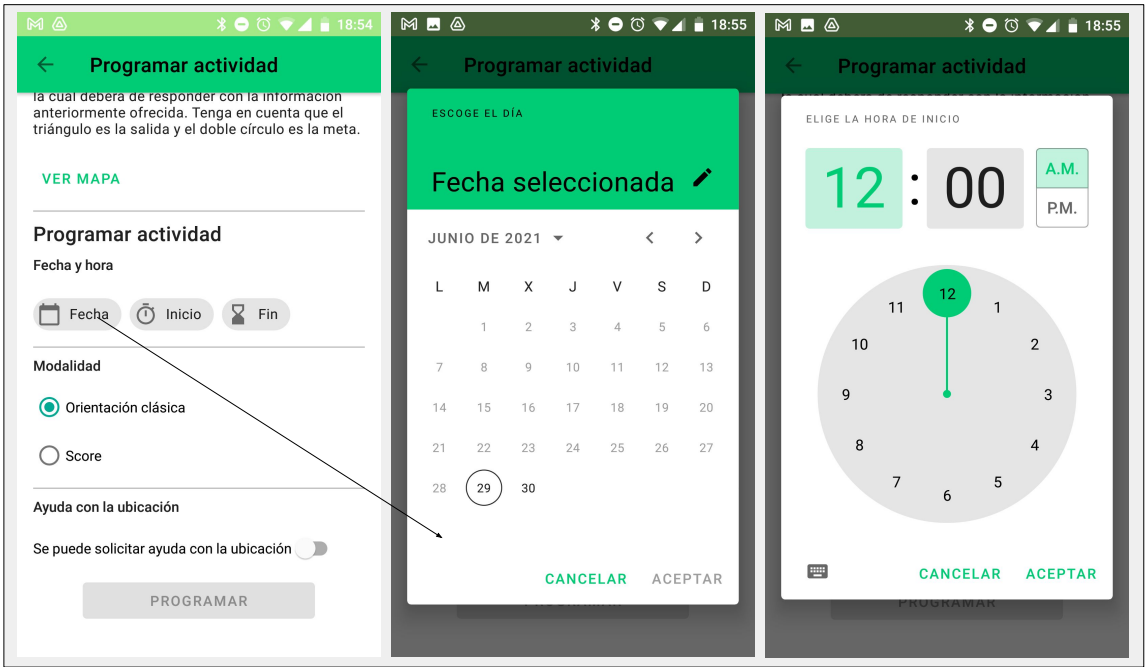

Figura A.6: Programación de la actividad. Elección de fecha y hora.

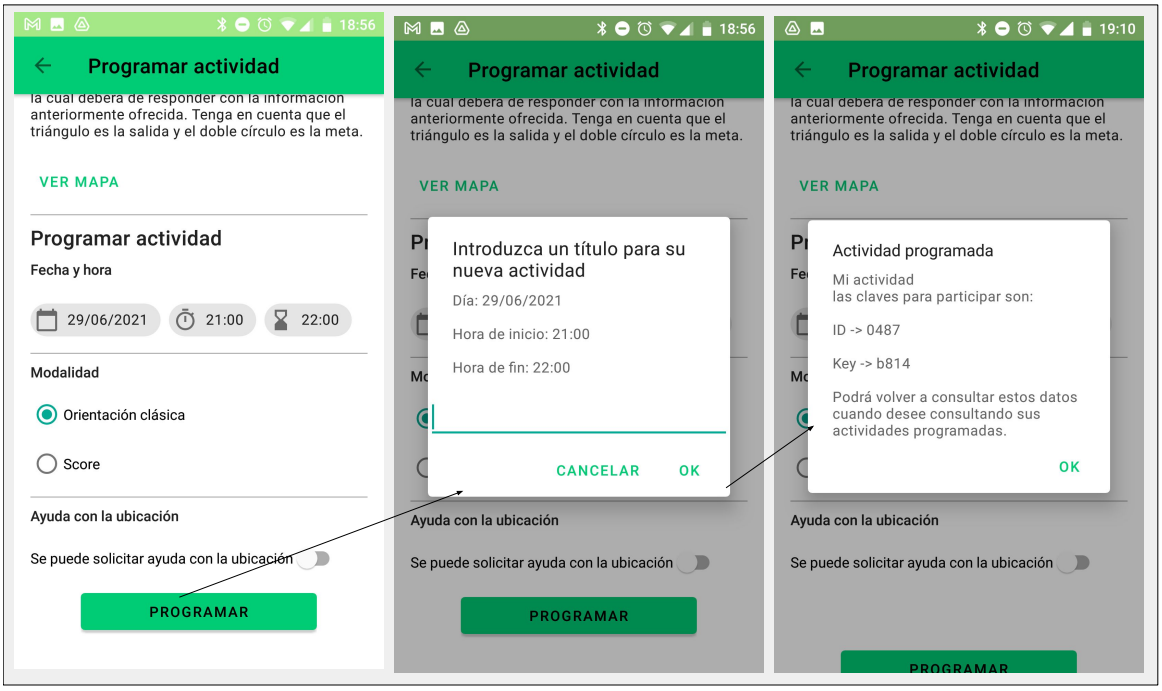

Figura A.7: Confirmación de la actividad programada. Claves de acceso generadas.

Finalmente, si volvemos a la pantalla de inicio y seleccionamos la etiqueta **previstas**, podremos ver la nueva actividad que hemos programado. Al pulsar sobre ella se nos mostrará la información que vemos en la Figura A.8.

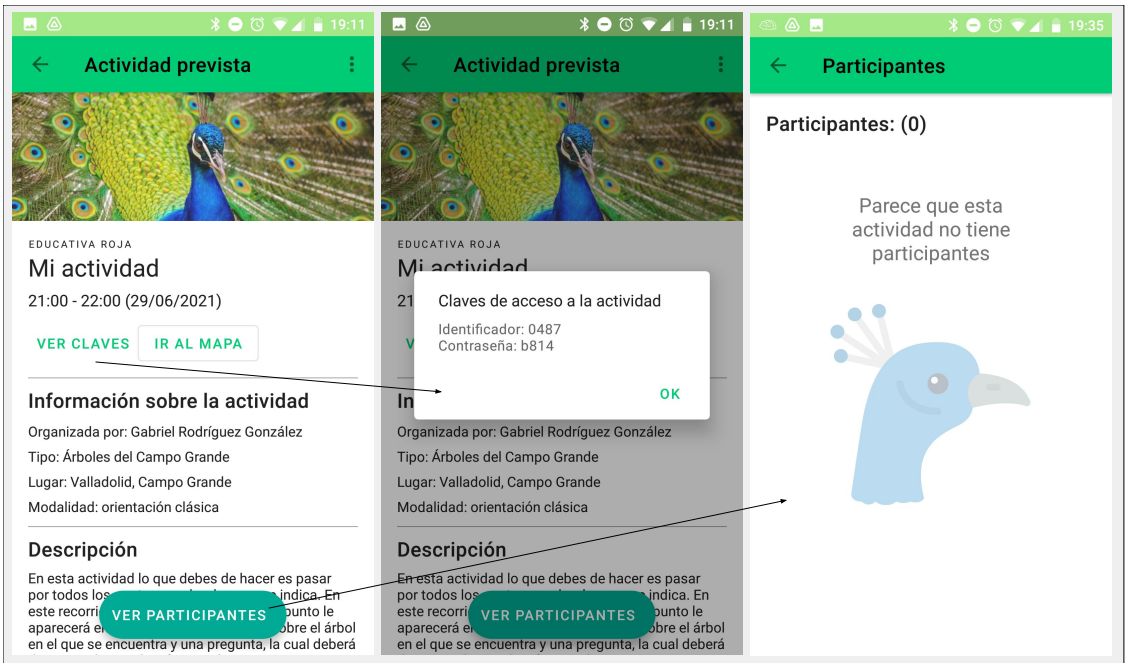

Figura A.8: Actividad programada, visualización de claves y *Empty State* indicando que aún no hay participantes inscritos.
(Icono de *Empty State* realizado por Smashicons y obtenido de www.flaticon.com)

A.5. Inscribirse en una actividad

A continuación se muestran los pasos que tiene que dar un usuario para inscribirse en una actividad como participante. Esta tarea tiene como *pre-condición* que el organizador de la actividad debe haber proporcionado al usuario las claves de acceso a la actividad (como las que se muestran en las Figuras A.7 y A.8). Una vez que el usuario conozca las claves, podrá buscar la actividad mediante la primera de ellas y completar su inscripción mediante la segunda, tal y como se muestra en las Figuras A.9 y A.10.

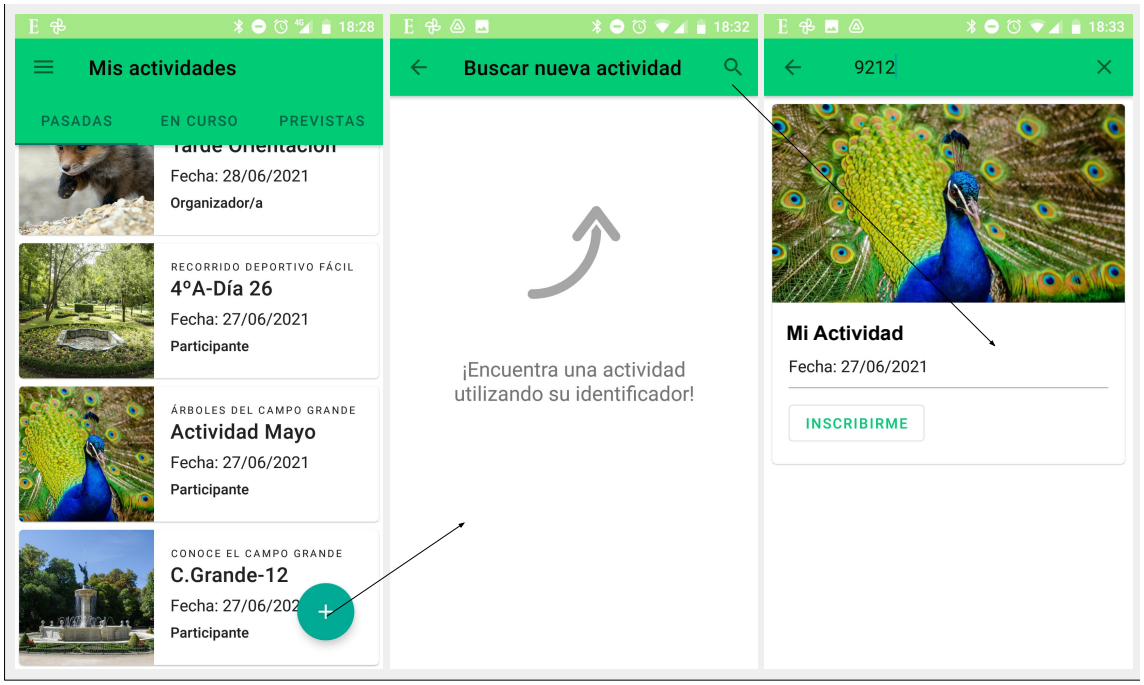

Figura A.9: Acceso mediante el *Floating Action Button* a la página de búsqueda de actividades. (La flecha hacia arriba es un icono realizado por Iconixar y obtenido de www.flaticon.com)

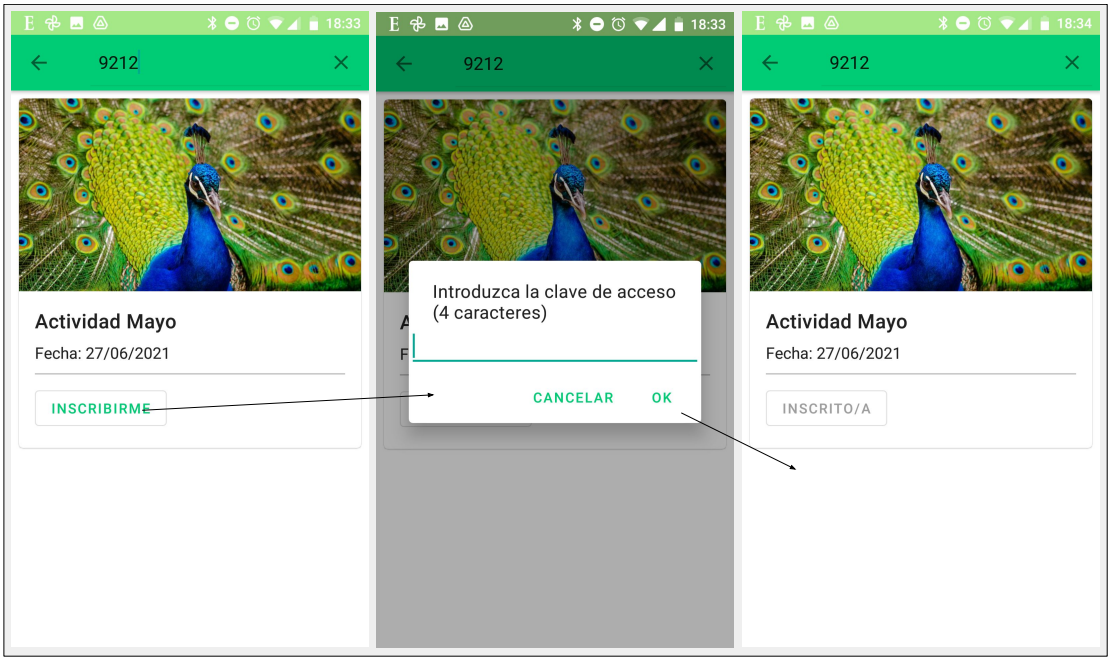

Figura A.10: Inscripción en una actividad.

A.6. Participar en una actividad

En este apartado veremos cómo pueden los usuarios realizar las acciones propias del transcurso de una actividad. En primer lugar, desde la pantalla de inicio, deben situarse en la etiqueta de actividades *en curso* y pulsar sobre aquella en la que quieran comenzar a participar. En la Figura A.11 vemos cómo basta con pulsar el botón **comenzar** para iniciar la participación en la actividad. Una vez que hemos comenzado, se nos mostrará el mapa y el cronómetro

empezará a correr. Como puede observarse, en la esquina inferior derecha de la pantalla **Mapa de la actividad** hay un botón llamado **balizas**, desde el que podemos acceder a las acciones que se muestran en la Figura A.12.

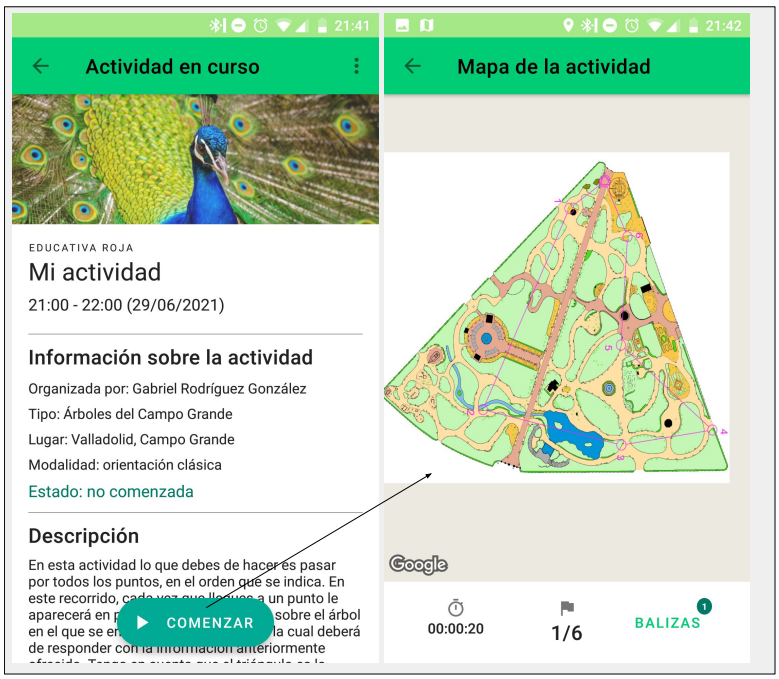

Figura A.11: Iniciar participación en una actividad.

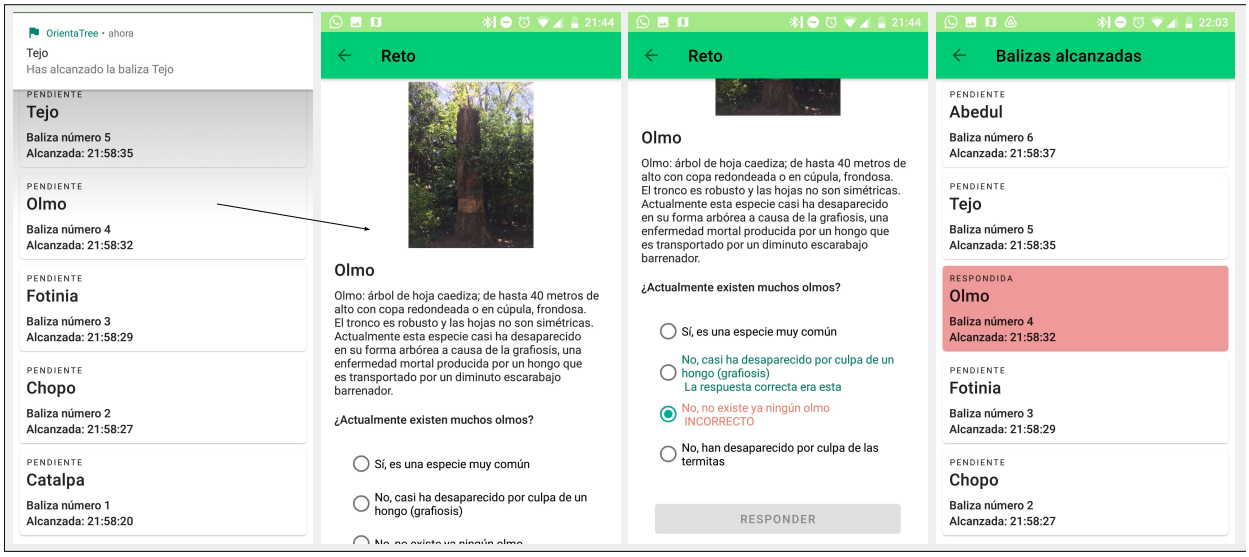

Figura A.12: Responder al reto de una baliza.

En la Figura A.13 vemos cómo el usuario puede visualizar los datos de su participación desde la pantalla **Mi participación**. Además, desde ésta pantalla puede volver a ver la información sobre las balizas y los retos que se mostraba en la Figura A.12, y también puede ir a la pantalla de **Track** desde la que visualiza el recorrido seguido durante la actividad.

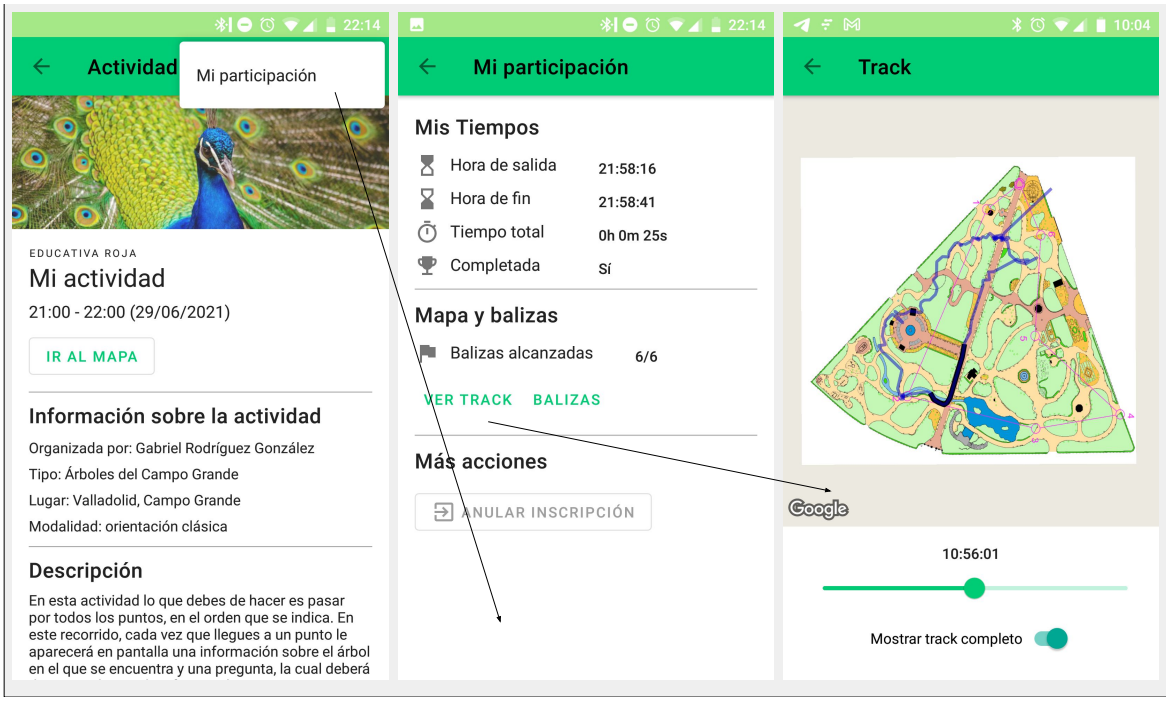

Figura A.13: Visualización de los detalles de *Mi Participación*.

A.7. Acciones del organizador

Veamos a continuación las principales acciones que puede realizar el organizador de una actividad mediante la aplicación. En primer lugar, en la Figura A.14 observamos las pantallas desde las que el organizador puede monitorizar en tiempo real las acciones que realizan los participantes durante una actividad. En la Figura A.15 vemos cómo el organizador puede observar la información sobre una actividad terminada. En el caso de la actividad de la imagen, se trataba de una actividad de tipo "deportiva", por lo que se puede ver una clasificación de los participantes de acuerdo a sus tiempos de finalización.

Figura A.14: Pantallas de monitorización de la actividad.

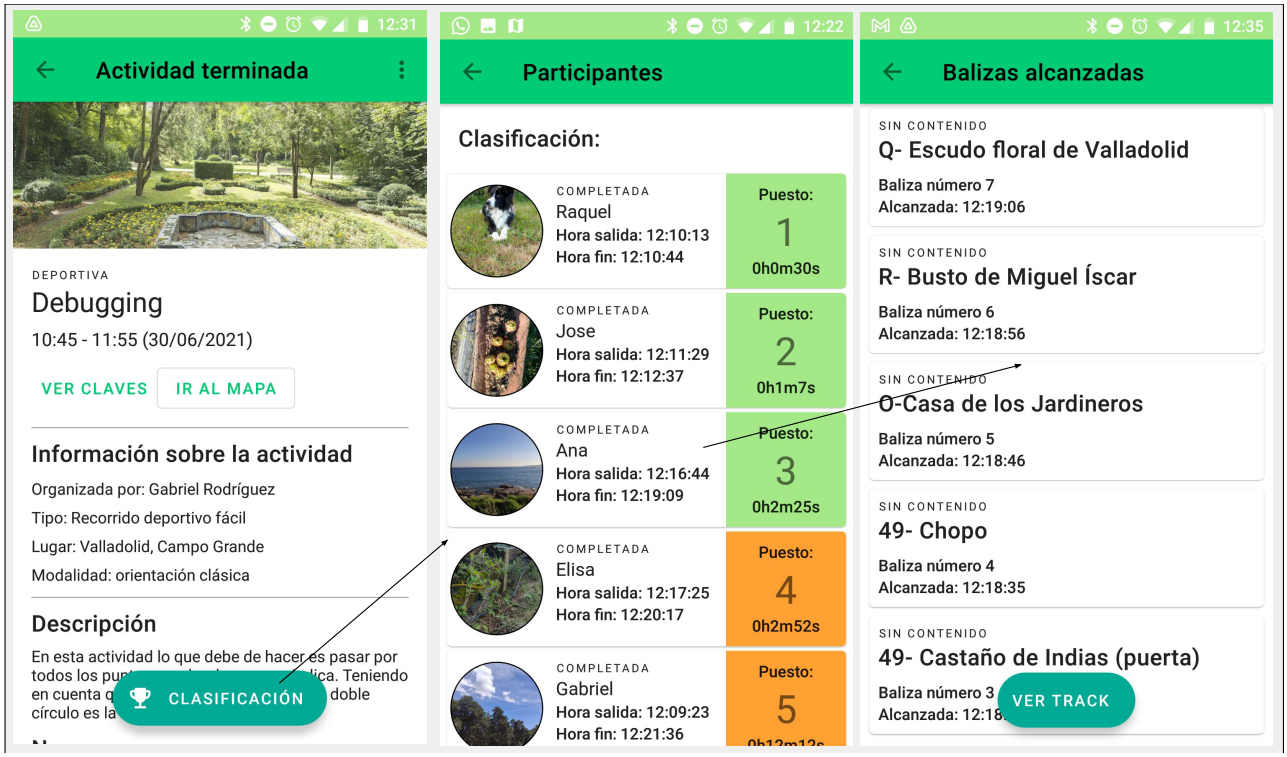

Figura A.15: Información sobre una actividad terminada.

Apéndice B

Plantillas de actividades

En este apéndice se describen las plantillas de actividades creadas a partir de la colaboración con dos profesores y un alumno de la Facultad de Educación y Trabajo Social.

B.1. Actividad de tipo educativa roja

Las plantillas de tipo "educativa roja" muestran a los usuarios preguntas de tipo test a su paso por las balizas. Esta plantilla tiene seis balizas cuyo contenido se muestra en la Tabla B.1.

Figura B.1: Mapa de la plantilla educativa roja.

Baliza	**Texto y pregunta**	**Respuesta**
1. Catalpa	Árbol de hoja caduca de hasta 20 metro de altura y copa ancha. Corteza gris con toques rojizos que se resquebraja. Sus hojas son muy grandes, en forma de corazón. Sus flores son blancas, con motas encarnadas y bandas amarillas en su interior, grandes y con forma de trompeta. El fruto es muy característico: muy alargado, estrecho y cilíndrico, de color verde al principio y después marrón. ¿Cómo es el fruto de este árbol?	Muy alargado, estrecho y cilíndrico
2. Chopo	Árbol de hoja caduca, puede llegar a los 40 metros de altura. Este árbol crece en zonas húmedas y se caracteriza porque tiene las hojas con forma romboidal a casi triangular y borde aserrado de color verde, las ramas son muy flexibles y la corteza es lisa. Un uso muy habitual de este árbol es para hacer pasta de papel. ¿Cómo son las ramas del chopo?	Flexibles
3. Fotinia	Arbolito de hoja perenne de hasta 7 metros de altura con copa amplia. El tronco es corto, ramificado y con corteza lisa, rojiza o marrón; parda en la madurez. Hojas ovaladas, verde oscuras con el borde irregular y del tamaño de una cuarta. Flores blancas semejantes a las del endrino que desprenden un agradable aroma. Los frutos son rojos y carnosos. ¿Qué forma tienen las hojas de la Fotinia?	Ovaladas
4. Olmo	Árbol de hoja caediza; de hasta 40 metros de alto con copa redondeada o en cúpula, frondosa. El tronco es robusto y las hojas no son simétricas. Actualmente esta especie casi ha desaparecido en su forma arbórea a causa de la grafiosis, una enfermedad mortal producida por un hongo que es transportado por un diminuto escarabajo barrenador. ¿Actualmente existen muchos olmos?	No, casi ha desaparecido por culpa de un hongo (grafiosis)
5. Tejo	Árbol de hoja perenne; de hasta 20 metros pero muchas veces tiene porte arbustivo en los ejemplares de jardinería. Hojas planas y estrechas, de hasta 3 centímetros, dispuestas en dos filas horizontales opuestas, una a cada lado de la rama. Sólo los ejemplares femeninos producen fruto. Se trata de una semilla rodeada de una cubierta roja muy llamativa que se denomina arilo. Toda la planta es venenosísima a excepción de la parte carnosa roja del fruto (arilo). Del Tejo se ha extraído taxol, empleado en tratamientos de quimioterapia contra el cáncer. ¿Para qué enfermedad se utiliza un extracto de esta planta en su tratamiento?	Cáncer
6. Abedul	Árbol de hoja caduca de hasta 30 metros de alto y con el tronco recto y ramificado. La corteza es muy blanca y distintiva; se agrieta y desprende en bandas horizontales. Sus hojas son romboidales o triangulares, con el borde doblemente dentado. Llaman la atención las flores masculinas, que agrupadas en pequeños cilindros cuelgan a principios de primavera. ¿De qué color es la corteza del Abedul?	Blanca

Tabla B.1: Actividad educativa roja

B.2. Actividad de tipo educativa naranja

Las plantillas de tipo "educativa naranja" muestran a los usuarios preguntas de respuesta corta a su paso por las balizas. Esta plantilla consta de cinco balizas:

Baliza	**Texto y pregunta**	**Respuesta**
1. Fotógrafo	Escultura a escala humana, obra del escultor vallisoletanos Eduardo Cuadro. Es un homenaje a Vicente Muñoz, quien fue fotógrafo oficial del Campo Grande de Valladolid durante más de medio siglo. ¿En qué año se colocó la estatua? (En número)	1994
2. Fuente del Cisne	La Fuente del Cisne, en la zona llamada "La Pérgola" del parque Campo Grande de Valladolid, fue proyectada por Gonzalo Bayón en 1887. El pilón circular está adornado con escudos de Valladolid y una inscripción que recuerda el año de su construcción. En su centro se alza un macizo del que surgen seis náyades, que sostienen en sendas manos peces que parecen lanzar al agua. La composición está coronada por la figura de un cisne con sus alas extendidas. Dentro del estanque y a su alrededor hay tritones que arrojan agua al conjunto central. ¿Cuántos peces se encuentran en la fuente? (En número)	6
3. Lago	Creado en 1879 por Ramón Oliva. El estanque está formado por dos islas y una montaña artificial imitando a una gruta de la que caía una cascada. En el estanque hay mucha vida animal, entre los que encontramos cisnes, patos, peces y reptiles. ¿Cuál era el mote del barquero?	El Catarro
4. Fuente de la fama	Fuente monumental proyectada por Antonio Iturralde y del escultor Mariano Chicote, erigida en 1883 en el centro de los jardines del Campo Grande, como homenaje a uno de los alcaldes de mayor trascendencia para la historia de Valladolid. ¿A quién está dedicada la fuente?	Miguel Íscar
5. Pajarera	Construida en los años treinta del siglo pasado. Tiene un diseño que adquiere un aire oriental de concepción modernista y se ha convertido en el epicentro de la vida animal que habita en el parque. Observando la pantalla que se encuentra en un lateral de la pajarera, nombra un ave que empiece por "P" y se encuentre en la pajarera del Campo Grande.	periquito

Tabla B.2: Actividad educativa naranja

Figura B.2: Mapa de la plantilla educativa naranja.

B.3. Actividad de tipo deportiva

Las plantillas de tipo “deportiva” no muestran ningún reto a los participantes al alcanzar las balizas. Tan solo tienen que alcanzarlas en el menor tiempo posible, como en una carrera de orientación normal. La plantilla de tipo “deportiva” que hemos creado consta de siete balizas:

1. Poeta L. Cano
2. Ciprés
3. Castaño de indias (puerta)
4. Chopo
5. Casa de los jardineros
6. Busto de Miguel Íscar
7. Escudo floral de Valladolid

Figura B.3: Mapa de la plantilla deportiva.

Apéndice C

Manual de instalación

Para instalar la aplicación se necesita disponer del archivo ejecutable (.apk) descargado en un dispositivo Android, cuya versión no debe ser inferior a Android 8. Luego basta con seguir los pasos que se describen a continuación en la Figura C.1.

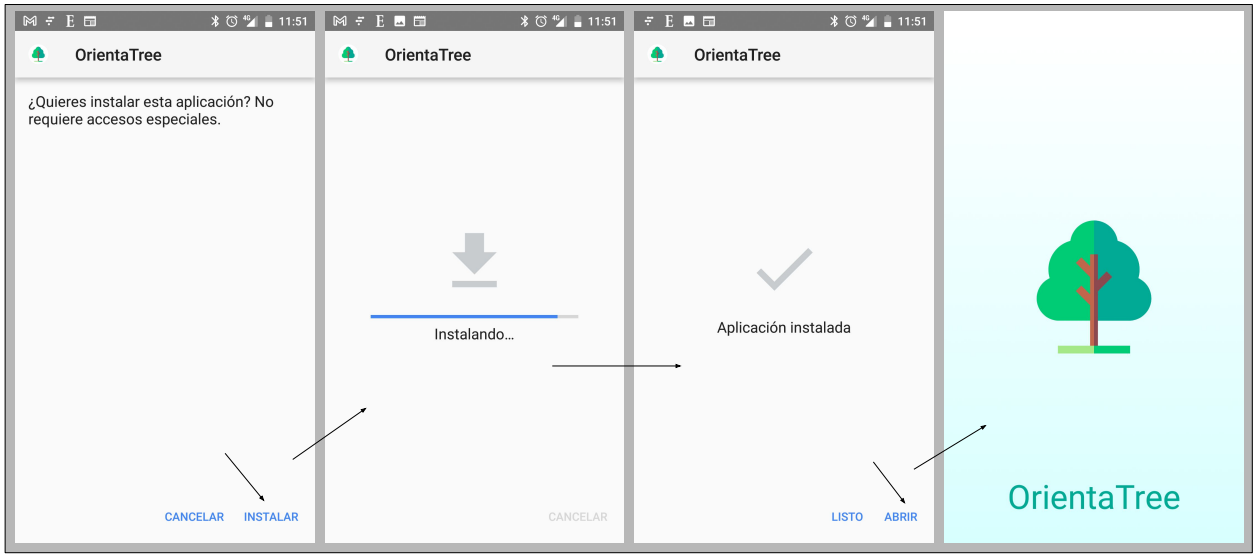

Figura C.1: Pasos para instalar la aplicación.

A la hora de descargar el archivo *apk* puede que haya que aceptar una autorización como la que aparece en la Figura C.2.

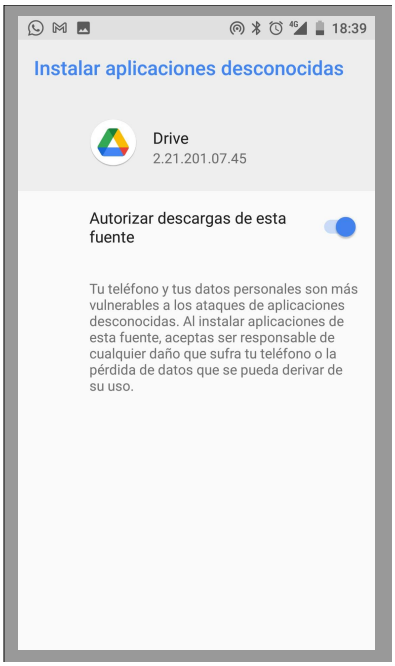

Figura C.2: Autorizar la descarga de la aplicación.

Apéndice D

Enlaces de interés

- Repositorio Github con el código de la aplicación:
  `https://github.com/GabrielRGBQ/OrientaTree.git`
- Descarga del archivo .apk de instalación de la aplicación:
  `https://drive.google.com/file/d/1Gzy9ZDzpk1bDVOXFDGDDVt03ui92BAbw/view?usp=sharing`