UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

APLICACIÓN PARA LA GESTIÓN DE BIBLIOTECA EN ANDROID, IOS Y WINDOWS PHONE

Fernando del olmo martín
Tutor: Jose Antonio Dongil

Febrero de 2017

APLICACIÓN PARA LA GESTIÓN DE BIBLIOTECA EN ANDROID, IOS Y WINDOWS PHONE

Autor: Fernando del olmo martín
Tutor: Jose Antonio Dongil

Escuela Politécnica Superior
Universidad Autónoma de Madrid

Febrero de 2017

Fernando del olmo martín
Aplicación para la gestión de biblioteca en Android, iOS y Windows Phone

Fernando del olmo martín
Avenida de la Dehesa 2 Portal 1 4ºB, Madrid

IMPRESO EN ESPAÑA – PRINTED IN SPAIN

“El esfuerzo de utilizar las máquina para emular el pensamiento humano siempre me ha parecido bastante estúpido. Preferiría usarlas para emular algo mejor”

– Edsger Dijkstra

Agradecimientos

Quisiera agradecer a mi familia, sobre todo a mis padres, el que hoy este aquí. Ellos me han dado siempre el apoyo moral y económico que he necesitado para dedicarme por completo a la carrera y no tener otras preocupaciones que no fuesen propias de un estudiante. A ellos en gran mayoría se lo debo todo, siempre me han ayudado en todo lo posible y siempre han estado para lo que fuera necesario.

Resumen

A lo largo de estos últimos años hemos ido viviendo un gran cambio tecnológico, pero el cambio más significativo ha sido el smartphone o teléfono inteligente. Este nuevo dispositivo móvil nos permite conectarnos a internet y a todos aquellos servicios que queramos. Actualmente, el número de dispositivos móviles a nivel global alcanza ya los 7,9 mil millones, más que personas hay en nuestro planeta.

Una vez que el smartphone se puso en auge, las compañías e incluso los desarrolladores no perdieron el tiempo y empezaron a desarrollar aplicaciones móviles o a desarrollar páginas web para ofrecer sus servicios más cómodamente en los smartphones. En la actualidad existen todo tipo de aplicaciones; desde mensajería instantánea o videollamadas hasta aplicaciones que te traen pedidos de comida a casa. Esto es una gran revolución puesto que permite a la gente facilitarle algunas tareas, y en cuanto a páginas web hay todo tipo de servicios ofrecidos a través de estas.

Pero aunque existan muchas aplicaciones o páginas web, no tienen porqué ser todas útiles para los usuarios. Puede haber alguna aplicación que intente ayudar a los usuarios a gestionar una lista de tareas, por ejemplo, pero que sea tan compleja y difícil de usar que los usuarios dejen de usarla.

Este segundo caso es el que sucede con algunas páginas web para las bibliotecas: son poco intuitivas o sencillas para que la experiencia de usuario al usarla sea buena y no acabé dejándola de usar. En relación a esto, ***The Book Manager*** es una herramienta diseñada con tecnologías web para dispositivos móviles. Permite que los usuarios puedan acceder a su biblioteca de una forma más sencilla que a través de la propia página web. Esta herramienta es intuitiva y fácil de usar: no genera dudas en los usuarios.

El sistema se ha implementado mediante tecnologías web para dispositivos móviles por ser uno de los medios en alza en estos momentos. Entre los motivos más importantes por los que se ha elegido esta tecnología destacan los bajos costes y la facilidad de ser implantada. ***The Book Manager*** es un sistema para agilizar la gestión de las bibliotecas. Todas las funcionalidades que implementa siguen el protocolo estándar RESTful. Por ello la implantación de esta herramienta conlleva unos costes casi nulos, lo que facilita su integración en el mercado.

Palabras clave

Aplicación móvil, Smartphone, Biblioteca, Gestión.

Abstract

Over the past few years, we have been experiencing a great technological change, but the most significant change has been the smartphone. This new mobile device allows us to connect to the Internet and all those services we want. Currently, the number of mobile devices reaches 7.9 billion worldwide, more than people on our planet.

Once the use of smartphones became popular, companies and even developers did not waste time and started developing mobile apps or developing web pages to offer their services on smartphones in a more comfortable way. Nowadays, there are all kinds of applications; from instant messaging or video calls to applications that bring you food orders at home. This is a great revolution since it allows people to make some tasks easily, and as for web pages there are all kinds of services offered through these.

Although there are many applications or websites, not all of them need to be useful for users. There may be some applications that try to help users to manage a list of tasks, for example, but that are so complex and difficult to use that users will stop using them.

This second case is what happens with some web pages for libraries: they are not intuitive enough or easy to use so that the user experience when using it is good and do not end up with the user leaving it unused. In relation to this, ***The Book Manager*** is a tool designed with web technologies for mobile devices. It allows users to access their library in an easier way than through the website itself. This tool is intuitive and easy to use: it does not generate doubts in the users.

The system has been implemented through web technologies for mobile devices because it is one of the rising technologies at the moment. Among the most important reasons for choosing this technology are low costs and ease of implementation. ***The Book Manager*** is a system to speed up the management of libraries. All the functionalities it implements follow the RESTful standard protocol. For this reason, the implementation of this tool entails almost zero costs, which facilitates its integration in the market.

Keywords

Mobile aplication, Smartphone, Library, Management.

ÍNDICE

Listas

Lista de códigos

Lista de figuras

Lista de tablas

1

Introducción

1.1. Marco del proyecto

La situación actual de las bibliotecas ha ido evolucionando a lo largo de los años. Inicialmente se centraba en una atención personalizada mediante los empleados, pero esto era hace muchos años. Actualmente, en la era digital en la que nos encontramos, estos servicios han evolucionado hasta un punto que antes era impensable.

Ahora mismo, gracias a los avances tecnológicos que se han ido dando, es posible acceder a la biblioteca a través de internet y realizar gestiones. Además, en esta época también hay muchas personas que disponen de un smartphone con el que pueden acceder a internet o ejecutar aquellas aplicaciones instaladas en este, gracias a eso, actualmente podemos leer libros en nuestros dispositivos móviles.

Todo esto supone un gran avance pero siempre y cuando la página web de la biblioteca sea intuitiva y clara de cara al usuario, o en el caso de las aplicaciones, que tengan una interfaz de usuario sencilla. ***The Book Manager*** se centra en solventar los problemas en aquellas bibliotecas donde la página web o aplicación es poco usable e intuitiva, ofreciendo como solución una aplicación sencilla, usable y compatible para todos los usuarios tengan el dispositivo que tengan.

1.2. Motivación

El trabajo de fin de grado es un proyecto de tal magnitud que decantarse por una rama de la informática no es tarea fácil. Además, como la realización de este proyecto es individual, eres tú mismo el que te pones las pautas de trabajo a seguir cada día. Es una oportunidad de demostrar los conocimientos adquiridos a lo largo del grado y de ponerlos en práctica. La rama elegida para desarrollar el proyecto fue el desarrollo web llevado a los dispositivos móviles para realizar aplicaciones híbridas.

Una vez elegida la rama sobre la cual íbamos a desarrollar el proyecto, faltaba decidir el “qué” queríamos desarrollar. Si la elección de la rama era complicada, esta lo era aún más, puesto que esta decisión, fuera la que fuera, tendría que mantenerla hasta el final, por lo que tenía que elegir una aplicación de la cual no me arrepintiese en un futuro de haberla elegido. En mi caso, tras darle muchas vueltas a la idea, decidí desarrollar una aplicación para gestionar los servicios que ofrecen las bibliotecas mediante una aplicación sencilla y usable.

Esta decisión la tome puesto que hay algunas bibliotecas, que aunque hayan evolucionado tecnológicamente, su página web no es nada intuitiva o accesible, además, los costes que supondrían implantar esta aplicación serían bastante bajos puesto que no requiere de una gran infraestructura para funcionar a diferencia de otras. A lo largo de este documento plasmaré las decisiones tomadas en cuanto a la arquitectura y tecnologías a usar y el desarrollo que he llevado a cabo para realizar dicha aplicación.

1.3. Objetivos

Los objetivos principales, por los cuales he llevado a cabo el desarrollo de este proyecto, han sido:

- Aprender a trabajar en cualquier parte de un proyecto moderno, tanto Frontend como Backend, poniendo en práctica un Stack moderno y vanguardista como es el full stack, comúnmente conocido como stack MEAN ((M)ongo + (E)xpress + (A)ngular + (N)ode).
- Poner en práctica los conocimientos, adquiridos de manera autodidacta, sobre desarrollo web mediante los frameworks AngularJS e Ionic.
- Aprender a desarrollar una API (Application Programming Interface) REST (Representational State Transfer) mediante tecnologías modernas, tales como Express y NodeJS, ya sea para aplicarlas en aplicaciones móviles o páginas web.
- Aprender a trabajar con bases de datos NoSQL para poder almacenar y manejar los datos de una forma totalmente distinta y de una manera mucho más ágil.
- Aprender a realizar un despliegue de una aplicación en un entorno real como lo son los Amazon Web Services y realizar peticiones desde un dispositivo móvil.
- Aprender y aplicar buenas prácticas de programación.

1.4. Estructura del documento

Este documento, donde vamos a redactar todo el proceso de desarrollo, se va a dividir en 9 capítulos incluyendo este. En este primer capítulo se hace una pequeña introducción al proyecto y se redactan los motivos por los cuales se ha decidido desarrollarlo.

En el capítulo 2 se hace una evaluación del estado del arte, se detalla la situación actual de las aplicaciones o páginas web relacionadas con las bibliotecas y servicios relacionados con los libros.

En el capítulo 3 se detalla un listado de todas las funcionalidades que se han detectado como principales y también otro listado con los objetivos que se esperan conseguir.

En el capítulo 4 se explica en detalle la metodología que se ha utilizado en el desarrollo, así como todas las herramientas, plataformas o lenguajes de programación, que han formado parte del mismo.

En el capítulo 5 se describe con detalle el análisis realizado, aquí se incluye el catálogo completo de requisitos, tanto funcionales como no funcionales y los casos de uso para los usuarios de la aplicación.

En el capítulo 6 se desarrollan el diseño y la implementación del proyecto. Para ello se detallan la arquitectura utilizada y sus componentes, el diseño que se ha seguido para la base de datos, la navegación entre pantallas, los diseños conceptuales de las maquetas, etc.

En el capítulo 7 se hace un análisis de las pruebas que se han realizado sobre la aplicación y se detallan los resultados y conclusiones obtenidas de las pruebas.

En el capítulo 8 se evalúa el resultado final por parte del equipo de desarrollo y los usuarios escogidos y se identificarán los beneficios que aporta la aplicación.

En el capítulo 9, como punto final, se detallan las conclusiones obtenidas del proyecto así como un breve planteamiento sobre el trabajo futuro.

2

Estado del arte

2.1. Servicios de lectura

Cuando se habla de servicios de lectura nos queremos referir a los servicios que ofrecen las bibliotecas, pero actualmente, no todos los servicios los ofrecen las bibliotecas. En esta época hay aplicaciones que ofrecen servicios para leer libros desde el propio móvil o dispositivos creados para la lectura. Por lo que podemos distinguir dos tipos, aquellos ofrecidos por las bibliotecas y los destinados a aplicaciones móviles o dispositivos para lectura.

2.1.1. Las bibliotecas

Las bibliotecas llevan años ofreciendo sus servicios a aquellos usuarios que no quieran comprar libros y que con tenerlos unos días es más que suficiente para leerlos. Antiguamente, las bibliotecas no se valían de las tecnologías para realizar las gestiones de sus usuarios, pero actualmente, incorporan páginas web para que los usuarios puedan acceder y gestionar los servicios.

2.1.2. Los dispositivos

Por otro lado, en cuanto a los dispositivos tenemos dos casos, o bien dispositivos fabricados específicamente para leer libros en ellos o bien los dispositivos móviles. Estos si que incorporan un mayor número de tecnologías en comparación con las anteriores, además agilizan las tareas de los usuarios.

The Book Manager está enfocado en solventar algunas carencias del primer caso ofreciendo los beneficios de este segundo.

2.2. Aplicaciones de lectura

Como ya hemos coentado previamente, existen una amplia variedad de aplicaciones para los dispositivos móviles, algunas de ellas con algunos objetivos similares a los nuestros. A continuación se van a citar los más relevantes.

Google Play Books

Esta aplicación creada por Google, pone a tu disposición una amplia gama sin límites de libros para leer. Puedes configurarlo a tu gusto, ya sea cambiando el tamaño de la letra por ejemplo. Además no hace falta conexión para poder acceder los libros y te permite leer los mismos libros ya sea en móvil, PC o a través de una tablet pudiendo retomar la lectura por donde la dejaste.

Kindle

Esta aplicación de la sección mobile de Amazon brinda el acceso a más de 1.500.000 libros de la tienda Kindle. También te permite personalizar tu lectura cambiando el tamaño de la fuente, cambiando el fondo incluso pudiendo leer en modo apaisado. Por otro lado también puedes hacer búsquedas dentro de un libro por si quieres buscar un personaje en concreto por ejemplo.

Nubico

Esta aplicación creada por Digicirtel S.L. nos permite leer miles de eBooks o libros electrónicos, e incluso nos permite leer revistas en la nube. En esta aplicación también se nos permite realizar búsquedas dentro de los libros, reordenar la biblioteca y leer eBooks o revistas en hasta 5 dispositivos sincronizados en una misma cuenta.

2.3. Observaciones

Las aplicaciones que hemos citado anteriormente, son sólo unas pocas dentro de la inmensa cantidad de aplicaciones existentes. Tras una larga investigación sobre estas y otras aplicaciones relacionadas con los libros y las bibliotecas, se ha advertido que las aplicaciones de cara al servicio para las bibliotecas no hay apenas.

En estos casos lo que hay son páginas web para acceder y algunas de ellas son poco usables y pueden dar una mala experiencia al usuario. Debido a que esta herramienta no supone ningún coste de implantación en la vida real, los usuarios utilizarían sus propios smartphones.

3

Objetivos y funcionalidades

3.1. Introducción

Como se ha especificado en el capítulo 1, el propósito de este Trabajo de Fin de Grado es definir, implementar y presentar un sistema software de gestión para los usuarios de las diferentes bibliotecas. Para cumplir estos objetivos se detallan las funcionalidades que dan forma a ***The Book Manager***.

3.2. Objetivos específicos y funcionalidades

3.2.1. Funcionalidades

F-1.– Realizar una búsqueda. En el catálogo el usuario podrá realizar una búsqueda para filtar los libros ya sea por autor, título o género.

F-2.– Realizar un préstamo. En el catálogo el usuario podrá ver los libros disponibles actualmente y podrá tomar prestado el que guste.

F-3.– Realizar una reserva. En el catálogo el usuario podrá ver los libros disponibles actualmente y podrá tomar prestado el que guste, en el caso de que no este disponible podrá reservar el libro para cuando esté disponible.

F-4.– Renovar un préstamo. El usuario tendrá un listado de sus préstamos actuales y podrá renovar aquel préstamo que quiera prolongar.

F-5.– Devolver un libro. El usuario podrá indicar que va a devolver un libro para que así pueda volver a estar disponible para los demás usuarios.

F-6.– Cancelar una reserva. El usuario podrá cancelar una reserva cuando ya no esté interesado en tomar prestado ese libro.

F-7.– Seleccionar idioma. El usuario podrá configurar el idioma de la aplicación a su gusto de forma persistente.

3.2.2. Objetivos

Las funcionalidades citadas anteriormente tienen como propósito cumplir los objetivos que se detallan en este apartado. Los objetivos de la aplicación son muchos pero pueden agruparse en los objetivos genéricos que se citan a continuación:

O-1.– Interfaz intuitiva. Esto se consigue haciendo que las pantallas de la aplicación tengan las menores acciones posibles, y aquellas acciones que tenga, que sean descriptivas de tal forma que el usuario solo viendo la pantalla sepa que hace esa acción.

O-2.– Ayudar a las bibliotecas. Esto se consigue desarrollando una aplicación usable para los usuarios, de esta forma las bibliotecas en las cuales la experienca de uso de su página web, por ejemplo, sea mala se podrá beneficiar de la aplicación.

4

Definición del proyecto

4.1. Metodología

En este apartado se definirán con detalle, y en orden de ejecución, cada una de las etapas que se han seguido durante el desarrollo del proyecto. El proceso requiere una metodología con 5 etapas [1] y han sido las correspondientes al cilo de vida en cascada.

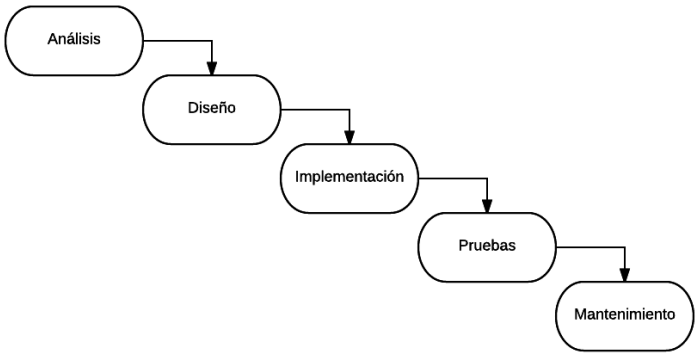


Figura 4.1: Ciclo de vida en cascada.

1.– **Análisis de requerimientos.** En esta primera fase, la tarea principal es extraer y analizar los requisitos del producto software. Es decir, analizar las necesidades de los usuarios para determinar que objetivos debe cubrir, de esta tarea surge el listado de requisitos funcionales y no funcionales que serán la base del proyecto.

2.– **Diseño y arquitectura.** En esta etapa se realizará el diseño de todos y cada uno de los componentes que conforman el proyecto, tanto la base de datos, la arquitectura y todo aquello que sea necesario. Esta etapa es muy importante puesto que es aquí donde se realizan todas las decisiones sobre las cuales se desarrollará el proyecto.

3.– **Implementación.** Esta etapa es en la que se desarrolla el proyecto basándose en las especificaciones y diseños realizados previamente. Además, es cuando realmente se aplican los diseños y se ve si son correctos o hay algo incompleto o ambiguo, en tal caso, se retrocede a las etapas previas para realizar los cambios necesarios. También se realizan algunas pruebas respecto a las implementaciones para ver que todo vaya correcto y terminar con la implementación.

4.– **Pruebas.** En esta etapa se comprueba que el software desarrollado cumpla los requisitos

especificados. Aquí se han realizado las pruebas que faltaban por hacer, ya que en la fase previa ya se realizaron algunas. Se han realizado pruebas tanto unitarias como de integración y pruebas de compatibilidad y funcionalidad.

5.– **Documentación.** En las etapas previas ya se ha ido realizando la labor de ir documentando las tareas realizadas, pero ha sido en esta fase donde se ha unificado cada documentación hecha en cada fase para dar lugar a la presente documentación. Este documento recoge además las líneas de trabajo futuro y conclusiones del proyecto.

6.– **Mantenimiento.** Esta es la última fase del proyecto y la más larga de todo el proceso. Durante esta fase se mejora y/o actualiza lo que se considere necesario de la aplicación. Se pueden realizar dos mantenimientos, correctivo o evolutivo.

Pero, como ya se ha mencionado, se ha decidido aplicar una variante que permite volver atrás en las fases para realizar modificaciones en el caso de que fuesen necesarias, así que, esta variante se llama desarrollo en cascada con realimentación.

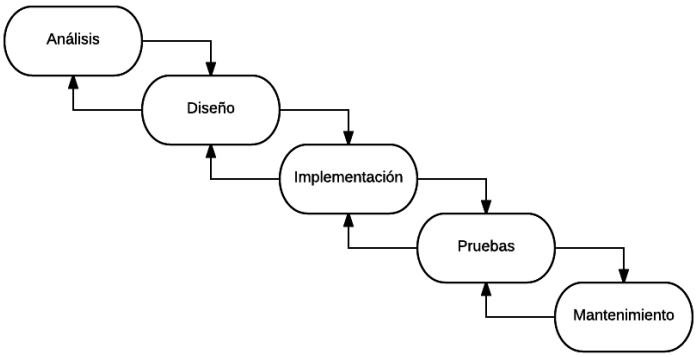


Figura 4.2: Ciclo de vida en cascada con realimentación.

4.2. Planificación orientativa

La planificación que se aporta a continuación es meramente orientativa puesto que se realizó en la fase de planificación del proyecto y es imposible planificar el desarrollo de un proyecto en el que trabaja uno solo debido a que si surge cualquier imprevisto se para el desarrollo. Por este motivo, y aunque se ha hecho todo lo posible por seguirla, en ocasiones ha sufrido alguna desviación.

4.2.1. Puntos de revisión

Puesto que el equipo de desarrollo estaba compuesto únicamente por un recurso humano, las revisiones que vale la pena mencionar son las que se han realizado en conjunto con el tutor del proyecto.

- **Semanas 1-2.** En este punto de revisión se comprueba que las decisiones tomadas respecto a las tecnologías a usar son las adecuadas para este tipo de proyecto. Tras esta revisión se cambió de idea respecto al stack que se iba a usar por uno más moderno y actual, acorde al tipo de proyecto con el cual iba a aprender a desarrollar tal y como se hace en un entorno de la vida real.

- **Semanas 20-22.** En este punto de revisión se comprueban los diseños como las definiciones de la base de datos, API e interfaz de usuario para poder pasar a la fase de implementación. Tras esta revisión fue necesario incluir modificaciones en la definición y la planificación del proyecto, sobre todo en la definición de la base de datos puesto que al ser NoSQL había que tener en cuenta diversos aspectos que no se tuvieron.
- **Semanas 28-29.** En esta revisión solo se comprueba la seguridad en cuanto a la forma de realizar las peticiones a la API, tras esta revisión fue necesario hacer un cambio en las peticiones para incluir seguridad en ellas mediante JWT (JSON Web Token).
- **Semanas 34-36.** En esta última revisión se comprueba que el funcionamiento es el correcto y el esperado en cada parte de la aplicación.

4.3. Herramientas utilizadas

4.3.1. Plataformas

Atom

Atom es un edior de texto moderno que se ha escogido en este proyecto para desarrollar la aplicación. La elección de esta herramienta frente a otras se debe a que está desarrollada por Github y además permite múltiples lenguajes.

Bitbucket

Es un sistema de alojamiento para repositorios que trabajan sobre Git, además te permite organizar los repositorios por proyectos.

Como este proyecto se ha llevado a cabo el desarrollo usando Git, hemos podido controlar y versionar todos los elementos del proyecto. Ha resultado útil en las revisiones el poder ver todas las subidas que se han ido haciendo porque junto con la descripción, nos permite de un primer vistazo saber que se está cambiando en la subida y en el caso de que un módulo falle se revisarían los cambios que se hicieron solo en ese módulo, ahorrándonos mucho tiempo de búsqueda.

4.3.2. Frameworks

AngularJS

Se ha escogido el framework AngularJS por delante de otras opciones tales como React o Ember por diversos motivos:

- Google: Este framework esta desarrollado por Google lo que conlleva un gran mantenimiento y comunidad.
- Ionic: Otro de los motivos que se eligió este framework es porque trabaja muy bien junto al otro framework usado, Ionic, del que hablaré a continuación.

Ionic

Este framework se ha escogido además de porque trabaja muy bien junto con AngularJS por otra serie de motivos:

- Estandár: Sigue los estándares en cuanto a las guías de estilo para las aplicaciones móviles nativas, además usa los SDKs nativos.
- Diseño: Es limpio, simple y funcional. Ionic ha sido diseñado para trabajar adecuadamente en todos los dispositivos y plataformas móviles actuales.
- Fácil aprendizaje: Es muy sencillo aprender a usarlo, solo necesitas saber HTML, CSS y Javascript.

4.3.3. Lenguajes de programación

Javascript

Es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

HTML

HTML (HyperText Markup Language) es el lenguaje de marcado básico de la World Wide Web para la elaboración de páginas web, actualmente en la quinta revisión.

CSS

CSS (Cascading Style Sheets) es el lenguaje utilizado para describir la presentación de documentos HTML o XML (eXtensible Markup Language), esto incluye varios lenguajes basados en XML como son XHTML o SVG.

SASS

Lenguaje de hoja de estilos inicialmente diseñado por Hampton Catlin y desarrollado por Nathan Weizenbaum. Una vez compilado los ficheros SASS (Syntactically Awesome Stylesheets), se generan los ficheros CSS correspondientes.

5

Análisis

5.1. Introducción

En este capítulo se presenta en detalle la fase de análisis del proyecto. Para ello se exponen los casos de uso que se han detectado como necesarios y un amplio registro de requisitos, tanto funcionales como no funcionales. Se podría decir que es un capítulo donde establecemos las bases de referencia respecto a lo que se espera que el proyecto ofrezca a los usuarios en temas de funcionalidad. Esta es una fase donde se realizan las decisiones que marcarán el transcurso del desarrollo del proyecto.

5.2. Casos de uso

Los casos de uso que se han detectado y definidos para el proyecto se presentan en el siguiente diagrama:

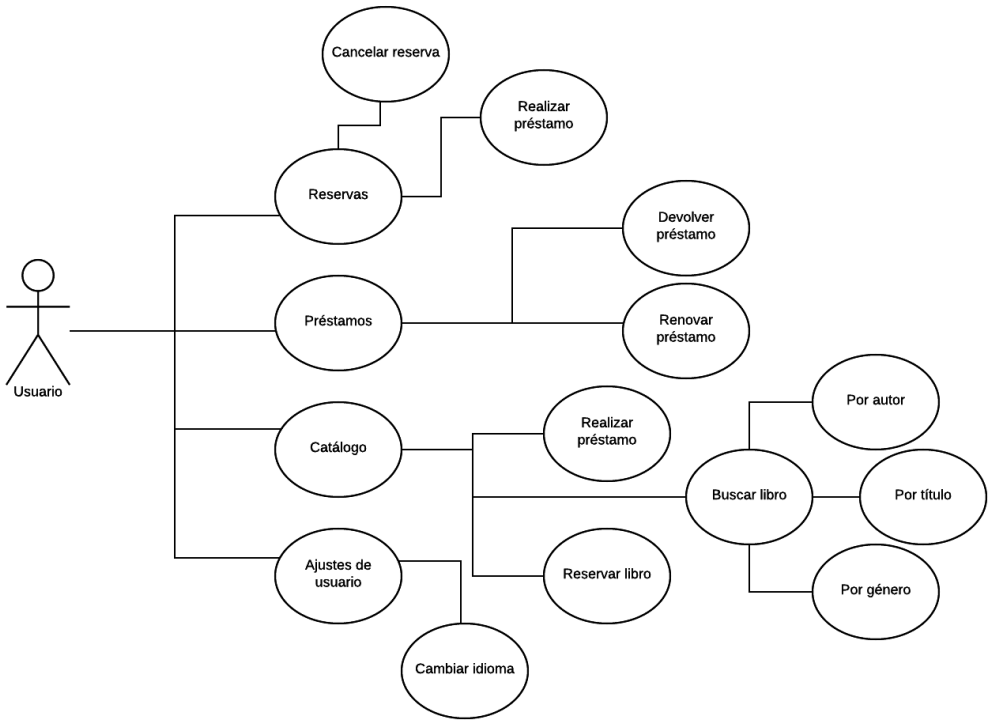


Figura 5.1: Diagrama de caso de uso para los usuarios

5.3. Catálogo y definición de requisitos

Los requisitos los hay de dos grupos, estos son:

Requisitos funcionales: son aquellos que definen una función del sistema de software, además de que son los que establecen el comportamiento del sistema.

Requisitos no funcionales: son aquellos que especifican criterios con los que juzgar la operación de un sistema, es decir, son los que describen las características de funcionamiento del sistema.

A continuación se define el catálogo de requisitos para detallar la funcionalidad de la aplicación y sus características.

5.3.1. Requisitos funcionales

RF-1.– Compatibilidad. La aplicación, al ser híbrida, deberá dar compatibilidad a los grandes sistemas operativos implantados en los móviles:

- **RF-1.1.– Android.** Para el caso de Android deberá ser compatible con las APIs y SDK que usen la mayor parte de los usuarios. Para concretar más sería a partir de la número 16.
- **RF-1.2.– IOS.** En el caso de iOS, la aplicación deberá dar compatibilidad a las últimas veriones de su sistema operativo. En este caso nos referimos a las versiones 8, 9 y 10.
- **RF-1.3.– Windos Phone.** Para Windows Phone, la aplicación deberá dar soporte a la versión más actual de su sistema operativo, con esto queremos decir que se dará soporte a la versión 8.

RF-2.– Conexión internet. La conexión a internet es fundamental para poder acceder a la base de datos alojada en la nube.

RF-3.– Registro. Para que el usuario pueda empezar a usar las ventajas de este proyecto deberá realizar un registro previo a través de la misma aplicación.

RF-4.– Login. Para que el usuario pueda acceder a los servicios de la aplicación deberá introducir el email y la contraseña con los que se registro previamente.

RF-5.– Reservas. El usuario podrá acceder a un listado donde se le muestren los libros que actualmente tiene reservados a la espera de que se encuentre disponible para realizar un préstamo.

- **RF-5.1.– Préstamo.** El usuario podrá realizar un préstamo de un libro que se encuentre disponible a través del listado de reservas, facilitando así que el usuario no tenga que buscar en el catálogo el libro de nuevo.
- **RF-5.2.– Cancelación.** El usuario podrá cancelar cualquier reserva que tenga hecha previamente.

RF-6.– Préstamos. El usuario podrá acceder a un listado donde se le muestren los libros que actualmente tiene en préstamo, además se le mostrará tiempo que le queda hasta que tenga que devolverlo. En el caso de que se haya vencido el préstamo también se le mostrará visiblemente.

- **RF-6.1.– Devolución** El usuario podrá a través de sus préstamos devolver uno en

concreto en cualquier momento.

RF-6.2.– Renovación. El usuario podrá renovar un préstamo siempre y cuando no le queden más de 15 dias para vencer o haya vencido el préstamo.

RF-7.– Catálogo. El usuario tendrá acceso a un listado de los libros de los que dispone la biblioteca para tomar prestados.

RF-7.1.– Búsqueda. El usuario dispondrá de una barra de búsqueda donde podrá filtrar los libros del catálogo.

- Autor. El usuario podrá filtrar el catálogo por el autor que ande buscando.
- Título. El usuario podrá filtrar por un título específico si busca un libro en concreto.
- Género. El usuario podrá filtrar por el género de los libros ya sea Juvenil, Fantástico, etc...

RF-7.2.– Reserva. El usuario podrá realizar una reserva de un libro cuando no queden unidades disponibles para realizar un préstamo.

RF-7.3.– Préstamo. El usuario podrá realizar un préstamo de cualquier libro que escoja siempre y cuando haya unidades disponibles.

RF-8.– Configuración. El usuario podrá configurar los datos con los que se ejecutará la aplicación:

- **Idioma.** El usuario podrá cambiar de idioma la aplicación para que le sea más cómodo utilizarla.

5.3.2. Requisitos no funcionales

RNF-1.– Seguridad. Las peticiones que se realizan a la API llevan un token en la cabecera para identificar el origen y asegurar que proviene de un usuario autenticado.

RNF-2.– Interfaz. Se ha seguido una interfaz sencilla e intuitiva donde las acciones sean las mínimas posibles para garantizar una buena experiencia de uso a los usuarios.

RNF-3.– Disponibilidad. Se ha asegurado que la base de datos tenga una alta disponibilidad alojandola en un servicio web que nos lo garantiza.

6

Diseño e implementación

6.1. Introducción

En este capítulo se presenta en detalle la fase de diseño e implementación del proyecto. Para ello se expone la arquitectura que se ha elegido y llevado a cabo tanto para la interfaz de usuario como para la base de datos o la API. El capítulo presenta tanto la fase de diseño como la de implementacion dado que al ser un unico desarrollador realizando el proyecto, ha sido bastante común el ir modificando el diseño a medida que el código lo requería.

Se podría decir que es un capítulo donde establecemos las decisiones tomadas en cuanto al cómo se desarrollara el proyecto y cómo se organizarán y comunicarán todas las piezas en él. Para empezar se va a explicar el diseño elegido e implantado para la arquitectura del proyecto debido a que es una forma de trabajo que, a fecha de proyecto, esta surgiendo con mucha fuerza.

6.2. Arquitectura de la aplicación

En cuanto a la arquitectura que se ha decidido llevar a cabo en el desarrollo del proyecto se ha elegido seguir una nueva corriente que está surgiendo en el desarrollo de proyectos en Javascript de principio a fin, el llamado ***stack MEAN*** [2]. Esta corriente se basa en realizar un desarrollo con Javascript en todas las partes que componen una aplicación web:

- **Frontend.** Ésta es la parte del proyecto que visualizará el usuario, podríamos decir que sería la interfaz de usuario.
- **Backend.** Ésta es la parte del proyecto que realiza la gestion de todos los datos y realiza las solicitudes a la base de datos a través de la API para enviar los datos al Frontend para que el usuario los visualize.
- **API.** Ésta parte es la encargada de gestionar todas las peticiones a la base de datos que se realicen y responderlas adecuadamente.

De esta manera, al usar un único lenguaje de programacion, se logra un desarrollo más ágil y permite a una persona manejarse en todos los ámbitos y no solo centrarse en desarrollo de frontend o backend. Todo esto favorece el desarrollo y la integración continua.

Alojamiento en la nube

Por otra parte, en cuanto al backend se decidió optar por alojarlo en los servicios de AWS para que los usuarios pudieran acceder a sus datos desde cualquier dispositivo unicamente introduciendo sus datos de login. Esta decisión se debio meramente a que AWS, al ofrecerme una máquina con linux en la nube, me daba mayor libertad para desplegar mi aplicación, además de que el primer año era totalmente gratis.

Para llevar a cabo toda esta configuración para el alojamiento en la nube [3], primero me registre en Amazon Web Services. Una vez registrado como usuario, lancé una instancia Elastic Compute Cloud (EC2) que básicamente consiste en una máquina virtual Linux. Además, Amazon nos proviene de una clave privada generada en un fichero para poder acceder a nuestra máquina virtual.

Una vez que tenía la instancia corriendo y la clave para conectarme a ella, genere un fichero de configuración para tener configurada una conexión ssh entre mi entorno local y la máquina virtual.Por último, lo que hice fue instalar todo lo necesario para ejecutar el proyecto dentro de la máquina virtual y descargué el proyecto del repositorio. Ahora ya estaba todo listo para poder realizar peticiones desde el móvil a la máquina virtual donde se encuentra la base de datos.

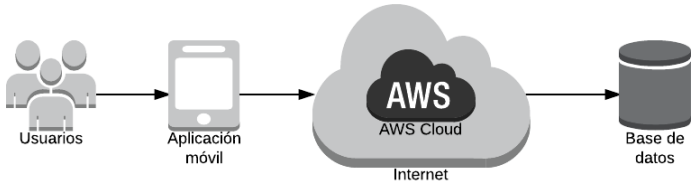


Figura 6.1: Arquitectura de la aplicación en la nube

6.3. Diseño de la base de datos

A la hora de diseñar e implementar la base de datos, dado que decidí seguir el stack MEAN, usé MongoDB [4] como base de datos NoSQL. Estas bases de datos nos presentan otra manera totalmente distinta de manejar y almacenar los datos. NoSQL es una amplia clase de gestión de bases de datos que difieren del modelo tradicional, RDBMS, en que no usan SQL como lenguaje para realizar consultas.

Las bases de datos NoSQL, no usan estructuras fijas lo que hace que no haya garantía completamente ACID (atomicidad, consistencia, aislamiento y durabilidad). Pero en cambio, las bases de datos NoSQL están altamente optimizadas para las operaciones recuperar y agregar, y normalmente no ofrecen mucho más que la funcionalidad de almacenar los registros. La pérdida de flexibilidad en tiempo de ejecución, comparado con los sistemas SQL clásicos, se ve compensada por ganancias significativas en escalabilidad y rendimiento cuando se trata con ciertos modelos de datos.

Teniendo en cuenta que al trabajar con una base de datos NoSQL no trataría con las típicas tablas, y que además, sería manejada por una API REST, distinguí 4 documentos o esquemas principales para controlar mediante REST.

El primero de todos es el esquema de un libro, este esquema contiene el avatar que se muestra en la página del libro, el título y nombre de su autor, el ISBN del libro, la editorial, una breve sinopsis de la historia, el número de copias disponibles del libro y por último la categoría a la que pertenece. Este esquema es de los principales puesto que todo el proyecto gira en torno a las aciones para gestionar los libros.

Otro importante es el esquema de un usuario, este esquema contiene el nombre del usuario, su email y contraseña y el idioma que tiene configurado el usuario para la aplicación.

Por último, cabe mencionar los recursos Préstamos y Reservas. Estos se encargan de la "relación" entre los usuarios y los libros, ya sea mediante un préstamo realizado o una reserva realizada.

Otro punto a mencionar es que se ha hecho uso de una característica que nos ofrece MongoDB, que es la de asignar un id único a cada instancia de un recurso que hagamos, de esta forma, podremos diferenciar a que recurso estamos accediendo y que recurso estamos modificando en las llamadas a la API.

Por último, cabe mencionar que también hemos hecho uso de las diversas querys que MongoDB nos facilita, tales como buscar un único objeto por un id, obtener todos los objetos de un recurso y obtener diversos objetos con una característica en común. Estas querys las ejecutara la API en función de la llamada que hagan, de esta forma obtendremos persistencia en los datos.

A continuación se muestra un esquema de todos los documentos creados en la base de datos NoSQL.

Figura 6.2: Documentos de la base de datos NoSQL

6.4. Diseño de la API

Para diseñar la API que controla las solicitudes y peticiones a nuestra base de datos y dado que se esta siguiendo el stack MEAN, he utilizado Express [5] junto con NodeJS [6] para diseñar y definir los diferentes endpoints de nuestra API a los cuales llamo desde la aplicación. Gracias a Express he podido definir los endpoints de una manera muy sencilla y en pocos pasos la tuve funcionando. Este framework es robusto, rápido, flexible y muy simple. Entre otras características, ofrece Router de URL (Get, Post, Put...), facilidades para motores de plantillas (Jade, EJS, JinJS...), Middleware vía Connect y un buen test coverage.

Al final de esta sección muestro una serie de tablas con los endpoints de la API para los distintos recursos o documentos en la base de datos, el método con el que se debe realizar la petición y una breve descripción de lo que devuelve.

La sintaxis que se ha seguido para las rutas de la API es muy intuitiva gracias a la sencillez de REST y a que se han aplicado buenas prácticas a la hora de definir las rutas [7]. Podemos apreciar que por cada recurso

que tenemos en la base de datos, tenemos ciertas rutas con sus métodos.

Para una misma ruta sobre un recurso, podemos realizar varias acciones dependiendo de qué método de llamada hagamos a la API. Además de la sencillez que nos proporciona REST a la hora de la sintaxis de las rutas y el manejo de los recursos, cabe mencionar que todo lo que recibe y devuelve son objetos de tipo JSON (Javascript Object Notation), por lo que facilita mucho a la hora de enviar datos y recibirlos.

También se pueden realizar pruebas de manera local ejecutando llamadas por terminal a la API y viendo los resultados en el navegador. Por último, cabe mencionar, que se le ha dotado a la API de una capa de seguridad añadiendo en las cabeceras de las peticiones un JWT [8]. Esto se ha logrado gracias a que cuando un usuario se registra o inicia sesión, se genera un token en el backend y se envía de vuelta al usuario junto con sus datos de sesión. Una vez que el usuario recibe esos datos, en todas las peticiones que haga a la API se enviará ese token de forma que se confirmará que es una petición realizada por el usuario.

Ruta	Método	Descripción
/libros	GET	Devuelve todos los libros
	POST	Añade un libro al catálogo
/libros/:idLibro	GET	Devuelve los datos de un libro
	PUT	Actualiza los datos de un libro
	DELETE	Borra los datos de un libro
/libros/:idLibro/prestamos	GET	Devuelve los préstamos de un libro
	POST	Añade un préstamo a un libro
/libros/:idLibro/reservas	GET	Devuelve las reservas de un libro
	POST	Añade una reserva a un libro
/libros/autor/:autor	GET	Devuelve los libros filtrados por autor
/libros/título/:título	GET	Devuelve los libros filtrados por título
/libros/género/:género	GET	Devuelve los libros filtrados por género

Tabla 6.1: Rutas de la API para los libros

Ruta	Método	Descripción
/usuarios	GET	Devuelve un listado de los usuarios
	POST	Añade un nuevo usuario
/usuarios/:idUsuario	GET	Devuelve los datos de un usuario
	PUT	Actualiza los datos de un usuario
	DELETE	Borra los datos de un usuario
/usuarios/:idUsuario/prestamos	GET	Devuelve los préstamos de un usuario
	POST	Añade un préstamo a un usuario
/usuarios/:idUsuario/reservas	GET	Devuelve las reservas de un usuario
	POST	Añade una reserva a un usuario

Tabla 6.2: Rutas de la API para los usuarios

Ruta	Método	Descripción
/reservas	GET	Devuelve todas las reservas del sistema
	POST	Añade una reserva al sistema
/reservas/:idReserva	GET	Devuelve los datos de una reserva
	DELETE	Borra los datos de una reserva

Tabla 6.3: Rutas de la API para las reservas

Ruta	Método	Descripción
/préstamos	GET	Devuelve todos los préstamos del sistema
	POST	Añade un nuevo préstamo
/préstamos/:idPrestamo	GET	Devuelve los datos de un préstamo
	PUT	Actualiza los datos de un préstamo
	DELETE	Borra los datos de un préstamo

Tabla 6.4: Rutas de la API para los préstamos

6.5. Diseño de la interfaz de usuario

En cuanto a la interfaz de usuario, he decidido seguir con el stack MEAN y utilizar el frameworks AngularJS [9] junto con Ionic [10]. Angular es un framework de Javascript para la parte cliente o Frontend de una aplicación web, que respeta el paradigma MVC y permite crear SPA (Single page application) (Aplicaciones web que no necesitan recargar la página), de manera más o menos sencilla aunque impone ciertas directrices a seguir para cumplir con el patrón de diseño establecido. Este patrón de diseño se llama MVC (Model-View-Controller).

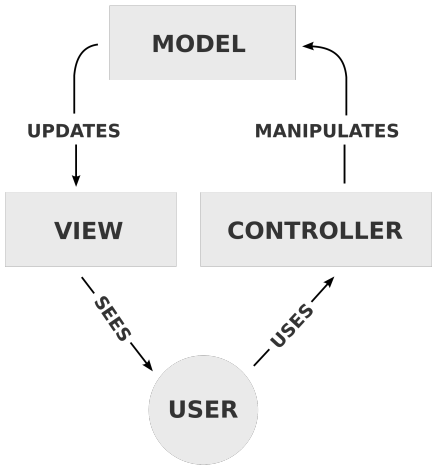


Figura 6.3: Patrón MVC

Este patrón de diseño tiene dos principales beneficios:

- Puedes cambiar el modelo sin necesidad de tener que cambiar la vista y viceversa.
- Los test unitarios son más fáciles de implementar y ejecutar.

Además, permite que el código se implemente de manera más fácil y en menos tiempo, esto junto con la facilidad de reusar el código de un controlador ahorrándonos tiempo son otros de los beneficios de este patrón de diseño. Una vez explicado cómo se implementara la interfaz de usuario, pasamos a definir los tres puntos clave, la vista, el modelo y, por último, el controlador.

Primero tenemos las vistas o pantallas que mostraremos a los usuarios, estas serán páginas implementadas en HTML haciendo uso de las herramientas facilitadas por los frameworks seleccionados, AngularJS e Ionic.

Luego estará el modelo, que será todo el flujo de datos que es gestionado por las factorías, ya sea este solicitado a la API o para actualizar datos enviándolos a la API. Éstas factorías nos sirven para realizar las llamadas HTTP de forma sencilla a la API REST mediante los métodos establecidos GET, POST, PUT y DELETE a la API.

Por ultimo están los controladores, estos se encargaran de pedir los datos a las factorías y de mostrarlos en las páginas según sea necesario.

El primer paso para definir la interfaz de usuario ha sido realizar unas maquetas a grandes rasgos de como seria cada página de la aplicación. Para la interfaz de usuario, se ha cuidado hasta el mínimo detalle ya que si queremos que la experiencia del usuario al navegar por la aplicación desde su dispositivo móvil sea buena necesitamos que la interfaz no tenga elementos descolocados o componentes que no funcionen como deberían.

Se ha adjuntado una sección en los anexos con las distintas maquetas de las pantallas de la aplicación, las cuales procederé a comentar a continuación, aunque como se verá adelante, se ha querido que la navegación sea muy intuitiva para facilitar su uso.

6.5.1. Diseño conceptual de las páginas

Login

Esta será la primera pantalla que mostraremos al usuario nada más iniciar la aplicación, aquí le pediremos o bien que inicie sesión con sus datos o que vaya a registrarse como nuevo usuario.

Registro

La pantalla de registro mostrara un sencillo formulario donde se le pedirán los datos requeridos para registrar un nuevo usuario en el sistema. Una vez rellenado el formulario podrá registrarse e ir al menú principal.

Menú principal

En el menú principal se le dará la bienvenida al usuario al sistema y se le explicara brevemente la aplicación.

Catálogo

En el catálogo el usuario dispondrá de todo un listado de los libros a su disposición. Además, el usuario podrá hacer un filtrado por autor, titulo o género.

Préstamos

En esta pantalla se mostrara un listado con todos los libros que tiene el usuario hasta ese momento prestados, además, debajo de cada libro le aparecerá cuanto le queda para que el préstamo llegue a su fin.

Reservas

En la pantalla de reservas se mostrara un listado con todos los libros que tiene el usuario hasta ese momento reservados, además, debajo de cada libro le aparecerá si ese libro se encuentra disponible o no para tomarlo prestado.

Ajustes

En los ajustes el usuario podrá ver su configuración y sus datos personales.

Libro

Esta será la pantalla que se le muestre al usuario con los datos de un libro cuando selecciones uno, ya sea para tomar prestado o reservar.

6.5.2. Esquema de navegación de las páginas

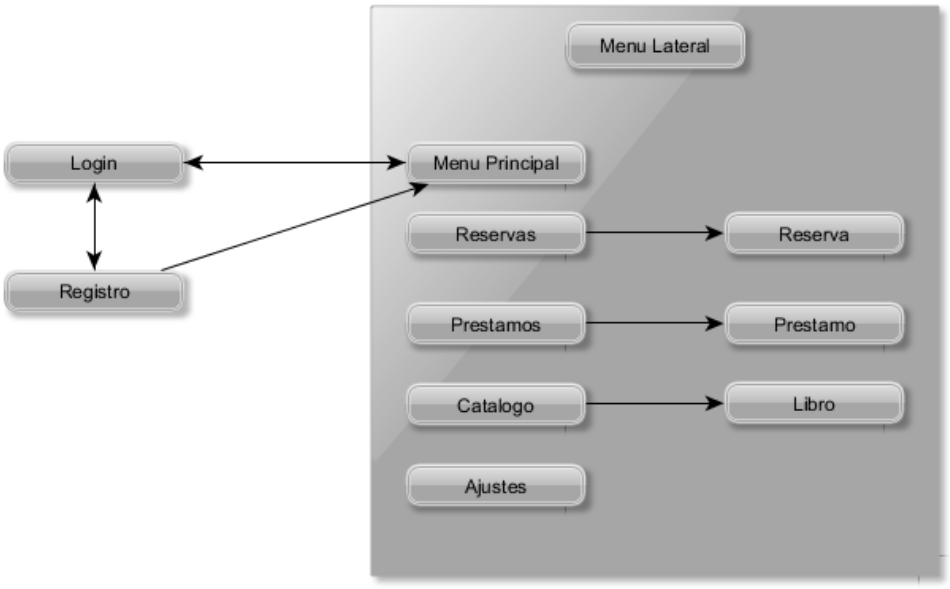


Figura 6.4: Diagrama de navegación

Tal y como se pude observar, en la figura anterior se muestra como se navegará por las pantallas de la aplicación.

No han sido mostradas todas las flechas con el fin de no saturar el esquema además de que la mayoría de ellas son redundantes. La página principal será la de Login, desde la cual el usuario podrá iniciar sesión o navegar a la pantalla de registro, si inicia sesión correctamente navegara a la pantalla de menú principal con los datos de su usuario cargados.

Si el usuario es nuevo y necesita registrarse, navegara a la pantalla de registro. En esta pantalla el usuario podrá registrarse en la aplicación por medio de un sencillo formulario, si los datos son correctos, el usuario será registrado y navegara al menú principal, y al igual que en el caso anteriormente descrito, tendrá los datos de su usuario cargados.

Como podemos observar, la pantalla de menú principal, y las demás, están dentro de un bloque llamado menú lateral, esto quiere decir que la aplicación dispone de un menú lateral con el que el usuario podrá navegar a cualquier pantalla desde donde quiera que se encuentre.

Lo único que cabría destacar, es la página de Libro. Esta página solo se puede navegar a ella a partir de catálogo, préstamos o reservas y esto se debe a que en esta página se cargaran los datos del libro buscado en el catálogo, el libro que se tiene prestado o del libro que se tiene reservado respectivamente. Por último, la página de ajustes es donde se le mostraran al usuario los datos de su cuenta y podría cambiar el idioma al que más guste.

7

Pruebas

7.1. Introducción

En este capítulo se presenta en detalle la fase de pruebas del proyecto. Para esto se van a exponer los tipos de pruebas que se han realizado en esta etapa, junto con la estrategia que se ha seguido para realizarlas. Por último se detallarán los resultados y se expondrán las conclusiones obtenidas.

7.2. Alcance de las pruebas

Como ya se ha ido comentando a lo largo del documento, a medida que se fueron terminando funcionalidades nuevas se iban realizando pruebas unitarias para ver que todo funcionaba correctamente. Una vez comprobado se añadía la nueva funcionalidad a la aplicación y se realizaban pruebas de integración para ver que todo siguiese funcionando como se esperaba.

Además de este tipo de pruebas, se han realizado otro tipo de pruebas centradas en la funcionalidad, compatibilidad y usabilidad. Debido a que solo se disponía de un recurso humano para el desarrollo, no se ha podido dedicar el tiempo necesario para realizar una fase completamente dedicada a realziar pruebas. En este apartado se listarán las pruebas que se han utilizado para validar y verificar el correcto funcionamiento de la aplicación.

7.3. Estrategia

Como se ha mencionado anteriormente, se han ido realizando pruebas unitarias sobre cada elemento de la aplicación. El objetivo de estas pruebas es garantizar que individualmente, todos estos elementos funcionan adecuadamente. Posteriormente se han realizado las correspondiente pruebas de integración, que al final del todo, las podríamos catalogar como las pruebas generales ya que prueban todos los elementos y su interacción entre ellos.

La estrategia que se ha seguido a la hora de realizar las pruebas de integración ha consistido en dar preferencia a aquellas funcionalidades más simples y atómicas, y una vez probadas, pasamos a las funcionalidades complejas formadas por la unión de varias simples, gracias a esto pudimos realizar las pruebas de integración fácilmente.

7.4. Tipo de pruebas realizadas

7.4.1. Funcionalidad

En primer lugar cabe mencionar las pruebas sobre funcionalidad, estas pruebas tienen como objetivo corroborar que la aplicación funciona correctamente según los requisitos funcionales mencionados anteriormente.

Como bien se ha mencionado, estas pruebas tienen como guía a los requisitos funcionales para comprobar que todos se cumplen y la funcionalidad es la deseada. Estas pruebas se han realizado cuando se añadía una nueva funcionalidad a la aplicación para asegurarnos de que todo fuese correcto.

Los resultados obtenidos indican el correcto funcionamiento de aproximádamente un 90 % de los requisitos planteados. Esto se debe a que el requisito funcional 1(RF1) no puede ser validado completamente, esto se detallara en el siguiente apartado, y el 7(RF7) por gestión de prioridades a la hora de acceder a realizar un préstamo de la única unidad disponible de un libro.

7.4.2. Compatibilidad

Al ser una aplicación híbrida, hay que tener en cuenta la compatibilidad de todos y cada uno de los sistemas operativos, ya sea Android, iOS y Windows Phone. Esta compatibilidad se basa en las distintas versiones que se usan para desarrollar y compilar.

En el caso de Android, esta aplicación se ha desarrollado en base a la API 16, es decir, que como mínimo, los dispositivos deberían tener la versión 4.1.2 en sus dispositivos para poder ejecutar la aplicación. Se decidió optar por esta como base porque había una gran cantidad de dispositivos android que tenían esta version o superiores. Posteriormente, se han ido sacando nuevas versiones de Android por lo que mantener la compatibilidad requiere de un constante trabajo en cuanto a desarrollo y adaptación de las nuevas APIs.

Las pruebas se han realizado en 4 dispositivos de 4 diferentes versiones cada uno de Android. Los resultados eran lo esperado, para aquellos dispositivos que disponían de la version base funcionaba correctamente, pero en aquellos que disponían una version inferior no permitía instalarla.

Para el caso de iOS, se ha desarrollado basándose en la versión más reciente a fecha de desarrollo, esta es la versión 8. Pero también sucede como en el caso de Android, desde la fecha de desarrollo se han ido sacando nuevas versiones de iOS por lo que mantener la compatibilidad con todas y cada una de ellas es un trabajo extra para la fase de mantenimiento.

En este caso no se han podido realizar las pruebas en los dispositivos iOS por falta de recursos, además de que para poder compilar la aplicación para iOS se requiere de un certificado de desarrollador de Apple que no era gratuito.

Y por último, en cuanto a Windows Phone también sigue el mismo patrón que las dos anteriores. En este caso se desarrollo basándose en la última versión lanzada, la 8.1.2. Y al igual que iOS, tampoco se ha podido realizar las pruebas por falta de recursos tales como dispositivos con Windows Phone instalado.

7.4.3. Unitarias

Este tipo de pruebas, las unitarias, son las que se han ido haciendo a lo largo del proyecto puesto que cada vez que se terminaba una funcionalidad completa o módulo, éste había que probarlo independientemente para ver si funcionaba correctamente. Para llevar a cabo estas pruebas se hizo uso de los frameworks Karma y

Jasmine [11].

Gracias a estos dos frameworks, podemos describir nuestras baterias de pruebas unitarias para cada componente, controlador o servicio creado fácilmente, y además, Karma se puede configurar para que nos muestre el porcentaje de cobertura que tienen nuestros test.

7.4.4. Integración

Por último, las pruebas de integración se han ido realizando una vez se comprobaba, mediante las pruebas unitarias, que el componente o módulo nuevo funcionanba correctamente. Una vez hecho esto, se añadía la funcionalidad nueva al conjunto del proyecto y se realizaban una serie de pruebas de integración para comprobar, que al igual que funcionaban de forma independiente, que también lo hacía de forma conjunta.

Para llevar a cabo estas pruebas de integración, también conocidos como pruebas End to End, se ha usado Protractor [12]. Lo bueno de este programa es que ejecuta pruebas end to end escritas en Javascript y además usa el framework Jasmine para la sintaxis de los test.

7.5. Resultados y conclusiones

Los resultados que se han obtenido al finalizar las pruebas nos indican que un 94 % de los requisitos funcionales que se definieron estan validados. Los fallos que no permiten llegar al 100 % están detectados pero no solventados por falta de recursos y tiempo.

El problema con las versiones y la compatibilidad tanto en Android, iOS y Windows Phone requieren de un constante trabajo de inspección en las diferentes funcionalidades, para controlar aquellas que estén declaradas obsoletas y aquellas que introduzcan cambios internos y sea necesario realizar esos cambios en la aplicación.

La solicitud para realizar préstamos tiene un problema muy puntual que se da cuando queda una única unidad disponible y varios usuarios entran a solicitarlo, este problema no se ha podido llegar a solventar pero necesitaría de una gestión de las peticiones por parte de la API.

Pese a esto, los resultados en general son buenos. Se ha logrado que los usuarios navegen por la interfaz de forma sencilla e intuitiva, los requisitos se han completado al 94 % y se han corregido los errores reportados por los usuarios. Finalmente se han recibido por parte de los mismos opiniones positivas.

8

Evaluación

8.1. Evaluación de los usuarios

The Book Manager se ha sometido a una multitud de pruebas y procesos de evaluación. La evaluación llevada a cabo por el equipo de desarrollo ha sido muy minuciosa, pero como este equipo es el que ha realizado el proyecto, poseen un conocimiento muy profundo de la aplicación. Esto conlleva a que en muchos casos se pasen por alto fallos que a lo mejor un usuario sin experiencia previa en la aplicacion le parezcan llamativos y visibles a simple vista.

Es por esto que ***The Book Manager*** ha sido evaluado por usuarios externos a este equipo de desarrollo. Estos usuarios se ha intentado que sean lo más parecidos a los usuarios finales por lo que se ha intentado que los usuarios no tuvieran muchos conocimientos sobre las tecnologías y herramientas utilizadas en el desarrollo de este proyecto.

Uno de los principales objetivos y requisitos comentados anteriormente, es el desarrollo de una interfaz de usuario intuitiva y sencilla para que aquellos usuarios que se dispongan a utilizar la aplicación, no necesiten ninguna guía de referencia o manual de uso sobre la misma. Sólo las bibliotecas que quieran llevar a cabo la implementación de esta aplicación en sus servicios necesitarán un contacto inicial con el equipo de desarrollo para configurar el servicio y después seran autónomos.

Una vez terminado los procesos de evaluación y recogidas las respuestas se ha visto que los usuarios, sobre todo, han dado sugerencias de como podría mejorar la aplicación y la experiencia de uso de cara a nuevos usuarios. A parte de las sugerencias, también se han recogido de algunos usuarios unas incidencias con algunos errores que al equipo de desarrollo se les pasó por alto, un caso de estos podría ser cuando dos usuarios intentan realizar un préstamo y además es la última unidad disponible. Pese a esto, fallos en situaciones extremas se salen del flujo de la aplicación y se podrían solucionar con un usuario experto.

En cuanto a la interfaz y la usabilidad, los usuarios no han reportado apenas errores, solamente algunas sugerencias, por lo demás han sido todos comentarios positivos refiriéndose a la interfaz y a su sencillez e intuitividad.

8.2. Beneficios de la aplicación

Los beneficios que ***The Book Manager*** puede ofrecer a aquellas bibliotecas que estén dispuestas a implantar esta aplicación en sus servicios son varios, algunos de ellos ya se han idoc comentando a lo largo del documento. Implantar esta aplicación ayudara a los usuarios de la biblioteca a tener un acceso más sencillo y con una interfaz más amigable e intuitiva de las ya disponibles por la propia biblioteca.

9 Conclusiones y líneas futuras

9.1. Conclusiones

En este trabajo se ha desarrollado y presentado una herramienta para dispositivos móviles y tabletas que ofrece una alternativa sencilla e intuitiva para gestionar los préstamos de los usuarios. Duante este proceso ha sido fundamental el trabajo de detección de las necesidades de los usuarios para elaborar una herramienta "útil".

Las conclusiones finales del presente Trabajo de Fin de Grado son las siguientes:

- He aprendido a aplicar un stack moderno en un proyecto real, por lo que he adquirido los conocimientos necesarios para desarrollar aplicaciones propias por mí mismo.
- He aprendido a realizar despliegues en entornos reales, en este caso los Amazon Web Services, lo que me ayudará en mi desarrollo personal y profesional.
- He aprendido a desarrollar aplicaciones que están pensadas para ser soportadas en los diferentes dispositivos ya tengan Android, iOS o Windows Phone , para que de esta manera poder llegar al mayor número de usuarios posibles.
- He aprendido a desarrollar proyectos aplicando reglas de buenas prácticas a la hora de programar.

9.2. Trabajo futuro

Las líneas de trabajo futuro que se han propuesto para el proyecto son las siguientes:

- Añadir un servidor dedicado exclusivamente para el manejo y tratado de las imágenes para mejorar la respuesta del servidor principal de cara a los usuarios.
- Ampliar las funcionalidades de la aplicación.
    - Añadir notificaciones push para avisar a los usuarios de cuando tienen que devolver el libro o de cuando está disponible un libro reservado por ellos.
    - Añadir la posibilidad de añadir comentarios y valoraciones a los libros leídos.
- Implementación de otras opciones para los usuarios. Estas ideas son fruto de las sugerencias que se han recibido durante la evaluación de la aplicación.
    - Realización de una página web para también acceder desde el navegador.
    - Añadir un flujo de registro mediante una red social, ya sea facebook o google, para facilitarles el registro.

- Añadir la posibilidad de valorar los libros ya leídos de cara a ayudar a los demás usuarios a decantarse por el libro o no.

Bibliografía

[1] "Las 5 etapas en la ingeniería del software," http://proyectosguerrilla.com/blog/2013/02/las-cinco-etapas-en-la-ingenieria-del-software/.

[2] "Qué es el stack MEAN," https://carlosazaustre.es/blog/desarrollo-full-stack-javascript-tambien-conocido-como-mean

[3] "Desplegar una app en amazon," https://scotch.io/tutorials/deploying-a-mean-app-to-amazon-ec2-part-1.

[4] "Qué es MongoDB," https://www.genbetadev.com/bases-de-datos/mongodb-que-es-como-funciona-y-cuando-poder

[5] "Qué es Express," http://expressjs.com/es/.

[6] "Qué es NodeJS," https://nodejs.org/es/about/.

[7] "Buenas prácticas a la hora de diseñar una API REST," https://www.genbetadev.com/desarrollo-web/unas-cuantas-buenas-practicas-cuando-hablamos-de-apis-rest.

[8] "Autenticación basada en JWT," https://code.tutsplus.com/es/tutorials/token-based-authentication-with-angularjs-noc

[9] "Qué es AngularJS," http://www.desarrolloweb.com/articulos/que-es-angularjs-descripcion-framework-javascript-con html.

[10] "Qué es Ionic," http://www.phonegapspain.com/que-es-y-como-empezar-con-ionic-framework/.

[11] "Test unitarios en AngularJS," https://www.adictosaltrabajo.com/tutoriales/angularjs-test-unitarios/.

[12] "Test end to end en AngularJS," https://docs.angularjs.org/guide/e2e-testing.

I

Glosario

Acrónimos

API	Application Programming Interface
CSS	Cascading Style Sheets
HTML	HyperText Markup Language
JSON	Javascript Object Notation
JWT	JSON Web Token
MEAN	(M)ongo + (E)xpress + (A)ngular + (N)ode
MVC	Model-View-Controller
REST	Representational State Transfer
SASS	Syntactically Awesome Stylesheets
SPA	Single page application
XML	eXtensible Markup Language

Definiciones

Aplicación Hibrida: aquellas aplicaciones que se desarrollan una vez y se compilan para varios destinos como Android, iOS y Windows Phone.

Cliente-servidor: modelo de arquitectura que separa las tareas de solicitud de recursos de las relacionadas con el proceso de proveer dichos recursos.

Desarrollo ágil: no es más que una metodología de gestión adaptativa, que te permite llevar a cabo, proyectos de desarrollo de software, adaptándote a los cambios y evolucionando en forma conjunta con el software.

Framework: estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concretos, que puede servir de base para la organización y desarrollo de software.

Frontend: son todas aquellas tecnologías que corren del lado del cliente, es decir, todas aquellas tecnologías que corren del lado del navegador web, generalizándose más que nada en tres lenguajes, HTML, CSS Y JavaScript.

II

Anexos

A

AngularJS

El framework de Google que cada día tiene más adeptos y que se apoya en una fortísima organización del código utilizando para ello diversos patrones de diseño a nivel de JavaScript (DI, Templates, Bindings etc).

Como se puede observar en la imagen siguiente:

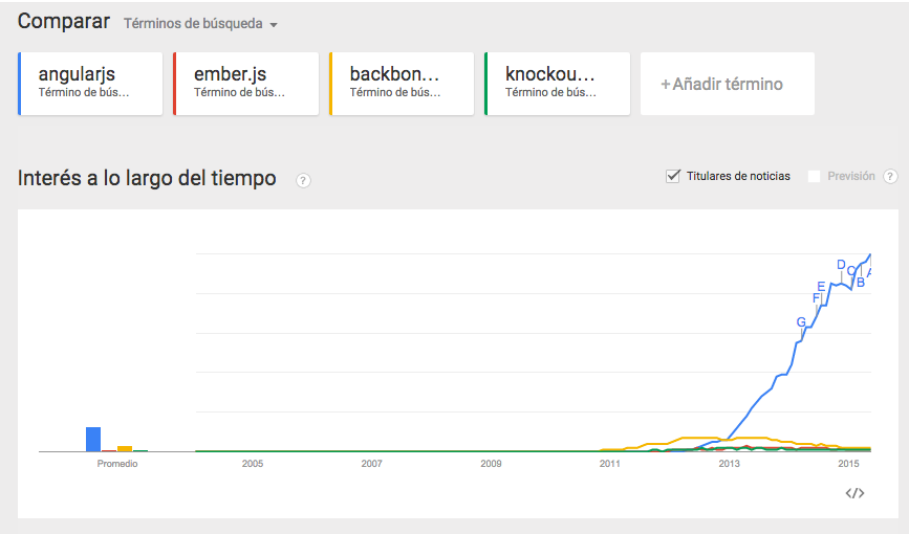


Figura A.1: Tendencia de los distintos frameworks de javascript.

La linea de AngularJS tiene un crecimiento más fuerte que los demás y esto se debe a dos importantes factores:

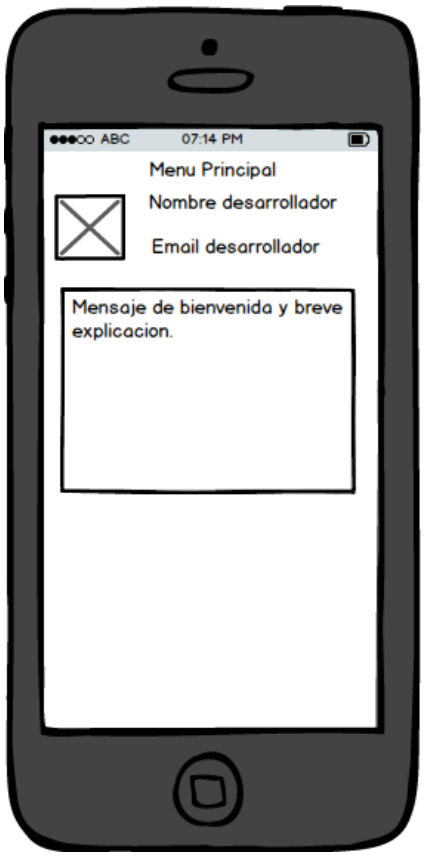
- Google: Muchos frameworks Open Source han tenido un gran éxito debido a que había un apoyo fuerte por parte de empresas tecnológicas que originalmente los crearon. Sin ese apoyo en muchas ocasiones es difícil simplemente sobrevivir. El hecho de que Google este detrás de Angular genera confianza en los desarrolladores.
- Organización: Angular.js aporta un nivel de organización del código muy alto, posibilitando la creación de aplicaciones complejas sin perder el control de lo que se esta construyendo.

B

Maquetas de la aplicación

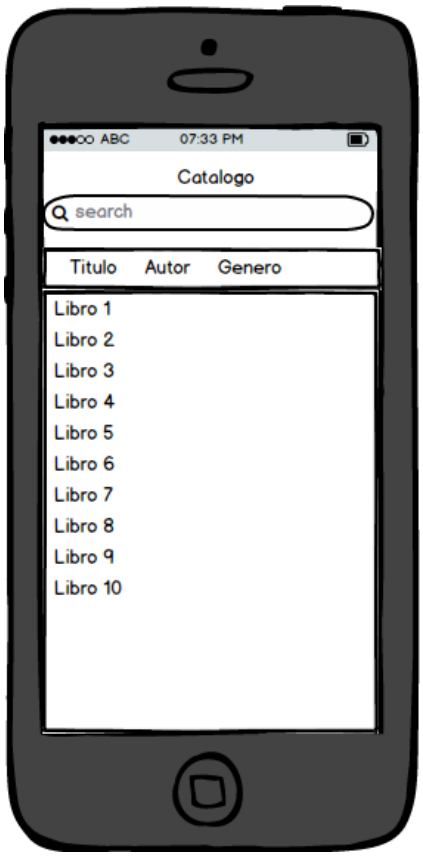
(a) Login (b) Registro

Figura B.1: Imágenes de la aplicación (parte 1): Login y registro



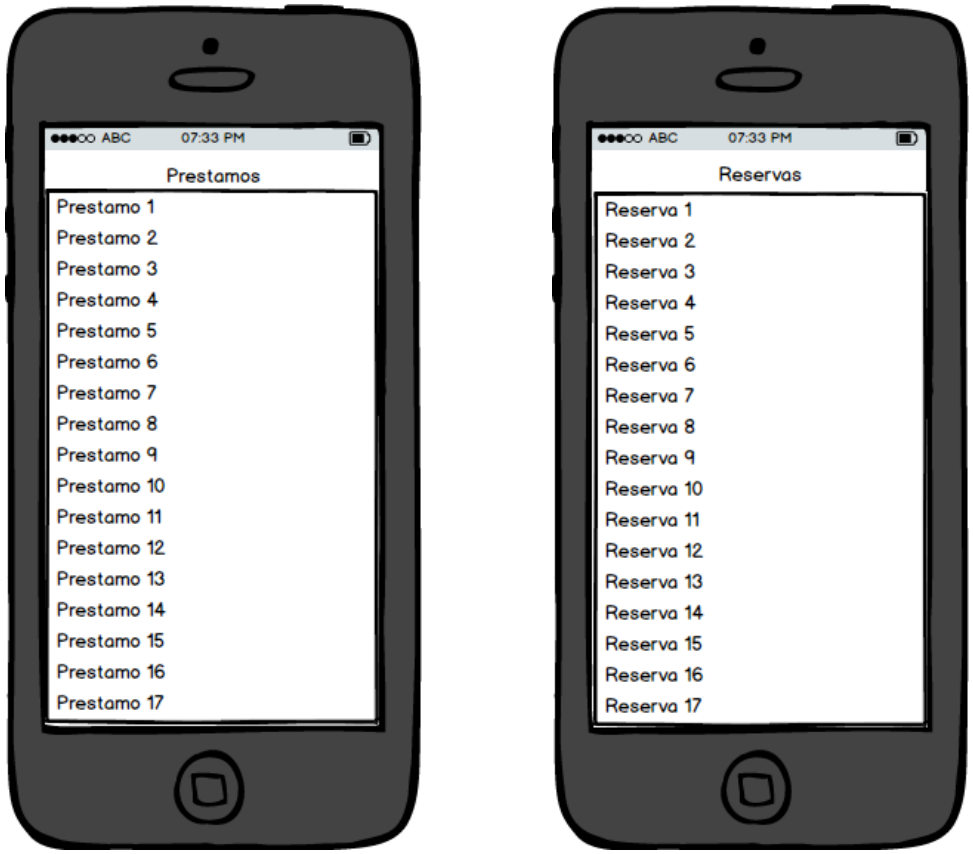
(a) Menú principal

Figura B.2: Imágenes de la aplicación (parte 2): Menú



(a) Catálogo

Figura B.3: Imágenes de la aplicación (parte 3): Catálogo



(a) Préstamos (b) Préstamos

Figura B.4: Imágenes de la aplicación (parte 4): Préstamos y Reservas

(a) Datos de un libro

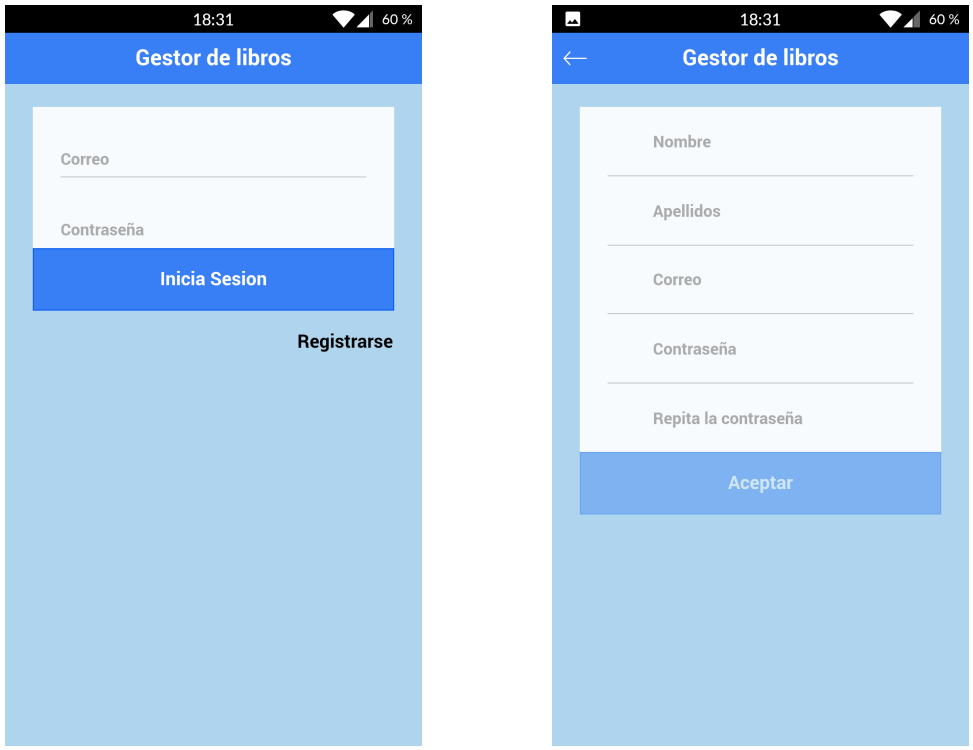
Figura B.5: Imágenes de la aplicación (parte 5): Datos de un libro

(a) Ajustes

Figura B.6: Imágenes de la aplicación (parte 6): Ajustes

C

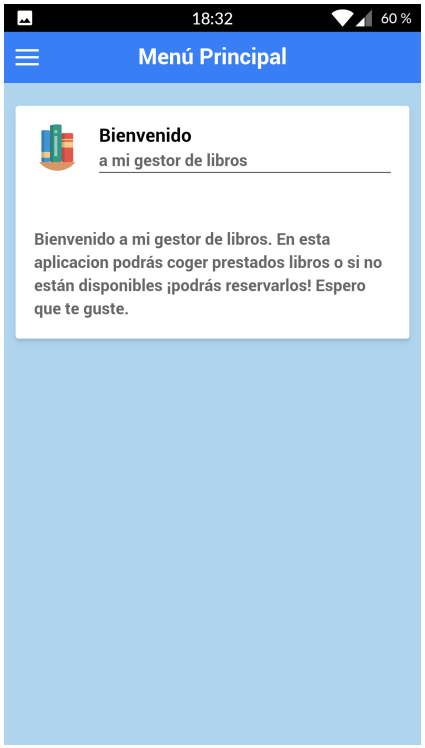
Imágenes de la aplicación



(a) Login

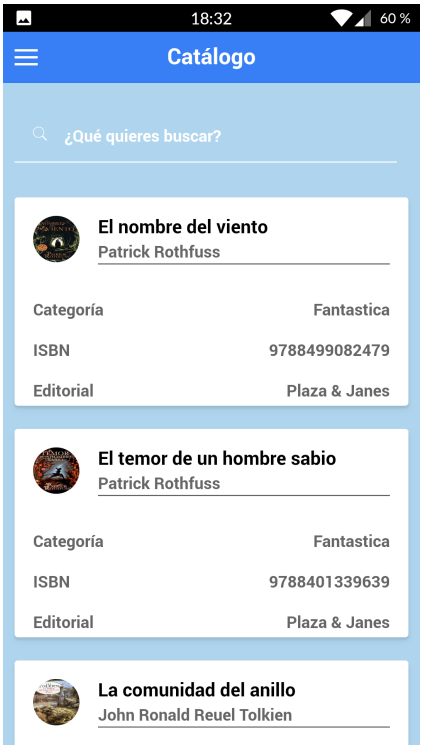
(b) Registro

Figura C.1: Imágenes de la aplicación (parte 1): Login y registro

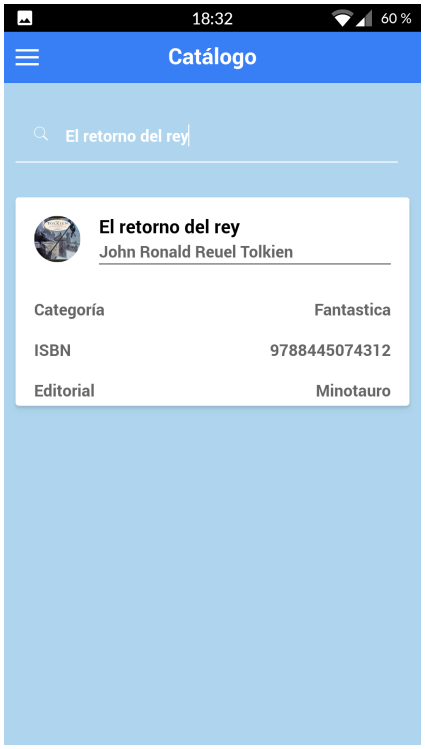


(a) Menú principal

Figura C.2: Imágenes de la aplicación (parte 2): Menú

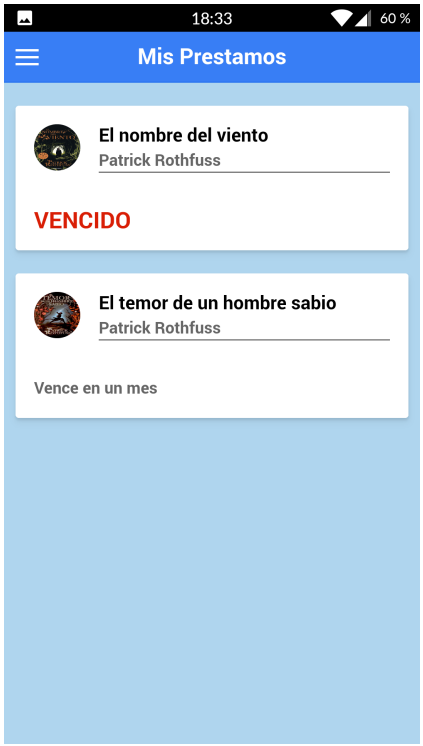


(a) Catálogo



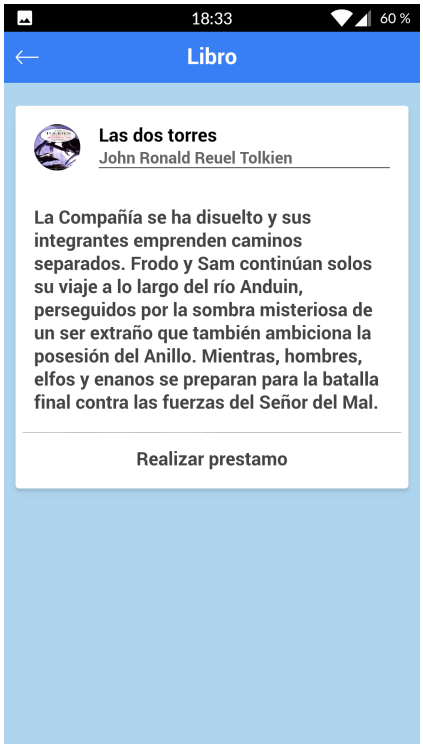
(b) Búsqueda en el catálogo

Figura C.3: Imágenes de la aplicación (parte 3): Catálogo



(a) Préstamos

Figura C.4: Imágenes de la aplicación (parte 4): Préstamos



(a) Datos de un libro

Figura C.5: Imágenes de la aplicación (parte 5): Datos de un libro

(a) Ajustes

Figura C.6: Imágenes de la aplicación (parte 6): Ajustes