



Book4Book: Desarrollo de aplicación Android para el intercambio de libros entre usuarios

Javier Sevillano Jiménez

Máster Universitario en Desarrollo de Aplicaciones para Dispositivos Móviles

Pau Dominkovics Coll

Carles Garrigues Olivella

Junio de 2018



Esta obra está sujeta a una licencia de Reconocimiento-NoComercial-SinObraDerivada [3.0 España de Creative Commons](https://creativecommons.org/licenses/by-nc-nd/3.0/es/)

FICHA DEL TRABAJO FINAL

Título del trabajo:	<i>Book4Book: Desarrollo de aplicación Android para el intercambio de libros entre usuarios</i>
Nombre del autor:	<i>Javier Sevillano Jiménez</i>
Nombre del consultor:	<i>Pau Dominkovics Coll Carles Garrigues Olivella</i>
Fecha de entrega:	<i>06/2018</i>
Titulación:	<i>Máster Universitario en Desarrollo de Aplicaciones para Dispositivos Móviles</i>
Resumen del Trabajo:	
<p>La lectura continúa gozando de buena salud en España, y a pesar de la existencia de los libros electrónicos, los lectores siguen prefiriendo el soporte impreso para disfrutar de su afición. Con el tiempo, muchos de esos libros quedan olvidados en cajas o estanterías, y con mucha probabilidad no se vuelven a leer, ocupando espacio innecesariamente, sin ser utilizados. El presente trabajo pretende mitigar este problema, mediante una herramienta que ayuda a poner en contacto a los aficionados a la lectura y les permite intercambiar sus libros usados, dándoles así nuevos usos y un mayor aprovechamiento.</p> <p>El trabajo tiene como objetivo cubrir esta necesidad mediante el desarrollo de una aplicación para el sistema operativo Android, denominada Book4Book, que permite compartir libros propios y buscar los que comparten otras personas mediante diversos criterios, como título, autor, género o distancia con el propietario, utilizando para ello la localización del dispositivo, e intercambiarlos.</p> <p>El desarrollo de la aplicación se ha realizado de forma nativa para la plataforma Android, utilizando Backendless como servicio de <i>backend</i> en la nube para implementar características como la persistencia de datos, la gestión de usuarios o el envío de correos electrónicos.</p> <p>La metodología aplicada ha sido el modelo en cascada, dado que los requisitos se han fijado claramente desde el inicio, y se ha empleado la filosofía de diseño centrado en el usuario para dar respuesta a las necesidades reales de los mismos, obteniéndose como resultado un producto completamente funcional que satisface los objetivos definidos al comienzo del trabajo.</p>	

Abstract:

Reading is in good health in Spain, and despite electronic books, readers still prefer printed format to enjoy their hobby. Over time, many of these books are forgotten in boxes or shelves, not being read again anymore and occupying space unnecessarily, without being used. This project aims to mitigate this problem with a tool that connects readers and allows them to swap their used books, to give them new uses and a better harnessing.

The project aims to meet this need by developing a mobile application for the Android operating system, called Book4Book, which allows users to share their own books and find the ones shared by other people through several criteria, such as title, author, gender or distance to the book owner, using the location of the device, and swap them.

The application has been developed natively for the Android platform, using Backendless as a *BaaS* (Backend as a Service) to implement features such as data persistence, user management or email sending.

The waterfall model was applied as the methodology for executing the project, since the requirements were clearly established during its early stages, and the user-centered design process has been used to meet the real needs of the users, resulting in a fully functional product that satisfies the objectives defined at the beginning of the project.

Palabras clave:

Intercambio, libro, permuta, mapa, Android.

Índice

1. Introducción.....	1
1.1 Contexto y justificación del Trabajo.....	1
1.2 Objetivos del Trabajo.....	5
1.3 Enfoque y método seguido.....	8
1.4 Planificación del Trabajo.....	11
1.5 Breve resumen de productos obtenidos.....	17
1.6 Breve descripción de los otros capítulos de la memoria.....	17
2. Análisis y diseño.....	18
2.1 Definición del público objetivo.....	18
2.2 Encuesta.....	21
2.3 Definición de personas y escenarios.....	22
2.4 Árbol de navegación.....	24
2.5 Prototipado.....	25
2.5.1 Pantalla de presentación (splash).....	26
2.5.2 Autenticación y registro de usuarios.....	26
2.5.3 Usuarios cercanos y búsqueda de libros.....	27
2.5.4 Libros compartidos por el usuario.....	31
2.5.5 Intercambios.....	32
2.5.6 Línea de diseño y prototipos de alta definición.....	35
2.6 Evaluación.....	36
2.7 Definición de casos de uso.....	37
2.8 Diseño de la arquitectura.....	44
2.8.1 Cliente-servidor.....	44
2.8.2 Modelo-Vista-Presentador.....	45
2.8.3 Entidades y base de datos.....	47
3. Implementación.....	51
3.1 Arquitectura.....	51
3.2 Herramientas y APIs utilizadas.....	52
3.2.1 Backendless API.....	52
3.2.2 OpenLibrary API.....	54
3.2.3 Retrofit.....	54
3.2.4 ZXing Android Embedded.....	54
3.2.5 Picasso.....	54
3.2.6 Google Maps API.....	55
3.2.7 Google Places API.....	55
3.2.8 Espresso.....	55
3.3 Aspectos relevantes del desarrollo.....	55
3.3.1 Estructura del proyecto de Android Studio.....	55
3.3.2 Diseño adaptativo para distintos tamaños de pantalla.....	61
3.3.3 Navegación principal.....	62
3.3.4 Almacenamiento de entidades en Backendless.....	63
3.3.5 Gestión de cuentas de usuario con Backendless User Service API.....	66
3.3.6 Obtención de imágenes y almacenamiento en Backendless Files.....	67
3.3.7 Búsqueda de libros en la base de datos de OpenLibrary.....	68
3.3.8 Lectura de códigos de barras (ISBN de libros) con ZXing embedded.....	69
3.3.9 Obtención de la localización actual del usuario con Google Maps.....	69
3.3.10 Localización del usuario con Google Places durante el registro.....	71

3.3.11	Búsqueda de usuarios y visualización de resultados en mapa	71
3.3.12	Mensajería	72
3.3.13	Valoración y actualización de puntuación media	74
3.3.14	Custom adapters.....	74
3.4	Pruebas	75
3.5	Funcionalidades desarrolladas.....	80
3.5.1	Splash, autenticación y registro de usuarios	81
3.5.2	Pantalla principal, mapa y búsqueda de usuarios	81
3.5.3	Detalle de usuario y detalle de libro compartido por otro usuario	81
3.5.4	Listado de libros del usuario, detalle de libro y nuevo libro	82
3.5.5	Listado de intercambios y gestión de intercambio.....	82
3.5.6	Valoración de intercambio y desconexión	82
4.	Conclusiones	83
4.1	Líneas de trabajo futuro.....	85
5.	Glosario	86
6.	Bibliografía	88
Anexo I:	Diagrama de Gantt de planificación	90
Anexo II:	Árbol de navegación con miniaturas	92
Anexo III:	Instrucciones de compilación	93
Anexo IV:	Manual de usuario	94
	Autenticación y registro	94
	Búsqueda de usuarios.....	96
	Iniciar un intercambio	98
	Gestionar libros propios.....	101
	Visualizar intercambios.....	104
	Aceptar un intercambio entrante	106
	Rechazar un intercambio entrante	107
	Valorar un intercambio	108
	Cierre de sesión	109

Lista de figuras

Figura 1: Porcentaje de lectores con frecuencia mínima trimestral [1]	1
Figura 2: Libros leídos en el último año por formato [1]	2
Figura 3: Libros almacenados en el hogar sin contar libros de texto [1]	2
Figura 4: Comparativa de tiempo de uso de aplicaciones frente a tiempo dedicado a navegación web en diferentes países [8]	9
Figura 5: Distribución de ventas de plataformas móviles en España en 2017 [11]	10
Figura 6: Fases del modelo en cascada	11
Figura 7: Estructura de descomposición del trabajo	14
Figura 8: Distribución de lectores habituales de libros según rango de edad [1]	18
Figura 9: Distribución de lectores habituales de libros por tamaño de municipio [1]	19
Figura 10: Distribución del idioma de lectura habitual u ocasional [1]	19
Figura 11: Uso del smartphone en España por franja de edad en 2017	20
Figura 12: El 93% de los españoles usaron el smartphone a diario en 2017	20
Figura 13: Árbol de navegación	25
Figura 14: Pantalla de presentación (splash)	26
Figura 15: Pantallas de autenticación y registro de usuarios	27
Figura 16: Pantalla de usuarios cercanos y menú principal de navegación	28
Figura 17: Pantalla de búsqueda de libros	29
Figura 18: Datos de usuario sobre el mapa y pantalla de detalle de usuario	30
Figura 19: Pantalla de detalle de libro compartido por otro usuario y solicitud del mismo	30
Figura 20: Listado de libros del usuario y pantalla de detalle de libro	31
Figura 21: Pantalla de alta de libro con funcionalidad de cambio de fotografía	32
Figura 22: Pantalla de lectura del código ISBN del libro	32
Figura 23: Listado de intercambios y pantalla de detalle de intercambio con el usuario como iniciador	33
Figura 24: Pantalla de detalle de intercambio con el usuario como destinatario	34
Figura 25: Intercambio aceptado y pantalla de valoración	34
Figura 26: Logotipo de la aplicación	35
Figura 27: Prototipos de alta definición para las pantallas de splash, autenticación y búsqueda sobre mapa	36
Figura 28: Prototipos de alta definición para las pantallas de detalle de libro, detalle de usuario y listado de intercambios	36
Figura 29: Diagrama de casos de uso	37
Figura 30: Arquitectura cliente-servidor	45
Figura 31: Diagrama del patrón MVP	46
Figura 32: Diagrama entidad-relación	47
Figura 33: Diagrama de entidades	50
Figura 34: Implementación del patrón MVP en la aplicación	51
Figura 35: Estadísticas de instalación de Android por versión [18]	56
Figura 36: Estructura de los paquetes principales del proyecto	56

Figura 37: Diagrama de clases de la implementación de MVP por subpaquete	58
Figura 38: Diagrama de clases de las vistas de la aplicación	59
Figura 39: Diagrama de clases de los presentadores de la aplicación	60
Figura 40: Diagrama de clases del modelo de la aplicación	61
Figura 41: Definición de la entidad User en Backendless	64
Figura 42: Definición de la entidad Libro en Backendless	65
Figura 43: Definición de la entidad Intercambio en Backendless	65
Figura 44: Proceso de obtención de la localización del usuario	70
Figura 45: Marcador de usuario sobre el mapa	72
Figura 46: Configuración del envío de mensajes en la consola de Backendless	73
Figura 47: Resultados de ejecución de pruebas automáticas	80
Figura 48: Splash, autenticación y registro de usuarios	81
Figura 49: Pantalla principal, mapa y búsqueda de usuarios	81
Figura 50: Detalle de usuario y detalle de libro compartido por otro usuario	81
Figura 51: Listado de libros del usuario, detalle de libro y nuevo libro	82
Figura 52: Listado de intercambios y gestión de intercambio	82
Figura 53: Valoración de intercambio y desconexión	82
Figura 54: Diagrama de Gantt con la planificación del trabajo	90
Figura 55: Árbol de navegación con miniaturas de los prototipos de baja fidelidad	92
Figura 56: Bienvenida	94
Figura 57: Autenticación	94
Figura 58: Registro de usuario	95
Figura 59: Herramienta de autocompletado de Google Places	96
Figura 60: Permiso de acceso a la localización y activación del servicio	97
Figura 61: Mapa con la posición de usuarios cercanos	97
Figura 62: Búsqueda de usuarios	98
Figura 63: Información del usuario sobre el mapa	99
Figura 64: Detalle de usuario	99
Figura 65: Detalle de libro de otro usuario	100
Figura 66: Listado de intercambios	101
Figura 67: Menú principal y listado de libros	101
Figura 68: Detalle de libro del usuario	102
Figura 69: Nuevo libro	103
Figura 70: Diálogo de selección de fotografía del libro	104
Figura 71: Menú principal y listado de intercambios	105
Figura 72: Detalle de intercambio saliente pendiente	106
Figura 73: Detalle de intercambio entrante pendiente	106
Figura 74: Selección de libro para aceptar intercambio entrante	107
Figura 75: Detalle de intercambio aceptado	107
Figura 76: Detalle de intercambio entrante rechazado	108
Figura 77: Valoración de intercambio	109
Figura 78: Detalle de intercambio con las dos valoraciones	109
Figura 79: Menú principal y cierre de sesión	110

Lista de tablas

Tabla 1: Planificación del trabajo	15
Tabla 2: Objetivos y entregables por bloque	16
Tabla 3: Lectores de libros en tiempo libre según nivel de estudios [1]	18
Tabla 4: Escenario #1	22
Tabla 5: Escenario #2	23
Tabla 6: Escenario #3	23
Tabla 7: Paleta de colores de la aplicación	35
Tabla 8: Caso de uso "Registrar cuenta"	38
Tabla 9: Caso de uso "Autenticar"	38
Tabla 10: Caso de uso "Buscar libros"	39
Tabla 11: Caso de uso "Ver libro"	39
Tabla 12: Caso de uso "Solicitar intercambio"	40
Tabla 13: Caso de uso "Ver libros compartidos"	40
Tabla 14: Caso de uso "Añadir libro compartido"	41
Tabla 15: Caso de uso "Ver libro compartido"	41
Tabla 16: Caso de uso "Eliminar libro compartido"	42
Tabla 17: Caso de uso "Ver intercambio"	42
Tabla 18: Caso de uso "Aceptar/rechazar intercambio"	43
Tabla 19: Caso de uso "Valorar intercambio"	43
Tabla 20: Entidad "Usuario"	48
Tabla 21: Entidad "Libro"	48
Tabla 22: Entidad "Intercambio"	49
Tabla 23: Entidad "Idioma"	49
Tabla 24: Entidad "Género"	49
Tabla 25: Entidad "Provincia"	49
Tabla 26: Checklist de pruebas de verificación	78

1. Introducción

1.1 Contexto y justificación del Trabajo

La lectura sigue siendo una de las aficiones preferidas en España para ocupar los ratos de ocio. A pesar de que existe la creencia popular de que se lee poco, el 65,8% de la población española mayor de 14 años afirma leer libros con una frecuencia al menos trimestral, y el 59,7% lo hace en su tiempo libre, como hobby [\[1\]](#):



Figura 1: Porcentaje de lectores con frecuencia mínima trimestral [\[1\]](#)

En general, además, los aficionados a la lectura disfrutan más haciéndolo en el clásico soporte impreso, como demuestra el hecho de que el 75,2% de los libros leídos en 2017 estaban en formato papel [\[1\]](#):

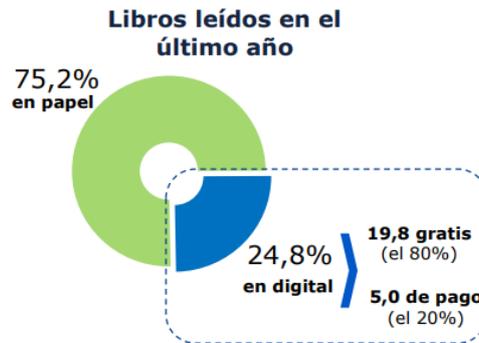


Figura 2: Libros leídos en el último año por formato [1]

Esos libros ocupan espacio físico de almacenamiento, y la tendencia es que se vayan acumulando a lo largo del tiempo, pasando a formar parte de la biblioteca personal. Éste es, además, un hábito generalizado, ya que el 54,2% de los aficionados a la lectura acumula más de 50 libros en su hogar, siendo la media global por hogar en España de 227 ejemplares en 2017, sin contar libros de texto [1]:



Figura 3: Libros almacenados en el hogar sin contar libros de texto [1]

Por diversas circunstancias (lectura ya realizada o abandonada, libros de texto de cursos pasados, regalos aceptados por compromiso, etcétera), muchos de estos libros quedan olvidados en una caja o estantería, donde con mucha probabilidad no se volverán a leer, ocupando espacio innecesariamente y sin ser aprovechados.

Por otra parte, según el Anuario de Estadísticas Culturales de 2017, publicado por el Ministerio de Educación, Cultura y Deporte [2], el gasto medio por hogar en libros se situó en 123,6 euros durante el año 2016. Si esos libros se acumulan y no se reutilizan el resultado es que, con el paso de los años, la cantidad de dinero invertido que se pierde se hace cada vez más importante.

La problemática expuesta puede ser tratada haciendo uso de los nuevos canales de comunicación y las posibilidades que estos ofrecen a través de una aplicación móvil. Los dispositivos móviles, y en especial los *smartphones*, son ya un elemento indispensable en nuestra vida diaria y nos facilitan casi cualquier tarea cotidiana. La necesidad planteada no es una excepción, ya que una aplicación móvil que permitiera el intercambio de libros usados entre usuarios ayudaría a rescatar los ejemplares olvidados y a darles una segunda oportunidad.

Este tipo de aplicación fomentaría la reutilización de los libros, al crear una comunidad de aficionados a la lectura dispuestos a compartir los títulos que ya no les son útiles y cambiarlos por los de otros usuarios, en los que sí estén interesados. El proceso de permuta podría repetirse indefinidamente, con lo que un mismo libro se reutilizaría en multitud de ocasiones. Esta circunstancia repercutiría también indirectamente en otros aspectos como:

- Ahorro económico.
- Reducción del impacto medioambiental.

De manera adicional, gracias al servicio de geolocalización que ofrece la mayoría de dispositivos móviles, la aplicación podría favorecer los intercambios por proximidad, de manera que se pongan en contacto usuarios que se encuentren cercanos entre sí, posibilitando hacer los cambios en persona y garantizando así transacciones más afines.

Con este planteamiento como referencia se ha indagado en Internet y en las tiendas de las plataformas móviles más usadas, iOS y Android, para buscar aplicaciones y sitios web dedicados al intercambio de libros usados entre usuarios, existiendo algunas alternativas:

BookMooch (bookmooch.com)

Se trata de un portal web, de registro gratuito, dedicado al intercambio de libros que tiene como base un original sistema de puntos, por el cual, un usuario los recibe cuando comparte libros con la comunidad o entrega alguno a otro usuario. Si no se dispone de puntos no es posible solicitar libros a otros usuarios. Es por tanto una forma de premiar a aquellos miembros de la comunidad que contribuyen en mayor medida, aunque no se trata de un sistema de intercambio 1 a 1, ya que un usuario que disponga de suficientes puntos puede solicitar un libro sin tener que entregar otro en la misma operación.

El portal tiene una comunidad bastante amplia, ya que aglutina usuarios de todo el mundo, por lo que los envíos, en algunos casos, se hacen internacionalmente. No obstante, no dispone de app ni de un diseño responsivo, lo que dificulta su visualización en un dispositivo móvil.

BookToBook (www.booktobook.net)

El concepto de BookToBook está orientado a la permuta entre usuarios, de manera que en cada operación el usuario entrega un libro y recibe el del otro participante. Se trata de un portal español que permite realizar búsquedas por ciudad, con lo que permite encontrar libros que se encuentren cercanos al usuario. Ambos enfoques se pretenden adoptar en el trabajo por lo que, en ese sentido, es una de las alternativas encontradas más parecidas.

El portal dispone de una funcionalidad que permite a los usuarios registrar los libros que desean y recibir un aviso cuando algún usuario los comparte con la comunidad, lo cual resulta muy útil y cómodo. Por el contrario, no tiene comportamiento responsivo, por lo que es complicado navegar por él en un dispositivo móvil, ni cuenta con app en las tiendas de aplicaciones.

BookInt (bookint.eu)

El portal BookInt se especializa en el intercambio y venta de libros de texto en los centros de enseñanza. En este caso la idea subyacente es ligeramente diferente, ya que los intercambios se realizan a través de entidades gestoras interesadas que actúan como intermediarias, típicamente las asociaciones de madres y padres de alumnos o los propios colegios. Los usuarios entregan los libros que desean compartir a estas entidades, y pueden recoger en ellas tantos ejemplares como hayan donado. El portal, por tanto, funciona como punto de encuentro entre los usuarios finales y los centros de enseñanza.

BookInt tampoco cuenta con app ni se visualiza correctamente en dispositivos móviles, al no tener un diseño responsivo.

Bookswap [\[3\]](#)

Bookswap es una app de reciente publicación disponible solo para el sistema operativo Android. La idea de la aplicación es muy similar a la que se pretende proponer en el trabajo: los usuarios registran en la aplicación los libros que quieren compartir y pueden solicitar lo que otros usuarios ofrecen. Por medio de la ubicación del dispositivo permite encontrar títulos cercanos, aunque los representa en forma de listado, no ubicados en un mapa.

Si bien la aplicación tiene un diseño atractivo, el registro de libros resulta tedioso, ya que es necesario introducir los datos de forma manual. Por otra parte, al ser una comunidad de creación reciente, y a tenor del número de descargas de la aplicación en Google Play Store, el número de miembros es aún pequeño, por lo que el catálogo de libros disponibles es reducido.

Booxup [\[4\]](#)

Se trata de una app para iOS y Android especializada en la reutilización de libros de segunda mano, pero a través de la compraventa. Los usuarios ponen a la venta en la plataforma los libros que ya no utilizan y reciben dinero por ello. Se trata, por tanto, de una aproximación diferente a la planteada en el trabajo. Se centra, además, en libros en los idiomas inglés y francés, lo que supone una limitación importante en nuestro país.

Otras alternativas

Si se amplía el ámbito de la aplicación se pueden encontrar numerosas apps dedicadas al intercambio de bienes en general, como Obsso [\[5\]](#), Wendoo [\[6\]](#) o Trueketeke [\[7\]](#). En todas ellas existen categorías de búsqueda que abarcan los libros, sin embargo, precisamente por no estar especializadas en este tipo de producto, los resultados que se suelen obtener son bastante escasos.

Aportación realizada

Tras la revisión realizada, se ha podido comprobar que ya existen aplicaciones y portales que dan respuesta a la problemática de la reutilización de libros mediante permutas entre usuarios. Por ello, se proponen las siguientes características diferenciadoras para el trabajo:

- **Movilidad:** Una interfaz diseñada específicamente para su correcta visualización en dispositivos móviles otorgaría una ventaja competitiva con respecto a los sitios web evaluados, que no ofrecen un comportamiento responsivo.
- **Proximidad:** El servicio de localización que ofrece el dispositivo móvil permitiría al usuario encontrar libros que se encuentren próximos a su ubicación geográfica, lo que supondría una ventaja frente al resto de alternativas que, o bien no dan esta posibilidad, o solo ofrecen filtrar por provincia.
- **Especialización:** El hecho de facilitar únicamente el intercambio de libros permitiría ofrecer una clasificación y categorización de los mismos de manera mucho más precisa que las aplicaciones de intercambio más generalistas, lo que supondría unas búsquedas más eficientes.
- **Atractivo visual y comodidad de uso:** La representación de los resultados de las búsquedas en un mapa geográfico o la introducción de los datos de los libros de manera automática, haciendo uso de la cámara del dispositivo, son características que representan un valor añadido sobre aplicaciones, como Bookswap, que ofrecen funcionalidades parecidas, ya que mejorarían la experiencia de uso.

1.2 Objetivos del Trabajo

Los objetivos generales que se persiguen con el trabajo son los siguientes:

- Favorecer la reutilización de libros ya leídos o cuya lectura se haya abandonado, y que de otra manera quedarían olvidados y ocupando espacio inútilmente.
- Ofrecer a los aficionados a la lectura la posibilidad de acceder a títulos descatalogados y difíciles de encontrar en otras circunstancias.
- Facilitar el contacto entre personas con una afición común como la lectura, aprovechando el tirón actual de los entornos colaborativos y las redes sociales.
- Contribuir al ahorro económico de los usuarios, ya que el intercambio ofrece la posibilidad de leer ejemplares sin necesidad de tener que comprarlos.
- Ayudar a los usuarios a descubrir nuevos géneros a los que probablemente no darían una oportunidad si tuvieran que adquirirlos en una librería.
- Colaborar en la sostenibilidad del medio ambiente por medio de la reutilización de los libros, reduciendo la tala de árboles y la emisión de gases derivada de los procesos de impresión y distribución.

Dichos objetivos generales se conseguirán a través de los siguientes objetivos específicos:

- Implementar una aplicación móvil que permita la creación de una comunidad en la que los usuarios puedan intercambiar sus propios libros.
- Incluir en ella un mecanismo que permita buscar a otros usuarios que se encuentren cercanos geográficamente, para incluso posibilitar que los

intercambios se realicen en persona, evitando así posibles problemas con los envíos.

- Conseguir que los intercambios realizados a través de la aplicación sean considerados como confiables, gracias a la incorporación de un sistema que permita puntuarlos. Estas valoraciones servirán para medir la reputación de los usuarios.
- Obtener una aplicación atractiva visualmente y con un diseño intuitivo que facilite su utilización por parte de cualquier tipo de usuario.

Alcance del Trabajo:

Como consecuencia de los objetivos enumerados anteriormente y para dar respuesta a los mismos de manera satisfactoria, la implementación de la aplicación móvil se enmarcará en el alcance definido por los siguientes requerimientos:

Requerimientos funcionales básicos:

- El acceso a la aplicación estará limitado a usuarios registrados, por lo que se deberá permitir la creación de nuevas cuentas a aquellos que no estén dados de alta. La información que se recogerá para la creación de una cuenta de usuario será muy básica. Constará, como mínimo, del correo electrónico y una contraseña de acceso, que serán los datos que el usuario deberá introducir para autenticarse en la aplicación.
- El usuario podrá compartir con el resto de miembros de la comunidad todos los libros que considere oportuno. Para ello contará con una funcionalidad que le permitirá gestionar los libros de su propiedad que desee intercambiar con otros usuarios, pudiendo:
 - Incluir un nuevo libro en su perfil.
 - Eliminar del perfil un libro existente.
 - Visualizar la lista de libros registrados. Estos libros serán los que estén disponibles para realizar los intercambios con otros usuarios.
- Existirá un buscador que permitirá a los usuarios encontrar los libros que otros miembros ponen a disposición de la comunidad para realizar los intercambios. Las búsquedas podrán acotarse utilizando diferentes criterios, como título, autor y género literario.
- Dado que se quiere favorecer los intercambios de libros en persona, los resultados de las búsquedas se mostrarán sobre un mapa, de manera que se pueda identificar fácilmente qué otros usuarios que comparten libros se encuentran próximos geográficamente, pudiéndose establecer un radio máximo para limitar la distancia de la búsqueda.
- Si un usuario encuentra un libro en el que está interesado podrá enviar una solicitud de intercambio a su propietario, el cual recibirá un aviso por correo electrónico. Cuando el destinatario de la petición acceda a la aplicación podrá gestionar la solicitud por medio de las siguientes acciones:
 - Consultar la lista de libros que comparte el solicitante, para comprobar si le interesa alguno de los que ofrece.

- Si está interesado en alguno podrá seleccionarlo y aceptar el intercambio. El solicitante recibirá un correo electrónico que le informará de que el intercambio ha sido aceptado y del libro con el que el destinatario desea hacer la permuta.
- Si no está interesado en ningún libro podrá rechazar el intercambio y éste no tendrá lugar. El solicitante recibirá un correo electrónico que le informará de que el intercambio ha sido rechazado.
- Los usuarios podrán valorar mediante una escala de puntuación las permutas que realicen a través de la aplicación. La media de estas puntuaciones aparecerá como dato público en el perfil de cada usuario, y será un reflejo de su reputación y confiabilidad de cara a realizar futuros intercambios.

Requerimientos funcionales secundarios:

Los siguientes requerimientos funcionales serían deseables, aunque no obligatorios, en una primera versión de la aplicación, ya que mejorarían la experiencia de uso, pero no añadirían ninguna funcionalidad adicional. Por ello, se incluyen en el alcance como opcionales, y se abordarán al final de la fase de implementación, tras los requerimientos básicos, si no se producen desvíos en la planificación propuesta:

- Facilitar al usuario el alta de un nuevo libro en su perfil obteniendo los datos del mismo a partir de su ISBN. De esta forma no sería necesaria su introducción manual, agilizándose el proceso. La lectura del ISBN se realizará por medio de la cámara del dispositivo móvil, que permitirá escanear el código de barras del libro. La búsqueda de los datos a partir del ISBN se realizaría por medio del servicio OpenLibrary.
- Enviar avisos a los usuarios a través de notificaciones *push* cuando se solicite un intercambio de libros o cuando se dé respuesta al mismo, ya sea aceptándolo o rechazándolo.

Requerimientos no funcionales:

- El desarrollo de la aplicación se realizará de forma nativa para dispositivos móviles con sistema operativo Android. Los motivos que justifican este requerimiento se expondrán en el siguiente apartado.
- El dispositivo necesitará disponer de algún mecanismo que permita obtener su ubicación, como GPS o Wi-Fi, para poder determinar la posición geográfica del usuario y así localizar libros que se encuentren cercanos.
- El dispositivo necesitará conexión a Internet para que la aplicación funcione correctamente, ya que toda la información se almacenará de manera persistente en un servidor remoto. Ningún dato se guardará localmente en el dispositivo.
- Todas las comunicaciones externas con el servidor de datos se cifrarán utilizando el protocolo HTTPS, para evitar que la información viaje en texto plano por Internet y, de esa forma, mejorar la seguridad.
- El idioma empleado en los textos de la aplicación será el español.

- El diseño de la interfaz debe ser intuitivo, para permitir una navegación ágil y un tiempo de aprendizaje por usuario reducido.

Límites y restricciones:

Dentro del alcance del trabajo se definen las siguientes restricciones:

- La aplicación no proporcionará ningún medio para realizar el envío de los libros, sino que deberán ser los usuarios los que acuerden entre ellos la forma de realizar los intercambios, ya sea presencialmente o a través de alguna empresa de mensajería. En cualquier caso, la logística del proceso queda fuera de la responsabilidad de la aplicación.
- Por lo ajustado de la planificación la interfaz de la aplicación se optimizará para su uso en dispositivos móviles. En futuras ampliaciones podría plantearse su adaptación a tabletas.

1.3 Enfoque y método seguido

La temática y objetivos propuestos en el trabajo, unidos a la alta tasa de penetración que tiene la tecnología móvil en la actualidad, sugieren plantear una solución adaptada a este tipo de dispositivos. Sin limitaciones de tiempo ni de recursos lo ideal sería dar respuesta a los requerimientos formulados mediante múltiples desarrollos, que cubrirían diferentes grupos de usuarios de manera simultánea:

- Sitio web responsivo.
- Aplicación móvil.

Lógicamente este supuesto no se verifica en el presente trabajo, en el que tanto tiempo como recursos son limitados, por lo que se hace necesario seleccionar la estrategia más apropiada.

Como ya se ha comentado con anterioridad, muchos de los productos existentes que tratan de dar solución a la problemática planteada en el trabajo son sitios web, no aplicaciones. Si bien es cierto que ninguno de los analizados dispone de diseño responsivo, lo que dificulta en gran medida su utilización en dispositivos móviles, parece que la necesidad está ya cubierta parcialmente desde el punto de vista de esta estrategia. Si además se tiene en cuenta que los usuarios solo pasan de media entre un 10 y un 20% del tiempo que utilizan el móvil en sitios web (80-90% en aplicaciones, 88% en el caso concreto de España) [8], y que la tendencia es que el uso de aplicaciones móviles siga creciendo en los próximos años [9], parece más adecuado enfocar el desarrollo del trabajo hacia esta segunda estrategia.

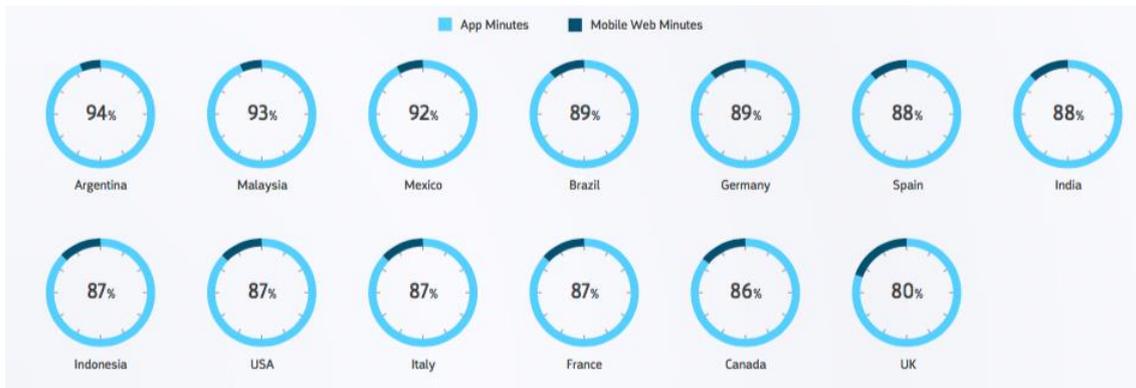


Figura 4: Comparativa de tiempo de uso de aplicaciones frente a tiempo dedicado a navegación web en diferentes países [8]

No obstante, el hecho de que la estrategia seleccionada sea el desarrollo de una aplicación móvil plantea una nueva disyuntiva, relacionada con la fragmentación de plataformas móviles existentes. Esta circunstancia obliga nuevamente a realizar múltiples desarrollos para los distintos sistemas operativos, en caso de seguir un enfoque basado en el uso de SDK nativos, o bien realizar un único desarrollo basado en la utilización de tecnologías multiplataforma, como PhoneGap o Appcelerator. Esta segunda aproximación, si bien permite llegar a varios sistemas operativos con un mismo código sin necesidad de portarlo, reduciendo así el tiempo y el coste del desarrollo, tiene algunos inconvenientes importantes:

- Se consigue una experiencia de usuario más pobre que con las aplicaciones nativas, ya que el rendimiento obtenido es peor.
- La interfaz de usuario nativa ofrece un mayor número de controles, por lo que, como norma general, será más rica que la de una aplicación multiplataforma.
- Determinadas funcionalidades y servicios de las plataformas móviles solo están disponibles a través de sus librerías nativas, por lo que una aplicación multiplataforma no podría hacer uso de ellos, limitando las capacidades del dispositivo.

Teniendo en cuenta estas desventajas y tomando como referencia algunos de los requerimientos planteados para la aplicación, como son el uso de la geolocalización, la cámara y una interfaz atractiva y ágil, se optará finalmente por realizar un desarrollo nativo. Por esta vía surge el problema de tener que desarrollar múltiples versiones de la aplicación para las distintas plataformas, pero nuevamente, al disponer de tiempo y recursos limitados, se hace imposible dentro del alcance del trabajo. En este sentido, atendiendo al criterio de selección del sistema operativo es claro que, en los últimos años, el dominio en el mercado de smartphones corresponde al duopolio formado por iOS y Android, copando entre ambos el 99,9% de las ventas a nivel mundial el pasado año 2017 [10]. No obstante, mientras en países como Estados Unidos o Reino Unido se reparten la cuota a partes aproximadamente iguales, en el caso concreto de España la penetración de Android es mucho más alta, contando con el 87,1% de las ventas, por el 12,7% de iOS en 2017 [11].

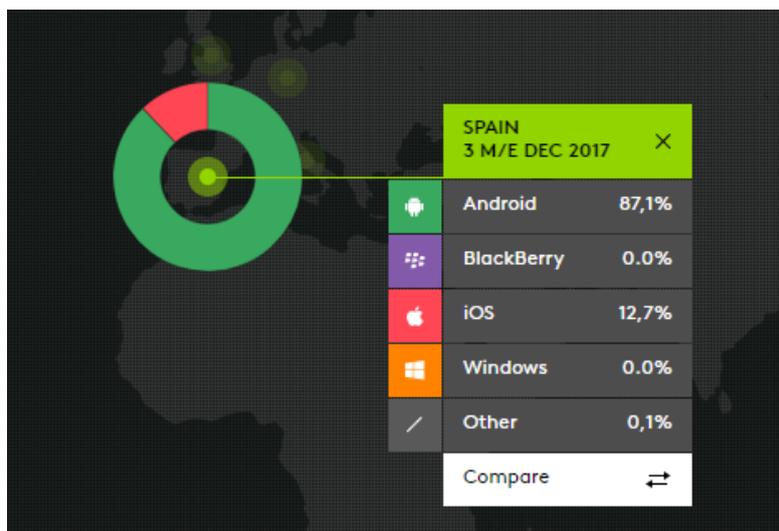


Figura 5: Distribución de ventas de plataformas móviles en España en 2017 [11]

Por todo lo comentado anteriormente se propone Android como sistema operativo de referencia para la aplicación que se está planteando y, de manera adicional, por los siguientes motivos:

- Contar de inicio con el mayor número posible de usuarios potenciales, ya que, al ser una aplicación colaborativa, cuanto mayor sea la base de personas que la utilicen mayor será también el número de libros que se comparten, por lo que podrá cumplir mejor su cometido.
- No hay mucha oferta de aplicaciones en Google Play Store especializadas en el intercambio de libros usados entre usuarios. La mayoría de las existentes son para comunidades anglo y francoparlantes, como Booxup [4] o Book Swap [12], siendo esta última de pago. La única que se ha encontrado en habla hispana en el mercado de aplicaciones de Google y con características similares ha sido Bookswap [3], pero por su escaso número de descargas dispone de una comunidad muy reducida y obliga a tener cuenta en Facebook para poder utilizarla, lo que supone una limitación para aquellos usuarios que no estén registrados en dicha red social. Obliga, además, a introducir los datos de los libros manualmente, lo que puede resultar tedioso si se dispone de una gran cantidad de libros. Por otra parte, ninguna de ellas prioriza el intercambio por proximidad, que es uno de los objetivos principales propuestos en el trabajo.

Teniendo en cuenta también que las alternativas revisadas en la tienda de aplicaciones de Google son soluciones cerradas, cuyo código fuente no está disponible y por tanto no pueden ser modificadas, la estrategia propuesta para el trabajo es el desarrollo de una nueva aplicación desde cero.

Finalmente, dado que los requerimientos del trabajo están fijados desde el inicio y no se espera que sufran demasiados cambios durante su ciclo de vida, se utilizará una metodología de desarrollo en cascada o *waterfall*. A través de ella el trabajo se dividirá en fases ligadas entre sí secuencialmente, de manera que la consecución de una depende irremediamente de la anterior, por lo

que cobra especial importancia el cumplimiento de las fechas propuestas para los hitos intermedios:



Figura 6: Fases del modelo en cascada

1.4 Planificación del Trabajo

Para la realización del trabajo es necesario contar con un conjunto de recursos que posibilite que se lleve a cabo correctamente. Para el caso del trabajo propuesto los recursos necesarios son los siguientes:

Humanos

El trabajo se desarrollará de manera individual, por lo que el equipo humano está compuesto por una única persona, el alumno autor del trabajo, que asumirá todos los perfiles involucrados en el ciclo de vida:

- Analista.
- Diseñador.
- Desarrollador.
- Verificador.
- Documentalista.

La dedicación estimada del recurso será de 3 horas los días laborables y 5 horas los días festivos y fines de semana, lo que arroja un mínimo de 25 horas semanales de trabajo.

Como recursos externos al equipo de desarrollo, y con el objetivo de revisar y tutorizar el progreso del trabajo, se encuentran los profesores D. Pau Dominkovics Coll y D. Carles Garrigues Olivella.

Software



Balsamiq Mockups

Herramienta para la construcción de *wireframes*. Se empleará durante el diseño de la aplicación para generar los primeros bocetos, en los que prima la representación esquemática de las distintas funcionalidades.

Web: balsamiq.com



Android Studio

Entorno de desarrollo integrado oficial para la construcción de aplicaciones nativas sobre la plataforma Android. Su uso estará ligado a las tareas de implementación y verificación.

Web: developer.android.com/studio



Microsoft Office

Suite ofimática que se empleará a lo largo de todo el proyecto para generar la documentación asociada al trabajo.

Web: www.office.com



Adobe Photoshop

Aplicación de diseño y tratamiento de imágenes que se empleará a lo largo de todo el proyecto para editar las ilustraciones incluidas en la documentación del trabajo.

Web: www.adobe.com/es/photoshop

Servicios



Backendless

Se trata de un servicio de *backend* en la nube que ofrece una serie de funcionalidades que permiten prescindir totalmente de una API y un servidor personalizados. Entre las funcionalidades que publica se encuentran la autenticación y registro de usuarios, el almacenamiento de archivos o el envío de notificaciones *push*, las cuales dan respuesta a los requisitos planteados para la aplicación. La gran ventaja de este tipo de servicios es que simplifican en gran medida el desarrollo de la parte de servidor, permitiendo centrar esfuerzos en la aplicación propiamente dicha.

Web: backendless.com



Google Services

Conjunto de servicios ofrecidos por Google y que, integrados en la aplicación, ayudarán a darle valor, como Google Maps, para la representación de mapas.

Web: developer.android.com/google



OpenLibrary

Base de datos libre y de código abierto que ofrece un servicio de consulta de libros que permite obtener información sobre los mismos.

Web: openlibrary.org

Hardware

Durante todas las fases del proyecto se requerirá el uso de una máquina de trabajo sobre la que poder ejecutar los diferentes recursos software. El equipo disponible es un portátil con procesador Intel® Core™ i5-4200U, 8 GB de memoria RAM y sobre el que corre un sistema operativo Windows 7 Professional de 64 bits. Las características del equipo superan los requisitos mínimos necesarios para ejecutar cualquiera de los recursos software del trabajo, por lo que está garantizado el correcto funcionamiento de los mismos.

Por otra parte, no es obligatorio, pero sí recomendable, disponer de dispositivos físicos con el sistema operativo Android sobre los que llevar a cabo el proceso de verificación de la aplicación. No es un asunto crítico, ya que las pruebas pueden realizarse por medio de emuladores, pero el hecho de contar con ellos ayuda a agilizar los procesos de desarrollo y verificación, al ofrecer un rendimiento mayor que los emuladores. En este sentido se dispone de un móvil Samsung Galaxy S4 con sistema operativo Android Nougat 7.1.2.

Planificación

Los recursos descritos van a permitir la correcta generación del conjunto de entregables que componen el trabajo. Dichos entregables son los siguientes:

1. La aplicación desarrollada.
2. Una memoria del trabajo, en la que se incluirá todo el proceso de diseño y desarrollo de la aplicación.
3. Una presentación en vídeo que resuma los aspectos más relevantes del trabajo.

Para facilitar la planificación de los entregables se presenta en el siguiente diagrama la estructura de descomposición del trabajo de cada uno de ellos, con las correspondientes tareas asociadas. En el caso concreto de la aplicación móvil, las tareas asociadas a requisitos secundarios se presentan en otro color (rojo) para su diferenciación:

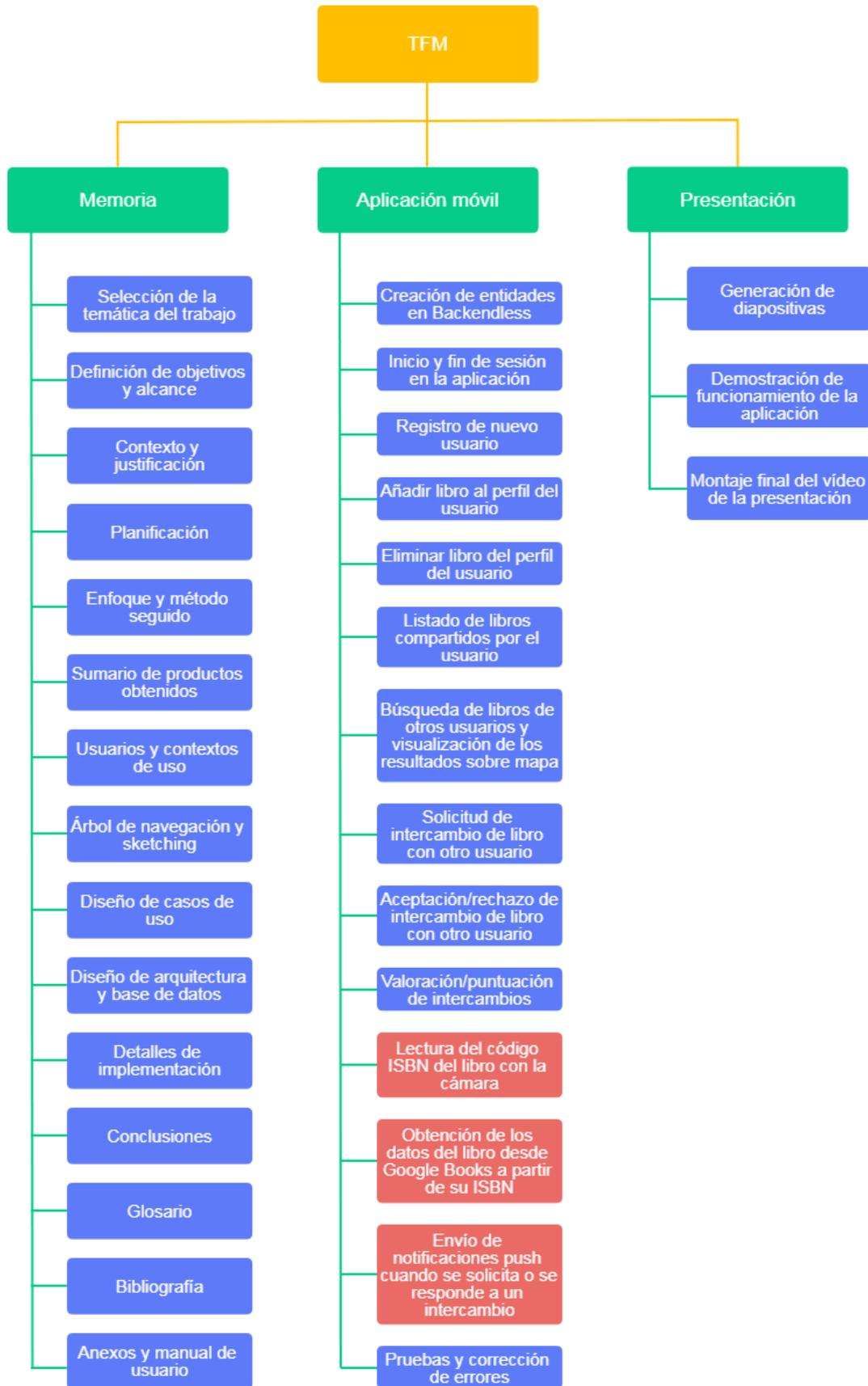


Figura 7: Estructura de descomposición del trabajo

En base al desglose de tareas planteado y a la temporización del plan docente, la planificación propuesta para la ejecución del trabajo es la que se muestra a continuación:

Tarea	Duración	Comienzo	Fin
BLOQUE 1: PLAN DE TRABAJO	76 horas	21-feb-18	14-mar-18
Selección de la temática	28 horas	21-feb-18	28-feb-18
Objetivos y alcance	12 horas	01-mar-18	04-mar-18
Contexto y justificación	8 horas	04-mar-18	06-mar-18
Planificación	16 horas	06-mar-18	10-mar-18
Enfoque y método seguido	8 horas	11-mar-18	12-mar-18
Sumario de productos obtenidos	4 horas	13-mar-18	14-mar-18
Entrega PEC1		14-mar-18	14-mar-18
BLOQUE 2: DISEÑO	76 horas	14-mar-18	04-abr-18
Usuarios y contextos de uso	12 horas	14-mar-18	17-mar-18
Árbol de navegación y <i>sketching</i>	24 horas	17-mar-18	24-mar-18
Evaluación	8 horas	24-mar-18	26-mar-18
Definición de casos de uso	16 horas	26-mar-18	31-mar-18
Diseño de arquitectura y BD	16 horas	31-mar-18	04-abr-18
Entrega PEC2		04-abr-18	04-abr-18
BLOQUE 3: IMPLEMENTACIÓN	150 horas	04-abr-18	16-may-18
Creación de entidades en Backendless	6 horas	04-abr-18	06-abr-18
Inicio/fin de sesión	8 horas	06-abr-18	08-abr-18
Registro de nuevo usuario	8 horas	08-abr-18	10-abr-18
Añadir/eliminar libro de perfil	12 horas	10-abr-18	14-abr-18
Listado de libros del usuario	16 horas	14-abr-18	18-abr-18
Búsqueda de libros / mapa	16 horas	18-abr-18	22-abr-18
Solicitud de intercambio	8 horas	22-abr-18	25-abr-18
Aceptación/rechazo de intercambio	8 horas	25-abr-18	27-abr-18
Valoración de intercambios	8 horas	28-abr-18	29-abr-18
Lectura de ISBN por código de barras	8 horas	29-abr-18	01-may-18
Integración API de búsqueda de ISBN	8 horas	02-may-18	04-may-18
Envío de notificaciones push	8 horas	04-may-18	06-may-18
Pruebas y corrección de errores	12 horas	06-may-18	09-may-18
Redacción detalles de implementación	24 horas	10-may-18	16-may-18
Entrega PEC3		16-may-18	16-may-18
BLOQUE 4: ENTREGA FINAL	74 horas	16-may-18	06-jun-18
Conclusiones	10 horas	16-may-18	19-may-18
Glosario	8 horas	19-may-18	21-may-18
Bibliografía	8 horas	21-may-18	23-may-18
Manual de usuario	12 horas	24-may-18	27-may-18
Anexos	12 horas	27-may-18	30-may-18
Generación de la presentación	24 horas	30-may-18	06-jun-18
Entrega PEC4		06-jun-18	06-jun-18

Tabla 1: Planificación del trabajo

Como se puede observar, la planificación se ha dividido en cuatro bloques, cada uno de ellos con un hito final coincidente con las PEC establecidas en la organización del semestre. En la fase de implementación, al igual que en la estructura de descomposición del trabajo, se han marcado en otro color las tareas opcionales, correspondientes a los requerimientos secundarios. Los objetivos a cumplir en cada bloque se resumen en la siguiente tabla:

Bloque	Objetivos	Entregables	Fecha
1. Plan de trabajo	Definición de los objetivos y planificación de las tareas del trabajo, describiendo a alto nivel los requerimientos funcionales y no funcionales que compondrán el alcance del mismo.	<ul style="list-style-type: none"> Capítulo 1 de la memoria, con el plan de trabajo. 	14/03/18
2. Diseño	Definición de los usuarios y contextos de uso de la aplicación propuesta, elaborando los escenarios de uso de la misma, su mapa de navegación y un primer diseño (<i>sketching</i>) de las pantallas que la componen. En este bloque se incluirá también el diseño técnico de la aplicación, en concreto, la definición de los casos de uso y el diseño de su arquitectura.	<ul style="list-style-type: none"> Actualización de la memoria con el capítulo 2, que incluye el análisis, diseño y <i>sketching</i> de la aplicación. 	04/04/18
3. Implementación	Implementación y verificación de la aplicación Android según el diseño realizado en el bloque 2, justificando en la memoria las decisiones tomadas.	<ul style="list-style-type: none"> Actualización de la memoria con el capítulo 3, que incluye las decisiones de implementación y las pruebas de verificación realizadas. Código fuente de la aplicación, junto con el binario ejecutable y las instrucciones de compilación. 	16/05/18
4. Entrega final	Finalización del resto de apartados de la memoria y generación de un vídeo de presentación, a modo de síntesis, que refleje el trabajo realizado y los resultados obtenidos.	<ul style="list-style-type: none"> Versión final de la memoria del trabajo. Versión final de la aplicación, incluyendo el código fuente, el binario ejecutable, las instrucciones de compilación y el manual de usuario. Vídeo de presentación. 	06/06/18

Tabla 2: Objetivos y entregables por bloque

La organización del trabajo encaja con la metodología de ejecución en cascada propuesta en el apartado anterior, existiendo un mapeo claro entre sus fases y los diferentes bloques:

- Fase de requisitos y planificación → Bloque 1.

- Fase de diseño → Bloque 2.
- Fase de implementación → Bloque 3.
- Fase de verificación → Bloque 3.
- Fase de cierre → Bloque 4.

1.5 Breve resumen de productos obtenidos

A la finalización del trabajo se habrán generado los siguientes entregables:

1. **Memoria**, en formato PDF, con la descripción y planteamiento del trabajo realizado, su desarrollo, las decisiones tomadas y el resultado obtenido.
2. **Código fuente** comentado con la implementación de la aplicación propuesta.
3. **Archivo binario ejecutable**, en formato APK, de la aplicación desarrollada.
4. **Manual de usuario**, en formato PDF, con la descripción del funcionamiento de la aplicación.
5. **Instrucciones para compilar y ejecutar la aplicación**, en formato PDF, con las librerías, versiones de software y servicios requeridos para ello.
6. **Vídeo de presentación** a modo de resumen de los aspectos más relevantes del trabajo.

1.6 Breve descripción de los otros capítulos de la memoria

El resto de capítulos que componen la memoria del trabajo son los siguientes:

Análisis y diseño

En este capítulo se identifican los perfiles y usuarios de la aplicación, examinando las condiciones en las que la utilizarán y definiendo su contexto y escenarios de uso. Dicha información permite diseñar el flujo de interacción en el sistema, así como los prototipos de baja fidelidad de las diferentes pantallas (*wireframes*), para definir la estructura de la información y la visualización e interacción básicas con la aplicación. Incluye también el establecimiento formal de las funcionalidades, por medio de la definición de los casos de uso, así como el diseño de la arquitectura del sistema, identificando las entidades de base de datos involucradas en la gestión de los diferentes procesos.

Implementación y pruebas

El capítulo documenta y justifica las decisiones tomadas durante el proceso de implementación de la aplicación. Incluye también la definición del conjunto de pruebas diseñadas para asegurar el correcto funcionamiento de la misma.

Conclusiones

Presenta las conclusiones generadas en base al producto final obtenido, revisando el nivel de logro de los objetivos propuestos inicialmente y planteando las posibles líneas de ampliación para trabajo en el futuro.

2. Análisis y diseño

2.1 Definición del público objetivo

Utilizando como referencia los datos estadísticos publicados en el Barómetro de Hábitos de Lectura y Compra de Libros en España de 2017, la distribución de lectores habituales de libros atendiendo a su edad es la mostrada en la siguiente ilustración:

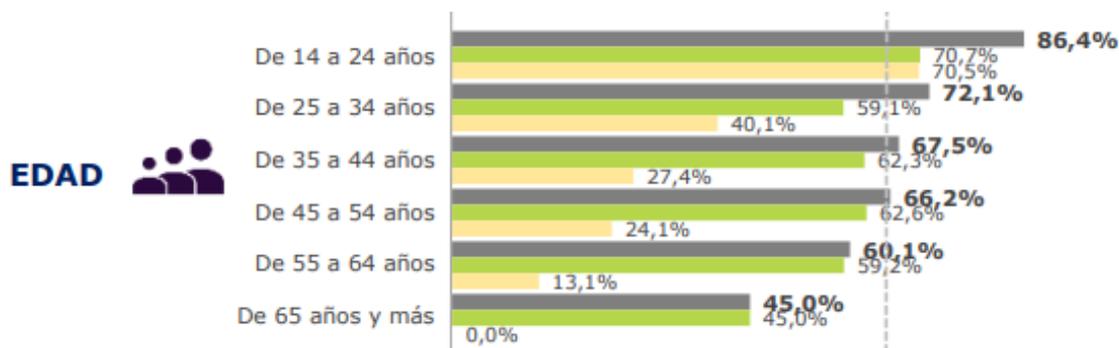


Figura 8: Distribución de lectores habituales de libros según rango de edad [1]

Como se puede observar, la lectura de libros se reduce con la edad, destacando muy claramente por encima de las demás la franja de población joven entre los 14 y 24 años, y seguida con no mucha diferencia por el grupo de población situado entre los 25 y los 54 años. Estos rangos de edad tienen también correlación con la ocupación, coincidiendo con la población estudiantil y la población activa, respectivamente. No obstante, para el caso concreto de la aplicación propuesta en el trabajo, se descartaría del perfil tipo el grupo de usuarios menores de edad, principalmente por dos motivos:

- No disponer de una biblioteca personal lo suficientemente grande.
- Necesitar la supervisión de un adulto, al requerirse el contacto personal con otros usuarios al realizar determinados intercambios.

Otro factor que determina el hábito lector es el nivel de estudios, aumentando con el grado de formación. En este caso, la población con estudios superiores se declara lectora en el 80,7% de los casos [1]:

	Hasta primarios	Secundarios	Superiores
Lector frecuente	28,2%	43,3%	69,5%
Lector ocasional	9,4%	13,6%	11,3%
Total	37,6%	56,8%	80,7%

Tabla 3: Lectores de libros en tiempo libre según nivel de estudios [1]

El tamaño de la población en la que se reside también condiciona la afición a la lectura, incrementándose en los municipios que tienen un mayor número de habitantes. Teniendo en cuenta, además, que la aplicación que se plantea se

basa en la colaboración entre usuarios, el hecho de vivir en núcleos con mayor número de habitantes aumenta la probabilidad de encontrar usuarios cercanos dispuestos a intercambiar sus libros con otras personas:

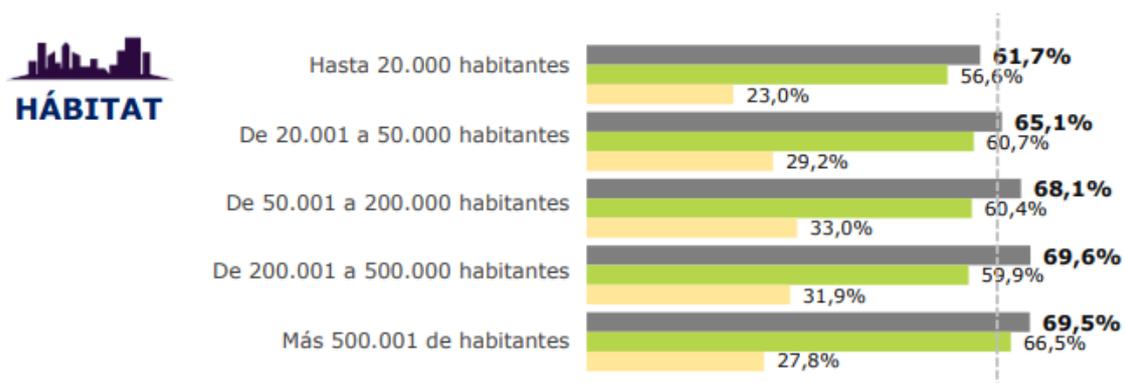


Figura 9: Distribución de lectores habituales de libros por tamaño de municipio [1]

Por otra parte, en lo referente a otros aspectos demográficos, como el sexo, si bien existen ciertas divergencias entre hombres y mujeres en cuanto a los hábitos de lectura, habiendo más mujeres lectoras que hombres (67,9% frente a 63,6%), la diferencia no es lo suficientemente grande como para incluir esta distinción en la definición del perfil.

Atendiendo al idioma de lectura se confirma un dato esperado *a priori*, y es que el 99,6% de los lectores habituales en España utiliza el castellano como idioma habitual u ocasional de lectura:

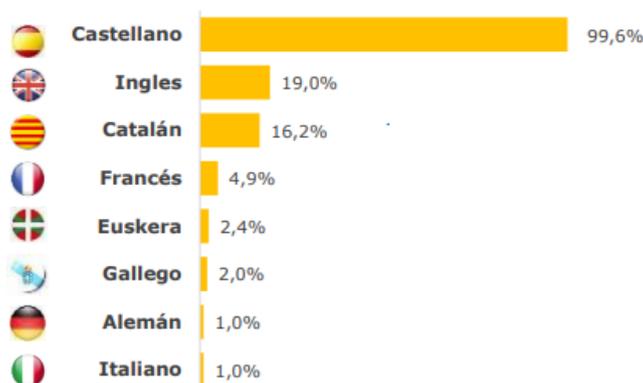


Figura 10: Distribución del idioma de lectura habitual u ocasional [1]

Cruzando esta información con otros datos estadísticos como, por ejemplo, el nivel de penetración del móvil, se puede ir construyendo el perfil demográfico tipo del usuario de la aplicación. Durante 2017 el 88% [13] de los españoles disponían de un *smartphone* para uso personal. Dentro de dicho porcentaje, la distribución por franja de edad es la mostrada en el siguiente gráfico:

Which devices do people use?

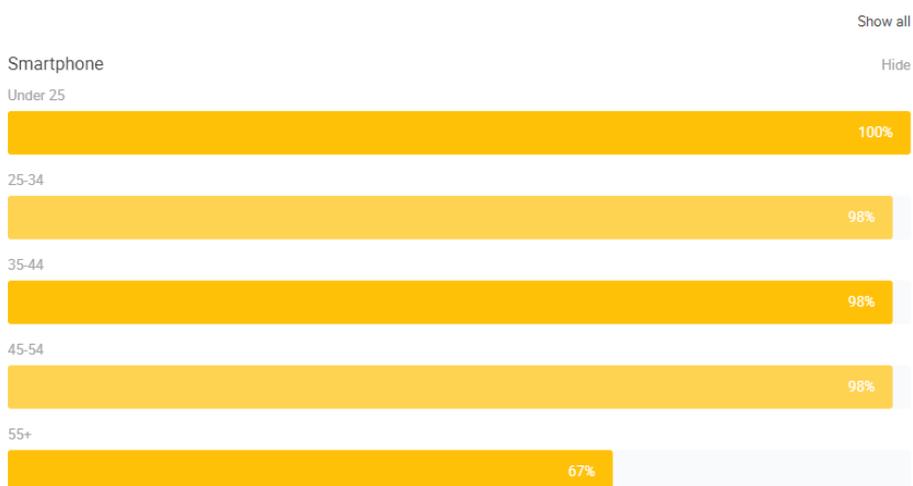


Figura 11: Uso del *smartphone* en España por franja de edad en 2017¹

Considerando, como se ha comentado anteriormente, que el rango de edad en el que se lee más a menudo se sitúa, aproximadamente, entre los 14 y los 54 años, parece claro que el público potencial al que iría dirigida la aplicación se sitúa dentro de las franjas de edad en las que la tasa de penetración del *smartphone* es más alta. Se trata, además, de un público joven o relativamente joven, habituado a las nuevas tecnologías y al manejo de dispositivos móviles.

How often do people go online (for personal Internet usage)?



Figura 12: El 93% de los españoles usaron el *smartphone* a diario en 2017²

De los datos anteriores se puede inferir un perfil objetivo aproximado de usuario potencial de la aplicación: aficionado a la lectura, preferentemente en idioma castellano, con edad comprendida entre los 21 y 54 años, con estudios superiores y ocupado laboralmente, que reside en un municipio de más de 20.000 habitantes y, por tanto, con más facilidad de acceso a las nuevas tecnologías de la información y la comunicación y con más probabilidad de interactuar con otros usuarios potenciales.

¹ Fuente: [Google Consumer Barometer](#).

² Fuente: [Google Consumer Barometer](#).

2.2 Encuesta

Para profundizar en las características del perfil objetivo representativo de la aplicación se ha diseñado una pequeña encuesta *online* por medio de la herramienta Google Forms, consistente en una serie de preguntas enfocadas a público aficionado a la lectura y con hábito de uso de un dispositivo móvil. La encuesta se encuentra disponible en el siguiente enlace:

https://docs.google.com/forms/d/e/1FAIpQLSejkqm9WRPz-g9qrtClhSbZRyMmBlb2q6zoQzgAh_D0NEw04A/viewform

La muestra de resultados obtenida no es grande (9 respuestas), por lo que los resultados no son totalmente representativos, pero ayudan a extraer algunas conclusiones sobre los posibles escenarios de uso y las funcionalidades de la aplicación:

- Casi la totalidad de los encuestados, el 88,9%, utiliza un dispositivo con sistema operativo Android, lo que refuerza la decisión tomada de comenzar desarrollando la aplicación en esta plataforma. El 11,1% restante utiliza iOS.
- Todos los usuarios coinciden en que estarían dispuestos a intercambiar libros que saben que no van a leer, siendo los principales motivos la falta de espacio en casa, el precio de los libros nuevos y el hecho de no leerlos una segunda vez como norma general.
- Los encuestados valoran positivamente la idea de intercambiar sus libros, sin embargo, ninguno de ellos declara haber utilizado anteriormente algún tipo de plataforma *online* para hacerlo, aunque todos estarían dispuestos a utilizarla en caso de existir. Esto induce a pensar que los usuarios no conocen herramientas que cubran esta necesidad, aunque muestran interés en ellas, por lo que su desarrollo parece justificado.
- En cuanto a la tipología de libros que los usuarios estarían dispuestos a intercambiar, casi todos coinciden en que serían novelas, el 77,8%, siendo los géneros de ciencia ficción y romántico los que más aceptación tendrían, con el 44,4% y el 33,3% de las respuestas, respectivamente. Llama también la atención que el 66,7% de las respuestas hacen referencia a la permuta de libros de texto escolares.
- Todas las respuestas valoran las funcionalidades propuestas como bastante útiles o muy útiles, y aportan ideas muy interesantes para mejorar la aplicación:
 - Valorar la reputación/nivel de confiabilidad de los usuarios con los que se hacen los intercambios (44,4%).
 - Valoración/reseña personal de los libros que cada persona ofrece para intercambiar (33,3%).
 - Disponer de una lista de deseos, de manera que la aplicación avise al usuario cuando alguien comparta un libro que se encuentre en ella (33,3%).

La primera de las sugerencias, a pesar de no aparecer entre las funcionalidades propuestas en la encuesta, sí que se encuentra dentro

del alcance definido inicialmente en el trabajo, por lo que, visto el interés mostrado por los usuarios, se implementará. El resto podrían incluirse como mejoras de la aplicación de cara a futuras ampliaciones.

2.3 Definición de personas y escenarios

A partir del perfil demográfico definido en anteriores apartados y de las respuestas obtenidas en la encuesta realizada a usuarios potenciales, se describen a continuación las *personas* que representarían los arquetipos de audiencia que harían uso de la aplicación, así como los posibles *escenarios* a través de los cuales se contextualiza el proceso de interacción de dichas *personas* con la problemática que se pretende solucionar con la aplicación móvil propuesta:

<p>Nombre: Yolanda Edad: 37 Profesión: Informática</p>
<p>Descripción: Yolanda vive con su pareja desde hace dos años en un pequeño apartamento en el centro de Sevilla, donde trabaja como consultora informática en una importante empresa de telecomunicaciones en una jornada completa de ocho horas de lunes a viernes. El poco tiempo libre que le permite su trabajo le gusta dedicarlo a viajar y a la lectura. Por su ocupación y por su naturaleza curiosa está habituada a utilizar el ordenador y el teléfono móvil para buscar información en Internet, especialmente sobre los temas que más le apasionan, que son las series y, por supuesto, los libros.</p>
<p>Escenario: Es sábado por la tarde y Yolanda ha decidido ir a dar un paseo por el centro de la ciudad. Al pasar por una librería ve en el escaparate que una de sus autoras preferidas ha publicado un libro nuevo. Le pilla por sorpresa, porque no sabía de su existencia. Junto a él hay expuestos otros títulos del mismo género, la literatura romántica, del que es una apasionada, de otros autores que no conoce. Le dan ganas de comprarlos todos, no tiene filtro para este tipo de novela, pero entonces recuerda que, literalmente, no tiene espacio para guardarlos y decide no comprarlos. Posee multitud de libros antiguos de temática similar que, tras leerlos, acumula en cajas guardadas en el cuarto trastero, ya que no sabe qué hacer con ellos y le da pena tirarlos, a pesar de que a la gran mayoría no les da una segunda lectura y solo están ocupando espacio.</p>

Tabla 4: Escenario #1

<p>Nombre: Elena Edad: 45 Profesión: Enfermera</p>
<p>Descripción: Elena está casada y tiene un hijo, Sergio, de 12 años. Vive en Barcelona, donde trabaja como enfermera de atención primaria en un centro de salud. En el último año la situación económica familiar se ha resentido debido a que su marido se ha quedado sin trabajo y hasta el momento no ha conseguido uno nuevo. Es por ello que durante los últimos meses han atravesado ciertas dificultades para afrontar los gastos domésticos, ya que el sueldo de Elena es la única fuente de ingresos de la que disponen en este momento.</p>
<p>Escenario: Es final de agosto y se aproxima la vuelta al colegio de Sergio tras las vacaciones de verano, lo que supone un gasto extra en material escolar, especialmente en libros de texto. Elena lo ha comentado en el centro de salud con sus compañeras, algunas de ellas también madres, y todas coinciden en que sería muy beneficioso disponer de alguna herramienta que pusiera en contacto a personas dispuestas a intercambiar sus libros de texto antiguos, ya que su reutilización supondría un importante ahorro económico.</p>

Tabla 5: Escenario #2

<p>Nombre: Francisco Edad: 51 Profesión: Profesor</p>
<p>Descripción: Francisco es profesor en un instituto de secundaria en Córdoba. Está divorciado y no tiene hijos lo que, unido a que no trabaja por las tardes, le deja mucho tiempo libre para dedicar a actividades culturales, que son su principal fuente de entretenimiento. Le gusta mucho la lectura, en especial la literatura clásica e histórica, así como los libros antiguos, a los que tiene dedicada una importante sección en la estantería del salón de su casa.</p>
<p>Escenario: Francisco lleva tiempo buscando una edición especial antigua de la <i>Gramática castellana</i> de Antonio de Nebrija. Está habituado a buscar este tipo de libros en tiendas de segunda mano y librerías anticuarias, pero no abundan en Córdoba, y este título se le está resistiendo más de la cuenta. Ha probado suerte también en plataformas <i>online</i>, como eBay o MilAnuncios, pero al no estar centradas en esta temática no es fácil encontrar en ellas lo que está buscando. Echa de menos algún tipo de aplicación o portal especializado que le permita ponerse en contacto directamente con otras personas con gustos literarios similares, y que puedan tener el ejemplar que está buscando, aunque esté usado.</p>

Tabla 6: Escenario #3

En base a este trabajo de conceptualización se identifican las siguientes funcionalidades principales para la aplicación:

- Compartir libros propios para poder intercambiarlos con los de otros usuarios.

- Poder contactar con usuarios cercanos geográficamente, para maximizar la probabilidad de éxito de los intercambios.
- Buscar libros de otros usuarios por género y autor.

Adicionalmente, y derivadas de las principales, se detecta la necesidad de incorporar también las siguientes funcionalidades secundarias:

- Registro de usuarios en el sistema, para poder asociar los libros que comparten a sus respectivos perfiles. Estos perfiles podrán ser consultados por el resto de usuarios.
- Autenticación de usuarios, como mecanismo de seguridad para acceder a la información almacenada en sus perfiles.
- Cierre de sesión, como consecuencia directa de lo anterior.
- Valoración numérica de los intercambios completados, lo cual servirá como medida de la reputación y confiabilidad de los usuarios.

Tomando como referencia este conjunto de funcionalidades se definirán a continuación el árbol de navegación de la aplicación y los prototipos en baja fidelidad de las pantallas principales de la misma.

2.4 Árbol de navegación

La aplicación se iniciará siempre con una pantalla de presentación, o *splash*, desde la que se dará paso a la pantalla de autenticación. En caso de no disponer aún de una cuenta, se dará la posibilidad al usuario de registrarse.

Una vez autenticado, la navegación principal se plantea a través de un menú desplegable lateralmente, desde el que se ofrece al usuario la posibilidad de acceder a las funcionalidades principales de la aplicación. Según los escenarios de uso identificados, la necesidad a cubrir más demandada por los usuarios potenciales es la de poder localizar a otros usuarios, preferiblemente cercanos geográficamente, que estén dispuestos a intercambiar sus libros. Por lo tanto, parece adecuado proponer que la primera pantalla que aparezca tras la autenticación muestre la ubicación de los usuarios que se encuentren más próximos y ofrezca un buscador para facilitar el poder encontrar los libros deseados.

Por otra parte, dado que la finalidad de la aplicación es facilitar el intercambio de libros entre usuarios, se deben proporcionar los medios para gestionar tanto los libros compartidos por cada usuario como los intercambios en los que un usuario participa. Son, por tanto, tareas fundamentales dentro de la aplicación, por lo que también aparecerán dentro del menú de navegación principal, en ambos casos, en forma de listado, de libros e intercambios, respectivamente, para proporcionar un acceso rápido a todos sus elementos.

Finalmente, dado que el usuario debe estar autenticado para acceder a las funcionalidades principales, se le debe proporcionar también una forma rápida de acceder a la desconexión de la aplicación, por lo que esta opción también se incluirá en el menú de navegación lateral.

Dentro de cada opción del menú principal la navegación entre las distintas pantallas se realiza de forma lineal, es decir, mediante avance y retroceso a la inmediatamente anterior.

En la siguiente figura se muestra de forma gráfica el árbol de navegación de la aplicación. Como complemento, y para mejorar su comprensión, se ha incluido como anexo una segunda versión utilizando como base miniaturas de los prototipos en baja fidelidad que se presentarán más adelante:

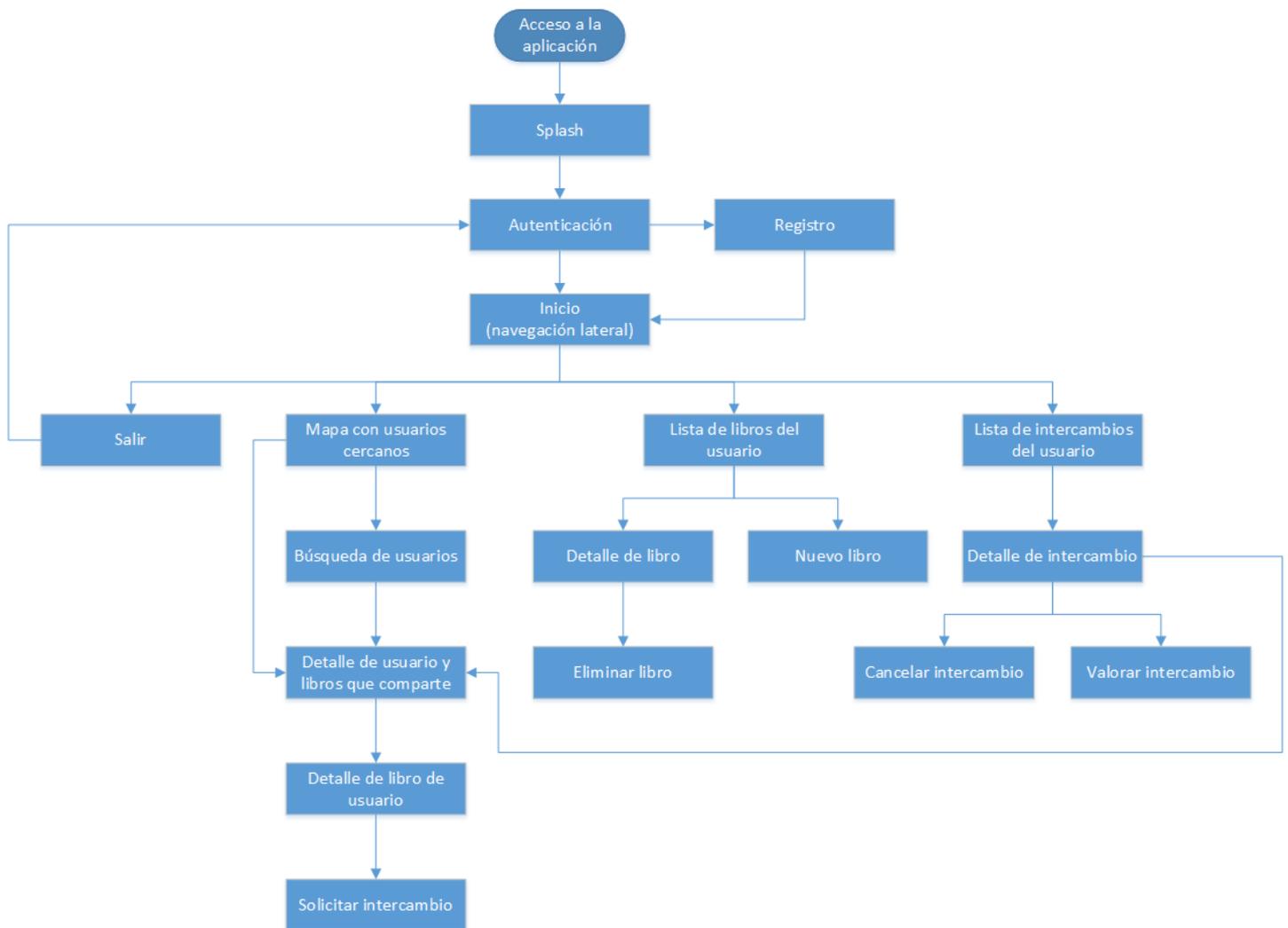


Figura 13: Árbol de navegación

2.5 Prototipado

Tal y como se ha argumentado en la presentación de la aplicación en el primer capítulo, el diseño de la misma se centrará en dispositivos móviles con sistema operativo Android. En base a esta premisa, y siguiendo las directrices sugeridas por Google para dicho sistema operativo [14], se ha tratado de seguir en los prototipos la base de la filosofía de diseño *Material Design*, en cuanto a los componentes empleados y su distribución dentro de la pantalla.

Se muestra a continuación el prototipado de las pantallas de la aplicación en baja fidelidad, agrupado por funcionalidades. Como norma general, por el tipo de información mostrado, y para favorecer la lectura y evitar excesivo desplazamiento vertical al usuario, el diseño propuesto se basará en una orientación vertical o *portrait*. Para aquellas pantallas que hagan uso de la cámara del dispositivo el diseño propuesto se basará en una orientación horizontal o *landscape*, para maximizar el espacio disponible en la pantalla.

2.5.1 Pantalla de presentación (*splash*)

La aplicación se iniciará con una pantalla a modo de presentación, como ocurre con la mayoría de aplicaciones del ecosistema Android. Si fuera necesario realizar algún tipo de inicialización o comprobación que pudiera demorar su arranque se realizaría en esta pantalla. El diseño será simple y tan solo contará con el logo de la aplicación y su nombre, ya que su función es meramente introductoria.

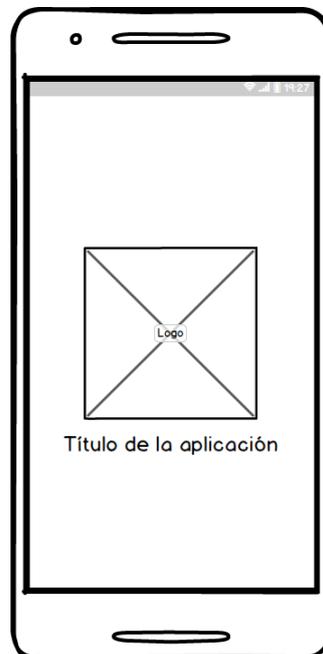


Figura 14: Pantalla de presentación (*splash*)

2.5.2 Autenticación y registro de usuarios

Para poder acceder a las funcionalidades principales de la aplicación los usuarios deberán autenticarse por medio de su correo electrónico y una contraseña. En caso de no disponer aún de una cuenta en el sistema se les ofrecerá la opción de registrarse. La información solicitada para realizar el registro del usuario será la siguiente:

- Nombre y apellidos, que lo identificarán de cara al resto de usuarios de la plataforma.

- Correo electrónico. Será utilizado como nombre de usuario y se empleará para enviarle avisos cuando intercambie algún libro, por lo que debe ser una cuenta existente.
- Contraseña de acceso a la aplicación.
- Provincia de residencia del usuario. Esta información será utilizada para mostrarle otros usuarios cercanos cuando el dispositivo no tenga activado el servicio de localización, es decir, actuará como una posición por defecto.

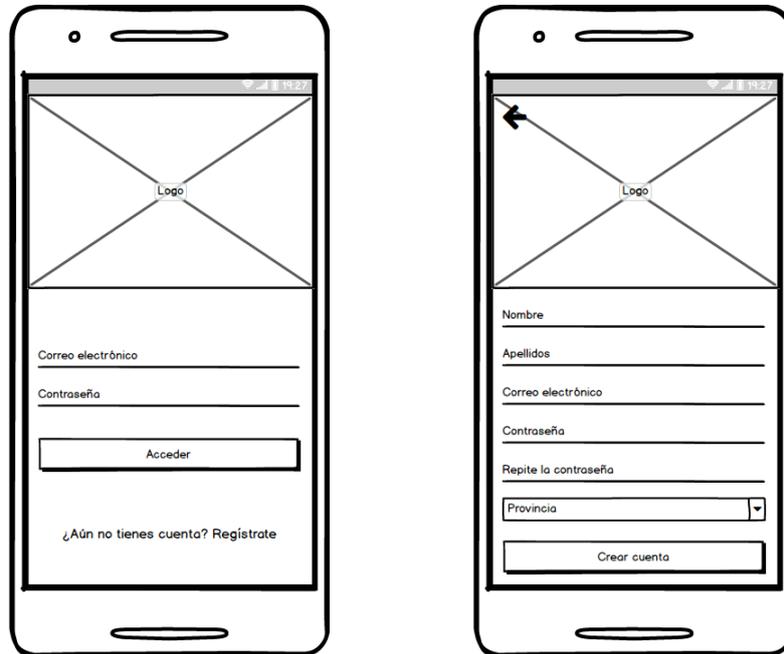


Figura 15: Pantallas de autenticación y registro de usuarios

Durante el proceso de diseño de la navegación se ha valorado también la posibilidad de que el usuario no necesite autenticarse nada más acceder a la aplicación, sino ofrecer de forma pública algunas funcionalidades y requerir el registro más adelante, para que el usuario pueda tomar contacto con ella y se haga una idea de lo que ofrece. No obstante, este planteamiento se ha descartado finalmente por tratarse de una aplicación de tipo colaborativo y que, por tanto, necesita tener un cierto volumen de usuarios para que pueda cumplir su cometido. Al ser de nueva creación y no contar con ninguno inicialmente, con el registro se fuerza a que los usuarios se vayan dando de alta, y de esta forma ir incrementando el censo. Si la aplicación tuviera éxito y se consiguiera una base consolidada de usuarios, como futura ampliación, se podrían publicar determinadas funcionalidades sin necesidad de registro previo como, por ejemplo, el poder visualizar los usuarios cercanos en el mapa, funcionalidad que se describe en el siguiente punto.

2.5.3 Usuarios cercanos y búsqueda de libros

La primera pantalla que se podrá visualizar, una vez se haya completado el proceso de autenticación en la aplicación, será la que muestra los usuarios

próximos geográficamente. Por ser el punto de entrada se ha tratado de diseñar de manera muy visual, mostrándolos sobre un mapa, para que el usuario, con solo un golpe de vista, pueda saber si tiene posibilidad de realizar algún intercambio de manera cercana.

Por el mismo motivo, esta pantalla se utiliza como punto de entrada al resto de funcionalidades de la aplicación. Ello se consigue a través de un menú lateral desplegable desde el que se accede a los bloques principales. Este tipo de menú es característico en las aplicaciones móviles Android basadas en *Material Design* [15] y ofrece una navegación cómoda y fácilmente ampliable en caso de que sea necesario añadir nuevas opciones. Como se verá más adelante, el acceso a este menú se mantiene en la primera pantalla de cada opción, para permitir también la navegación entre opciones principales en todo momento sin necesidad de que el usuario retroceda a la pantalla inicial para tal fin.

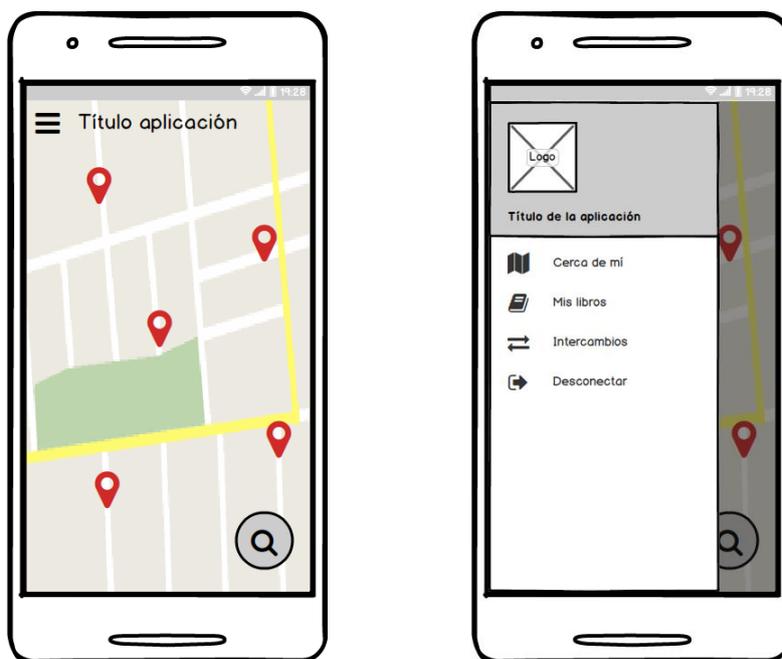


Figura 16: Pantalla de usuarios cercanos y menú principal de navegación

Una de las funcionalidades principales de la aplicación, la búsqueda de libros de otros usuarios, está también disponible desde la pantalla de inicio. Para destacarla se ha utilizado un botón flotante circular, que es el recurso que se suele utilizar para promocionar las acciones principales en los diseños basados en *Material Design* [16]. Al pulsar el botón se mostrará el filtro mediante el cual el usuario podrá acotar la búsqueda. Los campos empleados en el filtro son los siguientes:

- Título, autor y género literario del libro.
- Distancia máxima a la que se encuentra su propietario.

Cuando el usuario utiliza la herramienta de búsqueda los resultados se actualizan en el mapa, mostrándose únicamente los usuarios que se

encuentren dentro de la distancia máxima especificada y que compartan algún libro que verifique los criterios establecidos.



Figura 17: Pantalla de búsqueda de libros

Al tocar sobre alguno de los marcadores de posición ubicados sobre el mapa se visualizará un bocadillo o *callout* con información básica sobre el usuario. Si se pulsa sobre el bocadillo se presentará la pantalla de detalle del usuario seleccionado, en la que aparecerá el listado de libros que comparte, ordenados alfabéticamente por su título, así como datos de interés sobre él:

- Número de libros que comparte.
- Distancia aproximada a la que se encuentra.
- Valoración media de los intercambios en los que ha participado.

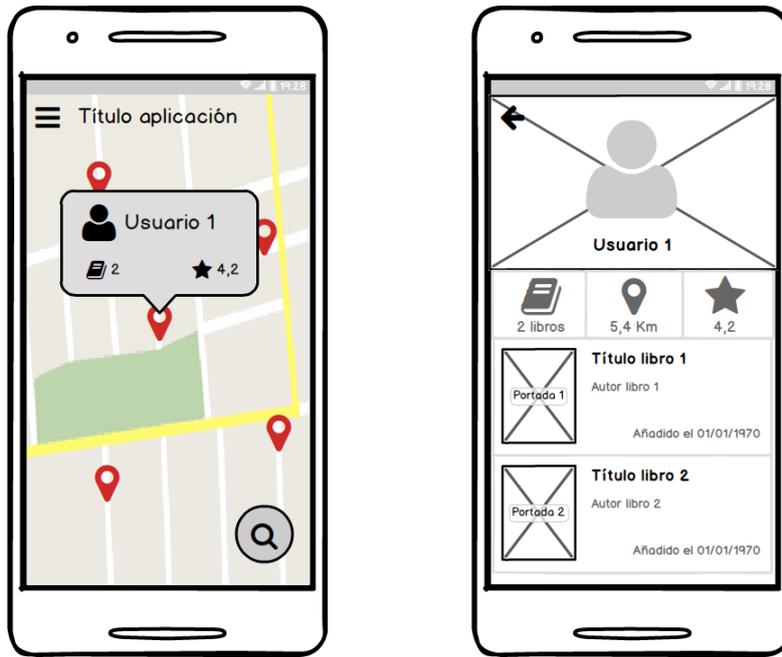


Figura 18: Datos de usuario sobre el mapa y pantalla de detalle de usuario

Los elementos de la lista de libros del usuario llevan a la pantalla de detalle del libro al ser pulsados. Desde dicha pantalla se podrán visualizar todos los datos del mismo y se podrá solicitar al usuario su intercambio con un libro propio. La acción de solicitud del libro se encuentra resaltada mediante un botón flotante circular, dada su importancia. El intercambio no se completará hasta que el otro usuario lo acepte, solicitando uno de los libros compartidos por el usuario iniciador del proceso.

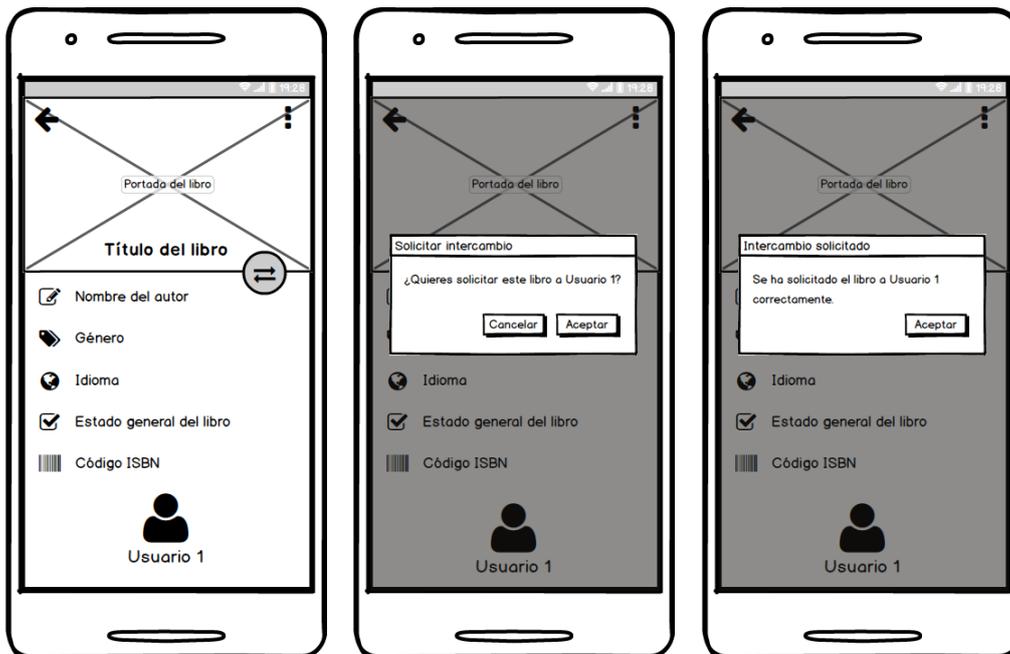


Figura 19: Pantalla de detalle de libro compartido por otro usuario y solicitud del mismo

En todas las pantallas interiores se ofrece la posibilidad de navegar hacia atrás, a la pantalla anterior, por medio de la flecha situada en la parte superior izquierda de la barra de título, tal y como se recomienda en las guías de diseño para aplicaciones Android [17].

2.5.4 Libros compartidos por el usuario

A través de esta funcionalidad el usuario podrá gestionar los libros de su propiedad que comparte con el resto de miembros de la plataforma. Esta característica estará disponible desde el menú principal de la aplicación.

La pantalla principal de libros del usuario mostrará una lista en la que podrá visualizar los que ya tiene registrados, ordenados alfabéticamente por su título. Desde ella se permitirá también:

- Añadir un nuevo libro. Sería la acción principal de la pantalla y, de nuevo, estaría destacada a través de un botón flotante circular, para estar disponible en todo momento, incluso cuando el usuario se desplace verticalmente sobre el listado de libros.
- Ver el detalle de un libro, pulsando sobre el libro deseado. Desde dicha pantalla se permitirá también su eliminación.

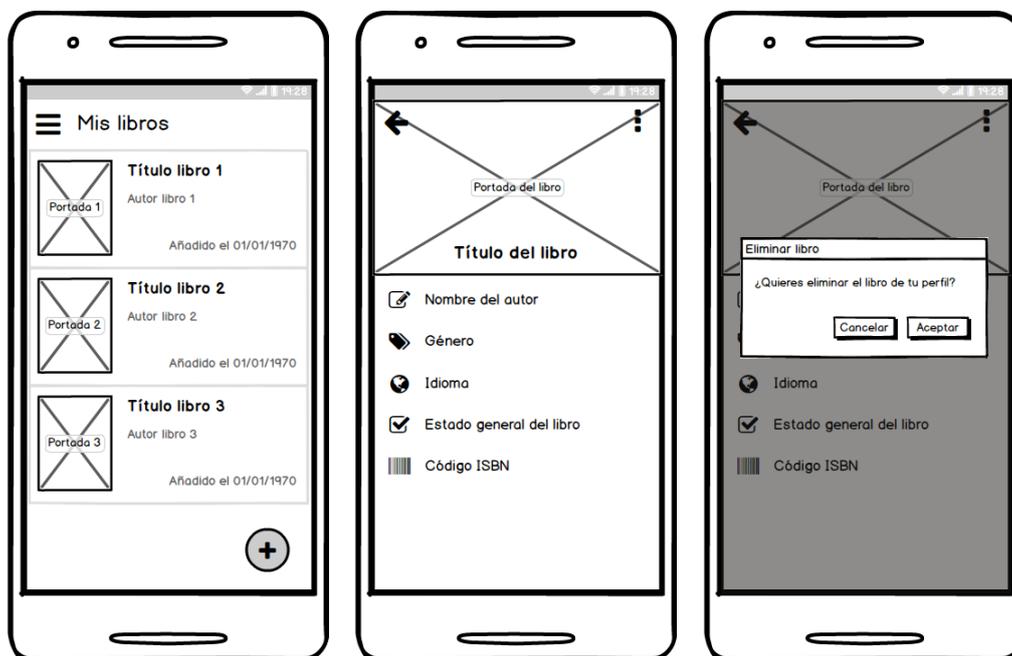


Figura 20: Listado de libros del usuario y pantalla de detalle de libro

A la hora de añadir un libro a su perfil el usuario deberá introducir su título, autor, editorial, género, idioma en el que está escrito, estado de conservación en el que se encuentra y, opcionalmente, podrá adjuntar una fotografía y rellenar su código ISBN. Para cumplimentar esta información dispondrá de dos alternativas:

- Introducir los datos manualmente.

- Cargar los datos automáticamente, leyendo el código ISBN con la cámara del dispositivo.

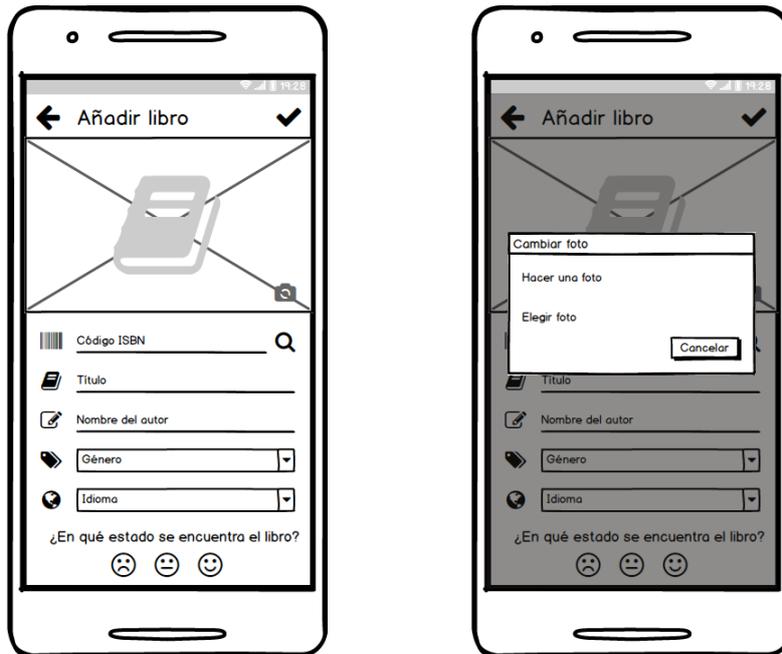


Figura 21: Pantalla de alta de libro con funcionalidad de cambio de fotografía

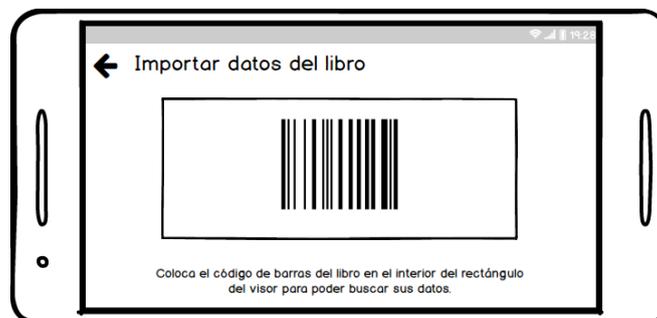


Figura 22: Pantalla de lectura del código ISBN del libro

El sistema de navegación hacia atrás se mantendría igual con respecto al resto de pantallas interiores de la aplicación, mediante la flecha situada en la parte superior izquierda de la barra de título.

2.5.5 Intercambios

La visualización y gestión de los intercambios en los que participa el usuario, bien como iniciador, bien como destinatario, está disponible desde el menú principal de navegación, siendo la pantalla principal de esta sección el listado de intercambios en los que el usuario está o ha estado involucrado, ordenados por fecha decreciente. Cada intercambio se representa mediante una tarjeta con la siguiente información:

- Un icono que indica si el usuario ha iniciado el intercambio (flecha hacia arriba) o es el destinatario del mismo (flecha hacia abajo).
- Título del libro solicitado.
- Nombre del otro usuario involucrado.
- Fecha de inicio del intercambio.
- Código de color que indica el estado del intercambio:
 - Pendiente: el destinatario aún no ha seleccionado el libro a intercambiar.
 - Rechazado: el destinatario ha cancelado el intercambio (el iniciador no tiene buena reputación, no le interesa ninguno de los libros que ofrece, etcétera).
 - Aceptado: el destinatario ha seleccionado el libro a intercambiar.

La pulsación sobre un determinado intercambio lleva al usuario a la pantalla de detalle del mismo, en la que podrá consultar los libros implicados y el estado en el que se encuentra. Los libros se representarán por medio de tarjetas con su fotografía y su título, mientras que el estado se presentará a través de un icono representativo.

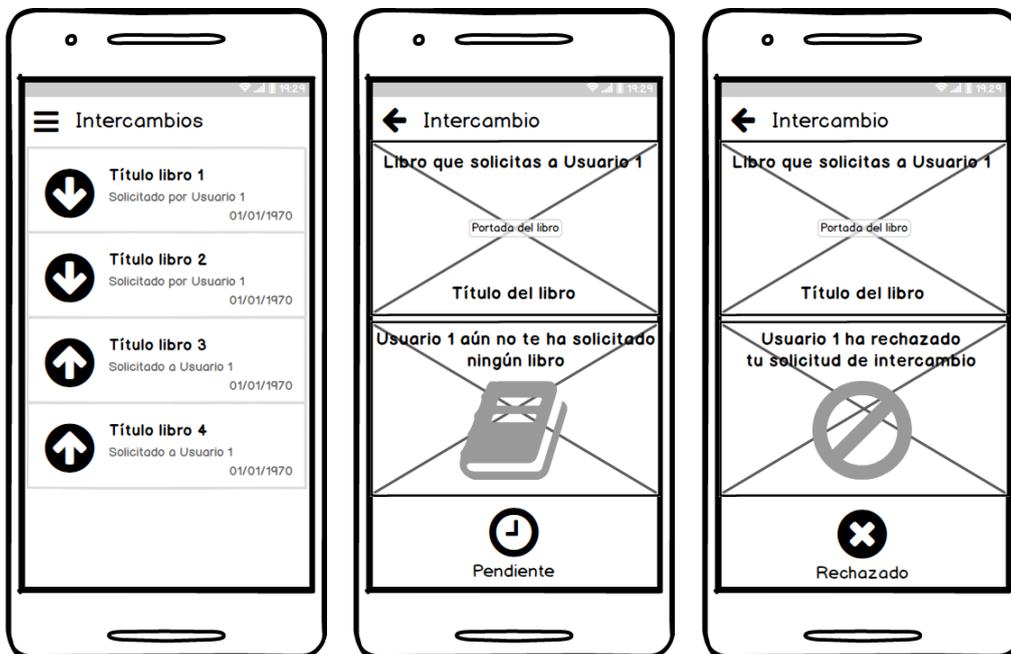


Figura 23: Listado de intercambios y pantalla de detalle de intercambio con el usuario como iniciador

Si el usuario es el destinatario del intercambio, desde la pantalla de detalle podrá seleccionar uno de los libros compartidos por el usuario iniciador, en caso de estar interesado en alguno. Si no es así, podrá rechazar el intercambio.

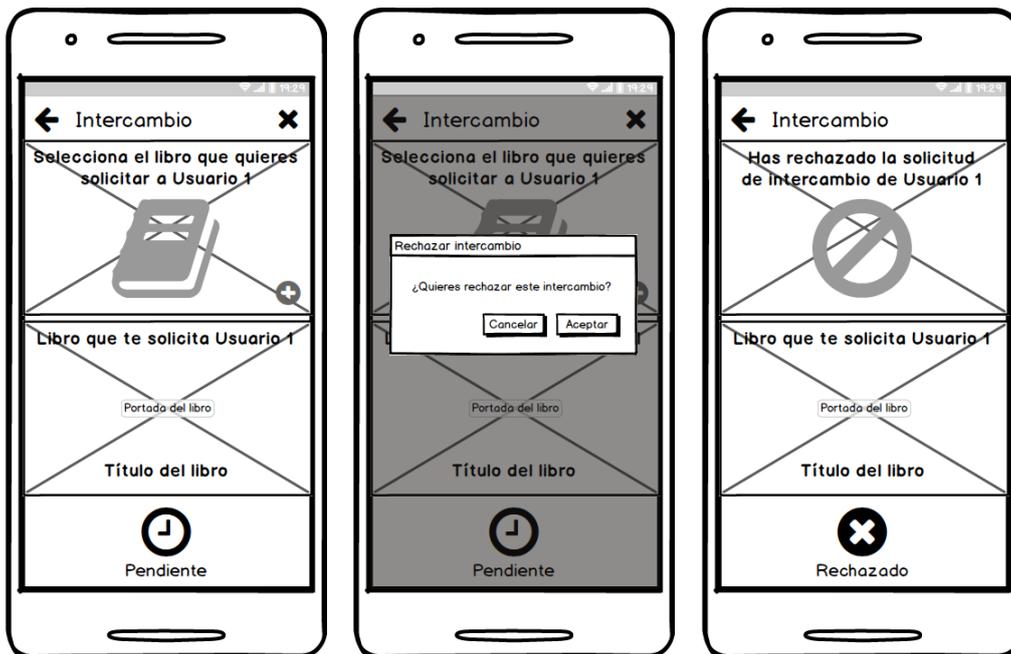


Figura 24: Pantalla de detalle de intercambio con el usuario como destinatario

Si ambos usuarios seleccionan un libro el intercambio tendrá el estado "Aceptado". Solo en ese caso se podrá realizar la valoración del otro usuario, mediante una ventana modal emergente, situada sobre la misma información de detalle, que permitirá otorgar una puntuación entre 1 y 5.

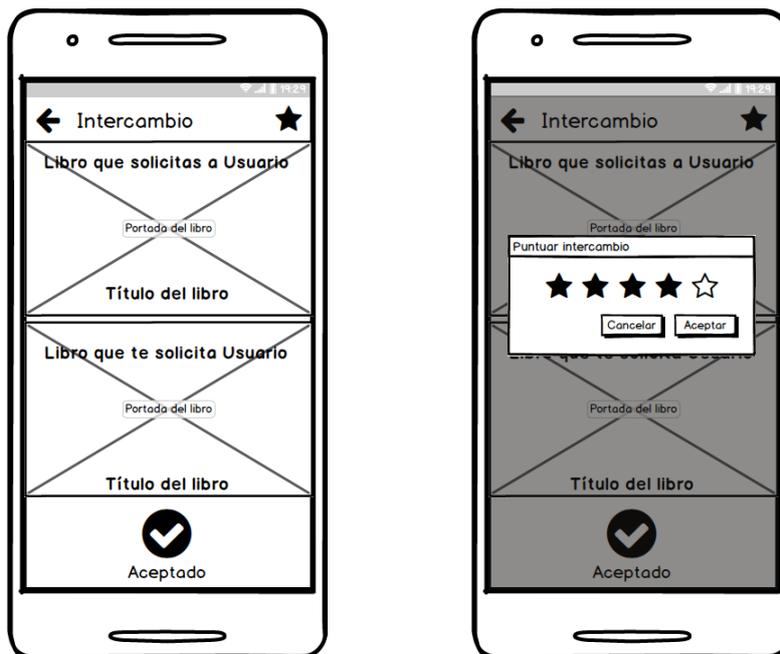


Figura 25: Intercambio aceptado y pantalla de valoración

2.5.6 Línea de diseño y prototipos de alta definición

Si bien los prototipos presentados anteriormente son de baja definición, se han construido de manera que se ajusten lo más posible al aspecto final de la aplicación. No obstante, como complemento a los mismos y para proporcionar una idea aún más realista, se plantea a continuación la línea de diseño que se seguirá durante el desarrollo, tomando como base la paleta de colores empleada en su logotipo y aplicándola a algunas de las pantallas diseñadas en los prototipos de baja definición:

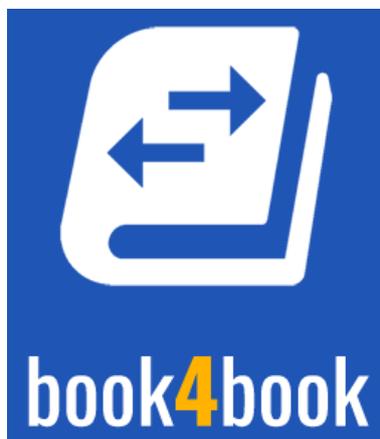


Figura 26: Logotipo de la aplicación

Utilizando como referencia el logotipo la paleta de colores propuesta es la siguiente:

Color primario	rgb(13, 71, 161)	
Color primario claro	rgb(84, 114, 211)	
Color primario oscuro	rgb(0, 33, 113)	
Color secundario	rgb(255, 179, 0)	

Tabla 7: Paleta de colores de la aplicación

Como resultado de aplicar la gama de colores a los diseños presentados, la propuesta de prototipado de alta definición para algunas de las pantallas de la aplicación es la mostrada a continuación:

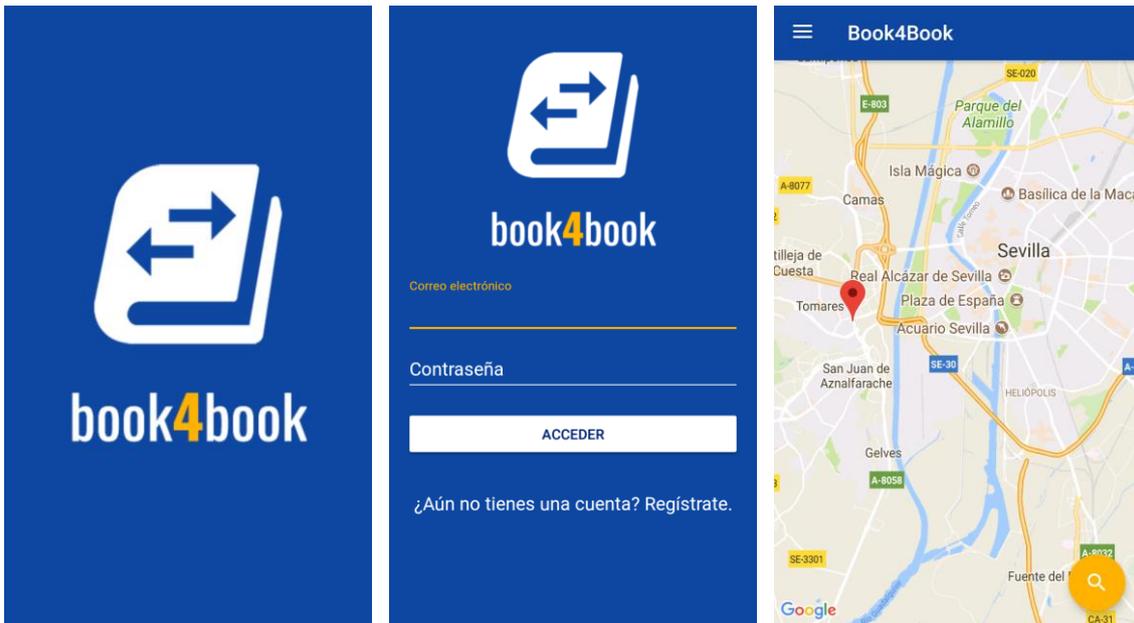


Figura 27: Prototipos de alta definición para las pantallas de *splash*, autenticación y búsqueda sobre mapa

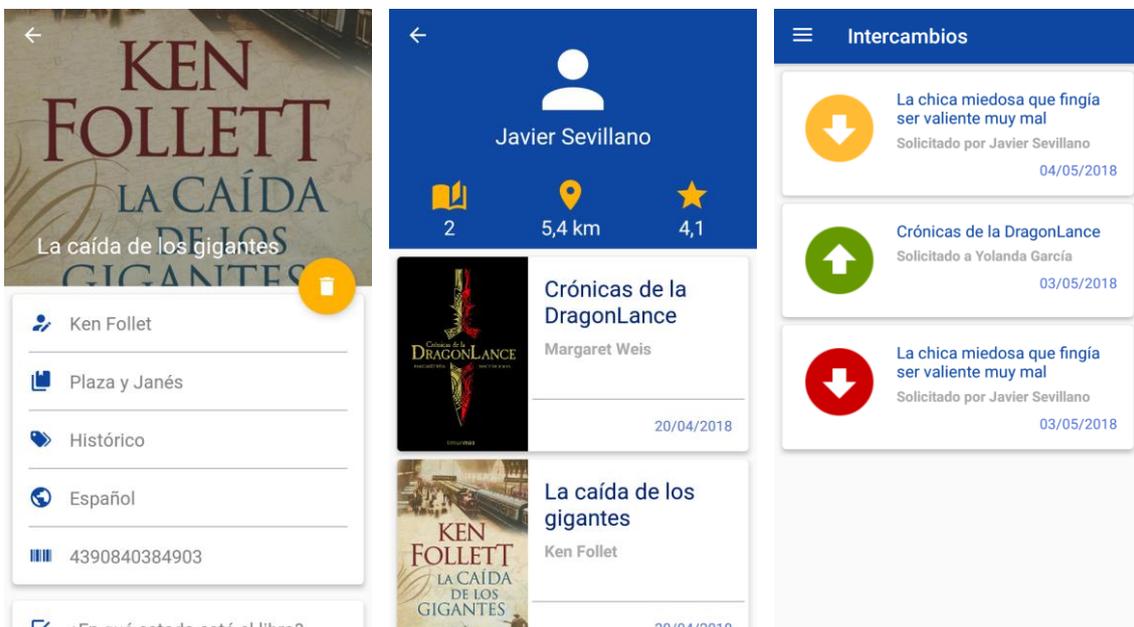


Figura 28: Prototipos de alta definición para las pantallas de detalle de libro, detalle de usuario y listado de intercambios

2.6 Evaluación

Para la evaluación del prototipo se han empleado esencialmente las siguientes técnicas:

- Inspección de estándares. Se ha revisado la interfaz para asegurar su adecuación a las convenciones de diseño e interacción establecidas

para la plataforma Android, basadas en la filosofía *Material Design*. La aplicación de estos estándares, asimilados ya por los usuarios habituales del sistema operativo, asegura una curva de aprendizaje menos pronunciada.

- Test de usuarios. El *feedback* de dos usuarios potenciales de la aplicación, obtenido a partir de su evaluación del árbol de navegación, ha permitido detectar carencias en el diseño inicial:
 - No se había tenido en cuenta la posibilidad de poder eliminar un libro registrado. Gracias a la evaluación se ha podido incorporar esta funcionalidad al diseño.
 - Dificultad excesiva a la hora de interactuar con los intercambios en el diseño planteado inicialmente, en el que intervenían demasiadas pantallas. Por ello, en el diseño propuesto se ha simplificado el proceso para que su uso sea más sencillo y claro para los usuarios.

Al tratarse de un proceso iterativo, durante la implementación de la aplicación se han proporcionado capturas de las diferentes pantallas conforme se han ido desarrollando a los mismos usuarios que han colaborado en la fase de diseño, así como al equipo docente del trabajo, para obtener retroalimentación y poder detectar errores o posibles mejoras de manera temprana.

2.7 Definición de casos de uso

Las necesidades funcionales de la aplicación que se derivan de los escenarios de uso identificados y del diseño de los prototipos se presentan de manera formal en el siguiente diagrama de casos de uso:

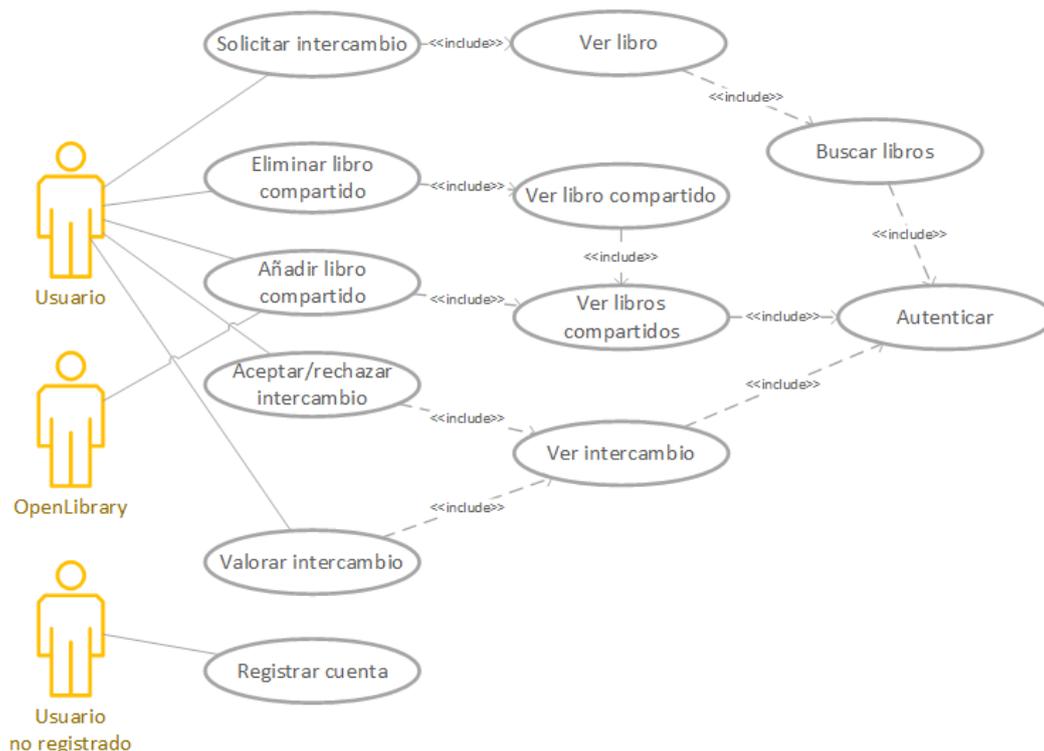


Figura 29: Diagrama de casos de uso

Se identifican tres actores diferentes en la aplicación:

- Usuario no registrado. Representa un usuario que aún no dispone de una cuenta registrada en el sistema.
- Usuario. Representa un usuario que dispone de una cuenta registrada en el sistema.
- OpenLibrary. Representa el servicio externo OpenLibrary, utilizado por la aplicación para obtener los datos de un determinado libro a partir de su código ISBN.

La descripción detallada de cada caso de uso es la mostrada a continuación:

Nombre	Registrar cuenta	CU-001
Descripción	Registrar una nueva cuenta de usuario para poder acceder a la aplicación.	
Actores	Usuario no registrado	
Precondiciones	<ol style="list-style-type: none"> 1. El dispositivo debe tener conexión a Internet. 2. El usuario aún no dispone de una cuenta registrada en el sistema. 	
Flujo básico	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación. 2. El usuario pulsa sobre la opción "Regístrate". 3. El sistema presenta la pantalla de registro, mostrando los campos a rellenar para la creación de la cuenta. 4. El usuario rellena los campos de creación de la cuenta: nombre, apellidos, correo electrónico, contraseña, confirmación de contraseña y localización habitual. 5. El usuario pulsa sobre el botón "Crear cuenta". 6. El sistema valida que los campos introducidos son correctos. 7. El sistema registra la nueva cuenta y autentica al usuario en la aplicación. 	
Flujo alternativo 1	<ol style="list-style-type: none"> 6. Al validar los campos introducidos el sistema detecta que alguno de ellos no es correcto. 7. El sistema muestra un mensaje informando del error para que el usuario pueda corregirlo. 	
Postcondiciones	La cuenta queda registrada en el sistema y el usuario puede utilizarla para acceder a la aplicación.	

Tabla 8: Caso de uso "Registrar cuenta"

Nombre	Autenticar	CU-002
Descripción	Autenticar al usuario en el sistema para que pueda hacer uso de las funcionalidades restringidas de la aplicación.	
Actores	Usuario	
Precondiciones	<ol style="list-style-type: none"> 1. El dispositivo debe tener conexión a Internet. 2. El usuario dispone de una cuenta registrada en el sistema. 	
Flujo básico	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación. 2. El sistema presenta la pantalla de autenticación, mostrando los campos a rellenar. 3. El usuario rellena los campos para autenticarse: correo electrónico, contraseña. 4. El usuario pulsa sobre el botón "Acceder". 5. El sistema valida que los campos introducidos son correctos. 6. El sistema autentica al usuario en la aplicación. 	
Flujo alternativo 1	<ol style="list-style-type: none"> 5. Al validar los campos introducidos el sistema detecta que alguno de ellos no es correcto. 6. El sistema muestra un mensaje informando del error para que el usuario repita de nuevo el proceso. 	
Postcondiciones	El usuario se autentica y accede a la aplicación.	

Tabla 9: Caso de uso "Autenticar"

Nombre	Buscar libros	CU-003
Descripción	Buscar en la aplicación libros compartidos por otros usuarios de la plataforma.	
Actores	Usuario	
Precondiciones	<ol style="list-style-type: none"> 1. El dispositivo debe tener conexión a Internet. 2. El dispositivo debe tener el servicio de geolocalización activado. 3. El usuario dispone de una cuenta registrada en el sistema. 	
Flujo básico	<ol style="list-style-type: none"> 1. El usuario realiza el caso de uso "Autenticar" (CU-002). 2. El sistema presenta un mapa con la ubicación de otros usuarios registrados en el sistema que se encuentran geográficamente próximos. 3. El usuario pulsa sobre alguno de los marcadores de ubicación situados sobre el mapa. 4. El sistema muestra un bocadillo con información básica del usuario seleccionado: nombre, número de libros compartidos y valoración media. 5. El usuario pulsa sobre el bocadillo. 6. El sistema presenta la pantalla de detalle del usuario seleccionado, en la que se muestra la lista de libros que comparte. 	
Flujo alternativo 1	<ol style="list-style-type: none"> 3. El usuario pulsa sobre el botón "Buscar". 4. El sistema presenta una pantalla a través de la cual el usuario puede acotar la búsqueda de libros. 5. El usuario introduce los criterios de búsqueda deseados. 6. El sistema muestra en el mapa la ubicación de otros usuarios que comparten libros que verifican los criterios de búsqueda indicados. 7. El caso de uso continua por el paso 3 del flujo básico. 	
Flujo alternativo 2	<ol style="list-style-type: none"> 2. El sistema no encuentra ningún usuario que se encuentre geográficamente próximo. 3. El sistema muestra un mensaje informando al usuario. 4. El caso de uso continua por el paso 3 del flujo alternativo 1. 	
Postcondiciones	El usuario consulta los libros compartidos por otros usuarios.	

Tabla 10: Caso de uso "Buscar libros"

Nombre	Ver libro	CU-004
Descripción	Ver en la aplicación el detalle de información de un libro compartido por otro usuario.	
Actores	Usuario	
Precondiciones	<ol style="list-style-type: none"> 1. El dispositivo debe tener conexión a Internet. 2. El usuario dispone de una cuenta registrada en el sistema. 	
Flujo básico	<ol style="list-style-type: none"> 1. El usuario realiza el caso de uso "Buscar libros" (CU-003). 2. El usuario pulsa sobre uno de los ítems mostrados en la lista de libros compartidos. 3. El sistema presenta la pantalla de detalle del libro seleccionado con la información relativa al mismo. 	
Postcondiciones	El usuario consulta el detalle de un libro compartido por otro usuario.	

Tabla 11: Caso de uso "Ver libro"

Nombre	Solicitar intercambio	CU-005
Descripción	Solicitar a otro usuario de la aplicación intercambiar uno de los libros que comparte.	
Actores	Usuario	
Precondiciones	<ol style="list-style-type: none"> 1. El dispositivo debe tener conexión a Internet. 2. El usuario dispone de una cuenta registrada en el sistema. 	
Flujo básico	<ol style="list-style-type: none"> 1. El usuario realiza el caso de uso "Ver libro" (CU-004). 2. El usuario pulsa sobre el botón "Solicitar intercambio". 3. El sistema muestra un mensaje de confirmación al usuario. 4. El usuario confirma que desea solicitar el intercambio pulsando sobre el botón "Aceptar". 	

	<ol style="list-style-type: none"> El sistema registra la solicitud de intercambio, que aparecerá en el perfil de los usuarios involucrados con el estado "pendiente". El sistema envía un aviso por correo electrónico al destinatario de la solicitud de intercambio.
Flujo alternativo 1	<ol style="list-style-type: none"> El usuario no confirma la solicitud del intercambio pulsando sobre el botón "Cancelar". El sistema no realiza acción alguna y el caso de uso continúa por el paso 2 del flujo básico.
Postcondiciones	Se registra en el sistema una solicitud de intercambio que aparecerá con el estado "pendiente" para los usuarios involucrados.

Tabla 12: Caso de uso "Solicitar intercambio"

Nombre	Ver libros compartidos	CU-006
Descripción	Ver el conjunto de libros que el usuario comparte con el resto de miembros de la comunidad.	
Actores	Usuario	
Precondiciones	<ol style="list-style-type: none"> El dispositivo debe tener conexión a Internet. El usuario dispone de una cuenta registrada en el sistema. 	
Flujo básico	<ol style="list-style-type: none"> El usuario realiza el caso de uso "Autenticar" (CU-002). El usuario pulsa sobre el botón de menú principal. El sistema despliega el menú principal de la aplicación con las diferentes opciones. El usuario pulsa sobre la opción de menú "Mis libros". El sistema presenta una pantalla que contiene el listado de libros que el usuario está compartiendo con el resto de miembros de la plataforma. 	
Postcondiciones	El usuario consulta por pantalla la lista de libros que está compartiendo con el resto de miembros.	

Tabla 13: Caso de uso "Ver libros compartidos"

Nombre	Añadir libro compartido	CU-007
Descripción	Añadir un libro a la lista de libros que el usuario comparte con el resto de miembros de la comunidad.	
Actores	Usuario, OpenLibrary	
Precondiciones	<ol style="list-style-type: none"> El dispositivo debe tener conexión a Internet. El dispositivo debe contar con una cámara para la captura de imágenes. El usuario dispone de una cuenta registrada en el sistema. El libro no se encuentra ya registrado en el perfil del usuario. 	
Flujo básico	<ol style="list-style-type: none"> El usuario realiza el caso de uso "Ver libros compartidos" (CU-006). El usuario pulsa sobre el botón "Añadir". El sistema presenta la pantalla para dar de alta un nuevo libro. El usuario pulsa sobre el botón "Fotografía" para tomar una imagen del libro con la cámara del dispositivo. El sistema abre la aplicación de cámara. El usuario toma una fotografía del libro y pulsa "Aceptar". El sistema almacena temporalmente la fotografía tomada de forma local. El usuario rellena el resto de campos manualmente con la información del libro: código ISBN, título, autor, editorial, género, idioma y estado en que se encuentra. El usuario pulsa sobre "Aceptar". El sistema valida que los campos introducidos son correctos. El sistema registra el libro en el perfil del usuario. El sistema presenta el listado de libros compartidos por el usuario, que incluye ya el libro recién registrado. 	
Flujo alternativo 1	<ol style="list-style-type: none"> El usuario pulsa sobre el botón "Buscar ISBN". El sistema abre la aplicación de cámara. El usuario captura una imagen del código de barras del libro. 	

	<ol style="list-style-type: none"> 7. El sistema realiza una búsqueda del código ISBN obtenido consultando el servicio OpenLibrary. 8. El servicio OpenLibrary devuelve al sistema los datos del libro asociado al ISBN consultado. 9. El sistema rellena los campos del libro con los datos obtenidos. 10. El usuario rellena de forma manual el campo con el estado en el que se encuentra el libro, el género y el idioma. 11. El caso de uso continua por el paso 9 del flujo básico.
Flujo alternativo 2	<ol style="list-style-type: none"> 4. El usuario pulsa sobre el botón "Buscar ISBN". 5. El sistema abre la aplicación de cámara. 6. El usuario captura una imagen del código de barras del libro. 7. El sistema realiza una búsqueda del código ISBN obtenido consultando el servicio OpenLibrary. 8. El servicio OpenLibrary informa al sistema de que no encuentra el libro asociado al ISBN consultado. 9. El sistema muestra un mensaje al usuario indicando que no ha sido posible obtener los datos del libro y que es necesario introducirlos manualmente. 10. El caso de uso continua por el paso 4 del flujo básico.
Flujo alternativo 3	<ol style="list-style-type: none"> 10. Al validar los campos introducidos el sistema detecta que alguno de ellos no es correcto. 11. El sistema muestra un mensaje informando del error para que el usuario repita de nuevo el proceso.
Postcondiciones	El nuevo libro queda registrado en el perfil del usuario y está disponible para compartir con otros miembros de la plataforma.

Tabla 14: Caso de uso "Añadir libro compartido"

Nombre	Ver libro compartido	CU-008
Descripción	Ver en la aplicación el detalle de información de un libro compartido por el usuario.	
Actores	Usuario	
Precondiciones	<ol style="list-style-type: none"> 1. El dispositivo debe tener conexión a Internet. 2. El usuario dispone de una cuenta registrada en el sistema. 3. El usuario dispone de algún libro registrado en su perfil. 	
Flujo básico	<ol style="list-style-type: none"> 1. El usuario realiza el caso de uso "Ver libros compartidos" (CU-006). 2. El usuario pulsa sobre uno de los ítems mostrados en su lista de libros compartidos. 3. El sistema presenta la pantalla de detalle del libro seleccionado con la información relativa al mismo. 	
Postcondiciones	El usuario consulta el detalle de uno de sus libros compartidos.	

Tabla 15: Caso de uso "Ver libro compartido"

Nombre	Eliminar libro compartido	CU-009
Descripción	Eliminar de la lista de libros compartidos uno de los libros del usuario.	
Actores	Usuario	
Precondiciones	<ol style="list-style-type: none"> 1. El dispositivo debe tener conexión a Internet. 2. El usuario dispone de una cuenta registrada en el sistema. 3. El usuario dispone de algún libro registrado en su perfil. 	
Flujo básico	<ol style="list-style-type: none"> 1. El usuario realiza el caso de uso "Ver libro compartido" (CU-008). 2. El usuario pulsa sobre el botón "Eliminar". 3. El sistema muestra un mensaje de confirmación al usuario. 4. El usuario confirma que desea eliminar el libro pulsando sobre el botón "Aceptar". 5. El sistema elimina el libro del perfil del usuario. 6. El sistema presenta el listado de libros compartidos por el usuario, que no incluye ya el libro recién eliminado. 	
Flujo alternativo 1	<ol style="list-style-type: none"> 4. El usuario no confirma la eliminación del libro pulsando sobre el botón "Cancelar". 5. El sistema no realiza acción alguna sobre el libro y el caso de uso 	

	continúa por el paso 2 del flujo básico.
Postcondiciones	El libro se elimina del sistema y, por tanto, del perfil del usuario.

Tabla 16: Caso de uso "Eliminar libro compartido"

Nombre	Ver intercambio	CU-010
Descripción	Ver detalle de un intercambio en el que el usuario ha participado, bien como solicitante o bien como receptor de la solicitud de otro usuario.	
Actores	Usuario	
Precondiciones	<ol style="list-style-type: none"> 1. El dispositivo debe tener conexión a Internet. 2. El usuario dispone de una cuenta registrada en el sistema. 3. El usuario ha participado en algún intercambio. 	
Flujo básico	<ol style="list-style-type: none"> 1. El usuario realiza el caso de uso "Autenticar" (CU-002). 2. El usuario pulsa sobre el botón de menú principal. 3. El sistema despliega el menú principal de la aplicación con las diferentes opciones. 4. El usuario pulsa sobre la opción de menú "Intercambios". 5. El sistema presenta una pantalla que contiene el listado de intercambios en los que el usuario ha participado. 6. El usuario pulsa sobre uno de los ítems mostrados en la lista de intercambios. 7. El sistema presenta la pantalla de detalle del intercambio seleccionado con la información relativa al mismo. 	
Postcondiciones	El usuario consulta el detalle de un intercambio en el que ha participado, ya sea como iniciador o como destinatario del mismo.	

Tabla 17: Caso de uso "Ver intercambio"

Nombre	Aceptar/rechazar intercambio	CU-011
Descripción	Aceptar o rechazar el intercambio de un libro propio que ha sido solicitado por otro usuario.	
Actores	Usuario	
Precondiciones	<ol style="list-style-type: none"> 1. El dispositivo debe tener conexión a Internet. 2. El usuario dispone de una cuenta registrada en el sistema. 3. El usuario está compartiendo algún libro en su perfil. 	
Flujo básico	<ol style="list-style-type: none"> 1. El usuario realiza el caso de uso "Ver intercambio" (CU-010), accediendo a la información de un intercambio del que es destinatario. 2. El usuario pulsa sobre el botón "Añadir libro". 3. El sistema presenta una pantalla con la lista de libros compartidos por el usuario que inició el intercambio. 4. El usuario pulsa sobre uno de los ítems mostrados en la lista de libros compartidos. 5. El sistema presenta la pantalla de detalle del libro seleccionado con la información relativa al mismo. 6. El usuario pulsa sobre el botón "Solicitar intercambio". 7. El sistema muestra un mensaje de confirmación al usuario. 8. El usuario confirma que desea solicitar el intercambio pulsando sobre el botón "Aceptar". 9. El sistema actualiza el intercambio registrando el libro solicitado. 10. El sistema actualiza el estado del intercambio a "Aceptado". 11. El sistema envía un aviso por correo electrónico al usuario iniciador de la solicitud de intercambio. 12. El sistema presenta la pantalla de detalle del intercambio con los datos actualizados. 	
Flujo alternativo 1	<ol style="list-style-type: none"> 8. El usuario no confirma la solicitud de intercambio pulsando sobre el botón "Cancelar". 9. El sistema no realiza acción alguna sobre el intercambio. 10. El usuario pulsa sobre el botón "Retroceder". 11. El sistema presenta la pantalla con la lista de libros compartidos por el usuario iniciador del intercambio. 	

	<p>12. El usuario pulsa sobre el botón "Retroceder".</p> <p>13. El sistema presenta la pantalla de detalle del intercambio.</p> <p>14. El usuario pulsa sobre el botón "Rechazar intercambio".</p> <p>15. El sistema muestra un mensaje de confirmación al usuario.</p> <p>16. El usuario confirma que desea rechazar el intercambio pulsando sobre el botón "Aceptar".</p> <p>17. El sistema actualiza el intercambio colocando su estado como "Rechazado".</p> <p>18. El sistema envía un aviso por correo electrónico al usuario iniciador de la solicitud de intercambio.</p> <p>19. El sistema presenta la pantalla de detalle del intercambio con los datos actualizados.</p>
Flujo alternativo 2	<p>16. El usuario no confirma el rechazo del intercambio pulsando sobre el botón "Cancelar".</p> <p>17. El sistema no realiza acción alguna sobre el intercambio y el caso de uso continua por el paso 2 del flujo básico.</p>
Postcondiciones	El usuario acepta o rechaza un intercambio que le solicita otro miembro de la plataforma, pasando su estado a ser "Aceptado" o "Rechazado", respectivamente.

Tabla 18: Caso de uso "Aceptar/rechazar intercambio"

Nombre	Valorar intercambio	CU-012
Descripción	Valorar el nivel de satisfacción con un determinado usuario tras haber realizado un intercambio con él.	
Actores	Usuario	
Precondiciones	<p>1. El dispositivo debe tener conexión a Internet.</p> <p>2. El usuario dispone de una cuenta registrada en el sistema.</p> <p>3. El intercambio debe haber sido aceptado por ambas partes.</p> <p>4. El intercambio no ha sido valorado anteriormente ya por el usuario.</p>	
Flujo básico	<p>1. El usuario realiza el caso de uso "Ver intercambio" (CU-010), accediendo a la información de un intercambio aceptado por ambas partes.</p> <p>2. El usuario pulsa sobre el botón "Valorar".</p> <p>3. El sistema presenta un campo para que el usuario introduzca la valoración.</p> <p>4. El usuario introduce la valoración y pulsa el botón "Aceptar".</p> <p>5. El sistema muestra un mensaje de confirmación al usuario.</p> <p>6. El usuario confirma que desea enviar la valoración pulsando sobre el botón "Aceptar".</p> <p>7. El sistema registra la valoración incluyéndola en el perfil del otro usuario implicado en el intercambio.</p>	
Flujo alternativo 1	<p>4. El usuario pulsa sobre el botón "Cancelar".</p> <p>5. El sistema no realiza acción alguna y el caso de uso continua por el paso 2 del flujo básico.</p>	
Flujo alternativo 2	<p>6. El usuario no confirma el envío de la valoración pulsando sobre el botón "Cancelar".</p> <p>7. El sistema no realiza acción alguna y el caso de uso continua por el paso 2 del flujo básico.</p>	
Postcondiciones	La valoración queda registrada en el perfil del otro usuario implicado en el intercambio.	

Tabla 19: Caso de uso "Valorar intercambio"

2.8 Diseño de la arquitectura

2.8.1 Cliente-servidor

Para dar soporte a las funcionalidades diseñadas se hace necesario implementar una arquitectura cliente-servidor, que permitirá que los diferentes usuarios compartan la información entre ellos. La aplicación planteada en el trabajo actuará como parte cliente, instalada en los terminales móviles de los diferentes usuarios, mientras que la parte de servidor se pondrá en funcionamiento a través de un servicio de *backend* en la nube, o BaaS (*backend as a service*). Este tipo de servicio proporciona un SDK unificado que permite acceder a las diferentes funcionalidades y gestionar los datos del *backend* de manera coherente, evitando el despliegue de la infraestructura necesaria para los servidores y el desarrollo de una API personalizada. La consecuencia inmediata de este planteamiento es que se reduce el tiempo que es necesario invertir en la parte de servidor, permitiendo poner el foco en la propia aplicación.

Existen múltiples servicios de *backend* en la nube, como Firebase, AWS o Microsoft Azure. No obstante, la alternativa que se utilizará en el ámbito del presente trabajo será Backendless, ya que dispone de un plan de uso gratuito y los servicios que ofrece dan cobertura a las funcionalidades diseñadas para la aplicación:

- Registro y autenticación de usuarios.
- Persistencia de la información en base de datos.
- Permisos sobre los datos, para limitar la visibilidad de los mismos entre los diferentes usuarios.
- Almacenamiento de archivos.
- Geolocalización y medición de distancias entre puntos geográficos.
- Mensajería a través de correo electrónico y notificaciones *push*.

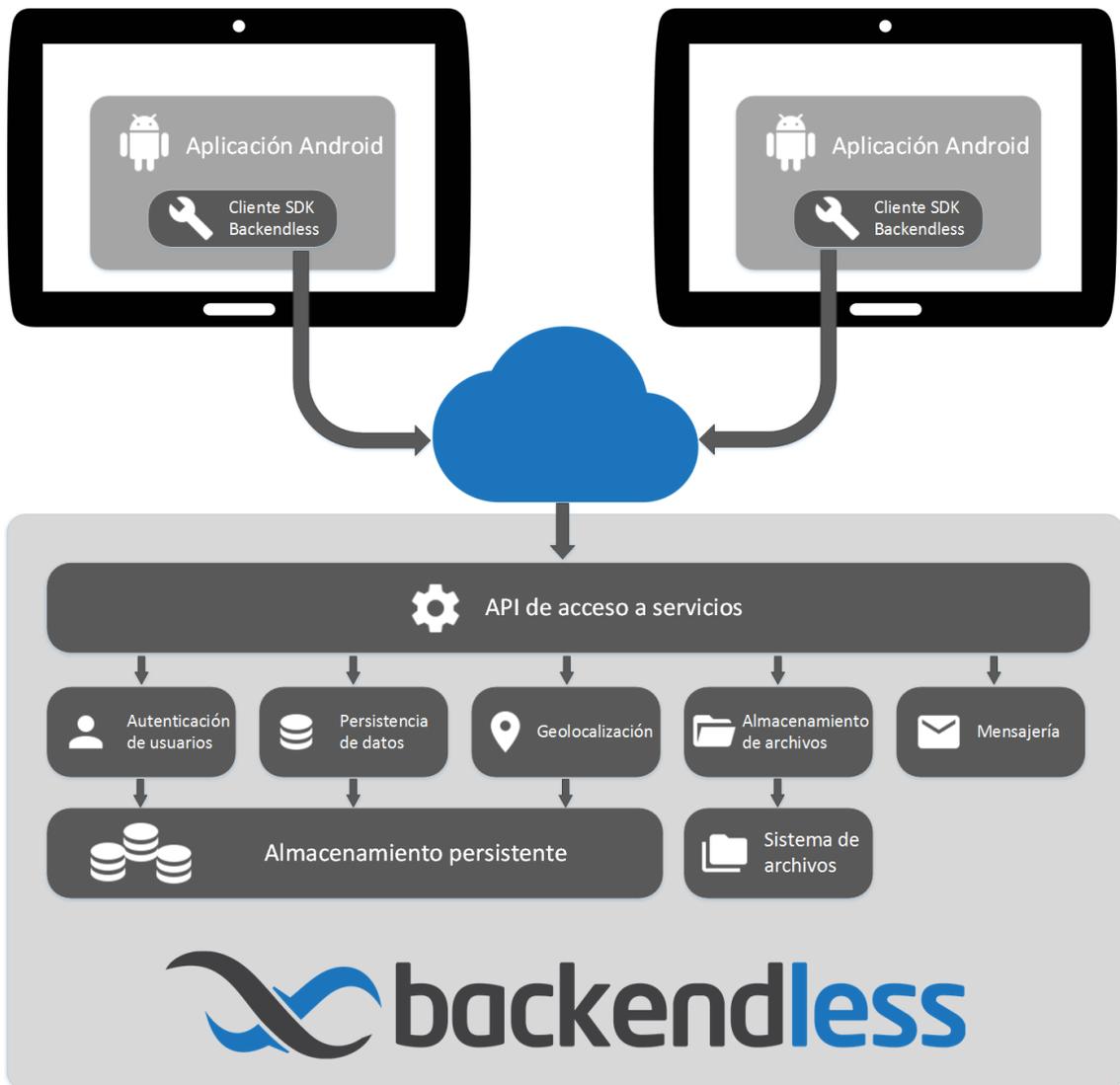


Figura 30: Arquitectura cliente-servidor

2.8.2 Modelo-Vista-Presentador

A la hora de codificar una aplicación es muy importante hacerlo de manera que su mantenimiento sea sencillo y que se pueda ampliar de forma fácil. La manera de llevar esto a cabo es definiendo capas bien diferenciadas que se encarguen solamente de la responsabilidad que tienen asignada. Esta filosofía es la que persigue el patrón de arquitectura de software modelo-vista-controlador (MVC), que separa de forma clara los datos y la lógica de negocio de su representación de cara al usuario en tres componentes:

- **Modelo.** Se encarga de la consulta y gestión de los datos (bases de datos, servicios web, sistema de archivos, etcétera), así como las reglas de negocio.
- **Vista.** Su responsabilidad es la representación visual de los datos del modelo.

- Controlador. Responde a los eventos lanzados por el usuario, solicitando los datos al modelo y comunicándoselos a la vista.

La idea subyacente, por tanto, es limitar la responsabilidad de cada componente de manera disjunta. Sin embargo, MVC presenta una complicación en una aplicación Android, y es que existen componentes de la propia plataforma, como las actividades y los fragmentos, que pueden actuar como vista y como controlador, con lo que se acaban mezclando responsabilidades y empobreciendo la calidad del código.

Para ayudar a evitar este acoplamiento entre controlador y vista, y favorecer la separación del código, surge el patrón modelo-vista-presentador (MVP). Se trata de una variación de MVC, más orientada a interfaces gráficas, que propone la división de la aplicación en las siguientes capas:

- Modelo. Su función es la misma que en MVC, es decir, actúa como proveedor de los datos que se muestran en la vista, sin embargo, solo se comunica con el presentador.
- Vista. Al igual que en MVC, se encarga de presentar visualmente los datos del modelo, pero en este caso se limita a invocar al presentador cada vez que se produce un evento en la interfaz de usuario.
- Presentador. Es el responsable de actuar como intermediario entre el modelo y la vista, obteniendo los datos de aquel y enviándoselos a ésta ya formateados. A diferencia de MVC, se encarga también de decidir lo que ocurre cuando el usuario interactúa con la vista. El presentador no tiene ningún tipo de conocimiento de cómo están implementados internamente la vista y el modelo.

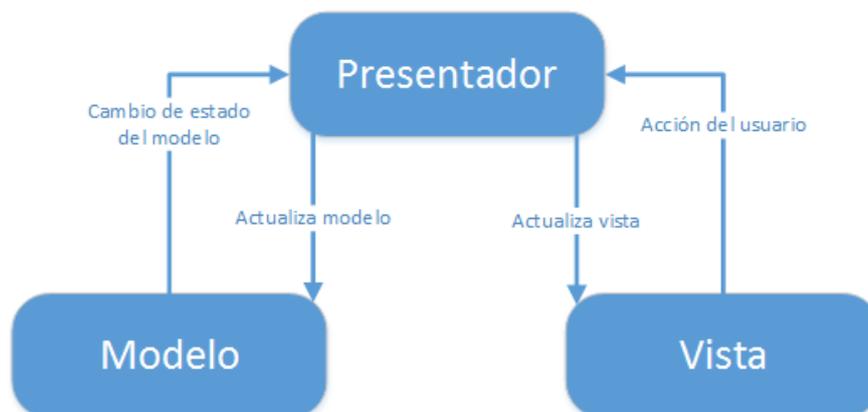


Figura 31: Diagrama del patrón MVP

Dado que el patrón MVP facilita el desacople entre las diferentes capas y se adapta mejor al desarrollo en la plataforma Android, será el que se adopte en la implementación de la aplicación.

2.8.3 Entidades y base de datos

Con el objeto de dar respuesta al conjunto de casos de uso planteados, las entidades que se manejarán en la aplicación, representadas de forma gráfica, serán las siguientes:

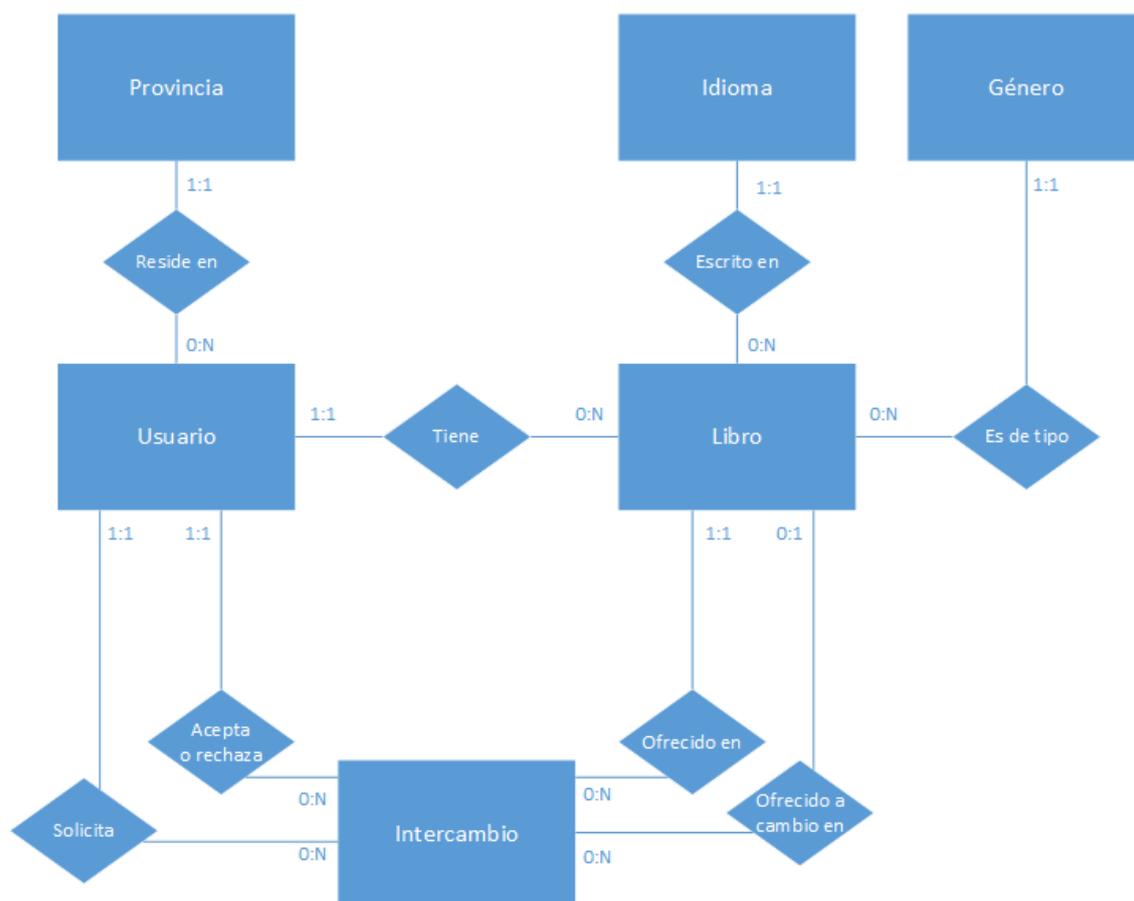


Figura 32: Diagrama entidad-relación

Como se puede apreciar en el diagrama existirá un conjunto de usuarios dados de alta en la aplicación, cada uno de los cuales registrará en el sistema un número indeterminado de libros. Dichos libros serán intercambiados entre los usuarios, circunstancia que quedará almacenada por medio de la entidad “Intercambio”. En cada intercambio intervienen dos libros, uno por cada usuario involucrado, de ahí la doble relación libro-intercambio. La relación usuario-intercambio es redundante, ya que a partir de un libro puede determinarse de forma unívoca su usuario propietario, pero se ha añadido por motivos de eficiencia y para facilitar las consultas al *backend*. Se trata también de una relación doble, por los mismos motivos expresados anteriormente.

Las entidades “Género”, “Idioma” y “Provincia” representan conjuntos de datos fijos, no actualizables desde la aplicación móvil, por lo que las tablas que se deriven de ellas actuarán como maestras.

De manera más detallada, las entidades estarán compuestas por los siguientes campos:

Usuario				
Representa un usuario registrado en la aplicación				
Campo	Tipo	Clave primaria	Obligatorio	Descripción
id	Entero	Sí	Sí	Clave primaria
email	Texto	No	Sí	Correo electrónico del usuario
password	Texto	No	Sí	Contraseña del usuario
nombre	Texto	No	Sí	Nombre de pila del usuario
apellidos	Texto	No	Sí	Apellidos del usuario
id_provincia	Provincia	No	Sí	Provincia de residencia del usuario. Se utilizará como ubicación aproximada cuando el usuario no haya activado nunca el servicio de geolocalización en su dispositivo
latitud	Decimal	No	No	Latitud de la coordenada geográfica de la última localización conocida del usuario
longitud	Decimal	No	No	Longitud de la coordenada geográfica de la última localización conocida del usuario

Tabla 20: Entidad "Usuario"

Libro				
Representa un libro compartido por un usuario en la plataforma				
Campo	Tipo	Clave primaria	Obligatorio	Descripción
id	Entero	Sí	Sí	Clave primaria
isbn	Texto	No	No	Código ISBN del libro
titulo	Texto	No	Sí	Título del libro
autor	Texto	No	Sí	Nombre del autor del libro
editorial	Texto	No	Sí	Editorial que publica el libro
url_fotografia	Texto	No	No	URL de descarga de la imagen del libro en el entorno de Backendless
id_usuario	Usuario	No	Sí	Usuario propietario del libro
id_idioma	Idioma	No	No	Idioma en que está escrito el libro
id_genero	Género	No	No	Género al que pertenece el libro
estado	Entero	No	Sí	Estado de conservación en el que se encuentra el libro: 0: Malo 1: Regular 2: Bueno
fecha_alta	Fecha	No	Sí	Fecha en la que el usuario registra el libro en la plataforma

Tabla 21: Entidad "Libro"

Intercambio				
Representa un intercambio de libros entre dos usuarios de la plataforma				
Campo	Tipo	Clave primaria	Obligatorio	Descripción
id	Entero	Sí	Sí	Clave primaria
fecha	Fecha	No	Sí	Fecha en la que se realiza la solicitud de intercambio
estado	Entero	No	Sí	Estado del intercambio: 0: Pendiente 1: Aceptado 2: Rechazado
id_usuario_origen	Usuario	No	Sí	Usuario iniciador de la solicitud de intercambio
id_libro_origen	Libro	No	No	Libro solicitado al usuario iniciador del intercambio
valoracion_origen	Entero	No	No	Valoración recibida por el usuario iniciador del intercambio
id_usuario_destino	Usuario	No	Sí	Usuario destinatario de la solicitud de intercambio
id_libro_destino	Libro	No	Sí	Libro solicitado al usuario destinatario del intercambio
valoracion_destino	Entero	No	No	Valoración recibida por el usuario destinatario del intercambio

Tabla 22: Entidad "Intercambio"

Idioma				
Representa un idioma en el que puede estar escrito un libro				
Campo	Tipo	Clave primaria	Obligatorio	Descripción
id	Entero	Sí	Sí	Clave primaria
nombre	Texto	No	Sí	Nombre del idioma

Tabla 23: Entidad "Idioma"

Género				
Representa un género al que puede pertenecer un libro (fantástico, terror, policíaco, etcétera)				
Campo	Tipo	Clave primaria	Obligatorio	Descripción
id	Entero	Sí	Sí	Clave primaria
nombre	Texto	No	Sí	Nombre del género

Tabla 24: Entidad "Género"

Provincia				
Representa una provincia de residencia de un usuario registrado. Se utilizará para acotar las búsquedas de libros por proximidad cuando los usuarios no tengan última localización conocida				
Campo	Tipo	Clave primaria	Obligatorio	Descripción
id	Entero	Sí	Sí	Clave primaria
nombre	Texto	No	Sí	Nombre de la provincia

Tabla 25: Entidad "Provincia"

Gráficamente, y a modo de resumen, el detalle de las entidades participantes en el sistema y el modo en que se relacionan entre ellas es el siguiente:

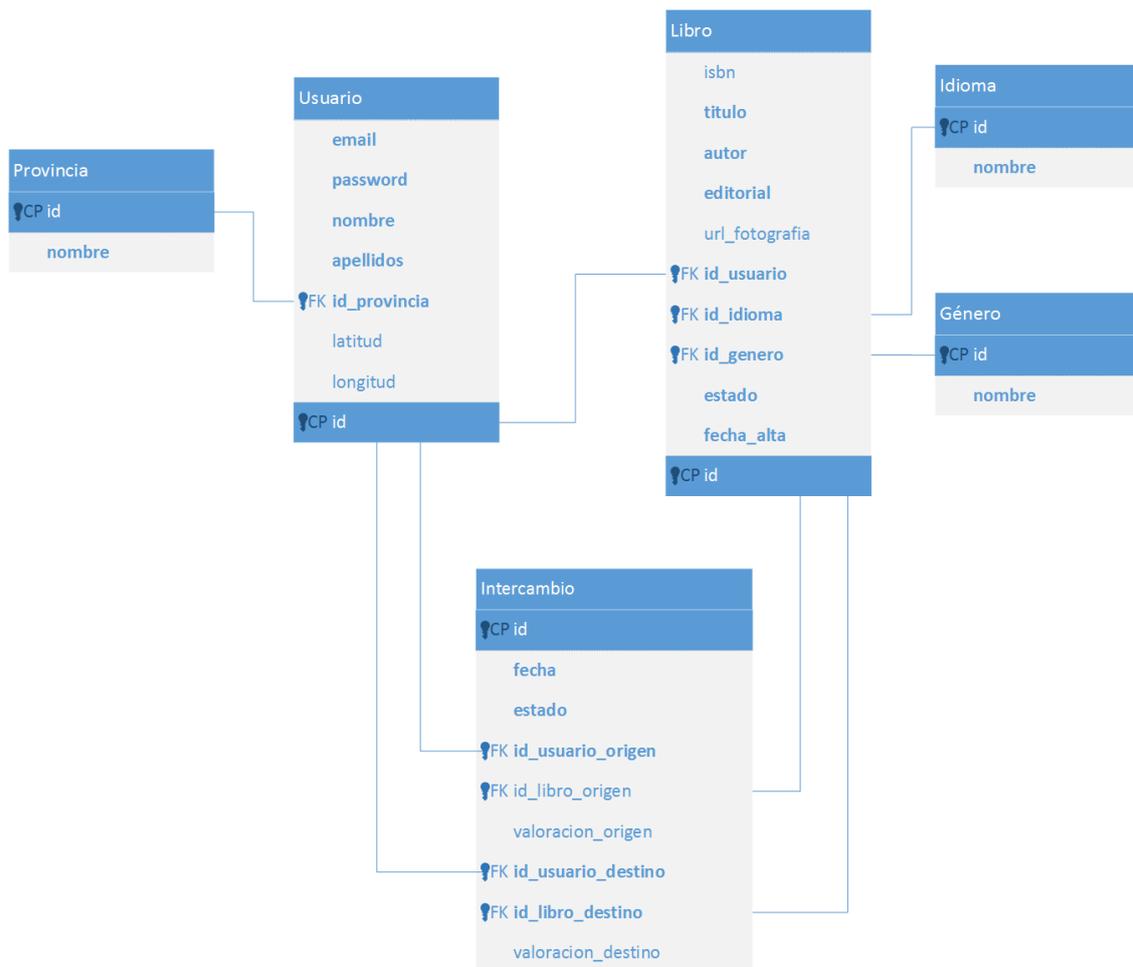


Figura 33: Diagrama de entidades

3. Implementación

3.1 Arquitectura

Tal y como se ha propuesto en la fase de diseño, la implementación de la aplicación se ha realizado siguiendo un patrón modelo-vista-presentador, que se ha materializado siguiendo las pautas mostradas a continuación:

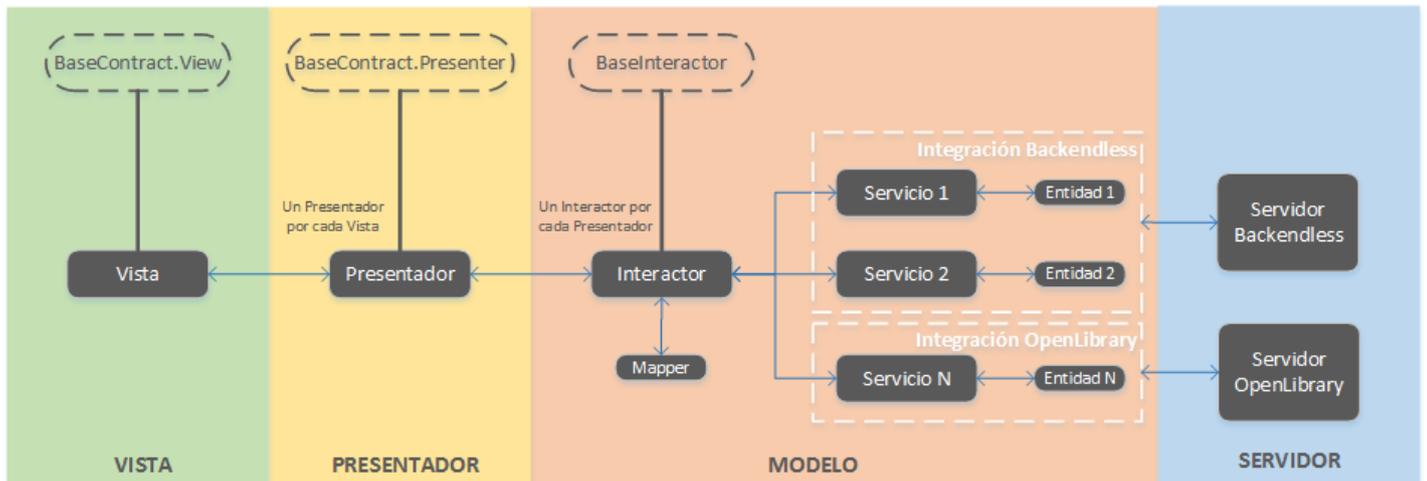


Figura 34: Implementación del patrón MVP en la aplicación

La vista se encarga simplemente de mostrar la información al usuario y trasladar los eventos que éste genera al presentador. La comunicación entre ambos se realiza por medio de interfaces, para que ninguno de ellos conozca los detalles de implementación del otro, y favorecer, de esta forma, el desacoplamiento. En el caso concreto de la aplicación, la vista es implementada por componentes de la plataforma Android, es decir, actividades (*activities*), fragmentos (*fragments*) y adaptadores (*adapters*). Cada uno de ellos dispone de su propio presentador e implementa una interfaz, que extiende a una interfaz común *BaseContract.View*, a través de la cual se puede comunicar con él.

El presentador recibe los eventos desde la vista, actualiza el modelo, si es necesario, y comunica de nuevo a la vista los cambios que debe realizar en la interfaz para informar al usuario. Existe un presentador por cada vista, realizándose la comunicación con ella por medio de una interfaz, que en todos los casos extiende a una interfaz común, *BaseContract.Presenter*. Al utilizarse interfaces para interactuar con la vista los presentadores son totalmente independientes de la plataforma, no teniendo conocimiento alguno sobre Android.

La comunicación del presentador con el modelo se realiza por medio de componentes denominados interactores. Existe un interactor por cada presentador, que le proporciona una interfaz de comunicación para poder manipular únicamente aquellos datos del modelo que son estrictamente necesarios para las funcionalidades que ofrece. De esta forma se delimitan mejor las competencias de cada presentador, al mismo tiempo que se consigue

modularizar la implementación del modelo. Como en los casos anteriores, la interfaz implementada por cada interactor extiende a una interfaz común, *BaseInteractor*, que permite ocultar a los presentadores el origen de los datos.

La manipulación de las entidades de negocio recae sobre los servicios, que son clases concretas que implementan la comunicación con los servidores de *backend*. Para evitar la generación de un único servicio monolítico se han desarrollado varios servicios más pequeños, cada uno de los cuales gestiona los procesos relacionados con una determinada entidad de negocio (usuarios, libros, intercambios, etcétera). Con ello se consigue también que cada interactor acceda únicamente a los servicios que realmente requiere para proporcionar la información a su correspondiente presentador. En el caso concreto de la aplicación del trabajo las clases de servicio acceden a dos orígenes de datos diferentes:

- API de Backendless, a través de la cual se consulta y persiste la información manejada por la aplicación.
- API de OpenLibrary, que permite consultar una base de datos abierta de libros para obtener información sobre los mismos por medio de su código ISBN.

Las entidades de negocio no traspasan el modelo. Al ser devueltas al interactor desde los servicios éste se encarga de transformarlas en entidades de interfaz de usuario por medio de componentes mapeadores (*mappers*). Estos objetos simples son los que se trasladan al presentador y a la vista, consiguiéndose así independizar totalmente la presentación de la implementación del modelo.

El anterior esquema se repite en los diferentes módulos de la aplicación, dado que existe una vista, como mínimo, por cada pantalla desarrollada, cada vista requiere de un presentador y éste, a su vez, dispone de su propio interactor. Más adelante, en el apartado de aspectos relevantes del desarrollo, se detallará la relación entre los distintos componentes dentro de cada módulo.

3.2 Herramientas y APIs utilizadas

Durante la fase de implementación se han utilizado diferentes herramientas y APIs para llevar a cabo el desarrollo de la aplicación. En los siguientes apartados se profundizará en las más importantes:

3.2.1 Backendless API

La API de Backendless permite acceder a los diferentes servicios ofrecidos por la plataforma BaaS. El acceso a la misma se realiza por medio de un SDK nativo proporcionado por el propio proveedor del servicio. Los módulos del SDK empleados por la aplicación son:

User Service API: Proporciona las funcionalidades relativas a la gestión de cuentas de usuario, tales como el registro en la aplicación, la autenticación o el

cierre de sesión. También provee de mecanismos para restringir el acceso a la información en función de los roles asignados al usuario, si bien, no se utilizan en el ámbito de la aplicación. La API proporciona una entidad genérica, *BackendlessUser*, que representa un usuario de la aplicación y cuyas propiedades pueden ser definidas por cada aplicación concreta.

Data Service API: Se trata de un sistema de almacenamiento de objetos altamente escalable, que comparte características de las bases de datos relacionales y de las bases de datos no estructuradas NoSQL. Los objetos son persistidos en tablas, cuyas columnas tienen asignado un determinado tipo de dato. Estos objetos se trasladan a la aplicación mediante clases cuyos campos se mapean con las columnas de las diferentes tablas.

Messaging API: Permite el envío de correos electrónicos a los usuarios cuando éstos intercambian libros. En este sentido, Backendless actúa como un mero intermediario, ya que los correos se envían a través del servidor configurado en la consola de administración del servicio, que en el caso de la aplicación es una cuenta de Gmail. No obstante, el punto fuerte del servicio de mensajería es su capacidad para publicar notificaciones *push* a los dispositivos suscritos, sin embargo, finalmente no se ha hecho uso de él, ya que la funcionalidad de envío de notificaciones, marcada en la planificación del proyecto como opcional, no se ha podido implementar por falta de tiempo.

Files API: Permite acceder al área remota de almacenamiento de archivos que la aplicación tiene dedicada en los servidores. Se trata de un servicio de almacenamiento potente y simple que permite subir y descargar contenido generado por los usuarios en la aplicación, en concreto, las fotografías asociadas a los libros que comparten.

Geolocation API: El servicio de geolocalización de Backendless es un sistema que permite el almacenamiento y búsqueda de coordenadas geográficas. La API ofrece funciones específicas que facilitan en gran medida la búsqueda de puntos que se encuentran dentro de una determinada distancia, lo cual encaja perfectamente con la funcionalidad de búsqueda de libros ofrecida en la aplicación, reduciendo los cálculos a realizar.

Todas las llamadas a la API de Backendless se realizan en segundo plano, para no bloquear el hilo principal de ejecución de la aplicación, recibándose las respuestas por medio de retrollamadas o *callbacks*.

La configuración de la conexión con el servicio se realiza en la clase de aplicación *B4BApplication*, ubicada en el paquete principal. En ella se establecen el identificador de la aplicación y la clave de API asignados por el servicio *BaaS*.

3.2.2 OpenLibrary API

OpenLibrary es un proyecto de código abierto cuyo objetivo es crear una base de datos de libros colaborativa y de acceso público. Actualmente consta de unos 30 millones de títulos. El proyecto proporciona una API REST, de acceso libre, que permite realizar consultas sobre la base de datos y obtener información sobre los libros en formato JSON. Entre las capacidades de la API se encuentra la de poder realizar búsquedas por código ISBN, circunstancia que se ha aprovechado en la aplicación desarrollada en el trabajo para facilitar a los usuarios la introducción de libros. En etapas iniciales se propuso la utilización de Google Books para tal fin, pero en el momento de desarrollar la funcionalidad se ha comprobado que OpenLibrary ha encontrado más resultados que Google Books en búsquedas por ISBN de libros en español, por lo que, finalmente, se ha descartado esta última y se ha optado por la primera.

3.2.3 Retrofit

Se trata de una librería de código abierto para desarrollar de manera sencilla clientes HTTP de tipo seguro. A través de esta herramienta se pueden realizar peticiones a APIs REST para obtener respuestas en formato JSON o XML y convertirlas automáticamente en objetos Java. Dado que la API de OpenLibrary puede consumirse por medio de peticiones HTTP de tipo GET, y a diferencia de Backendless no proporciona una herramienta nativa para hacerlo, se ha utilizado Retrofit como base para generar el cliente de esta plataforma y poder consultar los libros de la misma por su código ISBN.

3.2.4 ZXing Android Embedded

ZXing es un proyecto de código abierto cuyo objetivo es el desarrollo de una aplicación Android que permite que un dispositivo con cámara pueda escanear códigos de barras y códigos 2D y recupere los datos codificados en ellos. La librería ZXing Android Embedded es un proyecto paralelo, también de código abierto, basado en el núcleo de la aplicación desarrollada por ZXing, que permite integrar la funcionalidad de escaneo de códigos en aplicaciones de terceros de manera independiente. La librería ofrece soporte para decodificar multitud de formatos, incluidos los utilizados en las contraportadas de los libros para almacenar su ISBN, circunstancia que se ha aprovechado en la aplicación, junto con la API de OpenLibrary, para facilitar al usuario la introducción de los datos de libros.

3.2.5 Picasso

Se trata de una librería libre de código abierto que permite la carga de imágenes de manera rápida y eficiente. El uso de Picasso ayuda a que la carga se realice de manera optimizada en las listas de elementos, y se utiliza ampliamente en la aplicación para descargar de los servidores de Backendless las fotografías de los libros registrados en el sistema.

3.2.6 Google Maps API

Esta API permite incorporar en la aplicación mapas basados en los datos almacenados en Google Maps. La aplicación la utiliza para mostrarle al usuario de manera visual la localización de otros usuarios que están compartiendo libros a una determinada distancia de su ubicación.

3.2.7 Google Places API

Da la posibilidad de consultar información de los sitios almacenados en la base de datos de Google Maps y Google Local Guides, lo que permite recuperar los datos de millones de ubicaciones. Entre las funcionalidades ofrecidas por la API se encuentra la del autocompletado de sitios, que predice el nombre y/o dirección de un determinado lugar geográfico mientras el usuario lo escribe. Esta característica se emplea en la aplicación en el momento en que un usuario se registra, para que pueda proporcionar una ubicación por defecto que se utilizará en caso de que no desee activar los servicios de geolocalización del dispositivo para determinar su posición actual.

3.2.8 Espresso

Espresso es un *framework* de testing de código abierto de Google que provee una API para generar pruebas de interfaz de usuario simulando interacciones de usuarios en una aplicación Android. La principal ventaja de este *framework* radica en que permite la sincronización automática de las acciones de las pruebas con la interfaz de usuario de la aplicación, permitiendo detectar de manera anticipada resultados inesperados, tanto en dispositivos físicos como virtuales (emuladores).

3.3 Aspectos relevantes del desarrollo

3.3.1 Estructura del proyecto de Android Studio

La aplicación ha sido desarrollada de forma nativa con el IDE Android Studio 3.1.2, siendo compatible con dispositivos que dispongan de una versión del sistema operativo Android 4.4 – KitKat (API 19) o superior. Atendiendo a las estadísticas de instalación de Android publicadas por Google, esta elección permite que la aplicación funcione en el 95% del total de dispositivos activos que cuentan con el sistema operativo [\[18\]](#):

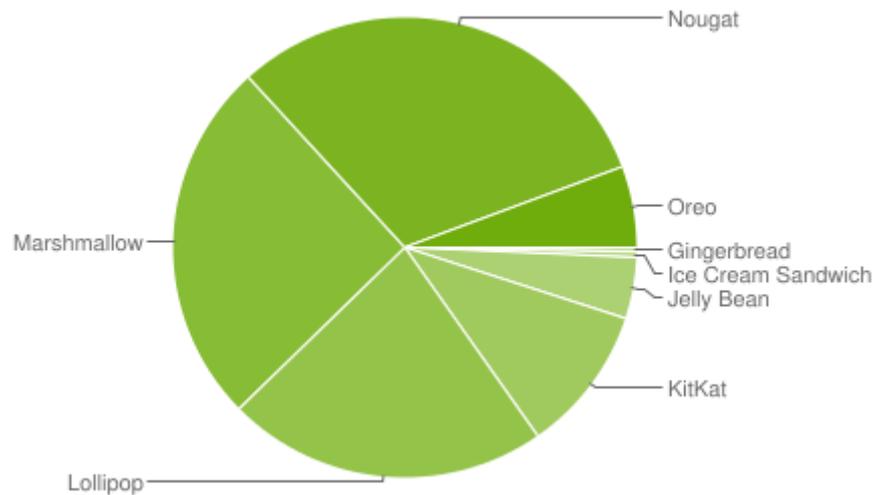


Figura 35: Estadísticas de instalación de Android por versión [18]

La selección de la versión mínima ha venido condicionada por la librería ZXing Android Embedded empleada en la aplicación, ya que la versión utilizada de la misma requiere un nivel de API del sistema operativo igual o superior al 19.

El proyecto se ha estructurado en tres subpaquetes que cuelgan del paquete principal, *edu.uoc.jsev.b4b*:

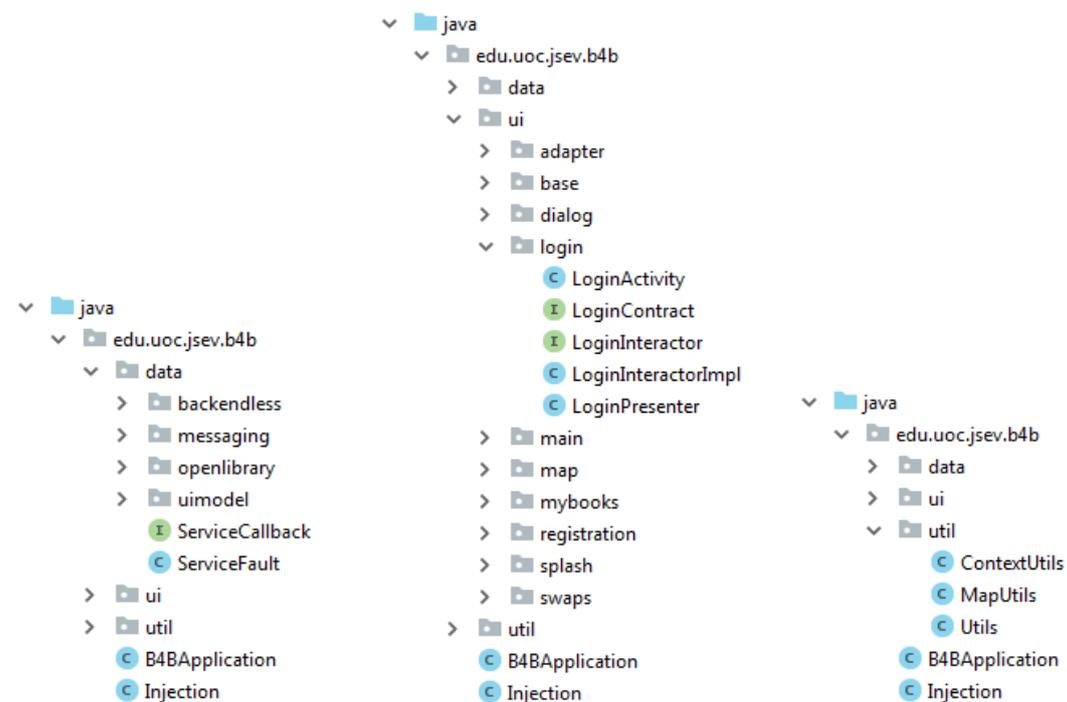


Figura 36: Estructura de los paquetes principales del proyecto

util: Incluye clases auxiliares con utilidades generales empleadas por el resto de componentes del proyecto, tales como validación de textos o formateo de cadenas y fechas.

data: Almacena las clases e interfaces de servicio que se comunican con los servidores de *backend* para la consulta y persistencia de las entidades de la aplicación. En este paquete se encuentran también las entidades de negocio y las entidades de interfaz de usuario que se trasladan a los presentadores, así como los componentes que permiten realizar el mapeo de un tipo de entidad a otro.

ui: Agrupa todas las clases e interfaces de la interfaz de usuario de las distintas pantallas de la aplicación. Dentro del paquete se ha generado un subpaquete por cada pantalla, que a su vez contiene las clases e interfaces que la implementan siguiendo el patrón MVP. La estructura de los subpaquetes es siempre la misma, contando cada uno de ellos con los siguientes elementos:

- Interfaces de la vista (*View*) y del presentador (*Presenter*), incluidas como internas en una interfaz común denominada contrato (*Contract*). Todas las vistas y presentadores descienden de una interfaz base común, *BaseContract.View* y *BaseContract.Presenter*, respectivamente.
- Clase de implementación de la vista. Dependiendo de cada pantalla esta clase será una actividad o un fragmento. Todas las actividades y fragmentos descienden de una clase base que implementa una serie de funcionalidades comunes a todas ellas: *BaseActivity* y *BaseFragment*, respectivamente.
- Clase de implementación del presentador utilizado en la vista correspondiente. Desciende siempre de una clase base *BasePresenter*.
- Interfaz del interactor empleado por el presentador para comunicarse con el *backend*. Desciende siempre de una interfaz base, *BaseInteractor*, que ofrece una serie de funcionalidades comunes a todos los interactores.
- Clase de implementación concreta del interactor asociado al presentador del paquete.

Adicionalmente a los correspondientes a cada pantalla, este paquete incorpora los siguientes subpaquetes:

- **adapter:** Incluye las clases adaptadoras (*adapters*) utilizadas para mostrar datos en el mapa y en los listados de elementos.
- **base:** Agrupa las clases e interfaces base de las que extienden todas las actividades, fragmentos, vistas, presentadores e interactores de la aplicación.
- **dialog:** Contiene la implementación de una ventana de diálogo genérica utilizada a lo largo de toda la aplicación para solicitar la confirmación al usuario de determinadas acciones, tales como el borrado de libros o la solicitud de intercambios, entre otras.

Se muestra a continuación de manera gráfica el diagrama que refleja las relaciones entre clases e interfaces que se producen en cada subpaquete de la interfaz de usuario. Las clases e interfaces propias del paquete aparecen en el diagrama con el prefijo genérico *Package*, que se correspondería con el nombre de cada subpaquete concreto. Se representa de esta forma por simplicidad, para no generar un diagrama individual para cada uno de ellos:

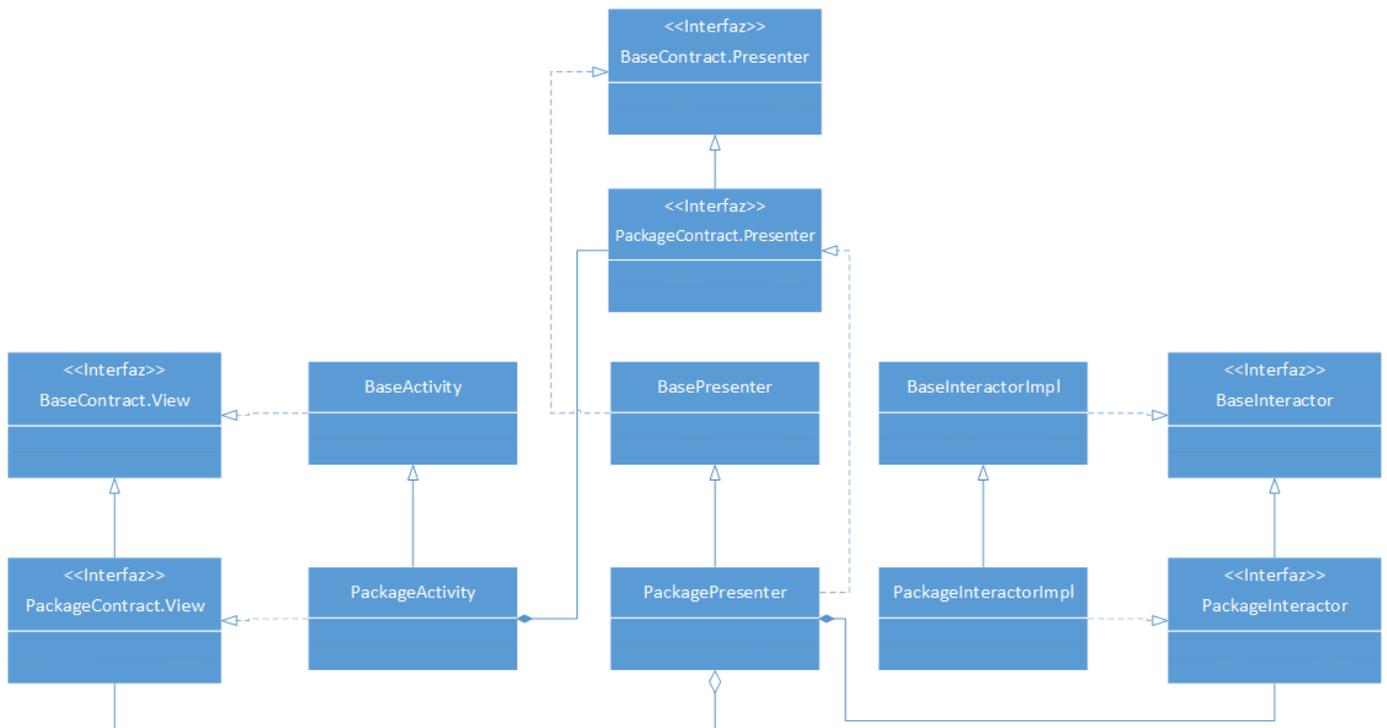


Figura 37: Diagrama de clases de la implementación de MVP por subpaquete

Los subpaquetes de la interfaz de usuario, correspondientes a cada una de las pantallas de la aplicación, son los siguientes:

- **login:** Autenticación en la aplicación.
- **main:** Pantalla principal, que se encarga de gestionar el menú de navegación y la desconexión del usuario (*logout*).
- **map:** Mapa para representar la posición de otros usuarios próximos.
 - **search:** Filtro de búsqueda de usuarios sobre el mapa.
 - **userdetail:** Pantalla de detalle de un usuario de la aplicación, con el listado de libros que comparte.
- **mybooks:** Listado de libros compartidos por el usuario.
 - **bookdetail:** Detalle de un libro compartido.
 - **newbook:** Pantalla para registrar un libro nuevo en el sistema.
- **registration:** Pantalla de creación de una nueva cuenta de acceso a la aplicación.
- **splash:** Pantalla de presentación.
- **swaps:** Listado de intercambios en los que participa el usuario, ya sea como solicitante o como destinatario.
 - **swapdetail:** Información de detalle de un intercambio.

Se muestra a continuación la relación existente entre las diferentes clases de la aplicación desde una perspectiva horizontal, es decir, tomando como referencia cada una de las capas del patrón MVP:

Vistas



Figura 38: Diagrama de clases de las vistas de la aplicación

Presentadores

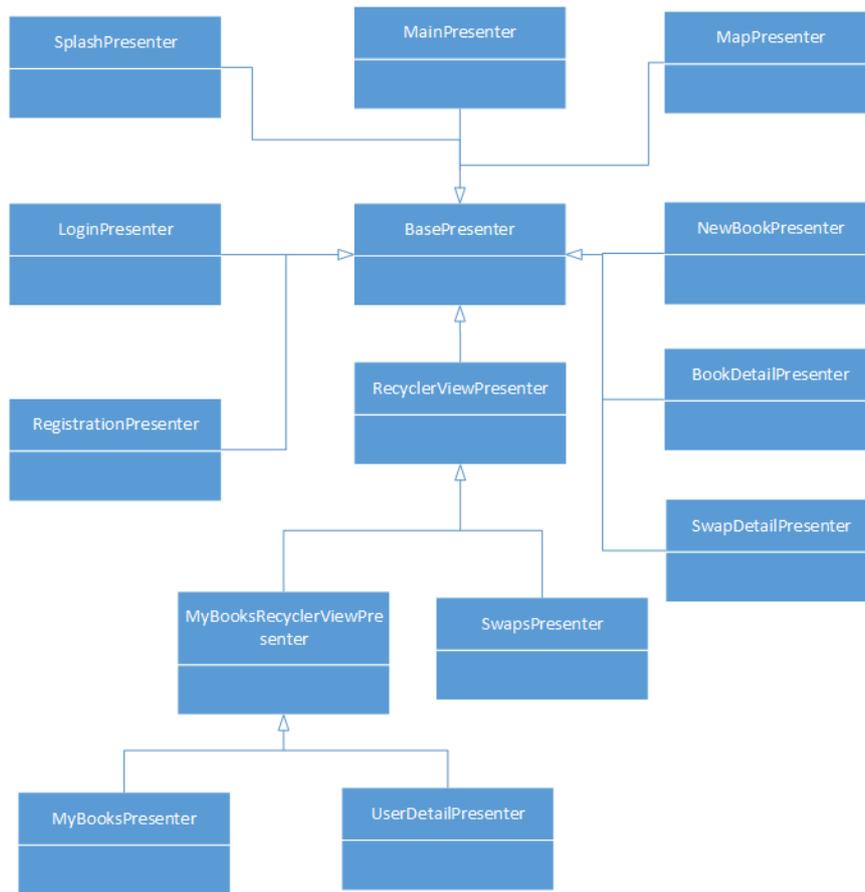


Figura 39: Diagrama de clases de los presentadores de la aplicación

Modelo

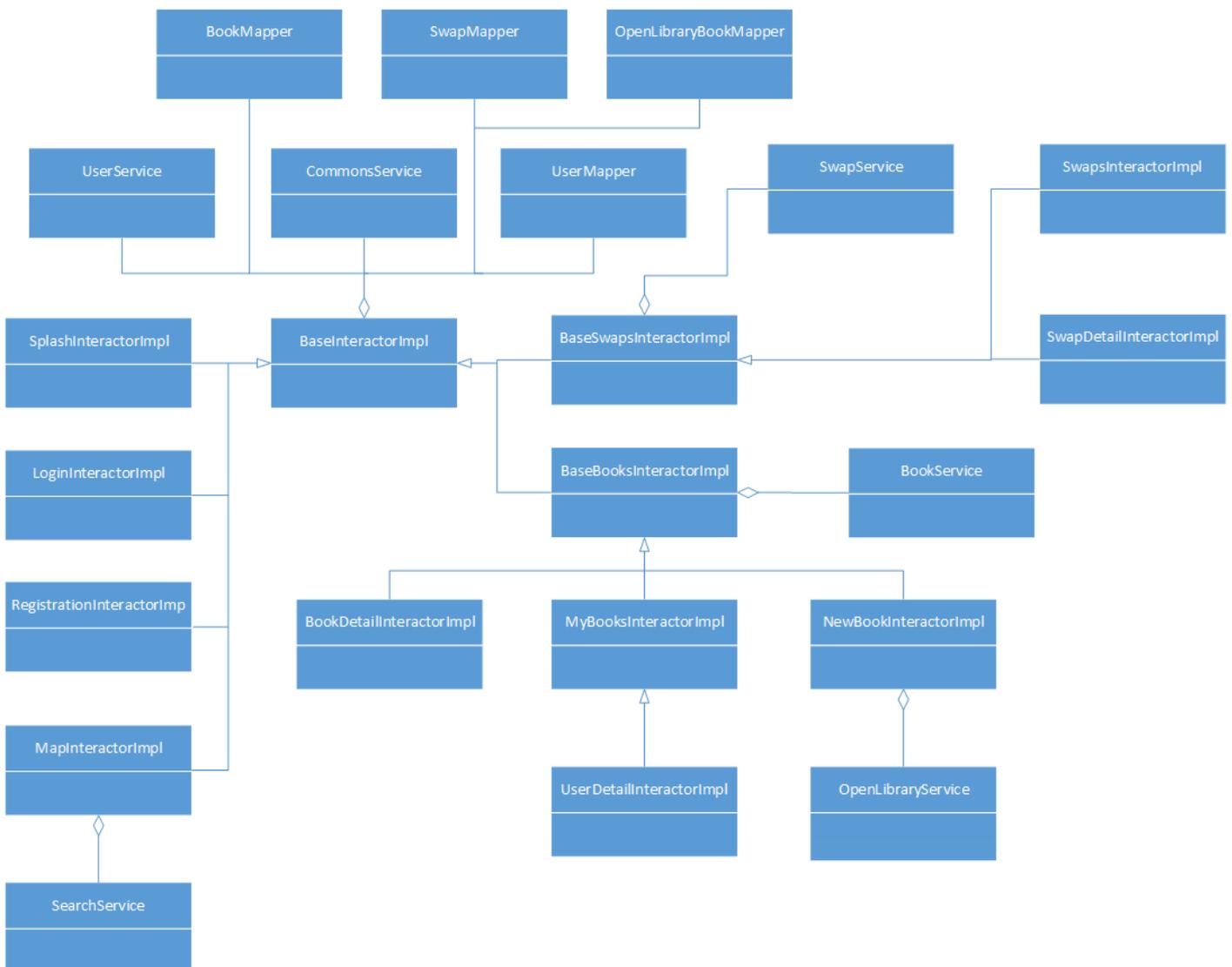


Figura 40: Diagrama de clases del modelo de la aplicación

3.3.2 Diseño adaptativo para distintos tamaños de pantalla

Ante la gran cantidad de tamaños y densidades de pantalla diferentes que existe en el ecosistema Android, y con el objetivo de maximizar el número de dispositivos compatibles con la aplicación desarrollada, se han llevado a cabo una serie de buenas prácticas recomendadas por Google en su documentación oficial [19].

Toda la interfaz gráfica de la aplicación está diseñada con *layouts* que posicionan las vistas de manera relativa (*RelativeLayout*, *LinearLayout*, *ConstraintLayout*) en función del tamaño de la pantalla. De la misma forma, para definir los tamaños de alto y ancho de las vistas de la interfaz de usuario se ha utilizado:

- *wrap_content* y *match_parent*. Estos valores permiten que las vistas ajusten el tamaño a su contenido o al de su vista padre, respectivamente.
- Píxeles independientes de la densidad (*dp*), para algunas vistas, y píxeles independientes de la escala (*sp*) para el caso de los textos.

No obstante, debido al tipo de información mostrado en la aplicación y para simplificar su desarrollo, se permite únicamente orientación vertical de la pantalla (*portrait*), y los diseños se han optimizado solo para dispositivos móviles, no tabletas, tal y como se determinó en la fase de diseño.

3.3.3 Navegación principal

La navegación principal se ha desarrollado por medio de un componente *NavigationView*, que es controlado y gestionado dentro de la actividad principal, *MainActivity*. La navegación da acceso a los tres módulos de la aplicación, así como al cierre de sesión:

- Cerca de mí: Búsqueda de usuarios en el mapa, implementada en el paquete *map*.
- Mis libros: Gestión de libros del usuario, implementada en el paquete *mybooks*.
- Intercambios: Gestión de los intercambios en los que participa el usuario, implementada en el paquete *swaps*.

La pantalla de entrada a cada uno de los módulos se ha implementado por medio de un fragmento (*MapFragment*, *MyBooksFragment* y *SwapsFragment*, respectivamente), que se carga dentro de la actividad principal al pulsar cada una de las opciones de la navegación. Los fragmentos se cargan únicamente la primera vez que el usuario accede a una determinada opción. Posteriormente, se van mostrando/ocultando en función de la opción seleccionada. De esta forma se consigue un comportamiento más fluido, al evitarse sincronizaciones de datos desde el servidor cada vez que el usuario pulsa en una opción del menú. No obstante, se pueden recargar los datos bajo demanda en los tres fragmentos, bien utilizando la función de búsqueda, en el caso del mapa, o bien haciendo el gesto de deslizar hacia abajo (*swipe*), en el caso de los listados de libros e intercambios.

Por otra parte, el diseño de esta pantalla está compuesto por varios *layouts* combinados de la siguiente forma:

- **activity_main.xml**: Layout principal de la pantalla. Incluye el *layout app_bar_main* como contenido principal de la misma y carga en el componente *NavigationView* el *layout nav_header_main* como cabecera, así como el menú *activity_main_drawer* como conjunto de opciones de navegación.
- **nav_header_main.xml**: Diseño de la cabecera del menú principal.
- **app_bar_main.xml**: *CoordinatorLayout* que contiene la barra de título de la pantalla, así como el botón flotante de acciones principales. Dentro de

este *layout* se inserta también un *FrameLayout*, a modo de contenedor, en el que se carga el fragmento correspondiente al pulsar cada una de las opciones del menú.

3.3.4 Almacenamiento de entidades en Backendless

Debido a las características intrínsecas al servicio de persistencia de datos ofrecido por Backendless, ha sido necesario realizar algunas transformaciones en el modelo de datos propuesto en el apartado de diseño del trabajo, para adaptarlo a sus peculiaridades.

La estructura de datos modelada en el servidor se ha orientado a facilitar las operaciones de lectura (aunque también se han tenido en cuenta las de escritura, obviamente), de manera que el proceso sea lo más simple posible, con el fin de mejorar el rendimiento de la aplicación. Para ello ha sido necesario desnormalizar la estructura de las entidades, mediante la duplicación de algunos datos, para optimizar así el número de peticiones realizadas al servidor.

Las modificaciones realizadas han sido las siguientes:

La **entidad *Provincia*** ha sido eliminada. El objetivo de esta entidad era el de almacenar las provincias españolas para que, en el momento de registrarse en la aplicación, el usuario pudiera seleccionar una de ellas y así el sistema poder determinar, de forma aproximada, su ubicación en caso de que éste no tuviera activado el servicio de geolocalización de su dispositivo. Finalmente, como se comentará más adelante, el usuario puede proporcionar una ubicación inicial, en el momento del registro, mediante la API Google Places, lo cual le proporciona mayor flexibilidad y precisión a la hora de elegir su situación más habitual. La ubicación seleccionada por el usuario se persiste en una tabla especial de Backendless, *GeoPoint*, específica para almacenar coordenadas geográficas, y que facilita las búsquedas por distancia por medio de la API de geolocalización del servicio.

La **entidad *Usuario***, que representa a un usuario registrado en la aplicación, se materializa por medio de la tabla *Users* de Backendless. Se trata de una tabla de sistema, no de creación propia, cuyo uso es obligatorio si se quiere utilizar la API de registro y autenticación de usuarios del servicio. Aparte del hecho de utilizar la tabla, los cambios introducidos en ella respecto al modelado inicial son los siguientes:

- Se incorpora un campo de tipo entero que actúa como contador del número de libros registrados por el usuario. Este campo se incrementa cuando el usuario añade un libro y se decrementa cuando lo elimina. De esta forma, al consultar en el mapa los usuarios cercanos, se evita tener que contar el número de libros de cada uno de ellos, reduciéndose el número de consultas enviadas al servidor.

- Se añade un campo de tipo decimal para almacenar la valoración media de los intercambios realizados por el usuario. El objetivo de este campo es completamente análogo al caso anterior.
- Se añade una relación a la tabla *GeoPoint*, para almacenar las coordenadas geográficas de la última localización conocida del usuario. En caso de que el usuario no active nunca la geolocalización en su dispositivo este campo contendrá la ubicación proporcionada en el momento de creación de la cuenta. Si está activada se actualizará con el último valor proporcionado por el terminal.

<input type="checkbox"/>	COLUMN NAME	DATA TYPE	DEFAULT VALUE	IDENTITY	CONSTRAINTS	VALIDATOR
<input type="checkbox"/>	booksNumber	INT	0	<input type="radio"/>	Select constraints	Select a validator
<input type="checkbox"/>	location	GEO Relation 1:1	not assignable	not assignable	Select constraints	not assignable
<input checked="" type="checkbox"/>	email	STRING	not assignable	<input checked="" type="radio"/>	NN UQ IDX	Select a validator
<input type="checkbox"/>	name	STRING		<input type="radio"/>	* NN	Select a validator
<input checked="" type="checkbox"/>	password	STRING	not assignable	not assignable	* NN	Select a validator
<input type="checkbox"/>	rating	DOUBLE	0.0	<input type="radio"/>	Select constraints	Select a validator
<input checked="" type="checkbox"/>	created	DATETIME	not assignable	not assignable	Select constraints	not assignable
<input checked="" type="checkbox"/>	updated	DATETIME	not assignable	not assignable	Select constraints	not assignable

Figura 41: Definición de la entidad *User* en Backendless

Sobre la **entidad *Libro*** se han realizado las siguientes adaptaciones:

- Se eliminan las claves foráneas a las entidades *Género* e *Idioma*, y se sustituyen por simples campos de texto que almacenan los nombres del género e idioma, respectivamente. El motivo para realizar este cambio es que cuando se almacena un objeto en Backendless que está relacionado con otra entidad es necesario realizar la operación en dos pasos, lo que implica dos llamadas al servidor: una primera para crear y almacenar el objeto en su tabla, y una segunda para actualizarlo y establecer la relación con la otra tabla implicada. En el caso de la entidad *Libro*, al tener dos relaciones sería necesario realizar tres llamadas. Como las entidades *Género* e *Idioma* almacenan información que no varía, y se utilizan como campos meramente informativos, se ha optado por esta solución para realizar una única petición al servidor a la hora de guardar un libro.
- Se añade un campo booleano, *eliminado*, para permitir el borrado lógico de los elementos de la tabla. De esta forma, si un usuario elimina un libro éste no se borra físicamente de la tabla, pero se marca como eliminado para que no aparezca en las búsquedas de otros usuarios, sin embargo, si el libro participó en algún intercambio los usuarios podrán seguir visualizando la información del mismo al consultar los datos. Con esta solución se consigue mantener la integridad de los datos a nivel físico, y se evita tener que borrar en cascada los intercambios asociados a un libro, reduciéndose nuevamente el número de peticiones al servidor.

<input type="checkbox"/>	COLUMN NAME	DATA TYPE	DEFAULT VALUE	CONSTRAINTS	VALIDATOR
<input type="checkbox"/>	isbn	STRING		Select constraints	Select a validator
<input type="checkbox"/>	autor	STRING		x NN	Select a validator
<input type="checkbox"/>	editorial	STRING		x NN	Select a validator
<input type="checkbox"/>	eliminado	BOOLEAN	FALSE	x NN	not assignable
<input type="checkbox"/>	estado	INT		x NN	RegExp [0-2]
<input type="checkbox"/>	foto	FILE REFERENCE		Select constraints	not assignable
<input type="checkbox"/>	genero	STRING		x NN	Select a validator
<input type="checkbox"/>	idioma	STRING		x NN	Select a validator
<input type="checkbox"/>	titulo	STRING		x NN	Select a validator
<input type="checkbox"/>	created	DATETIME	not assignable	Select constraints	not assignable
<input type="checkbox"/>	updated	DATETIME	not assignable	Select constraints	not assignable

Figura 42: Definición de la entidad *Libro* en Backendless

Finalmente, se han realizado las siguientes adaptaciones en la entidad ***Intercambio***:

- Se han eliminado las claves foráneas a la entidad *Usuario*, aprovechando que Backendless añade automáticamente a todas las tablas una columna de sistema, denominada *ownerId*, que identifica al usuario que insertó el registro. De esta forma, el usuario destinatario del intercambio se obtiene mediante el campo *ownerId* de la columna de relación *libroDestino* (que no puede ser nula, ya que para registrar un intercambio es necesario solicitar un libro), mientras que el usuario origen se determina a través del campo *ownerId* del propio intercambio. De esta forma, al igual que ocurría con la entidad *Libro*, se reduce el número de llamadas al servidor a la hora de insertar intercambios en la tabla, al no tener que invocarlo para establecer la doble relación con la entidad *Usuario*.
- Se incorporan dos campos de texto, *usuarioOrigen* y *usuarioDestino*, para almacenar los nombres de los usuarios participantes en el intercambio. De esta manera se simplifica la consulta necesaria a la hora de mostrar la información de los intercambios en los listados.

<input type="checkbox"/>	COLUMN NAME	DATA TYPE	DEFAULT VALUE	CONSTRAINTS	VALIDATOR
<input type="checkbox"/>	estado	INT	0	Select constraints	RegExp [0-2]
<input type="checkbox"/>	libroOrigen	Data Relation 1:1 Libro	not assignable	Select constraints	not assignable
<input type="checkbox"/>	usuarioOrigen	STRING		x NN	Select a validator
<input type="checkbox"/>	valoracionOrigen	INT	0	Select constraints	RegExp [0-5]
<input type="checkbox"/>	libroDestino	Data Relation 1:1 Libro	not assignable	Select constraints	not assignable
<input type="checkbox"/>	usuarioDestino	STRING		x NN	Select a validator
<input type="checkbox"/>	valoracionDestino	INT	0	Select constraints	RegExp [0-5]
<input type="checkbox"/>	created	DATETIME	not assignable	Select constraints	not assignable
<input type="checkbox"/>	updated	DATETIME	not assignable	Select constraints	not assignable

Figura 43: Definición de la entidad *Intercambio* en Backendless

3.3.5 Gestión de cuentas de usuario con Backendless User Service API

La aplicación utiliza el módulo de gestión de usuarios de Backendless para realizar la autenticación mediante correo electrónico y contraseña, el registro de nuevos usuarios y el cierre de sesión. Permite también la recuperación de contraseñas y la autenticación automática (recordar contraseña), sin embargo, estas funcionalidades no han sido incorporadas a la aplicación.

La validación de las credenciales introducidas por el usuario se realiza en la actividad de *login*, por medio de una invocación de su correspondiente interactor (*LoginInteractor*) al método de servicio. El servicio se encarga de realizar la petición al servidor Backendless, mediante la función *login(email, password)* de la clase *Backendless.UserService*. La respuesta se recibe mediante retrollamadas (*callbacks*), que se trasladan al interactor, y de él, al presentador, que se encarga de actualizar la vista en función de si el proceso de autenticación tiene o no éxito.

El registro de nuevos usuarios a partir de los datos introducidos se realiza desde *RegistrationActivity*. El proceso es similar a la autenticación, generándose la llamada al método de servicio desde el *RegistrationInteractor*. La creación de la cuenta se realiza en tres pasos:

- En primer lugar, se crea la nueva cuenta mediante la función *register(user)*. Esta llamada genera un nuevo registro en la tabla *Users*, con los datos proporcionados por el usuario.
- En segundo lugar, se inserta una nueva coordenada geográfica en la tabla *GeoPoint*, correspondiente a la ubicación que ha proporcionado el usuario en la entrada de datos, mediante la función *savePoint(point)* de la clase *Backendless.Geo*.
- Finalmente se actualiza el usuario recién creado, generando una relación con su coordenada geográfica, utilizando para ello el servicio de persistencia de datos por medio del método *setRelation()* de la clase *Backendless.Data*.

Como en el caso de la autenticación, los resultados se reciben por medio de *callbacks*. Si el proceso de registro finaliza correctamente se redirige al usuario a la pantalla principal de la aplicación, *MainActivity*, ya que tras la creación de la cuenta se invoca al método de *login* automáticamente, al producirse la autenticación de forma implícita.

Una vez el usuario está autenticado en la aplicación, sus datos son accesibles en todo momento sin necesidad de invocar a los servidores de Backendless, por medio del método *CurrentUser()* de la clase *Backendless.UserService*.

Finalmente, el cierre de sesión se realiza mediante la llamada al método *logout()* de la clase *Backendless.UserService*. Este método invalida el token de sesión en el servidor y hace que las llamadas a *CurrentUser()* devuelvan *null*, al no existir ningún usuario autenticado. El proceso de *logout* se desencadena cuando el usuario pulsa sobre la correspondiente opción del menú de navegación principal, situado en la actividad *MainActivity*. Al confirmarse la

desconexión el usuario es redirigido automáticamente de nuevo a la pantalla de *login*.

3.3.6 Obtención de imágenes y almacenamiento en Backendless Files

Cuando el usuario añade un libro a la plataforma la aplicación le permite asociar una imagen, que también podrá ver el resto de usuarios. Esta imagen puede obtenerse desde la galería de imágenes del dispositivo, o también mediante la realización de una fotografía con la cámara. La selección del origen de la imagen se realiza por medio de un *AlertDialogFragment*, que ofrece al usuario ambas opciones:

- Para realizar la captura de imágenes con la cámara es necesario comprobar previamente que se dispone del permiso de acceso a la misma, *CAMERA*, y en caso de no tenerlo, solicitarlo al usuario. La captura no se realiza accediendo al hardware del dispositivo directamente, sino que, a través de un *Intent*, se solicita al sistema operativo que se ejecute la aplicación que gestiona la cámara y devuelva los resultados a la aplicación.
- Cuando la cámara realiza la fotografía puede devolver a la aplicación una ruta del sistema de archivos del dispositivo, para acceder a la imagen tomada, o bien una miniatura de la misma (*thumbnail*) en baja resolución. En el primer caso es necesario solicitar permiso de lectura en el sistema de almacenamiento del dispositivo, sin embargo, en el segundo no, ya que la imagen es devuelta directamente a través de un objeto. En la aplicación se ha optado por la segunda opción, por no requerir permisos adicionales y porque la imagen tiene un peso reducido, lo cual minimiza el tráfico de datos al enviarla al servidor.
- La selección de imágenes desde la galería también se realiza a través de un *Intent*, de manera que es el sistema operativo el que selecciona la aplicación más adecuada para realizar la acción. De esta forma la imagen es devuelta a la aplicación sin necesidad de que ésta acceda directamente al sistema de archivos, evitándose así la solicitud del correspondiente permiso.

Cuando el usuario ha seleccionado la imagen ésta se establece en pantalla, para que pueda visualizarla. La aplicación coloca el título del libro sobre la imagen del mismo, por ello, para que el texto sea legible, incluso cuando la imagen tiene tonos predominantemente claros, se utiliza una vista con fondo semitransparente (definido en el recurso *image_overlay.xml*), que se superpone a la imagen oscureciéndola y facilitando la lectura del texto que hay sobre ella. Este mecanismo se emplea también en la pantalla de detalle de libro (*BookDetailActivity*) y en la de detalle de un intercambio (*SwapDetailActivity*).

Al pulsar la opción de guardar el libro la imagen es enviada al servidor por medio del método *saveFile()* de la clase *Backendless.Files*, a través del interactor asociado a la pantalla (*NewBookInteractor*). Esa llamada almacena la fotografía en el sistema de archivos remoto de Backendless, en una ruta única compuesta por el nombre de la aplicación, el ID del usuario propietario de la

imagen y un nombre de archivo único generado mediante la función Java `System.currentTimeMillis()` (`/b4b/<idUsuario>/<millis>.png`).

La forma de relacionar una entrada en el sistema de archivos remoto con una entidad de datos se realiza por medio de columnas de relación de tipo *file reference* (en el caso de la entidad *Libro* se trata de la columna *foto*). Este tipo de relación almacena en la entidad la URL a través de la cual poder realizar la descarga del archivo. En el caso de la aplicación, este tipo de descargas se realiza mediante la librería Picasso, como se ha comentado con anterioridad.

3.3.7 Búsqueda de libros en la base de datos de OpenLibrary

Aunque no son muy numerosos los campos que se deben rellenar para registrar un libro a través de la aplicación, resulta tedioso realizar este proceso de forma manual, especialmente si el usuario quiere dar de alta una gran cantidad de ellos. Para facilitar esta labor, y reducir la probabilidad de que se produzcan errores a la hora de dar de alta los libros, la aplicación permite buscarlos por su código ISBN en la base de datos OpenLibrary, obteniendo sus datos y precargándolos en la pantalla de alta del libro.

Las consultas a la base de datos se realizan por medio de una API REST, a la que se le realizan peticiones de tipo GET de la siguiente forma:

<http://openlibrary.org/api/books?bibkeys=ISBN:9788401337635&format=json&jscmd=data>

- El parámetro “bibkeys=ISBN:” indica el código ISBN del libro a buscar.
- El parámetro “format=json” informa a la API de que el resultado de la consulta se desea en formato JSON.
- Finalmente, el parámetro “jscmd=data” indica que se desea obtener información extendida del libro en caso de encontrarse. Esto es necesario para que la API devuelva determinados campos del libro, como la editorial o el autor.

Tras una petición como la anterior el servidor OpenLibrary devuelve un objeto JSON con el resultado de la consulta. Dependiendo del libro consultado puede devolver más o menos información, no obstante, independientemente de los datos devueltos, a la aplicación solo le interesan los relativos al título, autor, editorial y carátula. Una respuesta típica de la API es la siguiente (en negrita los campos utilizados por la aplicación):

```
{"ISBN:9788401337635": {"publishers": [{"name": "Random House Mondadori, S.A."}],  
"identifiers": {"isbn_13": ["9788401337635"], "openlibrary": ["OL25898442M"],  
"dep\u00f3sito_legal": ["M279612010"]}, "title": "La caída de los Gigantes", "url":  
"http://openlibrary.org/books/OL25898442M/La_caída_de_los_Gigantes", "number_of_pages":  
1017, "cover": {"small": "https://covers.openlibrary.org/b/id/8146891-S.jpg", "large":  
"https://covers.openlibrary.org/b/id/8146891-L.jpg", "medium":  
"https://covers.openlibrary.org/b/id/8146891-M.jpg"}, "publish_date": "2010", "key":  
"/books/OL25898442M", "authors": [{"url":  
"http://openlibrary.org/authors/OL229268A/Ken_Follett", "name": "Ken Follett"}],  
"publish_places": [{"name": "Barcelona"}]}}
```

Las peticiones a la API OpenLibrary se realizan por medio de un cliente HTTP implementado a través de la librería Retrofit. El cliente se desarrolla por medio del servicio *OpenLibraryService* y la interfaz *OpenLibraryApi*, que publica un método *getBookByIsbn(isbn)*.

Al invocar el método del cliente la librería se encarga de gestionar todo lo relacionado con la comunicación con el servidor a través de HTTP. Al recibir la respuesta la librería transforma automáticamente el objeto JSON en un objeto Java, de tipo *OpenLibraryBook*, que es mapeado en el interactor (*NewBookInteractor*) a una entidad de interfaz de usuario (*UiBook*) que se traspasa al presentador de la pantalla (*NewBookPresenter*), el cual se encarga de cargarlos en los campos de la actividad (*NewBookActivity*) que se muestran al usuario.

3.3.8 Lectura de códigos de barras (ISBN de libros) con ZXing embedded

Como complemento a la búsqueda de libros por ISBN, y para evitar también que el usuario tenga que introducir manualmente este código, la aplicación proporciona la posibilidad de leer los códigos de barras situados en la contraportada de los libros, decodificarlos, obtener el código ISBN y lanzar la consulta con el mismo a la API de OpenLibrary. Esta funcionalidad está disponible gracias a la utilización de la librería ZXing Android embedded.

La librería hace un uso directo del hardware de la cámara del dispositivo, por lo que para su utilización es necesario solicitar el correspondiente permiso (*CAMERA*) al usuario. No obstante, tanto la solicitud del permiso como la gestión de la respuesta del usuario es realizada por la propia librería, siendo totalmente transparente a la aplicación, que solo tiene que declarar el permiso en el archivo *AndroidManifest.xml*.

La utilización de la librería es muy sencilla. Basta con configurar los parámetros para realizar el escaneo del código (tales como el formato de código a leer o el mensaje a mostrar al usuario) mediante un objeto *IntentIntegrator*, e invocar al método *initiateScan()*. La librería gestionará el permiso de acceso a la cámara, y en caso de disponer de él abrirá una nueva actividad a través de la cual el usuario podrá escanear el código del libro con la cámara del dispositivo. Al realizarse la lectura el resultado de la decodificación, que es una cadena de caracteres con el código ISBN, se devuelve a la actividad llamante (*NewBookActivity*), por medio del *callback onActivityResult()*.

3.3.9 Obtención de la localización actual del usuario con Google Maps

La aplicación utiliza la ubicación del usuario para realizar búsquedas de otros usuarios que se encuentren próximos o a una distancia especificada. El proceso de determinación de la ubicación sucede en *MapFragment*, cuando el usuario accede a la aplicación tras autenticarse, y se utiliza para ello la API Location de Google Maps. Para poder acceder a la localización del dispositivo es necesario disponer del permiso *ACCESS_FINE_LOCATION*, por lo que hay

que solicitárselo al usuario en caso de que este disponga de una versión de Android 6.0 o superior. Lógicamente, el servicio de localización debe estar activado para poder acceder a ella. En base a esto, el proceso seguido para determinar la localización se muestra en la siguiente figura:

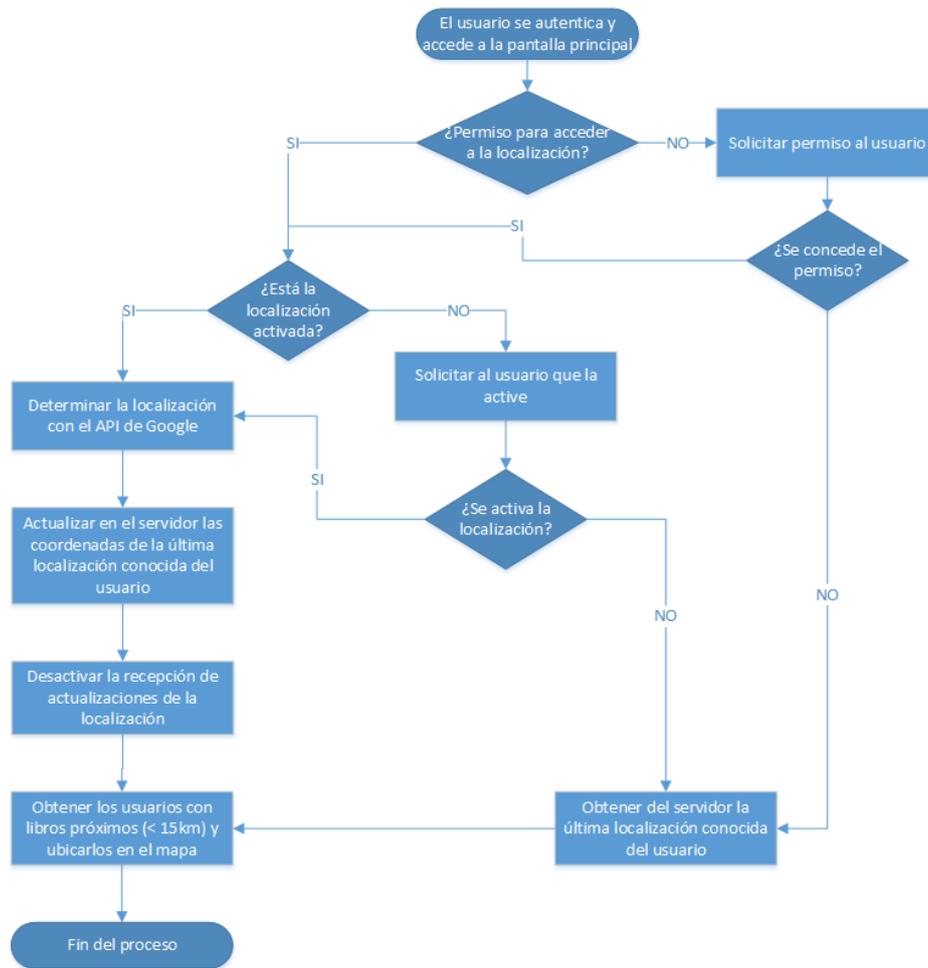


Figura 44: Proceso de obtención de la localización del usuario

Como se indica en el diagrama, en caso de que se active la localización y se disponga del permiso requerido, una vez se obtiene la ubicación del usuario se solicita a la API Location de Google Maps que desconecte y deje de recibir actualizaciones de la posición. De esta forma se consigue optimizar el rendimiento de la aplicación y reducir el consumo de la batería.

Por su parte, si no se dispone del permiso o la localización no está activada, se utiliza la última ubicación registrada en el servidor. En caso de que el usuario no haya actualizado esta ubicación nunca (ha denegado el permiso de acceso o tiene la localización desactivada siempre) se empleará la ubicación que proporcionó en el momento de registrarse en la aplicación, como se verá a continuación.

3.3.10 Localización del usuario con Google Places durante el registro

Como ya se ha mencionado, en el proceso de registro de una nueva cuenta en la aplicación se le solicita al usuario que proporcione una ubicación por defecto, que como se ha detallado en el anterior apartado, se utiliza para poder realizar búsquedas cuando no se tiene acceso a la localización actual del dispositivo. La herramienta que se le proporciona al usuario para ello es un buscador, basado en la API Google Places, y en concreto, en su servicio de autocompletado. Este servicio muestra predicciones de sitios en respuesta al texto que introduce el usuario, de manera que a medida que el usuario escribe, el servicio de autocompletado muestra sugerencias de lugares, como direcciones o municipios.

La integración del autocompletado en la aplicación se ha realizado por medio de un componente *PlaceAutocompleteFragment*, incluido en la API. Ha sido configurado con un filtro de tipo *AutocompleteFilter* para limitar los resultados obtenidos a localizaciones españolas. El componente proporciona una caja de búsqueda a través de la cual el usuario puede escribir la dirección deseada. La API va mostrando una lista con sugerencias a la búsqueda realizada. Cuando el usuario selecciona una de las opciones se genera un evento *PlaceSelection* que es capturado por la actividad de la aplicación (*RegistrationActivity*). Dicho evento contiene la información relativa al lugar seleccionado, entre la que se encuentra el nombre del mismo, que se muestra al usuario, y sus coordenadas geográficas, que serán almacenadas en el servidor cuando se produzca el registro de la cuenta.

La decisión de utilizar esta alternativa viene dada principalmente porque proporciona un mecanismo para obtener una ubicación precisa del usuario en caso de que éste no desee utilizar la localización del dispositivo, siendo a la vez cómoda para él, y permite prescindir del listado de provincias que se propuso en las fases iniciales del proyecto.

3.3.11 Búsqueda de usuarios y visualización de resultados en mapa

La búsqueda de usuarios en la aplicación se puede realizar en base a los siguientes parámetros:

- Título del libro.
- Autor del libro.
- Género literario del libro.
- Distancia máxima a la que se encuentra el usuario propietario del libro.

Para evitar que se arrojen excesivos resultados de búsqueda, el filtro impone la restricción al usuario de que, si se busca por título o autor, el texto introducido debe tener, al menos, 4 caracteres.

La búsqueda que se realiza por defecto al acceder a la pantalla por primera vez devuelve todos los usuarios situados en un radio de 45 kilómetros que estén compartiendo al menos un libro.

El filtro de búsqueda de usuarios se ha implementado a través de un fragmento de tipo *bottom sheet* (panel deslizante que se despliega desde la parte inferior de la pantalla), por medio de la clase *SearchFragment*. Este fragmento se limita a recoger la información introducida por el usuario y trasladarla al fragmento contenedor, *MapFragment*, desde donde se realiza la petición de búsqueda al servidor (en realidad todo el proceso es orquestado desde el correspondiente presentador, *MapPresenter*, que solicita las búsquedas al servidor a través de su interactor, *MapInteractor*).

La búsqueda y filtrado de los usuarios con los criterios establecidos se realiza en su totalidad en el *backend* de Backendless, mediante una única consulta, de forma que al obtenerse los resultados no es necesario aplicar ningún tipo de filtrado adicional dentro de la propia aplicación. El servicio proporciona un potente sistema de consultas, que incluye la realización de búsquedas de puntos geográficos por distancia (mediante la función *distance*) y la ejecución de consultas anidadas (*subqueries*). Todo ello integrado ha permitido reducir al máximo el número de peticiones al servidor en esta funcionalidad.

Cuando se obtiene respuesta del *backend* y se reciben los usuarios que verifican los criterios de búsqueda se envían al *MapFragment*, por medio de entidades de interfaz de usuario de tipo *UiUser*. Este fragmento incluye un *SupportMapFragment* de Google Maps, que visualiza un mapa sobre el que se añaden marcadores (*Marker*) en las ubicaciones de los usuarios encontrados, a través del método *addMarker()*, y utilizando un adaptador personalizado, *MapInfoWindowAdapter*, para mostrar la información asociada a cada uno de ellos.



Figura 45: Marcador de usuario sobre el mapa

3.3.12 Mensajería

La aplicación envía correos electrónicos a los usuarios cuando se producen determinados eventos, en concreto, los siguientes:

- Cuando un usuario inicia un intercambio y solicita un libro a otro usuario. El usuario destinatario de la petición recibe un mensaje indicándole que otro usuario le está pidiendo un libro.
- Cuando un usuario destinatario de un intercambio rechaza una petición. El usuario que inició el intercambio recibe un mensaje indicándole que su petición ha sido rechazada por el otro usuario.
- Cuando un usuario destinatario de un intercambio confirma la petición. En este caso se envían dos mensajes, uno a cada usuario, indicando que el intercambio ha sido aceptado y proporcionándoles la dirección de correo electrónico del otro participante para que se puedan poner en contacto y organizar la logística para realizar el intercambio físico de los libros³.
- Cuando un usuario valora un intercambio. Se envía un correo de aviso al otro usuario participante que le indica que ha recibido una valoración.

Para cada tipo de mensaje se utiliza una plantilla de envío predefinida. Las plantillas se definen por medio de la enumeración *MailTemplate*.

El envío de los mensajes se realiza por medio de la API de mensajería de Backendless (*Messaging API*), en concreto, por medio del método *sendTextMail()* de la clase *Backendless.Messaging*. Los datos del mensaje son enviados al servidor de *backend* que, a su vez, utiliza un servidor de correo, configurado a través de la consola de administración del servicio, para enviar el mensaje a los usuarios. En el caso de la aplicación se ha configurado una cuenta de Gmail para realizar el envío.



✉ Email Settings

SMTP Server:

Port:

From:

From Email:

User ID:

Password:

Show password

Security: SSL TLS

Figura 46: Configuración del envío de mensajes en la consola de Backendless

Inicialmente se planificó también la utilización de la API de mensajería para el envío de notificaciones *push* a los usuarios. No obstante, por falta de tiempo y por tratarse de una funcionalidad complementaria y opcional, su desarrollo finalmente se ha descartado, ya que la necesidad ha quedado cubierta a través del envío de correos electrónicos.

³ Este proceso es totalmente externo y ajeno a la aplicación.

3.3.13 Valoración y actualización de puntuación media

La valoración de un intercambio se puede realizar una vez que el usuario destinatario de la petición lo ha aceptado y ha establecido el libro que desea cambiar. Cuando se dan estas condiciones, desde la propia pantalla de detalle de un intercambio (*SwapDetailActivity*), se activa una opción de menú, ★, que permite a los usuarios valorar el intercambio.

La valoración se realiza por medio de un diálogo emergente, *RatingDialogFragment*, que extiende a la clase de diálogo genérica de la aplicación, *AlertDialogFragment*, añadiendo una vista de tipo *RatingBar* para dar la posibilidad al usuario de seleccionar una puntuación entera con valores comprendidos entre 1 y 5, ambos incluidos.

Con cada nueva valoración introducida se recalcula la valoración media del usuario que la recibe, siguiéndose el siguiente proceso:

- Se actualiza el registro correspondiente en la tabla *Intercambio*, añadiendo la valoración introducida.
- Se obtienen del servidor todos los intercambios que se encuentren en estado “ACEPTADO” en los que haya participado el usuario que recibe la valoración, ya sea como solicitante o como destinatario, y en los que ya haya recibido una valoración.
- Desde la propia aplicación se calcula la valoración media en base a los registros obtenidos en el paso anterior ($\sum \text{valoraciones} / \text{número de registros}$).
- Se actualiza el registro del usuario valorado en la tabla *Users*, estableciendo la nueva valoración media en el campo *rating*.

Como se puede comprobar, la valoración requiere de varias peticiones al servidor, y el cálculo de la misma se realiza en el cliente. Esto puede ocasionar problemas de escalabilidad y de concurrencia en el caso de que se valore a un mismo usuario desde varios dispositivos a la vez, ya que el proceso no es atómico. En la documentación de la API de persistencia de Backendless no se ha encontrado la manera de realizar esta operación de manera transaccional. En un entorno productivo este problema podría solucionarse realizando todos los cálculos en el propio servidor.

3.3.14 Custom adapters

Todos los *RecyclerView* que se utilizan en la aplicación son manejados por un adaptador personalizado, *RecyclerViewAdapter*, generado para utilizar *layouts* también personalizados como celdas de los listados. Las pantallas de la aplicación que incorporan listados son las siguientes:

- Lista de libros del usuario (*MyBooksFragment*). El *layout* empleado para los elementos del listado se define en el recurso *cardview_list_book.xml*.

- Pantalla de detalle de otros usuarios, para mostrar los libros que comparten (*UserDetailActivity*). El *layout* utilizado es el mismo que en el caso anterior.
- Lista de intercambios en los que participa el usuario (*SwapsFragment*). El *layout* para los ítems de la lista es *cardview_list_swap.xml*.

Aunque se utilicen varios *layouts* diferentes, la estructura de información que presentan es común a todos (título, subtítulo, imagen e información extendida), por ello, la clase de tipo adaptador empleada en todos los listados es la misma, *RecyclerViewAdapter*.

Todos los listados incorporan un *SwipeRefreshLayout*, que permite al usuario refrescar la información mostrada haciendo el gesto de desplazar el contenido de la pantalla hacia abajo.

El motivo de utilizar *RecyclerView* en lugar de *ListView* o *GridView* es porque recicla las vistas de los elementos del listado que no se están visualizando en la pantalla, para reutilizarlas en los que sí se están mostrando. Esto permite optimizar el rendimiento de la aplicación, especialmente cuando se presentan conjuntos de datos grandes.

3.4 Pruebas

Para garantizar el correcto funcionamiento del sistema se han llevado a cabo un conjunto de pruebas sobre la aplicación a lo largo de su desarrollo. Estas pruebas han sido, principalmente, de tipo manual, para verificar que el sistema implementado cumple con todos los requisitos especificados. Estas pruebas manuales han sido complementadas con pruebas automáticas de la interfaz de usuario, utilizando para ello el *framework* de *testing* Espresso que viene integrado con el IDE Android Studio.

La ejecución de las pruebas se ha realizado sobre el emulador proporcionado por el SDK de Android, con una imagen de Android 8.1 (*Oreo*) y simulando las características de un dispositivo Nexus 5X, así como sobre cuatro dispositivos físicos, para asegurar un comportamiento correcto en un entorno real. Los dispositivos utilizados han sido:

- Samsung Galaxy S4 con Android 7.1.2 (*Nougat*).
- Vodafone VFD 600 con Android 6.0.1 (*Marshmallow*).
- Samsung Galaxy J5 con Android 5.1.1 (*Lollipop*).
- Sony Xperia T3 con Android 4.4.4 (*KitKat*).

Para la realización de las pruebas manuales se ha ido construyendo un *checklist* que cubre los aspectos fundamentales de la aplicación, y que permite verificar el correcto funcionamiento de sus funcionalidades. El *checklist* resultante es el siguiente:

Pantalla	Prueba de verificación
Splash (<i>SplashActivity</i>)	<ul style="list-style-type: none"> • La pantalla se visualiza correctamente. • Se da paso automáticamente a la pantalla de autenticación.
Autenticación (<i>LoginActivity</i>)	<ul style="list-style-type: none"> • Se puede navegar a la pantalla de registro pulsando sobre el texto correspondiente. • Se valida el formato y la introducción de los campos. • Se muestra un mensaje de error cuando las credenciales de acceso no son válidas. • Mientras se validan las credenciales se muestra un indicador de progreso. • Se validan las credenciales y se da acceso a la pantalla principal.
Registro (<i>RegistrationActivity</i>)	<ul style="list-style-type: none"> • Se puede volver a la pantalla de autenticación pulsando sobre el texto correspondiente. • Se valida que el formato de los campos es correcto y la introducción de aquellos que son obligatorios. • Se pueden introducir localizaciones utilizando la herramienta de autocompletado de sitios. • Se muestra un mensaje de error si se intenta realizar el registro con una dirección de correo ya utilizada por otro usuario. • Mientras se registra la cuenta se muestra un indicador de progreso. • Se crea el nuevo usuario en Backendless y se le asocia la ubicación por defecto que ha seleccionado con la herramienta de autocompletado de sitios. • Se realiza el <i>login</i> automáticamente al crearse la cuenta y se navega hasta la pantalla principal de la aplicación.
Principal (<i>MainActivity</i>)	<ul style="list-style-type: none"> • Se muestra el mapa con los usuarios próximos al acceder. • La navegación principal se abre y cierra correctamente. • Aparece el nombre y dirección de correo electrónico del usuario autenticado en la cabecera de la navegación principal. • Al pulsar sobre la opción “Mis libros” de la navegación principal se muestran los libros del usuario. • Al pulsar sobre la opción “Intercambios” de la navegación principal se muestran los intercambios en los que participa el usuario. • Al pulsar sobre la opción “Cerca de mí” de la navegación principal se muestran los usuarios cercanos sobre un mapa. • Al pulsar sobre la opción “Salir” se muestra una ventana de diálogo. Si se pulsa en “Cancelar” se permanece en la aplicación. • Al pulsar sobre la opción “Salir” se muestra una ventana de diálogo. Si se pulsa en “Aceptar” se produce la desconexión del usuario y se navega hasta la pantalla de autenticación.
Mapa (<i>MapFragment</i>)	<ul style="list-style-type: none"> • Se solicita el permiso para acceder a la localización del dispositivo. • Se comprueba que el servicio de localización está activado. • Si la localización actual del usuario no está disponible o no se tiene permiso para acceder a ella se consultan y se muestran sobre el mapa los usuarios situados en un radio de 45 km alrededor de la última ubicación conocida registrada en el servidor. • Si la localización actual del usuario está disponible y se tiene permiso para acceder a ella se consultan y se muestran sobre el mapa los usuarios situados en un radio de 45 km alrededor de la ubicación actual del usuario, se almacena la nueva ubicación en el servidor y se desactiva la recepción de actualizaciones de la localización. • Mientras se carga el mapa se muestra un indicador de progreso. • Al pulsar sobre el botón lupa se despliega el panel de búsqueda. • Al pulsar sobre un marcador se visualiza la información del usuario. • Al pulsar sobre la información de un usuario se navega hasta la pantalla de detalle de dicho usuario.

Búsqueda de usuarios <i>(SearchFragment)</i>	<ul style="list-style-type: none"> • Se realiza la búsqueda con los filtros seleccionados, actualizándose los marcadores sobre el mapa. • Si se busca por autor o por título introduciendo menos de 4 caracteres se muestra un mensaje de aviso. • Se muestra un mensaje de aviso si no se encuentra ningún resultado con los criterios especificados. • Al aumentar el radio de búsqueda disminuye el <i>zoom</i> sobre el mapa.
Detalle de usuario <i>(UserDetailActivity)</i>	<ul style="list-style-type: none"> • Se muestra la información de resumen del usuario: nombre, número de libros que comparte, valoración media y distancia a la que se encuentra. • La <i>AppBar</i> se expande y se contrae mostrando y ocultando la información de resumen del usuario. • Se muestra la lista de libros que comparte con sus fotografías, ordenada alfabéticamente por título. • Mientras se carga la lista de libros se muestra un indicador de progreso. • Al hacer el gesto de deslizar hacia abajo se refresca la lista de libros. • Al pulsar sobre el botón de navegación atrás se vuelve a la pantalla principal si se procedía de ella. Si se navegó desde la pantalla de detalle de intercambio se vuelve a dicha pantalla. • Al pulsar sobre un libro del listado se navega hacia la pantalla de detalle del libro.
Mis libros <i>(MyBooksFragment)</i>	<ul style="list-style-type: none"> • Se muestra la lista de libros compartidos por el usuario con sus fotografías, ordenada alfabéticamente por título. • Mientras se carga la lista de libros se muestra un indicador de progreso. • Al hacer el gesto de deslizar hacia abajo se refresca la lista de libros. • Al pulsar sobre un libro del listado se navega hacia la pantalla de detalle del libro. • Al pulsar sobre el botón “+” se navega hacia la pantalla de alta de un nuevo libro.
Nuevo libro <i>(NewBookActivity)</i>	<ul style="list-style-type: none"> • Se solicita el permiso de acceso a la cámara si no se dispone de él. • Se realiza la foto con la cámara del dispositivo y se carga en la pantalla. • Se selecciona una foto desde la galería y se carga en la pantalla. • Se puede leer un código de barras de un libro y se cargan automáticamente los datos obtenidos, incluyendo la imagen de la portada. • Al escribir manualmente un código de barras y pulsar sobre la lupa se cargan automáticamente los datos obtenidos, incluyendo la imagen de la portada. • La <i>AppBar</i> se expande y se contrae mostrando y ocultando la imagen. • Si el código de barras buscado no encuentra ningún libro se muestra un mensaje de aviso. • Se carga la lista desplegable de géneros. • Se carga la lista desplegable de idiomas. • Se comprueba que se introducen todos los campos obligatorios. • Se muestra un mensaje de aviso si al guardar el libro su ISBN ya está registrado en el servidor. • Se guarda el libro correctamente en Backendless y se incrementa el contador de libros del usuario. • Al guardar el libro se vuelve a la pantalla de listado, en la que aparece el nuevo libro ordenado en su posición correcta. • Al buscar por ISBN se muestra un indicador de progreso. • Al guardar los datos en el servidor se muestra un indicador de progreso.
Detalle de libro <i>(BookDetailActivity)</i>	<ul style="list-style-type: none"> • Se carga la imagen del libro. • El título del libro se visualiza con claridad sobre la imagen. • La <i>AppBar</i> se expande y se contrae mostrando y ocultando la imagen. • Se muestran correctamente los datos del libro. • El icono de estado se corresponde con el estado correcto. • Si se procede del listado de libros: <ul style="list-style-type: none"> ○ El botón de navegación atrás vuelve al listado de libros. ○ Se muestra un botón para eliminar el libro. ○ Al pulsar el botón de eliminar libro se muestra un diálogo de confirmación. ○ Se elimina el libro correctamente en Backendless y se decrementa el contador de libros del usuario al pulsar sobre el botón eliminar.

	<ul style="list-style-type: none"> ○ Se muestra un mensaje de aviso si se trata de eliminar un libro que está involucrado en un intercambio pendiente. ○ Al eliminar el libro se muestra un indicador de progreso. ○ Al eliminar el libro correctamente se vuelve a la pantalla de listado, de la que también se elimina. ● Si se procede de la pantalla de detalle de usuario: <ul style="list-style-type: none"> ○ El botón de navegación atrás vuelve al detalle de usuario. ○ Se muestra un botón para solicitar el intercambio del libro. ○ Al pulsar el botón de solicitar libro se muestra un diálogo de confirmación. ○ Se registra el intercambio del libro en Backendless y se envían los correos electrónicos de aviso a los usuarios al pulsar sobre el botón de solicitud del intercambio. ○ Al solicitar el intercambio se muestra un indicador de progreso. ○ Se muestra un mensaje de aviso si se solicita un libro que ya está involucrado en un intercambio pendiente.
<p>Intercambios (<i>SwapsFragment</i>)</p>	<ul style="list-style-type: none"> ● Se muestra la lista de intercambios en los que participa el usuario, ordenada por fecha descendente. ● Mientras se carga la lista de intercambios se muestra un indicador de progreso. ● Los intercambios solicitados por el usuario aparecen con el icono de la flecha hacia arriba. ● Los intercambios solicitados al usuario aparecen con el icono de la flecha hacia abajo. ● Los intercambios pendientes aparecen con el icono de color naranja. ● Los intercambios cancelados aparecen con el icono de color rojo. ● Los intercambios aceptados aparecen con el icono de color verde. ● Al hacer el gesto de deslizar hacia abajo se refresca la lista de intercambios. ● Al pulsar sobre un intercambio del listado se navega hacia la pantalla de detalle del intercambio.
<p>Detalle de intercambio (<i>SwapDetailActivity</i>)</p>	<ul style="list-style-type: none"> ● Se muestran los datos de los libros y usuarios involucrados, así como del estado en que se encuentra el intercambio. ● Si el intercambio está pendiente y ha sido solicitado al usuario se muestra una opción de menú para cancelarlo. ● Al pulsar sobre el botón de cancelación se muestra un diálogo de confirmación. ● Al cancelar el intercambio se registra correctamente el cambio de estado en Backendless y se actualiza la información mostrada en pantalla si el proceso es correcto. ● Se muestra un indicador de progreso al cancelar el intercambio. ● Si el intercambio está pendiente y ha sido solicitado al usuario se muestra una opción para seleccionar el libro a intercambiar. ● Al seleccionar el libro a intercambiar se actualiza en pantalla la información del intercambio y se muestra una opción de menú para valorarlo. ● El botón de navegación atrás regresa al listado de intercambios. Si el intercambio ha sido editado los cambios se reflejan en el listado. ● Se envían los correos a los usuarios al cancelar o aceptar el intercambio.
<p>Valorar intercambio (<i>RatingDialogFragment</i>)</p>	<ul style="list-style-type: none"> ● Si el intercambio está aceptado y aún no ha sido valorado por el usuario se muestra una opción de menú para realizar la valoración. ● Al pulsar sobre el botón de valorar se muestra un diálogo para seleccionar la valoración. ● El intercambio se actualiza correctamente en Backendless, incluyendo la valoración. También se actualiza la valoración media del usuario valorado. ● Se actualizan los datos mostrados en pantalla con la valoración añadida. ● Se muestra un indicador de progreso al realizar la valoración. ● El botón de navegación atrás regresa al listado de intercambios. ● Se envía el correo al usuario que ha sido valorado.

Tabla 26: Checklist de pruebas de verificación

Algunos de los errores detectados durante el desarrollo al realizar estas pruebas han sido los siguientes:

- La pantalla de *splash* no se visualizaba correctamente en versiones anteriores a Android 6.0, apareciendo el logo ocupando toda la pantalla.
- Tras el registro de una nueva cuenta no se realizaba el *login* automáticamente, siendo necesario hacer la petición al servidor de Backendless de manera explícita.
- En caso de que el usuario denegara el permiso para acceder a su localización (en Android 6.0 o posterior) no se lanzaba la consulta al servidor con la última ubicación conocida, por lo que el mapa no aparecía centrado ni mostraba ningún marcador de usuario.
- Se producía un error al intentar realizar una fotografía para adjuntarla a un libro al darlo de alta, al no haberse solicitado el permiso de acceso a la cámara (en Android 6.0 o posterior).
- Al agregar un libro y regresar al listado éste no se insertaba de manera ordenada, sino al final del mismo.
- Al consultar los datos de un libro mediante su ISBN se mostraba el indicador de progreso, pero no se ocultaba al obtener respuesta del servidor.
- Tras valorar un intercambio la pantalla de detalle no reflejaba la puntuación otorgada.
- La *AppBar* de la pantalla de detalle de usuario no se ocultaba al deslizarlo hacia arriba.
- La *AppBar* de la pantalla de detalle de usuario no mostraba correctamente la distancia.

A la verificación a través del *checklist* de pruebas manuales se ha añadido un conjunto de pruebas automáticas implementadas por medio del *framework* Espresso, que permite la creación de pruebas de interfaz que simulan la interacción del usuario con la aplicación de una manera automatizada, y evaluar tanto las respuestas de la aplicación como los resultados obtenidos.

A través de la herramienta se han generado test automáticos para probar, de forma complementaria, un subconjunto de funcionalidades de la aplicación:

- **LoginActivityTest:** Verifica el funcionamiento de la pantalla de autenticación, comprobando el comportamiento de la aplicación tanto cuando las credenciales son correctas como cuando no lo son.
- **RegistrationActivityTest:** Prueba el comportamiento de la pantalla de registro, realizando un registro válido y comprobando que se muestran todos los mensajes correspondientes cuando no se introducen los campos obligatorios. Para el desarrollo de esta prueba ha sido necesario utilizar, de forma complementaria, el *framework* de *testing* UI Automator, para simular la interacción con la herramienta de autocompletado de Google Places, ya que Espresso no se integra correctamente con ella.
- **LogoutTest:** Valida el proceso de desconexión de la aplicación, autenticando al usuario desde la pantalla de *login* y cerrando sesión una vez se está dentro de la pantalla principal.

- **SearchTest:** Simula y valida el proceso de búsqueda de usuarios sobre el mapa, realizando previamente la autenticación del usuario. Como en el caso de la pantalla de registro, para esta prueba se ha utilizado complementariamente UI Automator, para simular la pulsación sobre un marcador del mapa y navegar a la pantalla de detalle de usuario, ya que Espresso no es capaz de simular esta interacción.
- **AddDeleteBookTest:** Verifica el funcionamiento de los procesos de registro y borrado de un libro. Para ello, se autentica al usuario en la aplicación, se registra un nuevo libro, se comprueba que se ha añadido y finalmente se elimina.

Para la realización de las pruebas automáticas se ha desarrollado una clase auxiliar, *ViewVisibilityIdlingResource*, que implementa lo que en Espresso se denomina un *IdlingResource*. Este concepto no es más que una manera de indicarle al *framework* de *testing* que la aplicación está ocupada realizando algún tipo de acción en segundo plano, como puede ser una llamada al servidor de Backendless, y que mientras se esté ejecutando no debe seguir simulando las interacciones definidas en la prueba ya que, en caso contrario, fallará, al no haberse completado la tarea en segundo plano. En el caso de la aplicación, cada vez que se lanza una acción en segundo plano se muestra un indicador de progreso, que se oculta cuando ésta termina. Esta circunstancia es la que se ha aprovechado para implementar el *IdlingResource*, a través de la clase mencionada, *ViewVisibilityIdlingResource*, que comunica a Espresso que la aplicación está ocupada cuando el indicador de progreso está visible, y ociosa cuando está oculto, permitiendo así la correcta ejecución de las pruebas automáticas.

Todos los tests automatizados han sido probados sobre la aplicación en diversas ocasiones con resultado satisfactorio:

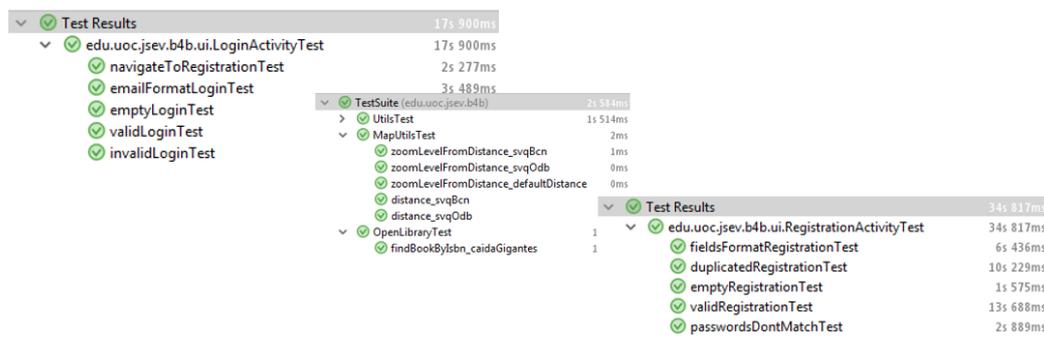


Figura 47: Resultados de ejecución de pruebas automáticas

3.5 Funcionalidades desarrolladas

A continuación, se muestran capturas de la ejecución de la aplicación en un dispositivo real, en las que se puede observar la implementación de las funcionalidades definidas en los casos de uso del proyecto:

3.5.1 Splash, autenticación y registro de usuarios

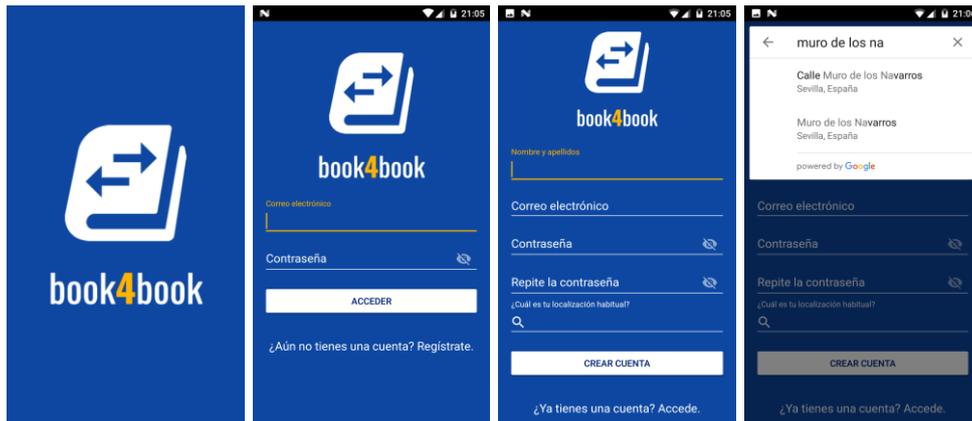


Figura 48: Splash, autenticación y registro de usuarios

3.5.2 Pantalla principal, mapa y búsqueda de usuarios

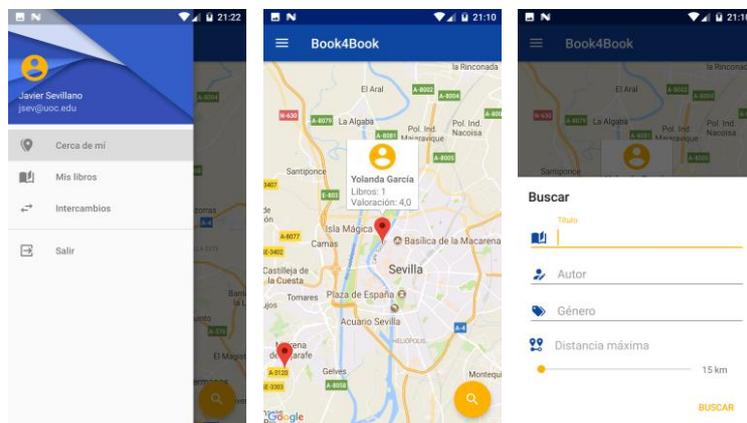


Figura 49: Pantalla principal, mapa y búsqueda de usuarios

3.5.3 Detalle de usuario y detalle de libro compartido por otro usuario

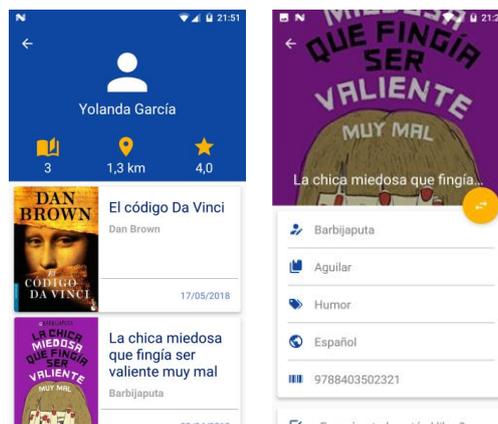


Figura 50: Detalle de usuario y detalle de libro compartido por otro usuario

3.5.4 Listado de libros del usuario, detalle de libro y nuevo libro

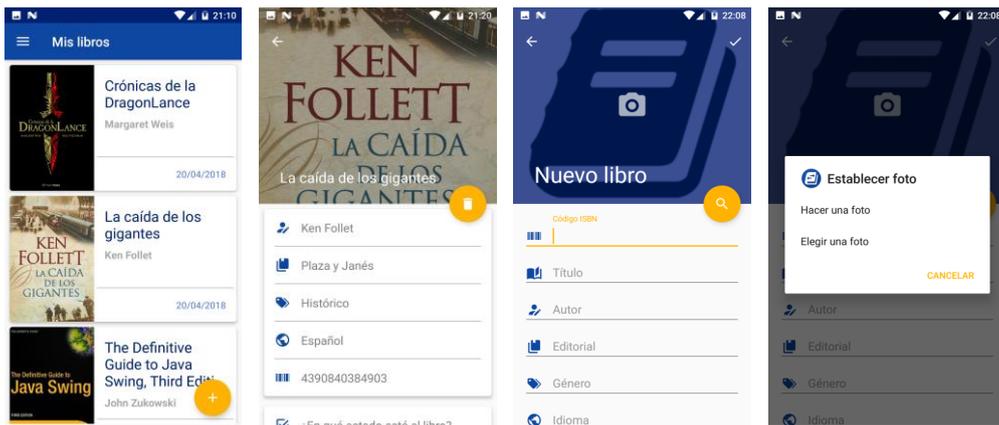


Figura 51: Listado de libros del usuario, detalle de libro y nuevo libro

3.5.5 Listado de intercambios y gestión de intercambio



Figura 52: Listado de intercambios y gestión de intercambio

3.5.6 Valoración de intercambio y desconexión

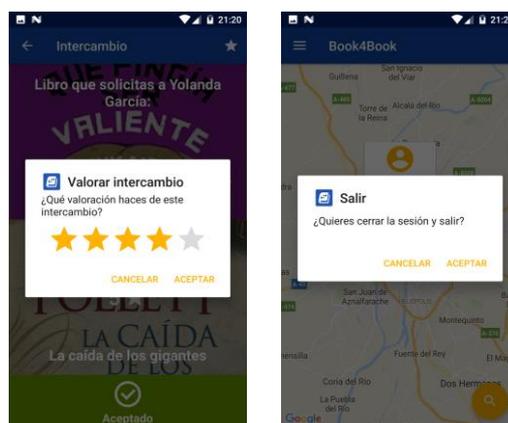


Figura 53: Valoración de intercambio y desconexión

4. Conclusiones

La idea de desarrollar la aplicación surgió tras conversación con una persona cercana que me comentó la cantidad de tiempo que necesitaba cada vez que hacía limpieza en su cuarto de lectura, debido a la enorme cantidad de novelas antiguas que acumulaba, muchas de las cuales, además, no sabía qué hacer con ellas, al no dejarle espacio para colocar nuevos ejemplares. Como me pareció una problemática bastante habitual la consideré un tema adecuado para desarrollar en el marco del trabajo final de máster.

La ejecución del trabajo ha supuesto un gran reto en lo personal, al implicar una gran dedicación y esfuerzo y tener que compaginarlo laboralmente, pero a la vez ha sido una experiencia gratificante, ya que me ha ofrecido la posibilidad de materializar una idea propia, desde sus etapas iniciales hasta cobrar forma con el producto final, la aplicación móvil. De la misma forma, me ha permitido afianzar y practicar los conocimientos adquiridos durante el máster, así como aplicarlos también en el ámbito profesional.

En la redacción de la memoria se han tratado de justificar todas las decisiones tomadas a lo largo del trabajo, lo que ha permitido constatar la importancia de decidir adecuadamente en las fases iniciales para minimizar el riesgo de desviaciones en la planificación.

La tarea más costosa y que más dudas ha planteado ha sido la adopción del patrón MVP en la construcción de la aplicación, puesto que no se disponía de ninguna experiencia previa en la utilización de esta filosofía de diseño dentro del desarrollo para la plataforma Android. A pesar de las dificultades, desde el primer momento se ha considerado como mala práctica condensar la mayoría del código de la aplicación en las actividades y fragmentos, por eso se ha hecho el esfuerzo de intentar separar las diferentes responsabilidades utilizando este patrón.

Revisando los objetivos definidos al inicio del trabajo se puede afirmar que han sido cubiertos en su totalidad con la aplicación desarrollada que, a través de las funcionalidades que ofrece, cumple con los requisitos funcionales y de diseño planteados. Únicamente se ha dejado de implementar la tarea de envío de notificaciones *push* a los dispositivos, para informar a los usuarios de cambios en el estado de los intercambios. Los motivos que han llevado a no ejecutar esta tarea han sido los siguientes:

- Se trata de una tarea marcada en la planificación como mejora opcional, ya que la necesidad que pretende cubrir ya está resuelta por medio del envío de correos electrónicos.
- El tiempo inicialmente planificado para ella se ha empleado en mejorar la determinación de la localización por defecto del usuario a la hora de registrarse en la aplicación, mediante la integración con la API Google Places. Durante el desarrollo se ha considerado que el diseño inicialmente propuesto para esta funcionalidad suponía una limitación de

cara al usuario, por lo que se ha preferido compensar esta carencia implementando esta mejora en lugar de las notificaciones *push*.

A nivel de gestión del trabajo se ha comprobado que resulta fundamental realizar una buena planificación al inicio del mismo para que concluya con éxito. En ese sentido, la planificación propuesta ha permitido cumplir escrupulosamente con las fechas de entrega y los objetivos fijados en cada una de ellas, a la vez que ha facilitado la detección temprana de desviaciones y su posterior corrección. Estos desvíos se han producido, fundamentalmente, en la fase de implementación, y han obligado a tomar medidas correctoras para asegurar el cumplimiento de la planificación, como aumentar la dedicación hasta las 5 horas en días laborables y hasta las 12 en días festivos y fines de semana, debido a problemas de salud de un familiar cercano y a una mayor carga laboral durante este período. Todo ello redujo el número de días disponibles.

Determinadas tareas han supuesto una dedicación superior a la estimada inicialmente, como, por ejemplo, el desarrollo de la pantalla de gestión de intercambios, sin embargo, ese retraso se ha visto compensado por otras tareas que se han ejecutado en un tiempo inferior al previsto, como la lectura de códigos ISBN con la cámara o la búsqueda de libros a través de la API de OpenLibrary.

A pesar de lo anterior, en general, la ejecución del trabajo se ha ajustado bastante a la planificación temporal propuesta, lo que ha permitido alcanzar los objetivos definidos para cada fase del proyecto sin retrasos en las fechas previstas.

En lo referente a la metodología empleada, la selección del modelo en cascada en un proyecto de estas características se considera que ha supuesto un acierto, ya que ha permitido:

- Desarrollar un producto completo en el tiempo establecido, siguiendo las diferentes etapas del desarrollo de software.
- Cumplir de manera más fácil con las fechas de los hitos intermedios.
- Comprender la importancia que tienen las fases iniciales de análisis y diseño, ya que son las que marcan la calidad del entregable final y permiten construir un producto centrado en las necesidades reales del usuario.
- Ayudar a reducir el tiempo total de desarrollo, gracias al esfuerzo dedicado en las etapas de análisis y diseño.

Como conclusión final, comentar que el desarrollo del trabajo ha supuesto un reto ilusionante, a la vez que muy motivador, ya que me ha permitido desarrollar una aplicación completa desde el inicio haciendo uso de una tecnología que me apasiona y que, con los conocimientos adquiridos, espero poder seguir aplicando en el ámbito profesional.

4.1 Líneas de trabajo futuro

Como el trabajo final de máster tiene una duración muy limitada, en las etapas iniciales del mismo se establecieron una serie de requisitos concretos para que la aplicación objetivo fuese completamente funcional a su conclusión. No obstante, existen multitud de aspectos y nuevas funcionalidades que se pueden añadir para complementarla y mejorarla:

- Permitir al usuario modificar los datos de su cuenta (nombre, fotografía, contraseña, etcétera), así como darse de baja de la aplicación, eliminando su perfil.
- Ofrecer al usuario la posibilidad de recordar su contraseña y permanecer autenticado, para evitar que tenga que introducir sus credenciales cada vez que acceda a la aplicación.
- Añadir una funcionalidad que permita al usuario restaurar su contraseña en caso de que la olvide.
- Añadir a los libros ofrecidos una valoración/reseña personal, para que el resto de usuarios disponga de más información sobre ellos.
- Permitir la edición de los datos de un libro registrado. Actualmente, una vez dado de alta, no se permite su modificación.
- Implementar una lista de deseos, de manera que la aplicación avise a los usuarios cuando alguien comparta un libro que se encuentre en ella.
- Mejorar la escalabilidad del sistema, la atonicidad de las operaciones y reducir el número de llamadas al servidor, implementando parte de la lógica de negocio en el propio *backend*.
- Informar a los usuarios sobre los cambios de estado de sus intercambios por medio de notificaciones *push*, adicionalmente al correo electrónico.
- Permitir el acceso a determinadas funcionalidades de la aplicación sin necesidad de registro previo de los usuarios como, por ejemplo, el poder visualizar los usuarios cercanos en el mapa.
- Ofrecer algún tipo de herramienta integrada en la aplicación para que los usuarios se pongan en contacto entre ellos una vez el intercambio ha sido aceptado, como un tablón de mensajes o un sistema de mensajería instantánea.
- Adaptar la aplicación para su utilización en tabletas, ya que se ha implementado teniendo como objetivo únicamente los teléfonos móviles.
- Desarrollar la aplicación en otras plataformas móviles, principalmente en iOS, que es el sistema operativo con más cuota de mercado tras Android.
- Estudiar la viabilidad de un plan de monetización para la aplicación, mediante la inserción de publicidad dentro de ella (publicidad *in-app*).

5. Glosario

- **Android.** Sistema operativo de Google para dispositivos móviles basado en el núcleo de Linux. Se trata de un proyecto de código abierto.
- **Android Studio.** Entorno de desarrollo integrado oficial para Android.
- **API (*Application Programming Interface*).** Interfaz de programación de aplicaciones. Conjunto de métodos publicado por una librería para ser utilizado por otro software como capa de abstracción.
- **APK (*Android aPplication pacKage*).** Formato de archivo empaquetado utilizado para distribuir e instalar aplicaciones para la plataforma Android.
- **App (*Application*).** Término utilizado para referirse a las aplicaciones desarrolladas para dispositivos móviles.
- **BaaS (*Backend as a Service*).** *Backend* como servicio. Modelo que permite vincular aplicaciones con servicios de *backend* gestionados en la nube, como análisis y almacenamiento de datos, notificaciones push o gestión de usuarios y roles.
- **Backend.** Parte de un sistema que se encarga de procesar las peticiones procedentes de la capa con la que interactúa el usuario, acceder a los datos y devolverle la información solicitada.
- **Backendless.** Plataforma BaaS empleada en el desarrollo de la parte servidora del trabajo.
- **Diagrama de Gantt.** Herramienta gráfica que muestra el tiempo de dedicación previsto para cada una de las tareas en las que se divide un proyecto.
- **Diseño centrado en el usuario (DCU).** Filosofía de diseño de productos y aplicaciones que sitúa al usuario en el centro de todo el proceso.
- **Estructura de descomposición del trabajo (EDT).** Desglose jerárquico del trabajo que tiene que ser ejecutado en el marco de un proyecto, orientado a organizar y definir el alcance total del mismo.
- **Framework.** Estructura conceptual y tecnológica de soporte definido, normalmente con módulos de software concretos, en base a la cual otro proyecto software puede ser más fácilmente organizado y desarrollado.
- **IDE (*Integrated Development Environment*).** Entorno de desarrollo integrado. Aplicación que integra un conjunto de servicios y utilidades para facilitar al programador el desarrollo de software.

- **Interactor.** Objeto simple de propósito único cuyo objetivo es encapsular la lógica de negocio asociada a un determinado caso de uso.
- **Intercambio.** Concepto central de la aplicación. Representa el cambio recíproco de dos libros entre dos usuarios de la aplicación, que la utilizan como medio de contacto para realizarlo.
- **Intercambio entrante:** Intercambio solicitado al usuario, iniciado por otro usuario distinto.
- **Intercambio saliente:** Intercambio iniciado por el usuario, solicitando un libro a otro usuario.
- **iOS.** Sistema operativo móvil de Apple para iPhone, iPad y iPod Touch.
- **Modelo-vista-presentador (MVP).** Patrón arquitectónico de interfaz de usuario que permite separar los datos y la lógica de negocio de la lógica de presentación, utilizando para ello un componente, el presentador, que actúa como intermediario entre la interfaz (vista) y los datos (modelo).
- **Notificación *push*.** Mensaje directo que se envía a uno o varios dispositivos móviles para informar a sus usuarios de un determinado evento en una aplicación.
- **Open source.** Código abierto. Modelo de desarrollo de software basado en la colaboración abierta, que se centra en la premisa de que al compartir el código fuente el programa resultante tiende a ser de una calidad superior al software propietario.
- **Participante:** Usuario que interviene en el intercambio y que no es el usuario que accede a la aplicación, es decir, la otra parte implicada en el intercambio.
- **SDK (Software Development Kit).** Conjunto de herramientas de desarrollo que permiten crear software para una determinada plataforma.
- **TFM.** Trabajo final de máster.
- **Usuario origen/iniciador.** Usuario de la aplicación que crea un nuevo intercambio, solicitando a otro uno de los libros que comparte. Se trata del usuario que inicia todo el proceso, y queda a la espera de que el otro usuario acepte o rechace la solicitud.
- **Usuario destino/destinatario.** Usuario de la aplicación que recibe la solicitud de otro usuario para realizar un intercambio de libros. Este usuario no genera un intercambio nuevo, sino que rechaza la solicitud recibida o la acepta seleccionando un libro del usuario origen.

6. Bibliografía

1. **Federación de Gremios de Editores de España** [Internet]. Barómetro de Hábitos de Lectura y Compra de Libros en España 2017. [Consultado en marzo de 2018]. Disponible en: <http://federacioneditores.org/img/documentos/HabitosLecturaCompraLibros2017.pdf>
2. **Ministerio de Educación Cultura y Deporte** [Internet]. Anuario de Estadísticas Culturales 2017. Disponible en: http://www.mecd.gob.es/dam/jcr:a8c481ba-e9d6-498c-ab97-4e199e3f1197/Anuario_de_Estadisticas_Culturales_2017.pdf
3. **Google Play Store** [Internet]. BookSwap. [Consultado en marzo de 2018]. Disponible en: <https://play.google.com/store/apps/details?id=com.smarttie.bookswap>
4. **Google Play Store** [Internet]. Booxup. [Consultado en marzo de 2018]. Disponible en: <https://play.google.com/store/apps/details?id=com.booxup>
5. **Google Play Store** [Internet]. Obsso. [Consultado en marzo de 2018]. Disponible en: <https://play.google.com/store/apps/details?id=com.obsso.obsso>
6. **Google Play Store** [Internet]. Wendoo. [Consultado en marzo de 2018]. Disponible en: <https://play.google.com/store/apps/details?id=com.wendoo>
7. **Google Play Store** [Internet]. Trueketeke. [Consultado en marzo de 2018]. Disponible en: <https://play.google.com/store/apps/details?id=com.trueketeke.trueketeke>
8. **Smart Insights** [Internet]. Mobile Marketing Statistics compilation. 20 de enero de 2018. [Consultado en marzo de 2018]. Disponible en: <https://www.smartinsights.com/mobile-marketing/mobile-marketing-analytics/mobile-marketing-statistics/>
9. **Flurry Analytics Blog** [Internet]. Flurry State of Mobile 2017. 10 de enero de 2018. [Consultado en marzo de 2018]. Disponible en: <http://flurrymobile.tumblr.com/post/169545749110/state-of-mobile-2017-mobile-stagnates>
10. **Gartner** [Internet]. Gartner Says Worldwide Sales of Smartphones Recorded First Ever Decline During the Fourth Quarter of 2017. 22 de febrero de 2018. [Consultado en marzo de 2018]. Disponible en: <https://www.gartner.com/newsroom/id/3859963>
11. **Kantar Worldpanel** [Internet]. Android vs. iOS. Smartphone OS sales market share evolution. [Consultado en marzo de 2018]. Disponible en: <https://www.kantarworldpanel.com/global/smartphone-os-market-share>
12. **Google Play Store** [Internet]. Book Swap. [Consultado en marzo de 2018]. Disponible en: <https://play.google.com/store/apps/details?id=com.bookswapdroid>
13. **Ditrendia** [Internet]. Informe Mobile en España y en el Mundo 2017. [Consultado en marzo de 2018]. Disponible en: https://www.amic.media/media/files/file_352_1289.pdf
14. **Android Developers** [Internet]. Design. [Consultado en abril de 2018]. Disponible en: <https://developer.android.com/design/>
15. **Material Design** [Internet]. Navigation drawer pattern. [Consultado en abril de 2018]. Disponible en: <https://material.io/guidelines/patterns/navigation-drawer.html>
16. **Material Design** [Internet]. Floating action button. [Consultado en abril de 2018]. Disponible en: <https://material.io/guidelines/components/buttons-floating-action-button.html>
17. **Material Design** [Internet]. Navigation patterns. [Consultado en abril de 2018]. Disponible en: <https://material.io/guidelines/patterns/navigation.html#navigation-up-back-buttons>

18. **Android Developers** [Internet]. Distribution dashboards – Google Play install stats. [Consultado en mayo de 2018]. Disponible en: <https://developer.android.com/about/dashboards/#google-play-install-stats>
19. **Android Developers** [Internet]. Screen compatibility overview. [Consultado en mayo de 2018]. Disponible en: https://developer.android.com/guide/practices/screens_support
- Android Developers** [Internet]. Documentation for app developers. [Consultado en abril-mayo de 2018]. Disponible en: <https://developer.android.com/docs>
- AndroidPub** [Internet]. RecyclerView in MVP – Passive view's approach. [Consultado en mayo de 2018]. Disponible en: <https://android.jelise.eu/recyclerview-in-mvp-passive-views-approach-8dd74633158>
- Antonio Leiva** [Internet]. MVP for Android: how to organize the presentation layer. [Consultado en abril de 2018]. Disponible en: <https://antonioleiva.com/mvp-android>
- Backendless** [Internet]. Backendless SDK for Android. [Consultado en abril-mayo de 2018]. Disponible en: <https://backendless.com/docs/android/doc.html>
- ErikJordan_Rey** [Internet]. Model View Presenter en Android. [Consultado en abril de 2018]. Disponible en: <https://erikcaffrey.github.io/ANDROID-mvp>
- Google Maps** [Internet]. Google Maps Android API. [Consultado en mayo de 2018]. Disponible en: <https://developers.google.com/maps/documentation/android-api/?hl=es-419>
- Google Places** [Internet]. Google Places API for Android. [Consultado en mayo de 2018]. Disponible en: <https://developers.google.com/places/android-api/?hl=es-419>
- Material Palette** [Internet]. Material Design Color Palette. [Consultado en abril de 2018]. Disponible en: <https://www.materialpalette.com>
- Mindorks** [Internet]. Android MVP Architecture Extension with Interactors and Repositories. [Consultado en abril de 2018]. Disponible en: <https://blog.mindorks.com/android-mvp-architecture-extension-with-interactors-and-repositories-bd4b51972339>
- StackOverflow** [Internet]. Varios posts. [Consultado en marzo-mayo de 2018]. Disponible en: <https://stackoverflow.com>
- UOC** [Internet]. Métodos para el desarrollo de dispositivos móviles. [Consultado en marzo de 2018]. Disponible en: http://materials.cv.uoc.edu/daisy/Materials/PID_00176750/pdf/PID_00176755.pdf
- UOC** [Internet]. Redacción de textos científico-técnicos. [Consultado en marzo-mayo de 2018]. Disponible en: http://materials.cv.uoc.edu/cdocent/_D_CBBU62JZTHQ9CQQGJ.pdf
- UOC** [Internet]. El trabajo final como proyecto. [Consultado en marzo de 2018]. Disponible en: <http://cvapp.uoc.edu/autors/MostraPDFMaterialAction.do?id=197260>
- UOC** [Internet]. Usuarios y sistemas interactivos. [Consultado en abril de 2018]. Disponible en: http://materials.cv.uoc.edu/daisy/Materials/PID_00143568/pdf/PID_00143566.pdf
- UOC** [Internet]. Diseño centrado en el usuario para dispositivos móviles. [Consultado en abril de 2018]. Disponible en: <http://cv.uoc.edu/webapps/xwiki/wiki/matm1202es/view/Main/WebHome>
- Wikipedia** [Internet]. Varios artículos. [Consultado en marzo-mayo de 2018]. Disponible en: <https://es.wikipedia.org>

Anexo I: Diagrama de Gantt de planificación

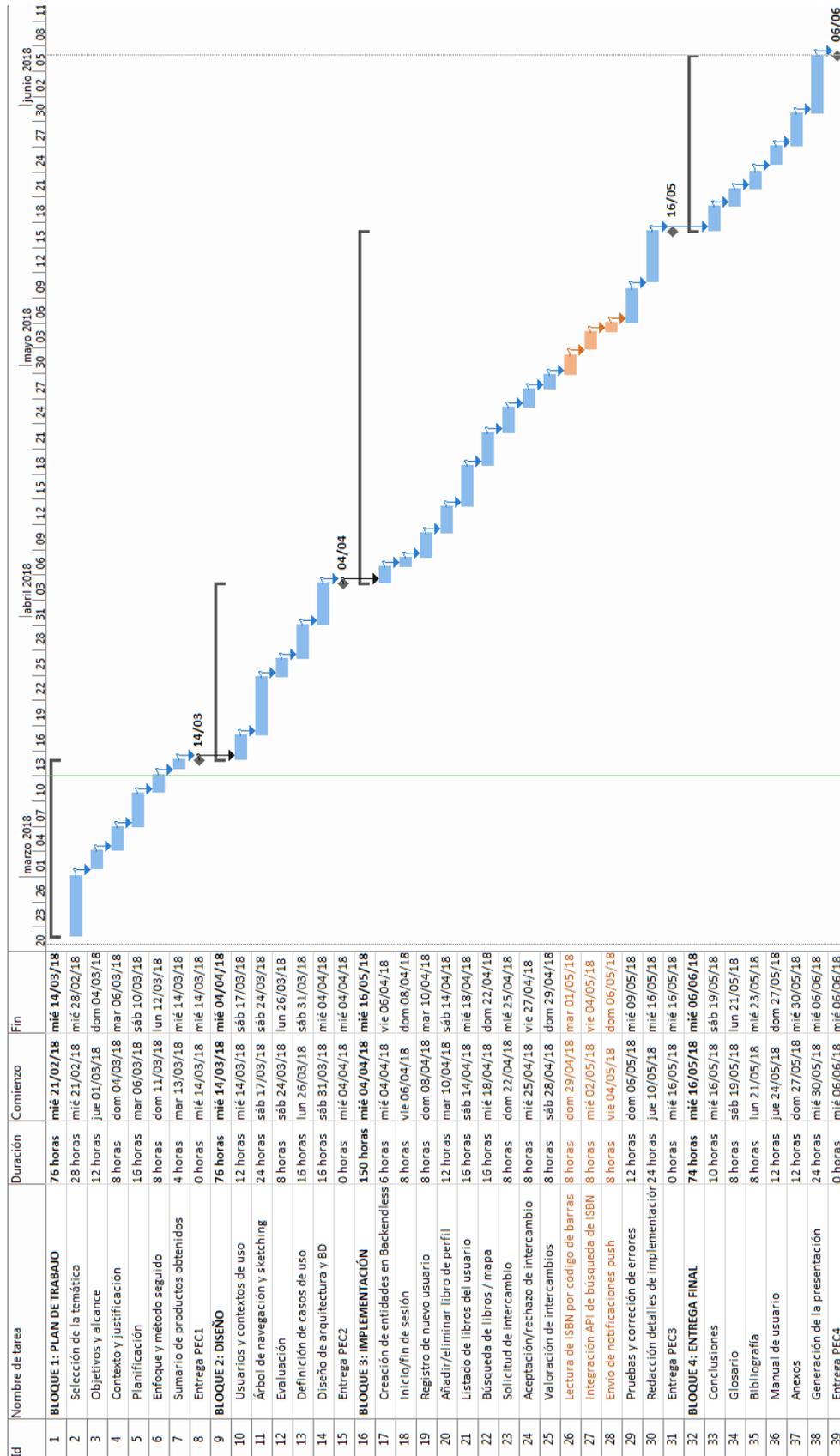


Figura 54: Diagrama de Gantt con la planificación del trabajo

Anexo II: Árbol de navegación con miniaturas

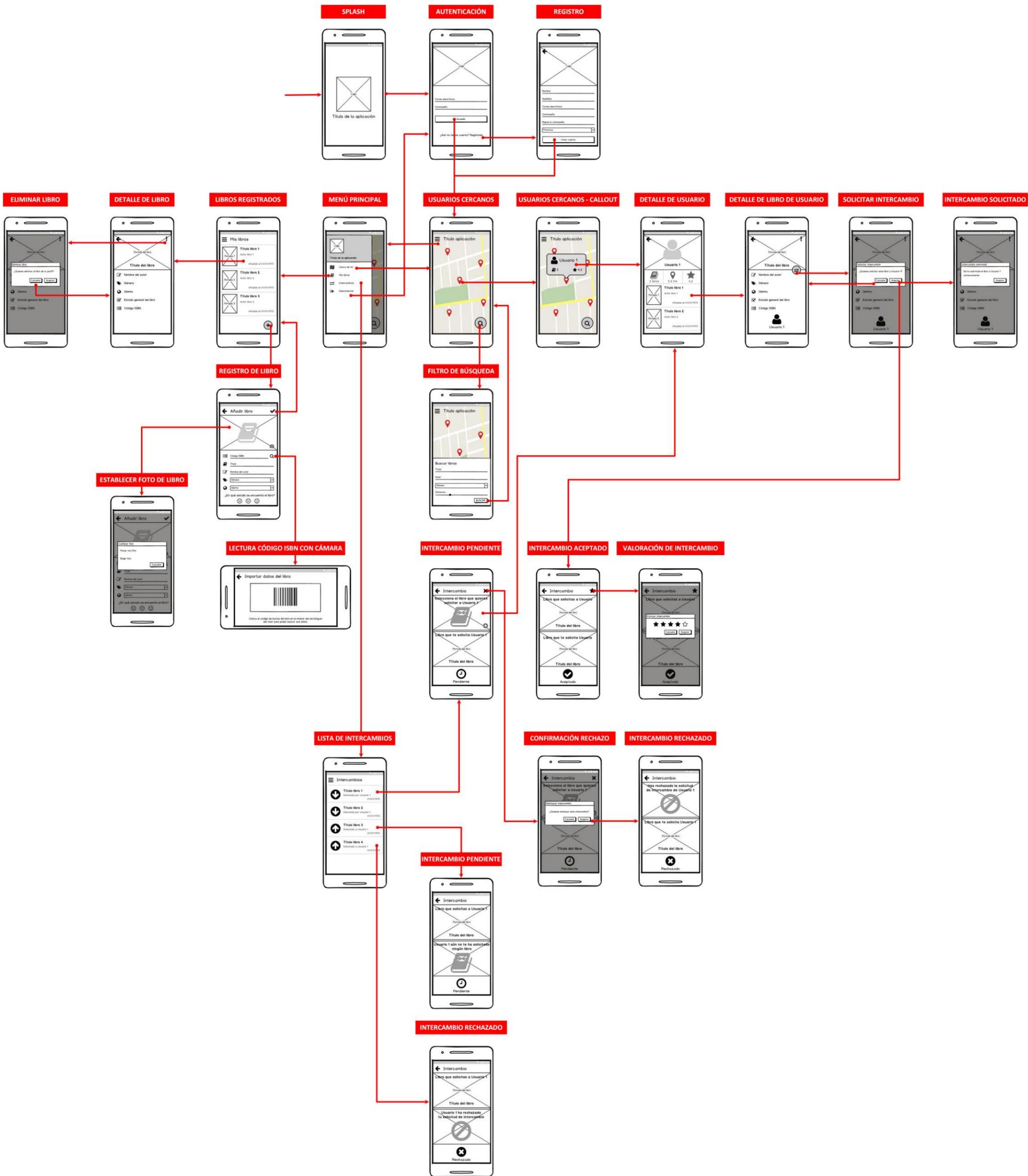


Figura 55: Árbol de navegación con miniaturas de los prototipos de baja fidelidad

Anexo III: Instrucciones de compilación

La aplicación ha sido desarrollada sobre Android Studio versión 3.1.2, utilizando para su compilación un nivel de API 27.

Las dependencias de la aplicación aparecen reflejadas en el archivo *build.gradle* del módulo *app*, empleándose la última versión de la librería de compatibilidad de Android, que en la fecha de entrega del trabajo es la 27.1.1.

La aplicación hace uso de las APIs de Google Maps y Google Places, utilizando para ello una clave de API generada con una cuenta personal de Google. La clave ha sido configurada sin restricciones de huella digital, por lo que las APIs pueden ser invocadas desde cualquier archivo binario APK generado, independientemente del certificado con el que haya sido firmado. Si se desea utilizar una clave de API diferente bastaría con editar el archivo de recurso *api_keys.xml* y establecerla en la propiedad *google_maps_key*.

La versión utilizada de la librería para acceder a los servicios de Google es la 15.0.1. Para que las llamadas a Google Places y Google Maps funcionen correctamente, si la aplicación se ejecuta desde un emulador, es necesario que éste disponga de dicha versión de los servicios. Para el caso concreto de Android Oreo, las últimas actualizaciones disponibles de las imágenes de los emuladores incluyen esta versión de los servicios de Google.

Para lanzar los tests automáticos se han empleado las últimas versiones disponibles del *framework* Espresso (3.0.1) y de la librería de reglas (1.0.1) en el momento de desarrollo de las pruebas, no habiéndose ejecutado con ninguna otra versión.

Aparte de las consideraciones anteriores, el proceso de compilación de la aplicación es el habitual, no siendo necesaria ninguna acción especial.

Anexo IV: Manual de usuario

Autenticación y registro

Cuando el usuario accede a la aplicación se encuentra con una pantalla de presentación o *splash*, a modo de bienvenida.

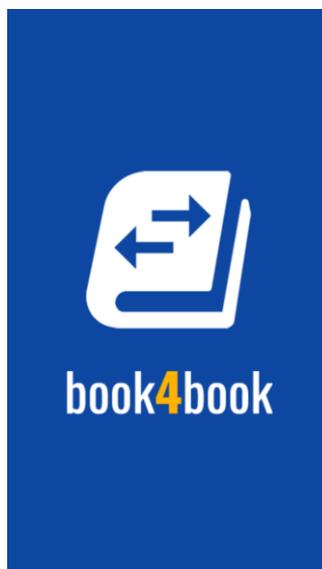


Figura 56: Bienvenida

La pantalla de presentación da paso automáticamente a la de autenticación. Es necesario disponer de una cuenta de usuario para poder utilizar la aplicación. Para autenticarse basta con introducir el correo electrónico del usuario y la contraseña, y pulsar sobre el botón "Acceder". Ambos campos son obligatorios. Si algún campo no se completa o las credenciales introducidas no son correctas se mostrará un mensaje de error y no se accederá a la aplicación. Por el contrario, si el correo y la contraseña son válidos el usuario se autenticará y entrará en la pantalla principal de la aplicación.



Figura 57: Autenticación

Si el usuario no dispone de una cuenta puede registrar una nueva pulsando en el texto “*¿Aún no tienes una cuenta? Regístrate*”, que mostrará la pantalla de creación de una nueva cuenta de usuario:



Figura 58: Registro de usuario

Para registrar la cuenta el usuario debe proporcionar la siguiente información, toda ella obligatoria:

- Nombre y apellidos.
- Correo electrónico, que debe tener un formato válido.
- Contraseña, que debe tener un mínimo de 6 caracteres.
- Ubicación habitual, que será utilizada por la aplicación en caso de que el usuario no active el servicio de localización de su dispositivo para determinar su posición actual.

Para seleccionar la ubicación habitual basta con pulsar sobre el campo de texto. Se mostrará un campo de búsqueda para que el usuario introduzca una dirección, lugar o municipio. Conforme el usuario vaya escribiendo se le irán mostrando las opciones coincidentes en un desplegable, de las que podrá seleccionar una de ellas para que se rellene el campo del formulario de registro.

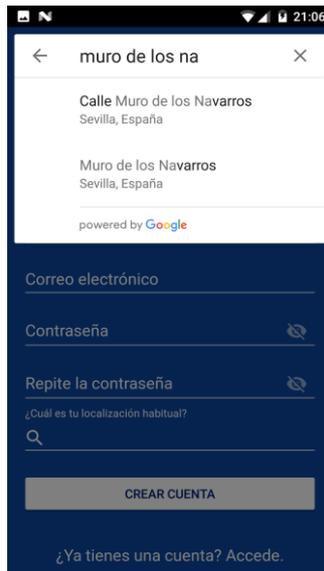


Figura 59: Herramienta de autocompletado de Google Places

Si no se completa alguno de los campos o se introduce algún valor no válido se informará al usuario por medio de mensajes de error. En caso de que todos los campos se hayan rellenado correctamente, al pulsar sobre el botón “*Crear cuenta*” la aplicación registrará los datos y llevará al usuario a la pantalla principal.

Se puede volver en todo momento a la pantalla de autenticación desde la pantalla de registro pulsando sobre el texto “*¿Ya tienes una cuenta? Accede*”.

Búsqueda de usuarios

Cuando el usuario se ha autenticado en la aplicación se le muestran en un mapa los usuarios que comparten algún libro y que se encuentran a 45 kilómetros o menos de su posición. Para poder determinar la ubicación la aplicación solicitará al usuario permiso para acceder a la misma, antes de mostrar el mapa. Si el usuario concede el permiso, la aplicación comprobará si el servicio de localización está activado. Si lo está, obtendrá la ubicación actual del usuario y la utilizará para ubicarlo en el mapa, si no lo está, le preguntará si desea activarlo. Si el usuario activa el servicio de localización la aplicación obtendrá su ubicación.

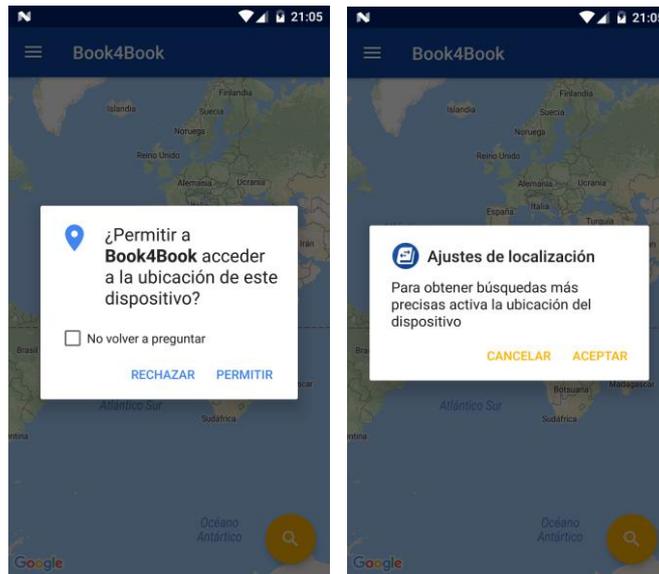


Figura 60: Permiso de acceso a la localización y activación del servicio

Si el usuario no activa el servicio de localización o no da permiso para acceder a la ubicación del dispositivo, la aplicación utilizará la última posición proporcionada por él para situarlo en el mapa.



Figura 61: Mapa con la posición de usuarios cercanos

El usuario puede acotar la búsqueda pulsando sobre el botón circular con la lupa. Al pulsar sobre él se mostrará un filtro para que establezca los criterios que desee, de entre los siguientes:

- Título del libro.
- Autor del libro.
- Género literario del libro.
- Distancia máxima a la que se puede encontrar el usuario.

Si se introduce como criterio el título o el autor, se debe proporcionar un mínimo de cuatro caracteres en cada uno. En caso contrario, se mostrará un mensaje de aviso al usuario al pulsar sobre el botón “*Buscar*”.

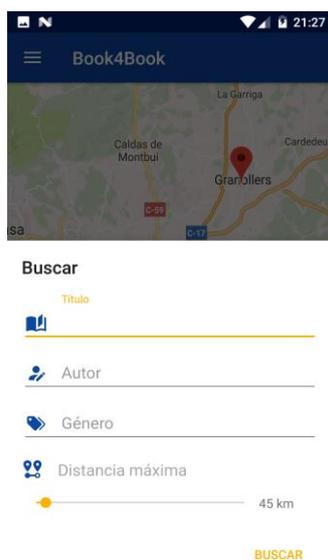


Figura 62: Búsqueda de usuarios

Cuando el usuario pulse sobre “*Buscar*”, la aplicación mostrará sobre el mapa aquellos usuarios que compartan algún libro que verifique todos los criterios, y que se sitúe dentro del radio de búsqueda proporcionado. Si no se encuentra ningún usuario que verifique los criterios se informará al usuario por medio de un mensaje.

Iniciar un intercambio

Para iniciar un intercambio es necesario seleccionar sobre el mapa el usuario al que se le quiere pedir un libro. Para seleccionar un usuario basta con pulsar sobre el marcador en el mapa. Aparecerá un bocadillo con información sobre el mismo, como su nombre, el número de libros que comparte y la valoración media que ha recibido de otros usuarios que han realizado intercambios con él.



Figura 63: Información del usuario sobre el mapa

Al pulsar sobre el bocadillo del usuario se navegará a la pantalla de detalle del mismo, en la que estará disponible la siguiente información:

- Nombre del usuario.
- Número de libros que comparte.
- Distancia a la que se encuentra.
- Valoración media recibida.
- Lista de libros que comparte. Por cada libro aparece:
 - Ilustración.
 - Título.
 - Autor.
 - Fecha en la que el usuario lo registró en la aplicación.



Figura 64: Detalle de usuario

Si el usuario desea obtener más información sobre alguno de los libros compartidos, al pulsar sobre la tarjeta correspondiente navegará a la pantalla

de detalle, donde podrá consultar información ampliada sobre el libro seleccionado:

- Ilustración.
- Título.
- Autor.
- Editorial.
- Género literario.
- Idioma.
- Código ISBN.
- Estado en que se encuentra (bueno, malo o regular).

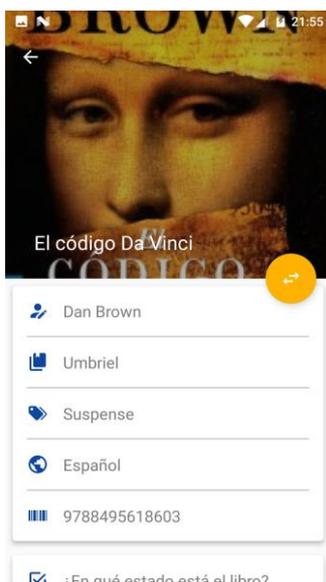


Figura 65: Detalle de libro de otro usuario

Si después de consultar los datos del libro el usuario está interesado en solicitarlo, bastará con que pulse sobre el botón circular con las flechas. Se le pedirá confirmación y, si acepta, se registrará la solicitud de intercambio, que quedará reflejada en el listado de intercambios en los que ha participado el usuario, al que se navegará automáticamente tras la solicitud del libro. El usuario destinatario del intercambio recibirá también un correo electrónico de aviso. No obstante, si el libro ya había sido solicitado anteriormente y el intercambio aún se encuentra pendiente, la aplicación informará al usuario y no permitirá realizar la solicitud.

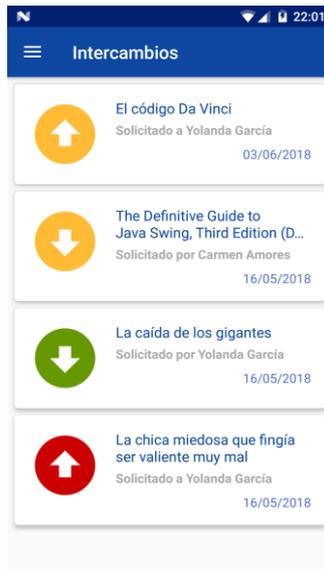


Figura 66: Listado de intercambios

Gestionar libros propios

Los libros registrados por el usuario están accesibles desde el menú principal de la aplicación, a través de “*Mis libros*”. Al pulsar sobre dicha opción de menú se mostrará el listado con los libros que el usuario comparte con el resto de la comunidad. La información mostrada para cada libro es la siguiente:

- Ilustración.
- Título.
- Autor.
- Fecha en la que el usuario lo registró en la aplicación.

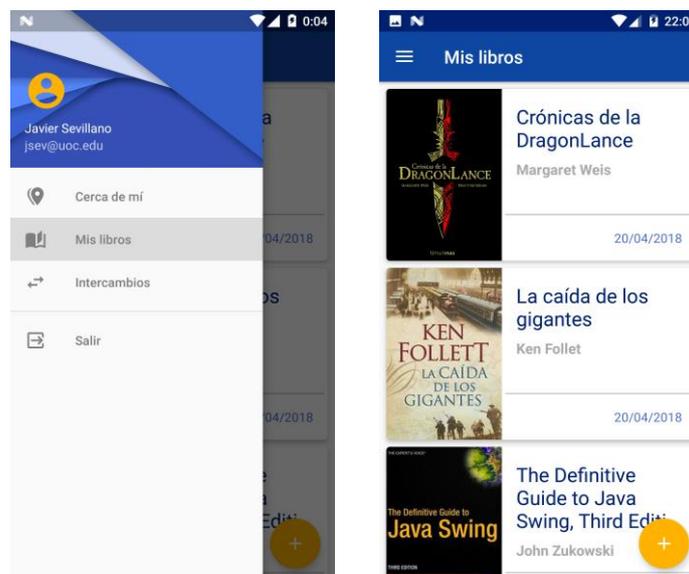


Figura 67: Menú principal y listado de libros

Si el usuario hace el gesto de deslizar hacia abajo sobre el listado su contenido se refrescará, obteniendo de nuevo la información desde el servidor.

Desde el listado de libros el usuario tiene disponibles dos opciones:

- Ver el detalle de un libro.
- Añadir un libro nuevo.

Para ver el **detalle de un libro** basta con pulsar sobre él. Se mostrará una nueva pantalla en la que se podrá consultar información ampliada sobre el mismo:

- Ilustración.
- Título.
- Autor.
- Editorial.
- Género literario.
- Idioma.
- Código ISBN.
- Estado en que se encuentra (bueno, malo o regular).

Desde esta pantalla el usuario podrá también eliminar el libro, pulsando sobre el botón circular con el icono de la papelera. Tras confirmar que desea borrarlo la aplicación lo eliminará, y dejará de estar disponible para intercambiar con otros usuarios, siempre y cuando el libro no esté ya siendo usado en algún intercambio que se encuentre pendiente, en cuyo caso, la aplicación mostrará un mensaje e impedirá su eliminación.

Cuando se borra un libro la aplicación regresa automáticamente al listado de libros, en el que ya no aparecerá el elemento recién eliminado.



Figura 68: Detalle de libro del usuario

Para **añadir un nuevo libro**, desde el listado, basta con pulsar sobre el botón circular con el símbolo “+”. Se mostrará una nueva pantalla para que el usuario introduzca los datos del libro que desea registrar. La información necesaria para registrar un nuevo libro es la siguiente:

- Ilustración (opcional).
- Código ISBN (opcional).
- Título (obligatorio).
- Autor (obligatorio).
- Editorial (obligatorio).
- Género literario (obligatorio). Se selecciona de una lista desplegable.
- Idioma (obligatorio). Se selecciona de una lista desplegable.
- Estado en el que se encuentra (obligatorio). Se proporcionan tres opciones, de las que el usuario debe seleccionar una: bueno, regular o malo.

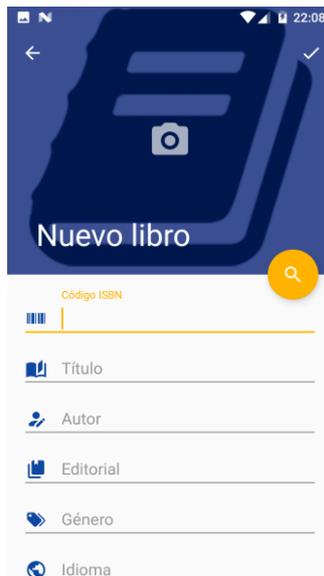


Figura 69: Nuevo libro

Para rellenar la información del libro el usuario dispone de tres alternativas:

- Introducir los datos manualmente uno a uno.
- Introducir el código ISBN del libro en el campo de texto correspondiente y pulsar sobre el botón circular de la lupa. La aplicación realizará una búsqueda del libro en Internet e intentará rellenar los campos título, autor y editorial, así como la ilustración.
- Pulsar directamente sobre el botón circular de la lupa, sin introducir ningún código ISBN. La aplicación pedirá al usuario que escanee el código de barras del libro, obtendrá su código ISBN y automáticamente realizará la búsqueda del mismo como en el caso anterior, rellenando los campos correspondientes. En caso de que aún no disponga de él, la aplicación solicitará permiso al usuario para acceder a la cámara del dispositivo. Si el usuario no da permiso para acceder a la cámara no se podrá escanear el código de barras.

Para asociar una ilustración al libro el usuario también dispone de varias alternativas:

- Realizar la búsqueda por ISBN comentada anteriormente. Si la aplicación encuentra los datos del libro intentará descargar su imagen, siempre y cuando esté disponible.
- Pulsar sobre el icono de la cámara. Se le presentará un diálogo para que seleccione el modo de obtener la ilustración del libro:
 - A través de la cámara, realizando una nueva fotografía. En este caso, la aplicación solicitará permiso para acceder a la cámara, en caso de que aún no disponga de él. Si el usuario no da permiso para acceder a ella no se podrá realizar la fotografía.
 - Seleccionando una imagen existente desde la galería de imágenes del dispositivo.

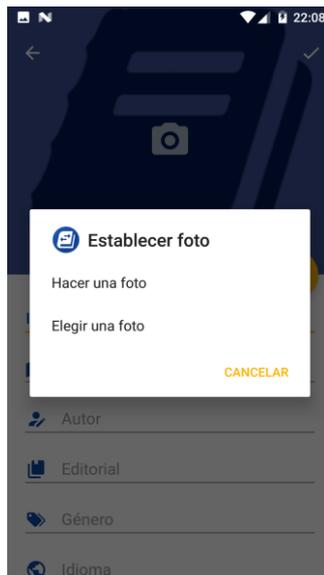


Figura 70: Diálogo de selección de fotografía del libro

Si el usuario decide no asociar ninguna imagen al libro la aplicación le asignará una por defecto.

Una vez completados los campos bastará con pulsar sobre la opción de menú “✓”, situada en la parte superior derecha de la pantalla, para dar de alta el nuevo libro. Si alguno de los campos obligatorios no ha sido rellenado se informará al usuario mediante el correspondiente mensaje. De la misma forma, si el usuario intenta añadir un libro con un código ISBN que ya ha registrado previamente la aplicación le informará mediante un mensaje y no le permitirá guardarlo.

Una vez registrado el libro la aplicación vuelve automáticamente al listado de libros del usuario, donde aparecerá el elemento recién insertado.

Visualizar intercambios

Para visualizar los intercambios en los que participa el usuario basta con seleccionar la opción “Intercambios” del menú principal. Los intercambios

mostrados son tanto los iniciados por el usuario como los que le han solicitado a él.

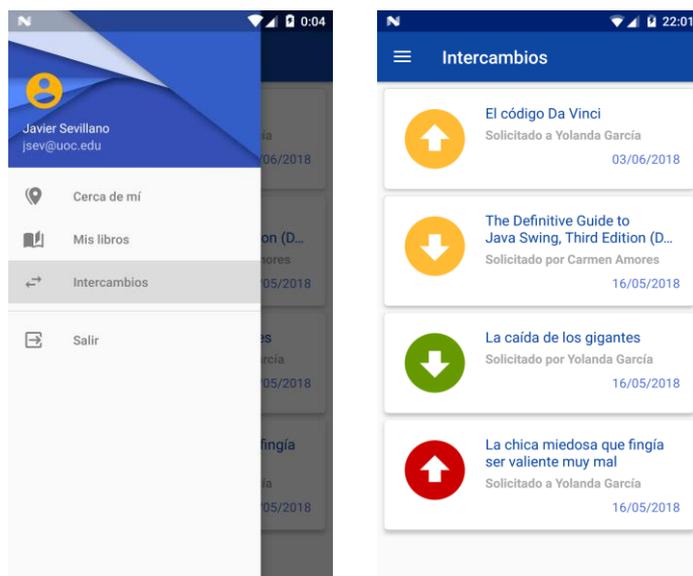


Figura 71: Menú principal y listado de intercambios

Los intercambios iniciados por el usuario aparecen representados con una flecha dirigida hacia arriba (saliente), mientras que los que han sido solicitados al usuario se muestran con una flecha dirigida hacia abajo (entrante).

Adicionalmente, por cada intercambio aparece representada la siguiente información:

- Estado del intercambio, mediante un código de colores:
 - Rojo: el intercambio ha sido **rechazado** por alguno de los participantes.
 - Verde: el intercambio ha sido **aceptado** por ambos participantes.
 - Naranja: alguno de los participantes aún no ha aceptado ni rechazado el intercambio, por lo que se encuentra **pendiente**.
- Título del libro solicitado al otro usuario (si es un intercambio saliente) o por el otro usuario (si es un intercambio entrante).
- Nombre del usuario al que se solicita el libro (si es un intercambio saliente) o que solicita el libro (si es un intercambio entrante).
- Fecha de registro del intercambio.

Si el usuario hace el gesto de deslizar hacia abajo sobre el listado su contenido se refrescará, obteniendo de nuevo la información desde el servidor.

Para ver la información ampliada de un intercambio basta con pulsar sobre el elemento del listado que se desea consultar. Se mostrarán los libros que participan en el intercambio, así como el estado actual del mismo.



Figura 72: Detalle de intercambio saliente pendiente

Aceptar un intercambio entrante

Para aceptar un intercambio entrante éste debe estar aún pendiente. Para ello basta con acceder al listado de intercambios y pulsar sobre el intercambio que se desea aceptar. Se mostrará la pantalla de detalle del mismo, indicando al usuario que aún no ha solicitado ningún libro al otro participante.



Figura 73: Detalle de intercambio entrante pendiente

Para solicitar un libro se debe pulsar sobre el botón circular con el símbolo "+". Esto llevará al listado de libros compartidos por el otro usuario, donde se podrá seleccionar el libro que se le desea solicitar. Para ello basta con pulsar sobre él en el listado, se abrirá la pantalla de detalle del libro, y en ella, sobre el botón circular con las flechas. La aplicación pedirá confirmación, y en caso de responder afirmativamente, el intercambio será aceptado. No obstante, si el

libro ya había sido solicitado anteriormente en otro intercambio aún pendiente, la aplicación informará al usuario y no permitirá realizar la solicitud.

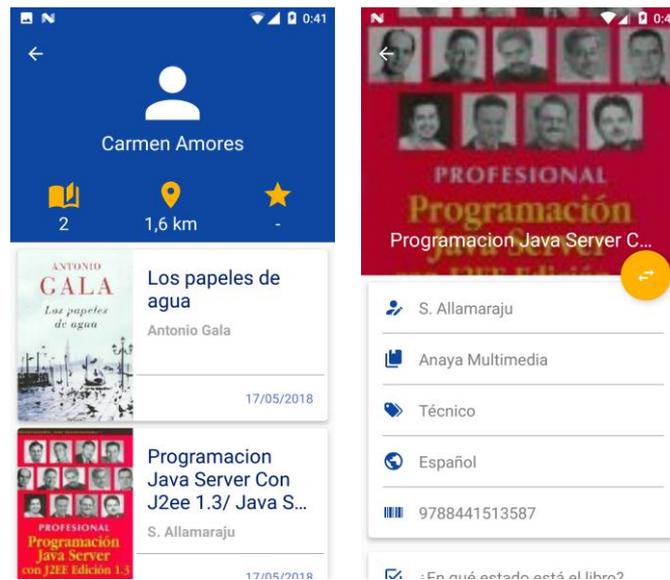


Figura 74: Selección de libro para aceptar intercambio entrante

Una vez el intercambio ha sido aceptado la aplicación vuelve automáticamente a la pantalla de detalle de intercambio, mostrando los datos actualizados e indicando que el intercambio ha sido aceptado.



Figura 75: Detalle de intercambio aceptado

Al completarse un intercambio ambos usuarios reciben un correo electrónico que les confirma que éste ha sido aceptado.

Rechazar un intercambio entrante

El usuario solo podrá rechazar los intercambios entrantes que se encuentren aún pendientes. Para ello basta con acceder al listado de intercambios y pulsar

sobre el intercambio que se desea rechazar. Se mostrará la pantalla de detalle del mismo, indicando al usuario que aún no ha solicitado ningún libro al otro participante. Para rechazar el intercambio basta con pulsar sobre la opción de menú situada en la parte superior derecha, “Ø”. La aplicación pedirá confirmación al usuario, y en caso de aceptar, el intercambio quedará rechazado. El otro usuario participante recibirá un correo indicándole que el intercambio ha sido rechazado.



Figura 76: Detalle de intercambio entrante rechazado

Valorar un intercambio

El usuario solo podrá valorar los intercambios aceptados, entrantes o salientes, que no haya valorado ya anteriormente. Para ello basta con acceder al listado de intercambios y pulsar sobre el intercambio que se desea valorar. Se mostrará la pantalla de detalle del mismo. Para valorar el intercambio basta con pulsar sobre la opción de menú situada en la parte superior derecha, “★”. La aplicación mostrará un diálogo para que el usuario puntúe el intercambio en una escala del 1 al 5. La puntuación seleccionada quedará reflejada en la información del intercambio, y el otro usuario participante recibirá un correo electrónico indicando que ha recibido una valoración.

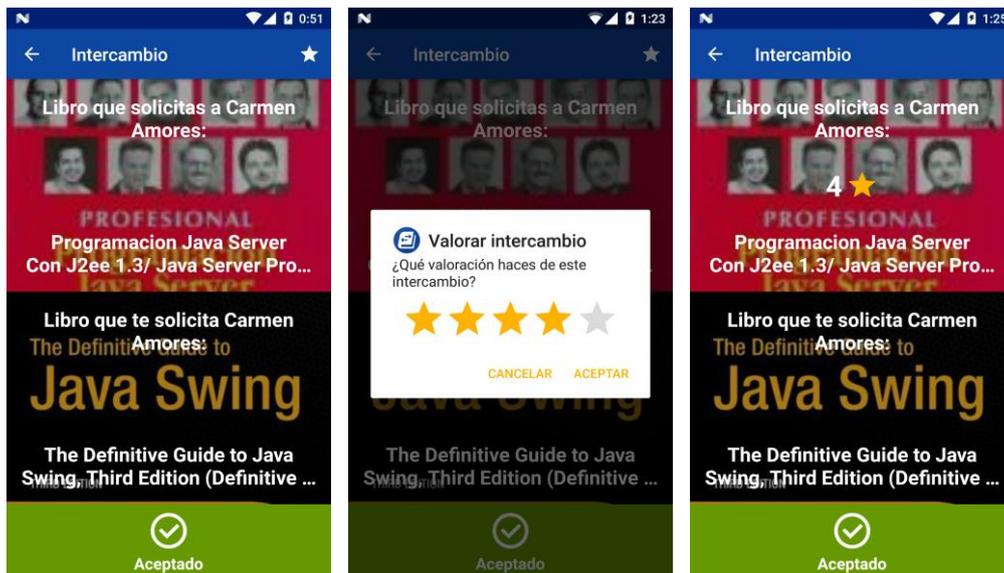


Figura 77: Valoración de intercambio

Cuando el otro participante valora el intercambio su puntuación también se ve reflejada en la pantalla de detalle, e igualmente el usuario recibe un correo electrónico avisándole de que ha recibido la valoración.



Figura 78: Detalle de intercambio con las dos valoraciones

Las valoraciones realizadas en el intercambio son tenidas en cuenta para calcular la puntuación media de los usuarios participantes.

Cierre de sesión

Mediante el cierre de sesión el usuario se desconecta de la aplicación. La desconexión está disponible desde el menú principal de navegación, en la opción "Salir". Al pulsar sobre ella se pide confirmación al usuario. En caso de responder afirmativamente se cierra la sesión y se le devuelve a la pantalla de

autenticación, donde deberá proporcionar sus credenciales para acceder de nuevo.

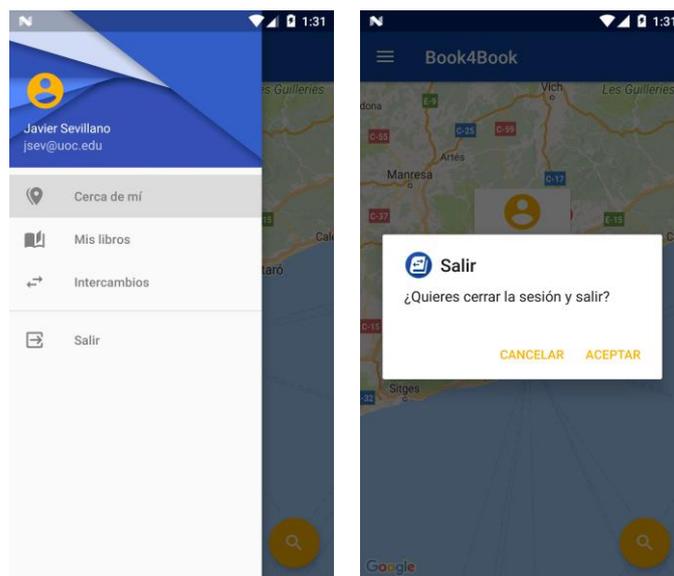


Figura 79: Menú principal y cierre de sesión