Universitat Oberta de Catalunya

uoc.edu

Desarrollo de una aplicación para facilitar la convivencia en pisos compartidos

Alberto Blasco Redondo
Máster en Ingeniería Informática
Desarrollo de Aplicaciones sobre Dispositivos Móvil

Jordi Ceballos Villach
Robert Clarisó Viladrosa
12 de junio de 2019

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Desarrollo de una aplicación para facilitar la convivencia en pisos compartidos*
Nombre del autor:	*Alberto Blasco Redondo*
Nombre del consultor/a:	*Jordi Ceballos Villach*
Nombre del PRA:	*Robert Clarisó Viladrosa*
Fecha de entrega (mm/aaaa):	06/2019
Titulación:	*Máster en Ingeniería Informática*
Área del Trabajo Final:	*Desarrollo de Aplicaciones sobre Dispositivos Móvil*
Idioma del trabajo:	*Castellano*
Palabras clave	*Convivencia, Hogar, Ionic*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados i conclusiones del trabajo.*

En España cada vez más personas tienen la necesidad de compartir piso, y no siempre se tiene la suerte de contar con compañeros adecuados, por tanto el objetivo de este trabajo es el de desarrollar una aplicación que pueda ayudar a estas personas en su convivencia y el día a día.

Esta aplicación está desarrollada con Ionic y Angular, entornos de desarrollo web con los que luego se ha generado una aplicación nativa.

La metodología empleada en la realización del trabajo es la de Diseño Centrado en el Usuario, poniendo así al usuario en el centro del desarrollo y entendiendo mejor cuales son sus necesidades a nivel de experiencia de usuario.

Además de la aplicación también se ha desarrollado un backend que permite la gestión de datos e información que utiliza la aplicación.

Por último, la aplicación resultante es un producto probado y que cumple con los requisitos establecidos en el presente trabajo, pudiendo afianzar los conocimientos adquiridos en el máster durante su desarrollo.

Abstract (in English, 250 words or less):

In Spain every day there are more people that need to share a house and not always are lucky enough to have good roommates. For this reason, the main objective of this work is to develop an app that can help these users in their coexistence and day to day.

This application has been developed using Ionic and Angular, two web frameworks that can be used to build a native application.

The methodology followed in the execution of this work is User Centered Design, which puts the users in the middle of the development to understand better what they need.

In addition to the application, a backend have been developed, that allows the management of the data and information used by the application.

Finally, the result of this work is a tested product that meets the requirements established at the start of the project, consolidating during the development the knowledge acquired in this course

Índice

Lista de figuras

1. Introducción

1.1 Contexto y justificación del Trabajo

El contexto social actual en España, derivado de la crisis económica iniciada en 2008, ha provocado que cada vez más personas opten por la opción de compartir vivienda.

La media de quienes comparten piso es, según un estudio publicado por el INE en enero de 2019, de 29 años (Navarro, 2019). Así mismo, el perfil más habitual de compañero de piso es el de poslicenciado o diplomado “que o están ampliando sus estudios o necesitan pagar un alquiler asequible por exigencias económicas” (expansion.com, 2015).

En algunas ocasiones, compartir piso es difícil, sobre todo cuanta más gente convive en esa vivienda. Hay gente que sabe organizarse mejor y otros peor, es complicado seguir la pista a todos los pagos, facturas, etc. En definitiva, la convivencia puede ser complicada y, actualmente, no existen muchas herramientas que ayuden a facilitar la convivencia a estas personas.

Algunos de estos problemas son, como recoge el portal Fotocasa (2016), la falta de control del dinero que se gasta, desorganización dentro de la vivienda, etc. Todo esto puede conllevar a disputas entre los compañeros y que se generen malos ambientes.

Esta aplicación pretende, por tanto, proponer una solución a estos problemas que se derivan de la convivencia. No obstante, sí que es verdad que realizando una búsqueda por las tiendas de aplicaciones de Android e iOS se han encontrado diversas soluciones parciales a esto, pero todas estas aplicaciones tenían ciertas carencias las cuales se pretenden suplir. La mayoría solo sirve para ayudar en uno de todos los problemas, ya sea gestionar el dinero o las tareas, pero no permiten mantener toda la vivienda y sus problemas. Dos ejemplos de este tipo de aplicaciones son:

- Splitwise - Cuentas y gastos (Google Play, 2019A): Es una aplicación que permite llevar el control de las cuentas y el dinero. Para ello se crean conceptos y se le asigna un precio. Además se puede elegir a que usuario corresponde hacer el pago. Como punto fuerte destaca una interfaz clara y la capacidad de tener un historial de todos los movimientos. Un punto en contra es que solo tiene esta funcionalidad, su único propósito es el de gestionar dinero.
- Bring! Lista de compras (Google Play, 2019B): Esta aplicación está orientada a la creación de listas de la compra y la gestión de las mismas. Destaca una interfaz limpia y la facilidad de uso de la misma. Como punto débil respecto a la propuesta, no tiene ninguna funcionalidad más aparte.

La aplicación propuesta para este TFM pretende facilitar en cierta medida la convivencia a aquellas personas que tienen que compartir piso. Se ofrecerán funcionalidades para que los usuarios puedan gestionar las tareas de la casa, los gastos y poder comunicarse entre ellos de una manera sencilla. La aplicación debe ser fácil de usar para que se pueda entender rápidamente.

1.2 Objetivos del Trabajo

Los objetivos que debe cumplir este trabajo y por tanto los requisitos de la aplicación son:

RF.1 La aplicación debe permitir al usuario iniciar sesión en su cuenta.

- RF.1.1 Será necesario un correo electrónico y una contraseña.

RF.2 La aplicación permitirá al usuario recuperar su contraseña en caso de que la haya olvidado.

- RF.2.1 Se deberá introducir el correo asociado a la cuenta al cual se mandarán instrucciones para cambiar la contraseña.
- RF.2.2 Se ofrecerá una web simple en la cual se podrá modificar la contraseña.

RF.3 La aplicación permitirá el registro de usuarios para crear una cuenta nueva.

- RF.3.1 Los datos requeridos serán el correo electrónico, una contraseña y un nombre.

RF.4 La aplicación permitirá a los usuarios que hayan iniciado sesión modificar su perfil.

- RF.4.1 En el perfil se podrán añadir datos como la fecha de nacimiento o los apellidos.

RF.5 La aplicación permitirá a los usuarios crear sus propias casas virtuales.

- RF.5.1 Se les ofrecerá un configurador simple en el cual podrán indicar cuántas habitaciones tiene la casa, de qué tipo es cada habitación y el responsable de dichas habitaciones.
- RF.5.2 El usuario que cree la casa obtendrá el rol de administrador de la casa.
- RF.5.3 Los usuarios podrán unirse a la casa usando un buscador.
- RF.5.4 El administrador de dicha casa será el que tenga que aceptar al usuario.

RF.6 Los usuarios dispondrán de dos roles diferentes dentro de la aplicación, administrador e inquilino.

- RF.6.1 El rol de administrador podrá gestionar todos los aspectos de la casa virtual y de los usuarios asociados dicha casa.
- RF.6.2 Un administrador puede cambiar el rol del resto de usuarios a conveniencia.
- RF.6.3 El rol de inquilino tendrá las funcionalidades más limitadas y solo podrá participar en ciertas tareas.

RF.7 La aplicación dispondrá de una opción de gestión de tareas.

- RF.7.1 Los usuarios administradores podrán crear tareas comunes para toda la casa y añadir a los responsables de llevarlas cabo.

RF.7.2 En caso de que el administrador no quiera asignar un responsable de la tarea, se podrá asignar de manera aleatoria. Si se usa esta opción se podrá elegir si se tiene en cuenta la aportación de cada usuario, esto quiere decir que un usuario que haya colaborado más (haya hecho más tareas) tendrá menos posibilidades de ser asignado como responsable.

RF.8 La aplicación deberá disponer de un tablón global de la casa.

RF.8.1 En este tablón se podrán poner todas las notas que quiera cada usuario y todos los usuarios podrán leerlas.

RF.8.2 Las notas aparecerán por orden de creación.

RF.8.3 Las notas podrán ser eliminadas.

RF.9 La aplicación permitirá a los usuarios crear listados, principalmente de compras.

RF.9.1 Los listados podrán ser públicos o privados.

RF.10 La aplicación notificará a los usuarios de diversas acciones que ocurran en la casa.

RF.10.1 Cuando alguien ha completado una tarea

RF.10.2 Cuando se añada o modifique una nota del tablón

RF.10.3 Cuando se haya asignado al usuario como responsable de una tarea.

RF.10.4 La aplicación dejará escoger a los usuarios que notificaciones reciben.

<u>Requisitos No Funcionales</u>

RNF.1 La aplicación debe ser sencilla de usar ajustándose a los datos obtenidos en los análisis de usuarios.

RNF.1.1 El tiempo de aprendizaje de la aplicación debe ser menor de 2 horas.

RNF.1.2 La aplicación debe proporcionar mensajes de error que sean informativos para que el usuario entienda que está ocurriendo.

RNF.3 La interacción con el usuario debe ser fluida, sin bloquear la interfaz más de lo necesario o avisando al usuario cuando se produzca una carga de datos.

RNF.4 Los datos se modificarán en el momento, no habrá un sistema de cache local en la aplicación, por tanto el usuario deberá estar conectado a internet en todo momento.

RNF.5 Solo los usuarios con rol de administrador podrán modificar los roles del resto de usuarios.

RNF.6 La conexión entre *frontend* y *backend* debe ir cifrada mediante HTTPS.

RNF.7 La aplicación estará desarrollada en español.

El sistema estará conformado principalmente por la aplicación (*frontend*) y el *backend*.

- La principal plataforma a la que se dirige la aplicación es Android con una versión mínima de Android 5.1. Sin embargo el desarrollo va a ser multiplataforma mediante el uso de Ionic + Angular, por lo que podría ser migrada a otros sistemas como iOS. La elección de Android se debe principalmente a su mayor cuota de mercado y los bajos costes económicos del desarrollo en comparación con otras plataformas.
- El backend deberá atender todas las peticiones realizadas por la aplicación, siendo el principal responsable del almacenamiento y el procesamiento de la información generada por los usuarios en la aplicación. Se contará con una base de datos MySQL para este propósito y se alojará en un servidor de AWS, el cual permite configurar ampliamente máquinas virtuales para este propósito.

1.3 Enfoque y método seguido

La estrategia que se ha decido llevar a cabo en este trabajo es la de desarrollar un producto nuevo con funcionalidades ya existentes. A pesar de que existen diversas aplicaciones que ya disponen de funcionalidades similares estas no están agrupadas, por lo tanto se quiere aprovechar ese hueco y crear un producto nuevo aunque ciertas funcionalidades ya existan. Esta es la estrategia más adecuada ya que se quiere hacer todo de cero, sin que haya similitudes con otras aplicaciones aunque si buscar cuales son sus puntos flacos y aprovecharlos.

En cuanto a la metodología, el proyecto se inicia realizando una breve investigación del mercado actual de aplicaciones relacionadas con la gestión de hogares y de la cantidad de usuarios que podrían llegar a usarla. Una vez se ha visto que la aplicación tiene sentido de ser, se ha procedido a la toma de requisitos, mediante la cual se van a obtener las funcionalidades mínimas que debe tener la aplicación y por tanto se deben incluir en el desarrollo.

El siguiente paso será realizar un análisis de usuarios y unos primeros prototipos para concebir la UX y la interfaz gráfica de la aplicación, para lo cual se seguirá el modelo de Diseño Centrado en el Usuario. Con esto se podrá realizar una evaluación de la usabilidad para entender mejor como realizar la aplicación focalizando en el público objetivo.

Una vez se tenga claro como plasmar la interfaz de usuario, se comenzará con el desarrollo, empezando con un análisis técnico y el establecimiento de la base de datos, la cual será MySQL. El siguiente paso será el desarrollo del API para lo que se utilizará Zend Framework 3, un *framework* de PHP que permitirá facilitar ciertas operaciones. Todo esto irá montado en un servidor Apache el cual estará desplegado en una máquina virtual EC2 de Amazon Web Services.

Una vez se haya avanzado con el *backend*, se comenzará a desarrollar la aplicación con Ionic 4 y Angular 7.

Durante el desarrollo se procederá a realizar diversos test para comprobar que todo funciona como es debido.

Además, se irá recopilando toda la información en la memoria del trabajo conforme se vaya generando contenido. Durante las fases finales del proyecto se procederá a una evaluación con usuarios, con la cual se buscarán posibles errores, recibir *feedback* sobre el funcionamiento y el diseño de la aplicación, etc.

Por último, se hará una revisión global de toda la aplicación y de la memoria antes de la entrega final para comprobar que todo es correcto y que no hay errores.

1.4 Planificación del Trabajo

Se han considerado las tareas mostradas en la siguiente tabla como las que se deben realizar en este TFM. Para cada tarea se ha estimado su duración y una fecha de inicio y de fin.

Tarea	Duración (h.)	Fecha inicio	Fecha fin
Toma de requisitos	2	01/03/2019	05/03/2019
Búsqueda bibliográfica	6	01/03/2019	05/03/2019
Planificación	4	01/03/2019	05/03/2019
HITO – PEC 1			06/03/2019
Análisis de usuarios	10	08/03/2019	10/03/2019
Diseño conceptual	16	10/03/2019	15/03/2019
Prototipado	20	16/03/2019	23/03/2019
Evaluación	16	24/03/2019	27/03/2019
Diseño de la BDD	8	28/03/2019	01/04/2019
Diseño de la arquitectura del sistema	8	28/03/2019	01/04/2019
HITO – PEC 2			03/04/2019
Creación BDD	4	05/04/2019	06/04/2019
Desarrollo del API	30	06/04/2019	14/04/2019
Desarrollo de la aplicación	70	14/04/2019	08/05/2019
Evaluación y corrección de errores	20	09/05/2019	13/05/2019
HITO – PEC 3			15/05/2019
Finalización del desarrollo	40	15/05/2019	26/05/2019
Evaluación y corrección de errores	20	27/05/2019	02/06/2019
Documentación	16	03/06/2019	06/06/2019

Redacción y revisión final de la memoria	20	07/06/2019	09/06/2019
Preparación de la presentación	10	09/06/2019	11/06/2019
HITO – Entrega final			12/06/2019

Viendo la anterior tabla, el total de horas para realizar el TFM es de 320 divididas a lo largo de casi 4 meses.

Las fechas se han estimado teniendo en cuenta la duración de cada tarea y que los días laborales se podrán dedicar unas 2-3 horas al trabajo y los días festivos entre 6 y 8 horas.

A continuación se muestra una imagen de cómo quedaría el diagrama de Gantt:

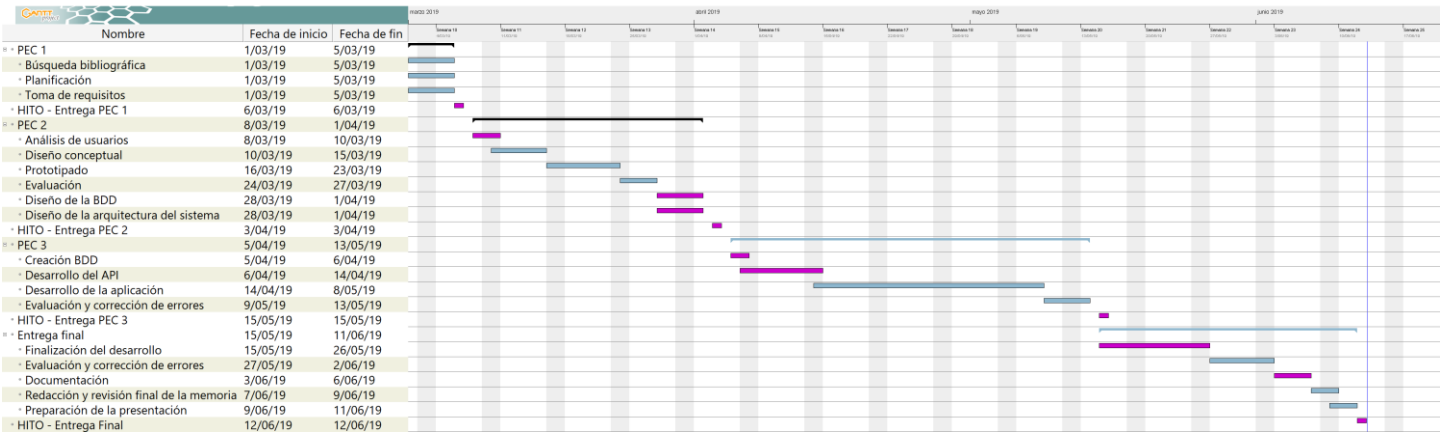


Ilustración 1. Diagrama de Gantt del proyecto

1.5 Breve sumario de productos obtenidos

Los productos que se van a generar al finalizar este trabajo son:

- Aplicación funcional que cumpla con los requisitos establecidos.
- Código fuente de la aplicación.
- Código fuente del *backend.*
- Fichero SQL con la estructura de la base de datos.
- Memoria final del trabajo.
- Manual de usuario.
- Presentación del proyecto en vídeo resumiendo los aspectos más esenciales del desarrollo.

1.6 Breve descripción de los otros capítulos de la memoria

Entre los siguientes capítulos de la memoria se dedica un capítulo a los aspectos relacionados con el diseño, el cual se divide en 3 apartados, análisis de los usuarios de la aplicación y el *target*, diseño conceptual y prototipado, y un último subcapítulo de evaluación para saber si los diseños usados son correctos o los fallos que puedan tener. En este capítulo por tanto se pretenderá entender quién va a usar la aplicación y diseñarla pensando en ese perfil de usuario.

Otro capítulo se centra en el apartado técnico de del trabajo donde se explicarán los casos de uso, el modelo de datos utilizado, un esquema de la base de datos y como se interconecta con toda la aplicación. Con esto se entenderá mejor la gestión de datos que hace la aplicación y de cuanta información dispone.

A continuación, un capítulo de implementación en el que se explica cómo se ha desarrollado la aplicación, qué tecnologías se han empleado y la estructura global usada. Se incluye tanto la parte de *frontend* como la de *backend.*

El último capítulo está dedicado a las conclusiones, donde se expone lo aprendido en la realización del presente trabajo, un análisis crítico del mismo y un apartado dedicado a las mejoras a futuro y cuáles son las posibilidades de la aplicación a largo plazo, nuevas funcionalidades o planes de expansión para entender cuáles son las salidas de la aplicación y cuál podría ser su recorrido.

Además en los anexos se incluyen manuales de usuarios con capturas de pantalla de la aplicación para poder conocer lo mejor posible su funcionamiento.

2. Diseño

2.1 Usuarios y contexto de uso

Para la indagación sobre los usuarios se han elegido dos técnicas diferentes, una cualitativa y otra cuantitativa. En primer lugar, se ha usado la técnica de *shadowing* ya que permite ver realmente cómo se comporta el usuario en su entorno natural, lo cual aporta mucho *feedback* para entender sus necesidades. Además, la posibilidad de plantear preguntas durante el proceso hace de esta una técnica muy versátil. En cuanto al segundo método de indagación, se ha realizado una encuesta a diferentes personas para valorar de una manera cuantitativa las necesidades de los usuarios y, por tanto, de la aplicación. Gracias a esta técnica el perfilado de usuarios ha sido más sencillo ya que con los datos obtenidos y analizando los resultados se han podido realizar diversos subconjuntos para generar los diferentes perfiles.

2.1.1. *Shadowing*

Se ha empleado esta técnica para ver realmente cuánto uso del móvil hacen los usuarios en su día a día y observar si es utilizado para alguna de las tareas que propone la aplicación, así como para entender cuándo lo usan y cómo. Además, se quería saber si realmente hay posibilidades de que usen la aplicación en el día a día o no ya que podría ser que lo que propone la aplicación no les interese o tengan alternativas más convenientes.

Se ha realizado la técnica sobre tres personas, dos entre 25 y 30 años y otra entre 40 y 49 años. Debido a la naturaleza de la aplicación, solo había que realizar el seguimiento en su casa o en los alrededores, por lo que no ha habido muchas objeciones ni impedimentos. El seguimiento se ha realizado durante unas 3 horas aproximadamente en las que el usuario hacía su vida normal, ambos casos por la tarde, después de que los usuarios acabasen su jornada laboral. No se vio necesario el uso de ningún método de grabación más allá de tomar notas. También se aprovecharon breves momentos de pausa de los usuarios para plantearles ciertas preguntas según lo que se había ido viendo, sin llegar a la profundidad de una entrevista, pero tampoco siendo preguntas superficiales.

- USUARIO 1

  Durante el seguimiento del primer usuario se pudo ver cómo este hacía la compra semanal y colaboraba con las tareas del hogar. Este usuario comparte piso con un conocido y, por lo tanto, ambos dividen las tareas y responsabilidades entre ambos. Mientras el usuario hacía la compra se pudo ver como tenía apuntados los productos que necesitaba comprar y los iba tachando conforme los iba metiendo en el carro. El siguiente momento destacado fue a la hora de fregar la vajilla ya que, como no disponen de lavavajillas, se van turnando cada vez uno, aunque se observó que el usuario lo hacía de mala gana, sobre lo cual se le preguntó más adelante.

  En una pausa que realizó el usuario se le preguntó por la aplicación que había usado para hacer la compra (Google Keep), la cual dijo que era bastante sencilla. También se le preguntó si tenía la lista compartida con su

compañero para mejorar la comunicación a lo que respondió que no, pero que le sería de gran ayuda (Google Keep sí que ofrece la funcionalidad pero el usuario lo desconocía). En cuanto a su actitud a la hora de fregar los platos, el usuario comentó que no estaba seguro de que fuese su turno y de que era un problema recurrente.

- USUARIO 2

  El segundo usuario sobre el cual se hizo el seguimiento comparte piso con su pareja y de este se pudo observar la realización de ciertas tareas del hogar, concretamente recoger y limpiar algunas habitaciones. Durante la realización de estas tareas, el usuario no hizo uso del móvil más que para mirar algún mensaje, pero nada relacionado con las tareas. Al finalizarlas se le preguntó que si creía que podría usar alguna aplicación a lo que respondió que no. También se le preguntó cómo se organizaba con su pareja o si tenían algún malentendido a la hora de repartir las tareas, ante lo cual sí que confesó que nunca estaba seguro de que el reparto fuese justo a partes iguales, pero que no tenía manera de saberlo. Aprovechando esto se le preguntó si utilizaría una aplicación que le ayudase con ese asunto a lo que respondió afirmativamente.

- USUARIO 3

  El tercer usuario se trata de una persona que vive con su familia, y se encarga principalmente del mantenimiento del hogar. Durante la observación se vio como realizaba diversas tareas como limpiar, hacer la compra o la cena. En ningún momento utilizó ni miró el móvil para nada, tan solo cuando había finalizado las tareas comprobó si tenía algún mensaje. Al acabar la sesión se le preguntó el porqué de esto a lo que este respondió que no tenía la necesidad. También se le preguntó si alguna vez había usado alguna aplicación que le ayudase en el día a día y contestó que no, pero que nunca se lo había planteado. Sin embargo, sí que comentó que se podría plantear el uso de alguna herramienta que fuese sencilla y que realmente le ayudase.

De este análisis se ha llegado a la conclusión de que los usuarios no utilizan el móvil para hacer las tareas cotidianas del hogar a pesar de que pueden tener problemas a la hora de organizarse. Sobre todo al estar en convivencia se ha observado como la organización parece un tema complicado al cual no se le presta demasiada atención y que afecta a dicha convivencia. Esto quedó plasmado con los dos últimos usuarios ya que ambos demostraron su descontento con la gestión de las tareas que llevaban a cabo. También se destaca el caso del tercer usuario que, aunque apenas usó el móvil durante el seguimiento, sí que era receptivo a usar herramientas adecuadas siempre y cuando no le complicasen la vida más de lo necesario.

2.1.2. Encuestas

Utilizando la herramienta de formularios de Google Drive, se han planteado ciertas preguntas, las primeras más genéricas, para entender la demografía de los usuarios. A continuación, se han realizado preguntas para entender los hábitos respecto al uso del móvil y las tecnologías. Por último, se ha preguntado

por aspectos más concretos de las posibles funcionalidades de la aplicación, como la gestión de tareas. Esta encuesta se ha distribuido entre conocidos y conocidos de los mismos durante una semana, tras lo cual se han recopilado las diferentes respuestas. La encuesta puede consultarse en el siguiente enlace: https://forms.gle/harFLLg1918zKyAf7:

El resultado global de la encuesta se puede ver en el anexo número 3.

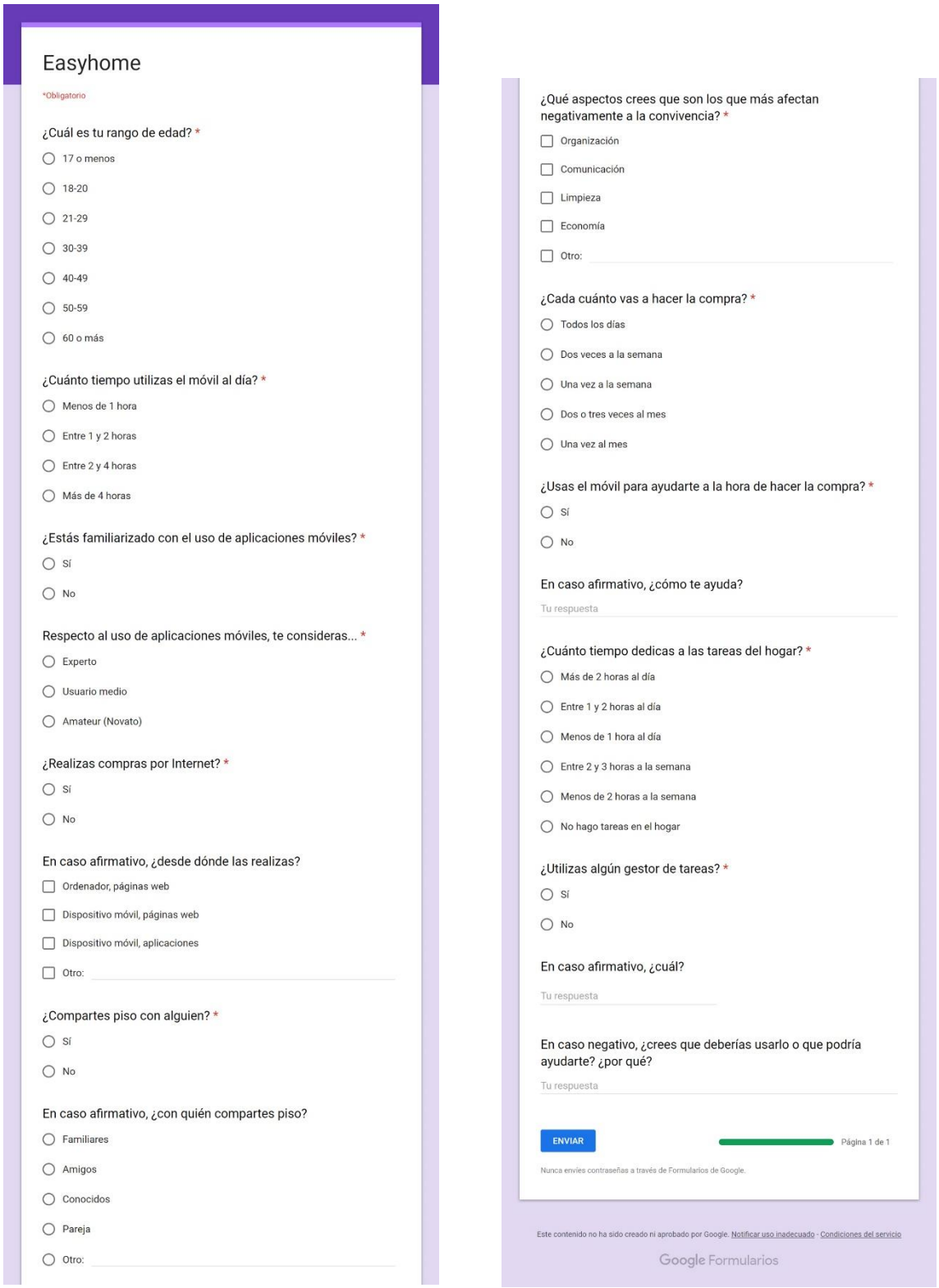


Ilustración 2. Encuesta de Google Forms

Lo primero que se ha podido observar es que el principal rango de edad que ha contestado la encuesta es de entre 21 y 39 años, por lo que los resultados se orientan a personas de este perfil. Esto concuerda también con el público objetivo de la aplicación, que pretende ser gente joven pero mayor de edad. De los resultados se obtiene también que la mayoría de los encuestados utiliza el móvil a diario y que están familiarizados con el uso de aplicaciones, con lo cual la aplicación puede diseñarse para el usuario medio, sin tener que ser demasiado simple. A pesar de que la aplicación podría ser usada por gente que vive sola, se quiere orientar para aquellos que comparten piso, y según los datos de la encuesta, dentro de ese perfil, la mayoría comparten piso con su pareja (68% de los encuestados), mientras que un 24% comparten piso con familiares, siendo este el segundo gran sector. Con esto ya se puede visualizar los dos grandes grupos a los que se va a orientar la aplicación. Un gran dato que se ha obtenido también es que los usuarios opinan que la mayor parte de los problemas de convivencia proceden en primer lugar de la limpieza (un 59,4% de los encuestados), en segundo lugar de la comunicación (un 46,9%) y en tercer lugar de la organización (un 43,8%). Con estos datos se puede ver que los dos grandes pilares de la aplicación tienen que ser mejorar la organización de los usuarios y la comunicación entre ellos (respecto a la limpieza no se puede hacer mucho de momento). También se ha observado que algunas personas utilizan aplicaciones de listados tipo Google Keep para realizar sus compras habituales, por lo que es una referencia a tener en cuenta durante el prototipado.

Por último, se observa que una cantidad importante de usuarios dedica gran tiempo a las tareas del hogar pero, sin embargo, muy pocos utilizan algún tipo de gestor que les ayude a organizar dichas tareas bien porque no conocen ninguno o bien porque no le ven utilidad. Esto puede deberse a que no han encontrado ninguno que pueda ofrecerles una utilidad real, por lo que se intentará ofrecer esta funcionalidad de la forma más sencilla posible en la aplicación.

2.1.3. Perfiles de usuario

Dados los resultados obtenidos, se han generado dos perfiles de usuario:

- Perfil nº1: usuarios que comparten vivienda con su pareja o con conocidos.

  Características: Usuarios entre 20 y 39 años que viven con su pareja o comparten piso con algún conocido. Son usuarios que están familiarizados con el uso de aplicaciones, su grado de experiencia con dispositivos móviles es medio-alto y sí que han usado otras herramientas de gestión en otras ocasiones. La principal motivación de este perfil es la de mejorar la convivencia disminuyendo los problemas de comunicación y de organización en la medida de lo posible.

  Contexto de uso:

  - Principalmente hará uso de la aplicación en casa, en los momentos libres del usuario que pueda dedicarle un momento o cuando se disponga a hacer alguna de las tareas de la aplicación, generalmente por las tardes/noches y/o los fines de semana. En cuanto al entorno, estos

estarán conectados a la red de su hogar, junto con las personas con las que convivan, en un ambiente distendido.

- De vez en cuando hará uso de la aplicación fuera de casa cuando necesite añadir o consultar alguna tarea o algo de las listas y tenga que recurrir a la aplicación. Esto puede ocurrir en cualquier momento del día.
- Cuando el usuario salga a hacer la compra, en el supermercado, estando conectado a una red de datos y en un ambiente principalmente ruidoso y cuyo entorno no permitirá prestar mucha atención al móvil ya que puede chocar con gente.

Análisis de tareas: El usuario deberá crearse una cuenta en la aplicación e iniciar sesión la primera. Deberá vincularse a una casa o crear una nueva. Deberá crear las listas que considere necesario y hacerlas públicas para que las vean el resto de usuarios. Deberá crear notas para mejorar la comunicación con sus compañeros. Deberá crear tareas para mejorar la organización de la casa.

- Perfil nº2: usuarios que comparten vivienda con sus familiares. Por los datos obtenidos en las encuestas, estos usuarios son menos propensos al uso de aplicaciones para la gestión de tareas u organización.

  Características: Usuarios entre 30 y 59 años que viven con su familia, su experiencia con el móvil es nivel bajo-medio y no suelen usarlo muy a menudo, aunque sí que están familiarizados con el uso de aplicaciones. Su principal motivación es la de ahorrar tiempo y autogestionarse mejor. Son personas que a pesar de convivir con más gente, suelen encargarse de la mayor parte de las tareas por sí mismas.

  Contextos de uso:

  - El usuario hará uso de la aplicación sobre todo en casa, cuando tenga tiempo libre para consultarla y en un entorno principalmente familiar, en el cual puede abundar el ruido y las distracciones. El dispositivo estará conectado a la red WiFi del hogar.
  - Cuando el usuario salga a hacer la compra, en el supermercado, estando conectado a una red de datos y en un ambiente principalmente ruidoso y cuyo entorno no permitirá prestar mucha atención al móvil ya que puede chocar con gente.

  Análisis de tareas: El usuario deberá crearse una cuenta en la aplicación e iniciar sesión la primera. Deberá vincularse a una casa o crear una nueva. Deberá crear tareas para mejorar la organización de la casa. Deberá asegurarse de que se cumplen las tareas y mantener los datos actualizados.

2.1.4. Características descubiertas

Gracias al análisis previo, se ha descubierto que los usuarios le dan mucha importancia a los listados, de primeras se pensaba como una opción secundaria,

pero a la vista de los resultados se ha decidido pasar a primer plano para que sea fácil de encontrar para los usuarios.

Debido a que uno de los principales problemas es la comunicación, el tablón de notas donde los usuarios pueden escribir mensajes al resto de usuarios, también pasa a ser una opción fundamental de la aplicación, ya que al igual que la función de listados, se pensaba incluir como un extra con menos protagonismo.

También se ha concluido que la función de gestionar tareas debe ser muy simple y fácil de entender ya que, en general, parece que los usuarios son bastante reticentes a perder el tiempo si tienen que hacer labores del hogar, por tanto se debe plantear la interfaz de esta sección con una cantidad mínima de opciones para que los usuarios enseguida sepan que pueden hacer.

2.2 Diseño conceptual

2.2.1. Escenarios de uso

- Escenario 1

  Perfil de usuario número 1. Un joven que vive con su pareja comprueba su nevera y se da cuenta de que es hora de ir a hacer la compra. Es por la tarde y los supermercados aún están abiertos. Su principal objetivo es comprobar el listado en la aplicación y añadir todo lo que le falta en la nevera. Además, también quiere que su pareja añada a la lista cosas que necesite comprar. Para conseguir esto, comienza comprobando el contenido de la nevera, con el móvil en la mano y la aplicación abierta. Conforme va haciendo repaso de lo que necesitaría tener, lo va apuntando en la lista de la aplicación. Tras esto, le comunica a su pareja que va a ir a hacer la compra y que puede entrar en la aplicación para añadir cosas mientras él se desplaza al supermercado.

  La principal funcionalidad que necesita este usuario es la de crear y editar listas, mediante la cual puede crear una lista publica para que su pareja pueda editarla al mismo tiempo.

- Escenario 2

  El mismo usuario que el escenario anterior se encuentra ya en el supermercado. Su objetivo ahora es el de recorrer el supermercado buscando los productos que ha anotado en la lista de la aplicación hasta tenerlo todo. Por tanto las tareas que desarrolla este usuario en este escenario comienzan con abrir la aplicación y acceder a la lista que ha creado anteriormente. Lo siguiente es comprobar los ítems para tener claro que debe añadir al carro. Tras esto, empieza a coger los productos e ir tachando en la lista. A veces tacha en el mismo momento que mete el producto en el carro y en otras ocasiones coge varias cosas y tacha varios de golpe. Para tachar simplemente marca el *check* del ítem correspondiente, con lo que la aplicación ya lo marca como añadido.

  Al igual que en el escenario anterior, la funcionalidad que necesita es la de listados, para poder consultar la lista creada anteriormente.

- Escenario 3

  La pareja del anterior escenario (perfil de usuario nº1) se da cuenta de que hace tiempo que no hacen limpieza en la casa y, la verdad, está todo desordenado. El usuario se encuentra en su casa, por la tarde tras haber llegado de trabajar. En este momento está solo por lo que nadie le molesta pero tampoco tiene ganas de ponerse a hacer las tareas, por lo que decide que al menos puede comenzar a organizar los próximos días para limpiar y recoger. Por tanto, su principal objetivo es pensar y tener claro que va a hacer y cuando lo va a hacer los siguientes días. Su objetivo secundario es asegurarse de que su pareja tenga claro también cuales van a ser sus tareas. Para conseguir esto, lo primero que hace, una vez abierta la *app*, es acceder a la sección de tareas. Una vez dentro, crea las tareas de una en una, añadiendo la información requerida por la aplicación y necesaria para facilitar el entendimiento de cada tarea. Además, cada vez que crea una nueva, asigna un responsable, a partes iguales entre el usuario actual y su pareja, para asegurarse de que ambos tienen la misma carga de trabajo. Una vez que ha finalizado la creación, comprueba las tareas que ha creado para ver si se ha dejado alguna y ver si realmente ha sido justo con la repartición.

  Las funcionalidades que necesita son las de creación y edición de tareas, visualización de las tareas y asignación de responsable a cada tarea.

- Escenario 4

  Perfil de usuario nº2. Un usuario de unos 48 años que vive con su familia se encuentra en su casa, un fin de semana por la mañana. Es momento de ocuparse de la casa. Quiere saber qué tareas tiene que realizar ese día, por lo que decide utilizar la aplicación en la que ha ido apuntando sus tareas. Su objetivo es realizar todas las tareas que se ha propuesto para ese día y completarlas lo antes posible para poder disfrutar del resto del día. Comienza abriendo la sección de tareas para consultar que cosas tiene pendientes. A pesar de vivir con más gente, el usuario figura como responsable de todas las tareas, aunque sí que le indica al resto de integrantes de la vivienda las tareas que debe realizar. Cada vez que el usuario u otra persona realizan una tarea, se encarga de marcarlo en la aplicación como que ha sido realizada y poco a poco va viendo el avance realizado.

  La principal funcionalidad que necesita este usuario es la visualización de las tareas, tanto en modo lista como los detalles de las mismas para entender correctamente lo que debe hacer. No parece necesitar la función de asignación de tareas a otros usuarios, pero en un futuro le podría ser útil ya que es algo que está haciendo aunque de manera manual (verbal).

2.2.2. Flujos de interacción

En el siguiente diagrama se puede apreciar a bajo nivel el flujo de interacción más básico de la aplicación. Al inicio de la aplicación lo primero que se deberá hacer es crear una cuenta o iniciar sesión si se creó con anterioridad.

Una vez que se ha entrado en la aplicación se deberá tener una casa creada, si no se tiene, se deberá crear.

Tras esto ya se podrá acceder a las funcionalidades principales de la aplicación, como la sección de listas, el tablón o las tareas. También se puede observar en el diagrama como hay ciertas acciones que quedan reservadas a los administradores de la casa.

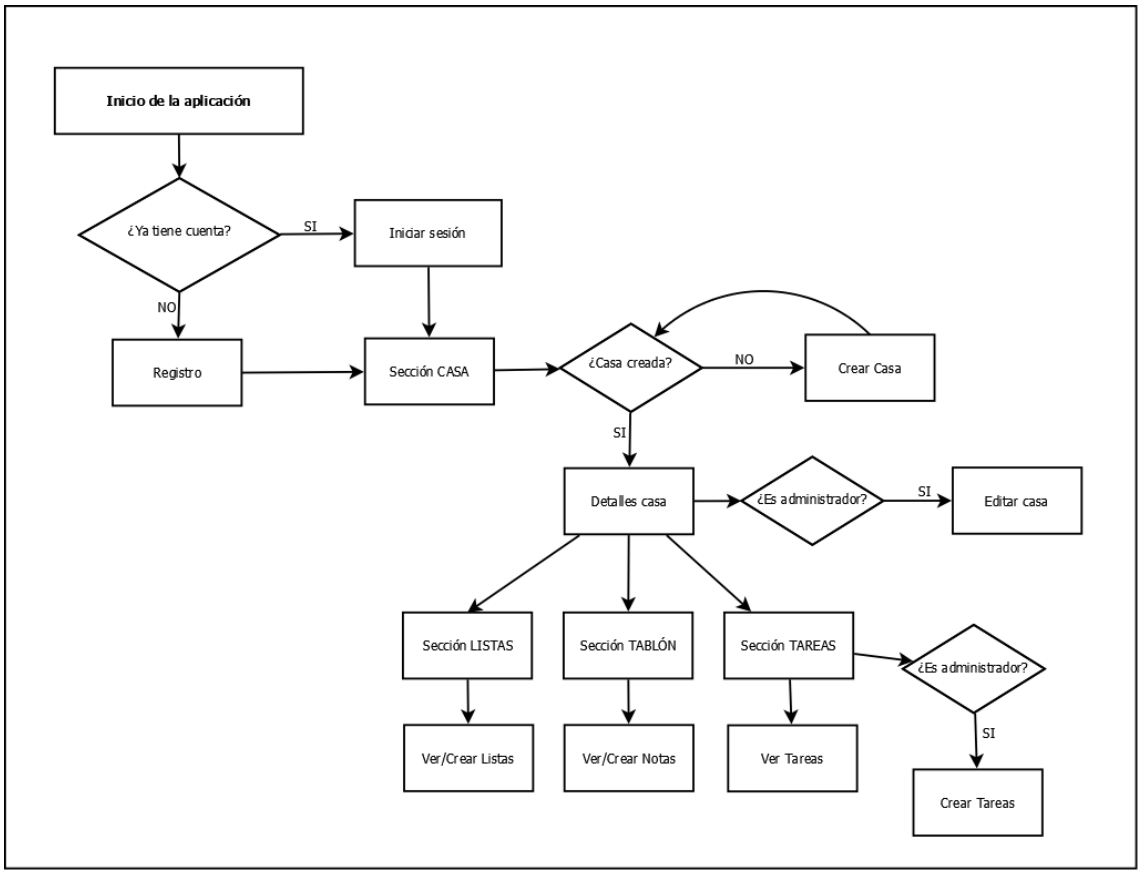


Ilustración 3. Flujo de interacción de la aplicación

2.3 Prototipado

Tras el diseño conceptual de la aplicación y teniendo claras las funcionalidades y necesidades de los usuarios, se ha procedido al diseño de la interfaz de la aplicación. En primer lugar, se han realizado unos bocetos de baja calidad con papel y boli, tras los cuales se ha utilizado la herramienta Justinmind de prototipado para realizar los esquemas de alta fidelidad.

2.3.1. Sketches iniciales

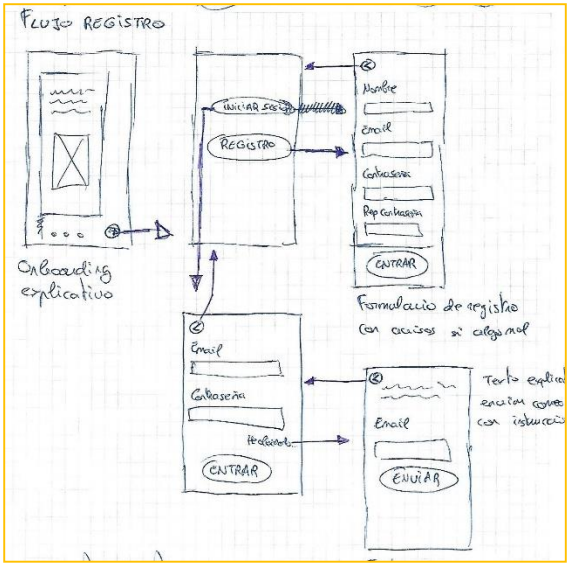


Ilustración 5. Boceto - Registro/Inicio sesión

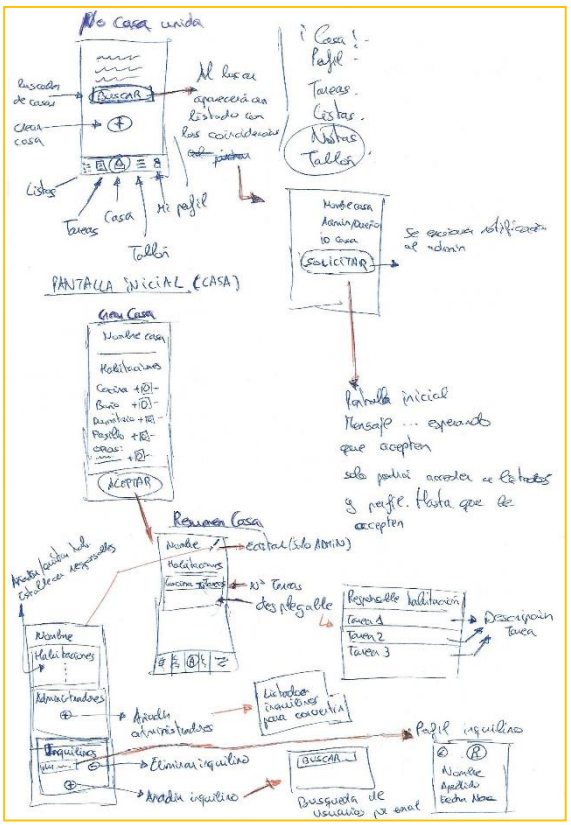


Ilustración 4. Boceto - Pantallas Casa

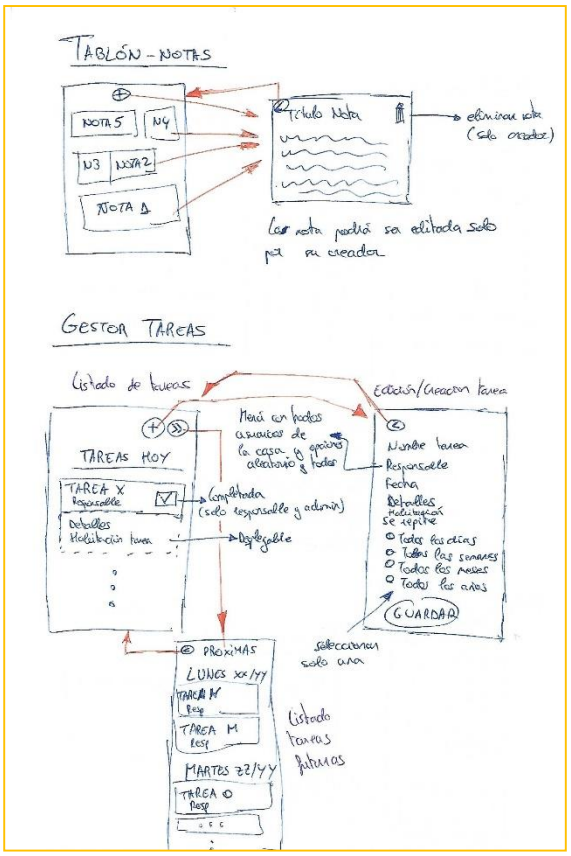


Ilustración 7. Pantallas de notas y tareas

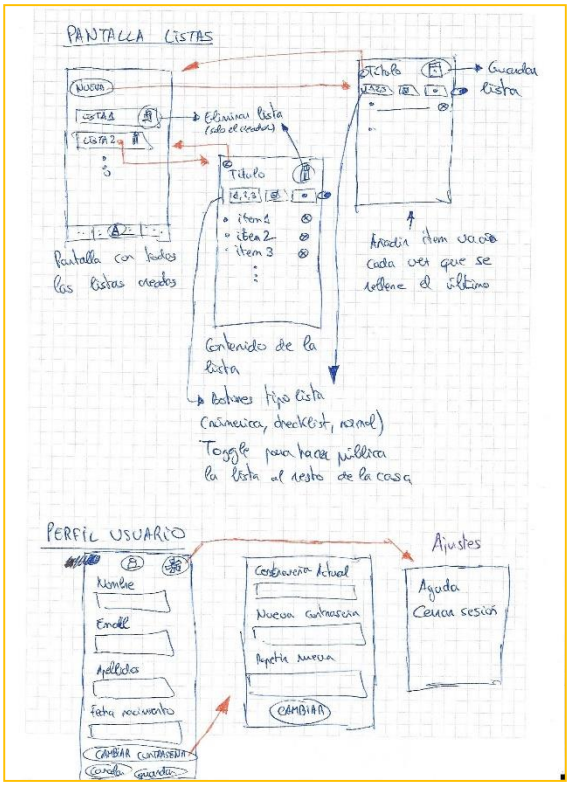


Ilustración 6. Pantallas de listas y perfil

2.3.2. Prototipado horizontal de alta fidelidad

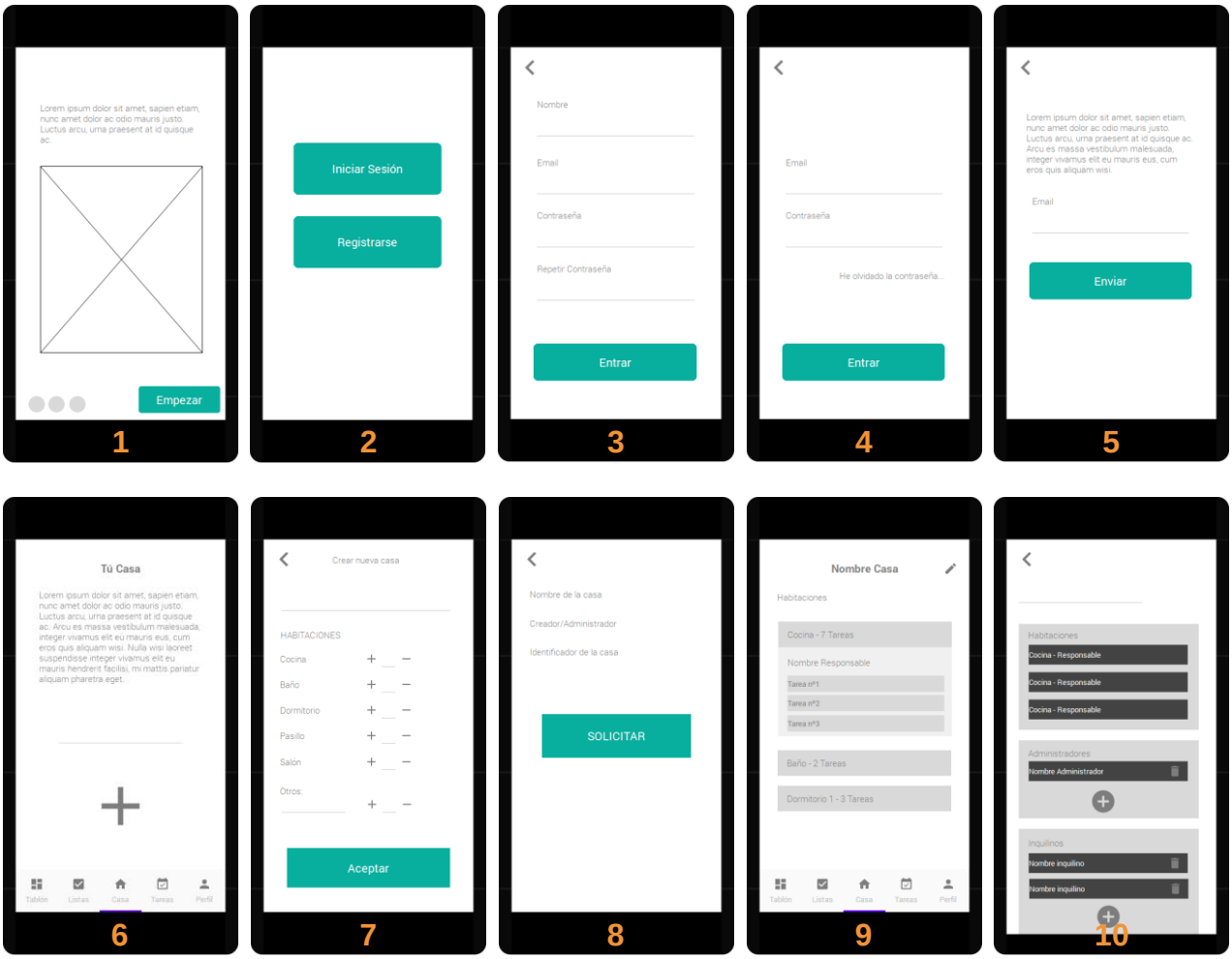


Ilustración 8. Prototipado alta fidelidad - Pantallas 1 a 10

Descripción de las pantallas:

1. Breve *onboarding* descriptivo en el cual se explicará brevemente la aplicación.
2. Pantalla con opciones de inicio de sesión (pantalla 4) y registro (pantalla 3).
3. Pantalla de registro en la cual se solicitará los datos al usuario (email, nombre, contraseña y confirmar contraseña). Al pulsar en el botón, si los datos son correctos, se creará la cuenta y se irá a la sección de casa (pantalla 6).
4. Pantalla de inicio de sesión en la cual el usuario podrá entrar en la aplicación utilizando su correo y contraseña. Para ello tiene que haberse creado una cuenta previamente. Al pulsar el botón se irá a la sección de casa (pantalla 6). Si se pincha sobre el enlace de “He olvidado la contraseña” se irá a la pantalla de recuperar contraseña (pantalla 5).
5. Pantalla de recuperar contraseña. En esta pantalla se le explicará al usuario los pasos a seguir y podrá introducir su correo para que se le envíe un email a dicha dirección para recuperar la contraseña.

6. Pantalla inicial de la sección Casa. En esta pantalla el usuario todavía no se ha unido a ninguna casa y se le ofrecerá la posibilidad de crear una casa nueva o buscar una ya existente. El menú inferior será común para el resto de la aplicación y desde el cual se podrá acceder al resto de secciones de la aplicación.
7. Pantalla de creación de casa, donde se deberá indicar el nombre de la casa y las habitaciones de la misma. Al pulsar en el botón, la casa será creada y el usuario pasará a la pantalla de resumen de la casa (pantalla 9).
8. Pantalla de solicitud de entrada en una casa, en la cual se ven detalles de la casa seleccionada tras haber hecho la búsqueda. Al pulsar el botón se enviará una notificación a los administradores de la casa para que confirmen la solicitud.
9. Pantalla resumen de la casa, donde se verán algunos detalles de la casa como el número de habitaciones y las tareas pendientes de la semana. Si el usuario es administrador dispondrá de la opción de editar la casa (icono en la zona superior derecha).
10. Pantalla de edición de casa, donde se podrán cambiar las habitaciones de la casa, el nombre, los administrados y los inquilinos de la misma.

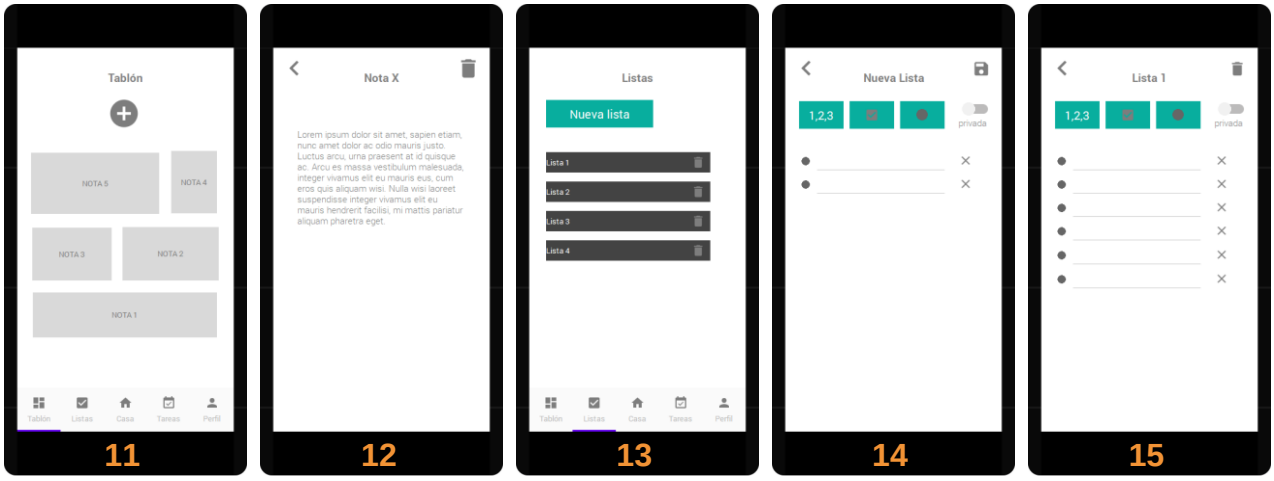


Ilustración 9. Prototipado alta fidelidad - Pantallas 11 a 15

11. Sección tablón, donde se podrán ver las notas creadas por los usuarios de la casa. Si se pulsa el botón “+” se accederá a la sección de creación (pantalla 12). Si se pulsa sobre una nota se accederá a la sección de edición (pantalla 12).
12. Pantalla de creación/edición de notas, donde el usuario podrá añadir un título y contenido a la nota. Si es modo de edición, se podrá eliminar la nota.
13. Sección listados. En esta sección se podrán ver las listas creadas por el usuario y las listas públicas del resto de usuarios de la casa. Pulsando sobre el botón se accederá a la creación de listas (pantalla 14), pulsando sobre una lista, se accederá a la edición de la misma (pantalla 15).

14. Pantalla de creación de listas, donde el usuario podrá añadir un título a la nueva lista, añadir todos los elementos que necesite la lista, indicar si es pública o privada y configurar el tipo de lista.
15. Edición de lista, similar a la pantalla anterior, pero con la opción de eliminar la lista.

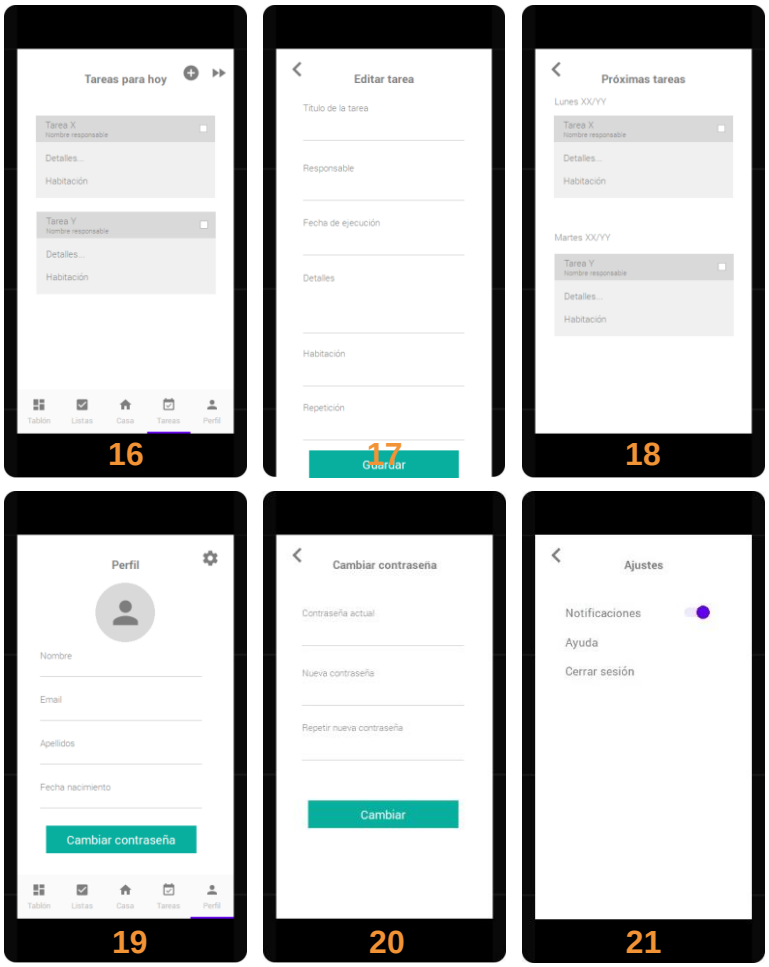


Ilustración 10. Prototipado alta fidelidad - Pantallas 16 a 21

16. Sección de tareas, en la cual se podrán ver las tareas que ha para el día actual, ver sus detalles o indicarlas como hechas. Pulsando sobre el botón “+” se accederá a la pantalla de creación de tareas (pantalla 17). Pulsando sobre el botón “>>” se podrán ver las tareas de los siguientes días.
17. En esta pantalla se podrá crear una tarea, indicando el título de la misma, un responsable, fecha en la que debe hacerse, los detalles, si tiene alguna habitación en la que hacerse y si debe repetirse (diaria, semanal…). Pulsando en el botón, la tarea quedará registrada y le llegará una notificación al responsable.
18. Pantalla siguientes tareas, donde se verán las tareas de los siguientes días, agrupadas por el día en que deben hacerse.

19. Sección perfil de usuario, donde se podrán ver los datos del usuario, nombre, correo, apellidos, fecha de nacimiento. Además se podrá cambiar la contraseña (pantalla 20).
20. Pantalla de cambio de contraseña, en la cual el usuario deberá introducir su contraseña actual, la nueva y confirmar la nueva. Pulsando sobre el botón y si la contraseña actual es correcta, se cambiará la contraseña.
21. Pantalla de ajustes, en la cual se podrá elegir si recibir notificaciones, ver una sección de ayuda o cerrar la sesión, con lo cual se volverá a la página inicial (pantalla 1).

2.3.3. Comentarios

A pesar de que no se haya incluido en los prototipos, todas las acciones destructivas que pueda ejercer el usuario serán seguidas de un *popup* que pregunte si realmente quiere realizar la acción para evitar que el usuario borre algo sin querer. Además, en los formularios se mostrarán mensajes de error en el caso de que el valor introducido en cada campo no sea correcto y se evitará que se puedan guardar los datos si los valores son inválidos.

En cuanto al menú inferior, si el usuario aún no pertenece a una casa, deshabilitará ciertas secciones para evitar que realice acciones no permitidas, como la creación de tareas.

La interfaz se adaptará ligeramente cuando el usuario sea administrador, ya que este cuenta con más opciones, como la edición de casa. Por tanto, los usuarios con rol de inquilino no verán los botones de editar casa o editar/crear tareas, entre otras cosas.

2.4 Evaluación

En esta fase se ha realizado una evaluación sobre el prototipo diseñado en la anterior fase y por tanto comprobar la experiencia de usuario, navegación entre pantallas, colocación de elementos, etc.

Para ellos se ha realizado un test de usuario con varios usuarios, con lo cual, además de conocer la eficacia y facilidad de uso de los prototipos, también se puede evaluar el grado de satisfacción del usuario.

2.4.1. Cuestiones sobre el usuario (pre-test)

Antes de comenzar con el test en sí, se ha realizado un cuestionario para conocer mejor a cada usuario y tener en cuenta el perfil de cada uno a la hora de realizar la recopilación de los resultados:

- Nombre
- Edad
- Nivel económico
- Número de personas en la vivienda
- Profesión
- ¿Desde dónde te conectas a Internet habitualmente?
- ¿Sueles realizar compras desde tu móvil?
- ¿Sueles utilizar aplicaciones móviles?
- ¿Utilizas el móvil para ayudarte en el día a día?
- A la hora de realizar tareas en el hogar, ¿prefieres organizarte las tareas o improvisar sobre la marcha?

2.4.2. Tareas a realizar

Tras el cuestionario pre-test, se ha indicado a los usuarios que ejecuten ciertas acciones sobre el prototipo de la aplicación. Se ha intentado que las tareas sean generales y sin dar demasiadas pistas al usuario de cómo ejecutarlas, por lo que principalmente se va a comprobar el funcionamiento general del prototipo sin entrar en detalles específicos:

1. Imagina que vas a comenzar a compartir piso con más personas y te preocupa la organización de la vivienda ya que es gente nueva que todavía estás conociendo y con la cual apenas tienes confianza. ¿Podrías configurar una nueva casa y añadir a tus nuevos compañeros de piso a esta?
2. Una vez que has completado la anterior tarea quieres seguir con tus labores de coordinación mediante la creación de una lista de la compra común a todos los inquilinos y así intentar que nunca falte de nada en la casa. ¿Podrías crear una lista de la compra y hacer que la vean todos los usuarios?
3. ¿Podrías colgar una nota en el tablón de la aplicación para que el resto de tus compañeros se enteren de que has creado el listado anterior?

4. Tras realizar estas gestiones, te das cuenta de que el papel de coordinador es algo pesado, por lo que decides añadir otro administrador para que pueda gestionar la casa. ¿Podrías realizar los pasos para conseguir esto?
5. Por último, decides programar una limpieza semanal para aseguraros de que la casa mantenga un estado adecuado. ¿Podrías crear una tarea para conseguir esto?

2.4.3. Cuestionario sobre las tareas (post-test)

Una vez finalizadas las tareas, se les ha indicado a los usuarios que rellene un cuestionario final sobre las tareas que han realizado. En este cuestionario hay tanto preguntas cerradas en las que tienen que evaluar del 1 al 10, como preguntas abiertas para que contesten libremente.

Cerradas:

- En rasgos generales, ¿Cómo de fácil es utilizar la aplicación?

(Muy difícil) 1 - 10 (Muy fácil)

- ¿Cómo de intuitiva te ha parecido la navegación?

(Nada intuitiva) 1 - 10 (Muy intuitiva)

- ¿En qué medida has podido completar tus tareas?

(Sin completar) 1 - 10 (Al completo)

- ¿Cuánto te ha costado encontrar los contenidos que deseabas?

(Lentamente) 1 - 10 (Rápidamente)

- ¿Cómo de útil te ha parecido la aplicación?

(Nada útil) 1 - 10 (Muy útil)

- Gráficamente, la aplicación te ha ayudado a completar las tareas

(Poca ayuda) 1 - 10 (Gran ayuda)

Abiertas:

- ¿Qué es lo que más te ha gustado de la aplicación?
- ¿Qué es lo que menos te ha gustado de la aplicación?
- Describe brevemente qué destacarías de la navegación de la aplicación
- ¿Te han parecido pesados los pasos para conseguir alguna tarea?
- ¿Hay algo que no te guste del diseño?
- ¿El contenido es fácil de encontrar y de entender?
- ¿Echas algo en falta?
- Después de usar la aplicación, ¿la usarías en tu día a día?
- ¿Crees que la aplicación puede ser útil a la gente que convive o necesita alguna funcionalidad extra?

3. Apartado técnico

3.1 Casos de uso

Se diferencian dos actores principales: el usuario que tiene el rol de administrador y el usuario que tiene el rol de inquilino. Ya que ambos son usuarios de la aplicación heredan parte de las funcionalidades de un actor padre llamado “usuario”, que es la persona que interacciona con la aplicación. Por tanto un “usuario” puede ser tanto un administrador como un inquilino.

Los casos de uso definidos son los siguientes:

Identificador	CU-001
Nombre	Inicio de sesión
Prioridad	Alta
Descripción	Un usuario quiere iniciar sesión en la aplicación para comenzar a usarla
Actores	Usuario
Precondiciones	El usuario ya se registró y tiene un correo y contraseña
Flujo	Paso 1: Al iniciar la aplicación se selecciona la opción de iniciar sesión Paso 2: El usuario rellena los campos requeridos (email y contraseña) Paso 3: El usuario pulsa sobre el botón “Entrar” Paso 4: El sistema comprueba la información y devuelve una respuesta al usuario Paso 5: La aplicación carga los datos del usuario
Poscondiciones	El usuario inicia sesión y comienza a ver sus datos en la aplicación
Notas	El usuario no iniciará la sesión si introduce erróneamente sus datos

Identificador	CU-002
Nombre	Registro de usuario
Prioridad	Alta
Descripción	Un usuario quiere registrarse en la aplicación para comenzar a usarla
Actores	Usuario
Precondiciones	El usuario no tiene una cuenta creada todavía
Flujo	Paso 1: Al iniciar la aplicación se selecciona la opción de registro Paso 2: El usuario rellena los campos requeridos (nombre, email, contraseña y repetir contraseña) Paso 3: El usuario pulsa sobre el botón “Entrar” Paso 4: El sistema comprueba la información y devuelve una respuesta al usuario Paso 5: La aplicación muestra una página sin datos de usuario
Poscondiciones	El usuario rellena los datos y el sistema los registra
Notas	

Identificador	CU-003
Nombre	Recuperar contraseña
Prioridad	Media
Descripción	Un usuario quiere recuperar la contraseña de la aplicación ya que no se acuerda de ella
Actores	Usuario
Precondiciones	El usuario ya se registró y tiene un correo almacenado en el sistema, pero no ha iniciado sesión en el sistema
Flujo	Paso 1: Al iniciar la aplicación se selecciona la opción de iniciar sesión Paso 2: En la pantalla de iniciar sesión se pulsa sobre “he olvidado la contraseña” Paso 3: El usuario rellena el formulario con su correo electrónico y pulsa sobre “Enviar” Paso 4: El sistema envía un correo a la dirección proporcionada con instrucciones para la recuperación de contraseña
Poscondiciones	El usuario recibe en su correo electrónico personal instrucciones para reiniciar su contraseña
Notas	El correo con instrucciones tendrá caducidad

Identificador	CU-004
Nombre	Crear casa
Prioridad	Alta
Descripción	Un usuario quiere crear una casa en la aplicación
Actores	Usuario
Precondiciones	El usuario ha iniciado sesión correctamente con su email y contraseña y no está vinculado a ninguna casa todavía
Flujo	Paso 1: El usuario pulsa sobre el botón de crear casa nueva Paso 2: El usuario rellena la información requerida del formulario de creación Paso 3: El usuario pulsa sobre el botón de “Guardar” Paso 4: El sistema recoge y almacena la información proporcionada por el usuario Paso 5: La aplicación muestra los datos de la nueva casa creada
Poscondiciones	El usuario ha rellenado los datos para la creación de la casa y el sistema los ha registrado
Notas	Al crear la casa, este usuario quedará registrado como creador y administrador de la misma

Identificador	CU-005
Nombre	Ver usuarios
Prioridad	Baja
Descripción	El usuario quiere ver un listado de los usuarios de la casa
Actores	Usuario
Precondiciones	El usuario está unido a una casa y ha iniciado sesión
Flujo	Paso 1: El usuario va a la pantalla de “Casa” Paso 2: El usuario pulsa sobre el botón de editar Paso 3: La aplicación le muestra los detalles de la casa, entre ellos un listado de usuarios
Poscondiciones	El usuario ve el listado de usuarios
Notas	

Identificador	CU-006
Nombre	Crear tarea
Prioridad	Alta
Descripción	El administrador quiere crear una tarea nueva
Actores	Administrador
Precondiciones	El administrador está unido a una casa y ha iniciado sesión

Flujo	Paso 1: El administrador va a la pantalla de “Tareas” Paso 2: El administrador pulsa sobre el botón de crear nueva tarea Paso 3: La aplicación le muestra al administrador un formulario con los datos requeridos para crear una nueva tarea Paso 4: El administrador rellena el formulario y pulsa sobre el botón de “Guardar” Paso 5: El sistema comprueba la información y la almacena Paso 6: La aplicación devuelve al usuario a la pantalla anterior
Poscondiciones	El usuario rellena los datos requeridos para crear una tarea y el sistema los registra correctamente
Notas	

Identificador	CU-007
Nombre	Cambiar responsable tarea
Prioridad	Media
Descripción	Un administrador quiere cambiar el responsable de una tarea y asignársela a otro usuario
Actores	Administrador
Precondiciones	El administrador está unido a una casa, ha iniciado sesión y existen tareas asociadas a dicha casa
Flujo	Paso 1: El administrador accede a la sección de “Tareas” Paso 2: El administrador selecciona una tarea Paso 3: La aplicación le muestra al administrador los detalles de la tarea Paso 4: El administrador edita el campo de responsable y pulsa sobre el botón de guardar Paso 5: El sistema almacena el cambio y devuelve al administrador a la pantalla anterior
Poscondiciones	El administrador edita la tarea y cambia el responsable asignado por otro. El sistema registra correctamente el cambio
Notas	El responsable de la tarea puede ser cualquier usuario que esté vinculado con esa casa

Identificador	CU-008
Nombre	Crear nueva lista
Prioridad	Alta
Descripción	Un usuario quiere crear una lista nueva

Actores	Usuario
Precondiciones	El usuario ha iniciado sesión
Flujo	Paso 1: El usuario accede a la sección de “Listas” Paso 2: El usuario pulsa sobre el botón de crear nueva lista Paso 3: La aplicación le muestra al usuario un formulario con el título y el contenido de la nueva lista Paso 4: El usuario rellena el formulario y pulsa sobre el botón de guardar Paso 5: El sistema registra los datos enviados y devuelve al usuario a la pantalla anterior, donde aparecerá la nueva lista creada
Poscondiciones	El usuario crea una lista y añade algunos elementos a esta. El sistema registra los datos correctamente.
Notas	Si el usuario indica que es pública, el resto de usuarios de la casa podrán ver la lista

Identificador	CU-009
Nombre	Eliminar lista
Prioridad	Media
Descripción	Un usuario quiere eliminar una lista
Actores	Usuario
Precondiciones	El usuario ha iniciado sesión y ha creado alguna lista
Flujo	Paso 1: El usuario accede a la sección de “Listas” Paso 2: El usuario pulsa sobre el botón de eliminar de la lista que desea borrar Paso 3: El sistema confirma la acción, elimina la lista y el usuario no podrá verla
Poscondiciones	El usuario elimina la lista
Notas	El usuario solo podrá eliminar listas de las cuales sea el creador

Identificador	CU-010
Nombre	Crear nueva nota
Prioridad	Alta
Descripción	Un usuario quiere crear una nota nueva
Actores	Usuario
Precondiciones	El usuario está unido a una casa y ha iniciado sesión
Flujo	Paso 1: El usuario accede a la sección “Tablón” Paso 2: El usuario pulsa en el botón de crear nueva nota

	Paso 3: La aplicación le muestra un formulario con el título y el contenido de la nueva nota Paso 4: El usuario rellena los datos requeridos y pulsa en el botón de “Guardar” Paso 5: El sistema registra los datos y devuelve al usuario a la pantalla anterior, donde aparecerá la nueva nota creada
Poscondiciones	El usuario crea la nota y añade algo de contenido. El sistema registra los datos correctamente
Notas	

Identificador	CU-011
Nombre	Eliminar nota
Prioridad	Media
Descripción	Un usuario quiere eliminar una nota
Actores	Usuario
Precondiciones	El usuario está unido a una casa, ha iniciado sesión y ha creado alguna nota
Flujo	Paso 1: El usuario accede a la sección de “Tablón” Paso 2: El usuario pulsa sobre la nota que desea eliminar Paso 3: La aplicación muestra el contenido de la nota Paso 4: El usuario pulsa sobre el botón de eliminar de la nota Paso 5: El sistema confirma la acción, elimina la nota y ningún usuario podrá volver a verla
Poscondiciones	El usuario elimina la nota
Notas	El usuario solo podrá eliminar notas de las cuales sea el creador

Identificador	CU-012
Nombre	Editar perfil
Prioridad	Media
Descripción	Un usuario quiere modificar los datos de su perfil
Actores	Usuario
Precondiciones	El usuario ha iniciado sesión en la aplicación con su correo y contraseña
Flujo	Paso 1: El usuario accede a la sección “Perfil” Paso 2: La aplicación muestra los datos del usuario Paso 3: El usuario edita los campos que le interesen y pulsa sobre el botón de “Guardar” Paso 4: El sistema registra los cambios

Poscondiciones	El usuario edita los campos del formulario del perfil. El sistema registra los cambios correctamente.
Notas	Los cambios solo se almacenarán si no hay errores en el formulario como dejar el nombre o el email vacíos.

Identificador	CU-013
Nombre	Desvincular usuario
Prioridad	Baja
Descripción	Un administrador quiere desvincular a un usuario que pertenece a su casa
Actores	Administrador
Precondiciones	El administrador pertenece a una casa, ha iniciado sesión y hay más usuarios enlazados a dicha casa
Flujo	Paso 1: El administrador accede a la sección “Casa” Paso 2: El administrador pulsa sobre el botón de editar casa Paso 3: La aplicación muestra los detalles de la casa, entre ellos el listado de usuario vinculados Paso 4: El administrador pulsa sobre el botón de eliminar del usuario que desea eliminar Paso 5: El sistema confirma la acción y desvincula al usuario de esa casa Paso 6: El sistema recarga la lista de usuarios mostrando el listado sin el usuario eliminado
Poscondiciones	El administrador accede a la lista de usuarios de la casa y elimina al que le interesa
Notas	No podrá eliminar al creador de la casa, pero si a otros administradores

Identificador	CU-014
Nombre	Cambiar contraseña
Prioridad	Media
Descripción	Un usuario quiere cambiar su contraseña
Actores	Usuario
Precondiciones	El usuario ha iniciado sesión en la aplicación con su correo y contraseña
Flujo	Paso 1: El usuario accede a la sección “Perfil” Paso 2: La aplicación muestra los datos del usuario Paso 3: El usuario pulsa en el botón de ajustes Paso 4: El sistema muestra la sección de ajustes Paso 5: El usuario pulsa sobre la opción de “Cambiar contraseña”

	Paso 6: El sistema solicita al usuario la contraseña actual, una contraseña nueva y confirmación de la misma. Paso 7: El usuario introduce los datos y pulsa en el botón de “Guardar” Paso 8: El sistema comprueba si los datos son correctos, en cuyo caso cambia la contraseña del usuario
Poscondiciones	El usuario ha rellenado la información solicitada correctamente y se ha actualizado su contraseña
Notas	Si la contraseña actual introducida no es válida, el sistema no hará ningún cambio en el usuario

3.2 Modelo de datos

El esquema de la base de datos que se ha realizado para el desarrollo de la aplicación es el siguiente:

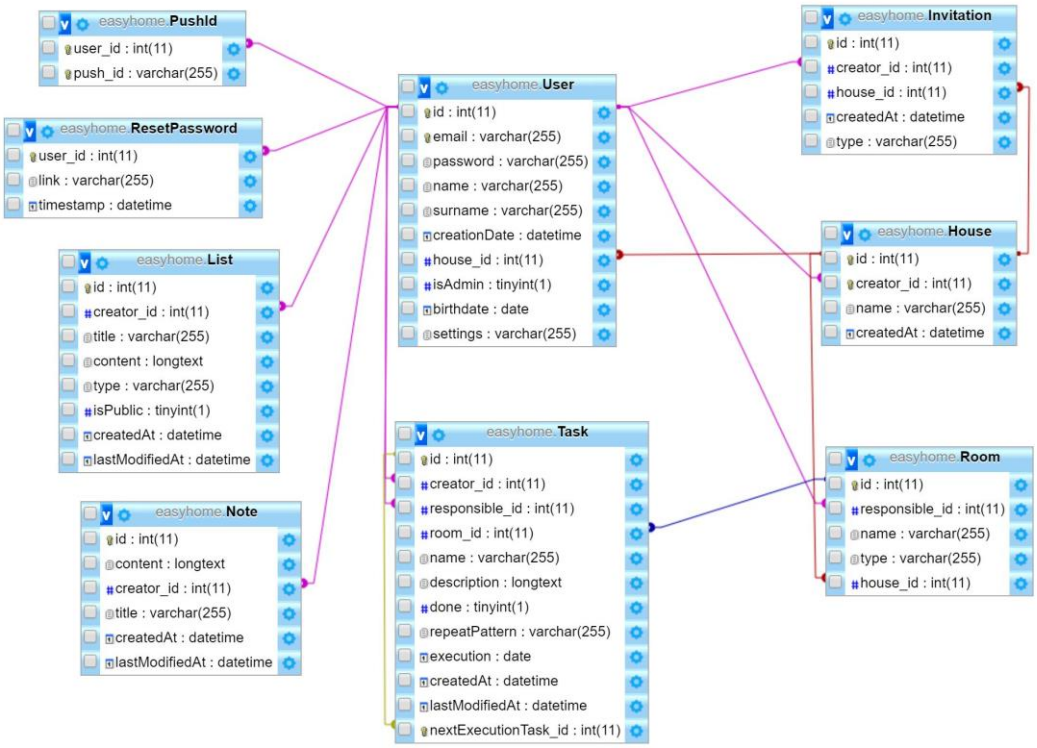


Ilustración 11. Esquema base de datos

En el *backend* se va a hacer uso de un ORM (Object Relational Mapper) llamado Doctrine, el cual se encargará de asociar cada una de las tablas de la base de datos a una clase (de PHP en este caso). Por tanto, las clases utilizadas tienen una correspondencia directa con las entidades mostradas en el esquema de la base de datos.

3.3 Arquitectura del sistema

La arquitectura seguida para el sistema es de Cliente-Servidor por lo que se pueden extraer dos grandes bloques:

- Servidor: Este componente del sistema se divide a su vez en otros dos, el *backend* y una página web.
    - *Backend*: Esta parte del sistema es la encargada del procesamiento, la gestión y el almacenamiento de los datos generados por la aplicación. Para ello, se utiliza una base de datos MySQL. Para conseguir esto se expone un API REST que permite a la aplicación conectarse y realizar peticiones con los datos que necesita para su correcto funcionamiento. Además, se ha separado el Modelo del Controlador, ya que en este caso los modelos son clases relacionadas directamente con las entidades de la base de datos.
    - Página Web: Esta es una web muy simple desplegada en el mismo servidor que permitirá a los usuarios recuperar su contraseña. Mediante un *token* único la web identificará al usuario y le pedirá que introduzca la nueva contraseña. El desarrollo de esta web se ha realizado con Angular.
- Aplicación: El principal componente del sistema y encargada de toda la interacción con el usuario y la recopilación de la información necesaria para cumplir con los requisitos del trabajo. Se conecta al *backend* mediante el API para obtener los datos requeridos, por tanto el procesamiento de información en este parte del sistema es mínimo.

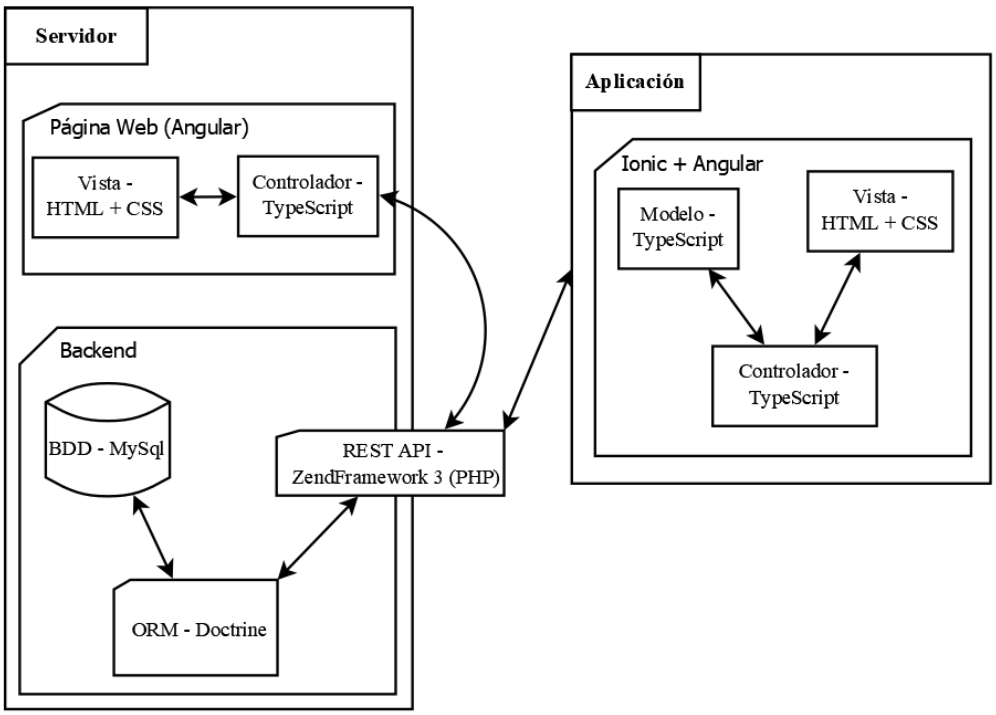


Ilustración 12. Arquitectura global del sistema

3.3.1. Arquitectura de la aplicación

Esta es una aplicación web multiplataforma desarrollada con Ionic que luego se ha compilado para ser ejecutada en dispositivos Android.
El patrón de diseño utilizado es MVC (Modelo Vista Controlador) ya que es un patrón bastante sencillo de implementar con Ionic y Angular.

- Modelo: Mediante el uso de *Injectables* (Angular) se han generado servicios que componen la capa de modelo. Estos servicios se conectan al *backend* para obtener los datos necesarios y realizan un procesamiento mínimo para el correcto entendimiento de la aplicación**.**
- Vista: Esta capa se ha desarrollado con HTML y SCSS, usando los datos provistos por el controlador y se encarga de presentar la información al usuario y de transmitir los eventos derivados de la interacción del usuario con la aplicación al controlador.
- Controladores: Usando *Components* (Angular), se ha creado la lógica de la aplicación, relacionando un componente con una única vista, aunque cada controlador puede obtener datos de diversos modelos. Cuando un controlador solicita información sobre un modelo, el modelo (servicio) realiza una petición al *backend* para obtener la información actualizada y la devuelve al controlador.

Gracias a esta arquitectura, se ha podido modificar la interfaz de usuario libremente sin necesidad de modificar la lógica de la aplicación ni el *backend.*

4. Desarrollo

4.1 Herramientas y tecnologías

Tal y como se ha mencionado en los capítulos anteriores, las principales tecnologías empleadas en el desarrollo de esta aplicación son Ionic, Angular y Zend Framework.

4.1.1. Aplicación

El desarrollo de la aplicación se ha realizado enteramente con Ionic, una plataforma basada en Javascript que permite el desarrollo de aplicaciones móviles mediante el uso de tecnologías web. En este trabajo se ha utilizado la última versión de esta herramienta (v4) la cual permite la selección del *framework* a usar o directamente no usar ninguno. Por facilidad y conocimientos de uso se ha integrado Angular v7 en Ionic. Esta versión por tanto hace uso de TypeScript, un superconjunto de Javascript que añade más funciones al lenguaje como el tipado de variables.

Para la correcta utilización de esta plataforma es necesario instalar en primer lugar NodeJS y su gestor de paquetes NPM.

El *frontend*, por tanto, se ha desarrollado enteramente usando Ionic + Angular y para hacer la conversión a app móvil se ha utilizado Cordova, una herramienta que permite envolver aplicaciones desarrolladas con tecnologías web para que sean capaces de ser ejecutadas en otras plataformas (Android, iOS, etc.). La elección de esta herramienta, más allá de los conocimientos previos que se tenía, también se ha debido a que es una herramienta potente y flexible, con la cual se pueden desarrollar fácilmente aplicaciones híbridas multiplataforma usando un lenguaje muy extendido. Además, al estar integrado con Angular, la documentación, la comunidad y las librerías disponibles son bastante amplias. En cuanto a *plugins* utilizados, principalmente se ha usado ngx-translate para gestionar traducciones en un futuro y ngx-restangular, el cual facilita las peticiones REST al API:

- ngx-translate: Mediante esta librería, se pueden cargar varios ficheros .json con texto en varios idiomas. El formato del archivo es { ‘cadena original’: ’cadena traducida’,…}. La cadena original tiene que ser la misma que se haya empleado en el código. Al inicio de la aplicación, este *plugin* carga el fichero json del lenguaje que se haya indicado para poder devolver las traducciones que sean necesarias. Para ello se puede utilizar un filtro de Angular para obtener las traducciones en la parte de la vista (HTML) o utilizar el método que ofrece para TypeScript.
- ngx-restangular: Este *plugin* permite realizar peticiones al *backend* de una manera sencilla, pudiendo usar los métodos básicos de HTTP (GET, POST, PUT, DELETE,…), pero además convirtiendo los objetos que recibe de las peticiones en objetos *restangularizados*, lo cual significa que cuando el *backend* envía un objeto, en el *frontend* se puede llamar a un método put o delete sobre dicho objeto y la librería se encarga de realizar la petición correspondiente al *backend*.

Además, se ha utilizado el servicio OneSignal y su *plugin* de Ionic para la gestión de notificaciones. Este servicio es multiplataforma y gratuito, además de ser sencillo de usar. Una vez integrado en la aplicación, se pueden realizar peticiones a su API REST para enviar las notificaciones deseadas. Por ejemplo, cuando un usuario crea una nota nueva y el *backend* recibe esta petición, a su vez manda una petición al servicio de OneSignal para que envíe una notificación a todos los usuarios que tengan activas las notificaciones y pertenezcan a la misma casa que el usuario que ha creado la nota.

La organización básica de la aplicación se basa en servicios, componentes y plantillas.

Los servicios son clases cuya funcionalidad es la de estar conectada con los componentes que las necesiten y gestionar los datos, esta sería la capa de Modelo, ya que es gracias a estos servicios que cualquier componente puede acceder a la información que ofrece. Estas clases se conectan al *backend* para recuperar los datos necesarios y cada clase se encarga de gestionar su propia información, por tanto hay un servicio para los usuarios, otro para la casa, otro para las tareas…

Los componentes son las clases encargadas de la lógica de la aplicación, desde donde se realiza el procesado de los datos antes de ser mostrado a los usuarios, se gestionan las entradas de los usuarios, etc. En este caso se ha asociado un componente para cada página de la aplicación, por tanto esta está altamente modularizada, y cada componente se ocupa únicamente de su propia página.

La vista se compone de HTML para la estructura y SCSS para el estilo de cada pantalla. En este caso también hay una plantilla para cada página creada y se relacionan directamente con el componente asociado a dicha pantalla.

4.1.2. Backend

- Zend Framework: Este es un *framework* para PHP que permite la creación de páginas web y la configuración de diferentes APIs. Se ha utilizado la última versión (v3) y, a pesar de que cuenta con capacidades para desarrollar páginas en el *frontend*, solo se ha utilizado para recibir peticiones de *backend*. Esto incluye la comunicación con la base de datos, el middleware encargado de la autorización y autenticación de usuarios o la gestión de datos.

  Para ejecutar este *framework* se ha desplegado en un servidor con un Apache. La comunicación con la base de datos se realiza mediante Doctrine, un ORM que permite la abstraer esta comunicación a nivel de objetos, con lo cual se tiene un objeto por cada tabla de la base de datos.

  El principal motivo para la elección de este *framework* es el conocimiento sobre el mismo. Esto ha permitido que el *endpoint* se desarrolle con algo más de rapidez. Aunque también hay muchos *plugins* que se pueden utilizar (como el ORM o un servidor OAuth2) y le dan mucha versatilidad por lo que es una buena elección. Los principales *plugins* utilizados en el *backend* son los mencionados previamente, Doctrine ORM para la gestión de la conexión con la base de datos y ZF-OAuth2 para contar con una capa de seguridad basada en OAuth2.

Esta capa de seguridad comprueba que las peticiones tengan un *token* de acceso y la validez del mismo, por tanto, sin este *token* las peticiones serán denegadas.

Los principales *endpoint* que se exponen en el api REST son:

Endpoint	**Operaciones**
/auth	POST
/houses	GET, POST
/houses/{houseId}	GET, PUT, DELETE
/houses/{houseId}/notes	GET, POST
/houses/{houseId}/notes/{noteId}	GET, PUT, DELETE
/houses/{houseId}/rooms	GET, POST
/houses/{houseId}/rooms/{roomId}	GET, PUT, DELETE
/houses/{houseId}/tasks	GET, POST
/houses/{houseId}/tasks/{taskId}	PUT, DELETE
/invitations	GET, POST
/invitations/{invitationId}	GET, PUT, DELETE
/users	GET, POST
/users/{userId}	GET, PUT, DELETE
/users/{userId}/lists	GET, POST
/users/{userId}/lists/{listId}	GET, PUT, DELETE
/users/{userId}/pushid	POST
/users/{userId}/pushid/{pushId}	DELETE

Se puede acceder a estos endpoints desde https://easyhome-aws.tk/api/

4.1.3. Web

La web se ha realizado con Angular, sin incluir ningún *plugin* ni librería adicional ya que esta es muy sencilla. Simplemente se ha creado una aplicación nueva con la consola de Angular y se han creado dos páginas, una principal y la de gestión de recuperar contraseña.

Esta última tiene un parámetro *token* obligatorio y antes de dejar continuar con el proceso al usuario, comprobará con el *backend* la validez del mismo. Solo en el caso de que este devuelva que es correcto, se podrá realizar el cambio de contraseña.

La dirección de acceso a esta web es https://easyhome-aws.tk/

4.1.4. Herramientas

- Edición de código: Para el desarrollo del código se han usado dos editores, Visual Studio Code para el *frontend* y Sublime Text 3 para el *backend*. Cada

editor tiene instalados *plugins* para facilitar el desarrollo de cada una de las partes. La elección de dos editores diferentes (aunque con muchas funcionalidades similares) se debe a que es más fácil distinguir dónde y cómo se está escribiendo cada parte visualmente.

- Apicurio: Herramienta para el diseño de APIs. Se ha utilizado esta herramienta para diseñar las peticiones que se iban a gestionar en el *backend* antes de empezar a programarlo. Esto ha permitido tener una estructura más clara a la hora de gestionar el API.
- Servidor AWS: Para poder tener el *backend* accesible desde cualquier lugar, se ha generado una instancia EC2 en los servidores de Amazon. En esta máquina se ha desplegado un sistema CentOS 7 configurado con PHP, MySQL y Apache.
- MySQL Workbench: *Software* de gestión de bases de datos, con el cual se han gestionado los datos almacenados tanto en la base de datos local como la del servidor remoto.
- Apache: *Software* que permite servir contenido vía web, por tanto es el encargado de exponer los servicios del API y la web de recuperación de contraseña. Se ha configurado de tal manera que la web principal sea la web hecha con Angular y el api se exponga en /api.
- WinSCP: Herramienta para la conexión vía scp a un servidor y realizar gestión archivos y transferencias. Se ha utilizado principalmente para subir el *backend* al servidor de AWS.

4.2 Pruebas

Se han realizado dos tipos de pruebas sobre la aplicación, pruebas de funcionamiento con usuarios y pruebas unitarias sobre el código.

4.2.1. Pruebas con usuarios

Con las pruebas de usuarios se quiere encontrar posibles fallos junto con alguna mejora en la UX diseñada. Para ello se ha hecho uso del cuestionario de evaluación planteado en el apartado 2. Estas pruebas se han realizado sobre dispositivos reales, instalando la aplicación en los dispositivos de los usuarios y a los cuales se les pidió primero que ejecutasen las tareas indicadas y luego que fuesen utilizando libremente la aplicación, mientras tanto anotaban sus hallazgos.

Algunos puntos clave a destacar son:

- Los usuarios se dieron cuenta de que había ciertos fallos de coherencia entre las pantallas.
- Ciertos elementos pulsables de la aplicación no lo parecían, por lo que los usuarios no sabían continuar con ciertas acciones.
- Errores en los roles, en ocasiones los inquilinos podían acceder a funcionalidades de los administradores
- Varios usuarios destacaron la necesidad de que el teclado usase automáticamente las mayúsculas al empezar a escribir en ciertos campos, como los títulos o los nombres.

Todos estos errores y mejoras se corrigieron en la aplicación, pero no se han actualizado los prototipos iniciales debido a la falta de tiempo.

4.2.2. Pruebas unitarias de código

Respecto a las pruebas unitarias, Ionic y Angular integran herramientas como Jasmine y Karma que facilitan este tipo de pruebas. Mediante Karma se han ejecutado los test desarrollados mientras que con Jasmine se facilita la escritura de los mismos, usando una estructura tipo *describe, it, expect*. Primero se describe la tarea que se va a ejecutar (*describe*), luego se le indica lo que debería hacer (*it*) y por último el resultado esperado de ejecutar la operación (*expect*).

Las pruebas unitarias se han realizado en la propia máquina de desarrollo, ya que al ser tecnologías web, se hace uso de un navegador web para ejecutarlas.

Los test realizados cubren sobre todo la correcta creación de los componentes que usa la aplicación y su adecuada inicialización, para asegurarse que no se comienza teniendo ningún valor erróneo ni invalido.

El resultado de la ejecución de estos test es el siguiente:

Ilustración 13. Resultado de los tests unitarios

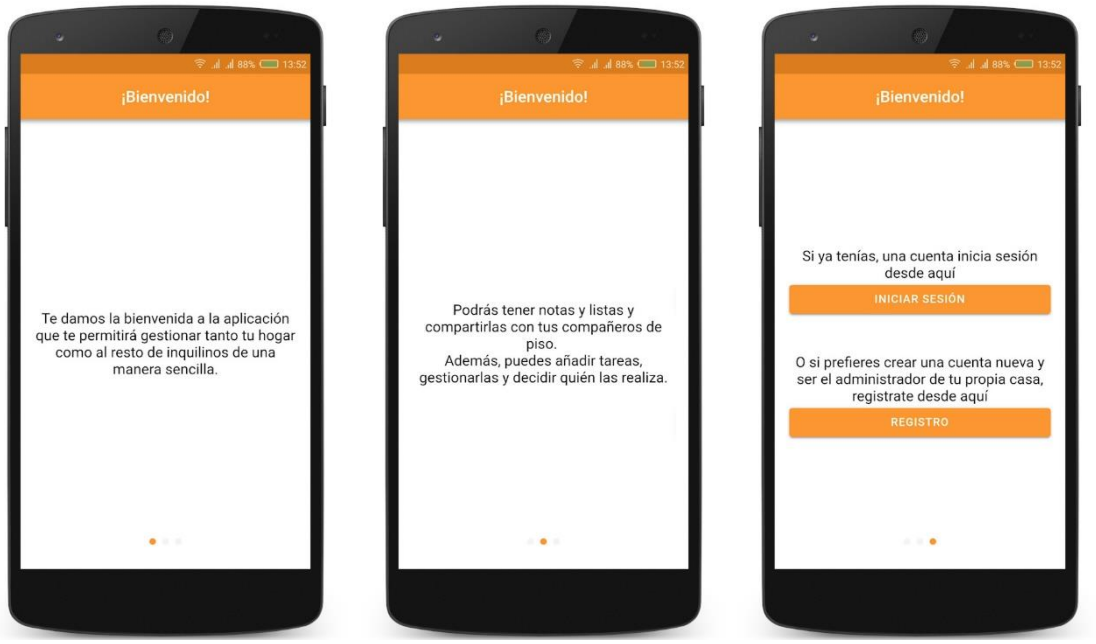
Además se ha realizado un análisis de la cobertura de código conseguida con estos test, y se ha obtenido que un 50% de las líneas de código y de las declaraciones están cubiertas por los test.

```
============================== Coverage summary ==============================
Statements   : 48.27% ( 920/1906 )
Branches     : 15.67% ( 47/300 )
Functions    : 29.14% ( 190/652 )
Lines        : 49.32% ( 795/1612 )
================================================================================
TOTAL: 65 SUCCESS
TOTAL: 65 SUCCESS
```

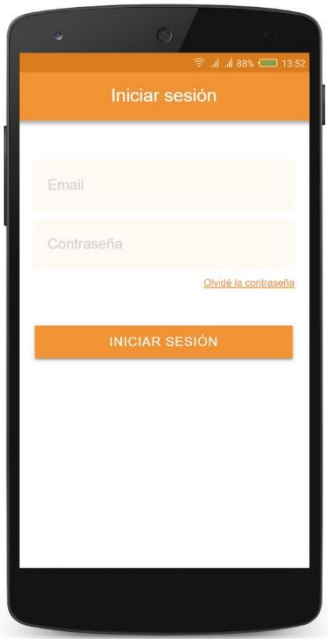
Ilustración 14. Resumen de la cobertura de código

4.3 Ejemplo de funcionamiento

4.3.1 Pantallas iniciales



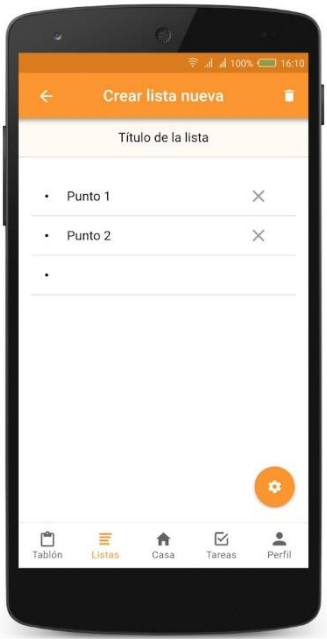
4.3.2 Inicio de sesión y registro



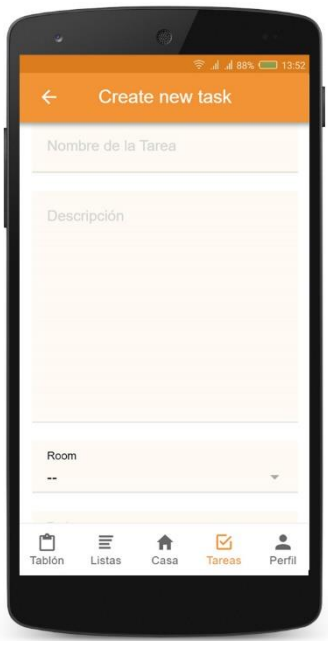
4.3.3 Pantalla inicial de casa

4.3.4 Sección Tablón

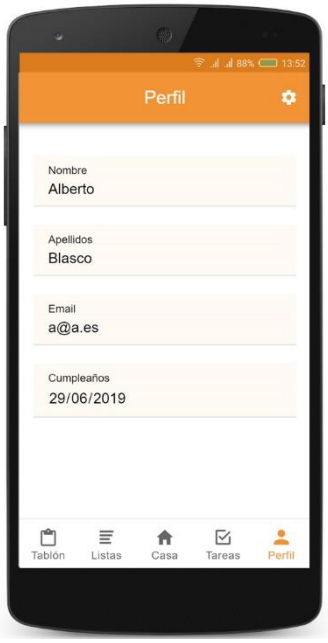
4.3.5 Sección Listas



4.3.6 Sección Tareas



4.3.7 Sección Perfil



5. Conclusiones

Como conclusiones finales del trabajo, se quiere destacar lo aprendido sobre todo en la fase de diseño. En comparación con otros desarrollos llevados a cabo anteriormente, no se le había dado tanta importancia al usuario, ni se había utilizado una metodología DCU. Este trabajo al tener un fuerte componente de diseño, ha permitido al estudiante aprender y practicar ampliamente.

Además, destacar que gracias a este trabajo se ha podido poner en práctica mucho de lo aprendido durante el presente Máster, pudiendo así afianzar diversos conocimientos.

También se ha aprovechado el trabajo para aprender y mejorar sobre las tecnologías empleadas, ya que, a pesar de que eran tecnologías que el estudiante ya conocía, gracias a este trabajo se ha podido profundizar más en ellas.

Respecto a los objetivos planteados al principio del trabajo, se puede decir que se han cumplido todos y la aplicación satisface los requisitos funcionales y no funcionales establecidos.

En cuanto al seguimiento y la planificación, ha sido un apartado que ha costado mantener al día ya que debido a ciertos impedimentos y errores de cálculo, no se ha podido dedicar todo el tiempo esperado al desarrollo del trabajo y de la aplicación, aunque eso no ha impedido que finalmente el resultado cumpla con lo propuesto.

Por último, el proyecto ha sido asumido también como un reto personal y que se podría mejorar, tal y como se comentará en el siguiente apartado. A poder ser se intentará seguir desarrollando la aplicación y cubrir las debilidades que hayan podido quedar por cuenta propia.

5.1 Líneas de futuro

Debido a la limitación de tiempo establecida, varias ideas se han quedado en el tintero, de hecho, en la primera entrega hubo que reformular los requisitos ya que en principio la aplicación era más ambiciosa. Algunas de las ideas que mejorarían la aplicación y sus funcionalidades son:

- Incluir una funcionalidad de gestión económica, en principio sin manejar dinero real, solo para la autogestión y por tanto llevar más control de que inquilinos pagan las facturas, colaboran económicamente a la casa, etc.
- Se podría implementar una sección de analíticas, para que los administradores pudiesen ver que usuarios colaboran más o menos.
- Otro aspecto que se podría mejorar bastante es el apartado gráfico. Ahora mismo se dispone de una interfaz muy básica, pero que podría ser mejorada si se contase con ayuda de un experto.
- Opción de deshacer, ya que mucho contenido lo genera el usuario, y este es propenso a cometer errores, estaría bien darle la opción de rectificar.

- Una funcionalidad muy interesante que se podría implementar es la de chat, a pesar de que hay otras aplicaciones de mensajería, se aseguraría de que este chat solo está relacionado con los temas del hogar.
- Soporte multiidioma, a pesar de que se ha preparado la aplicación para introducir nuevos idiomas, es necesario realizar una revisión de todos los textos para poder traducirla.
- Añadir fotografías al perfil de usuario y de la casa, para que cuando se realicen búsquedas de los mismos, sea más fácil reconocerlos.

6. Glosario

- API REST: Interfaz de aplicación que permite la recepción de peticiones para procesarlas y devolver una respuesta. Concretamente un API REST utiliza el protocolo web HTTP para realizar esta comunicación, junto con los métodos que ofrece (GET, POST, PUT, DELETE…)
- App: Diminutivo de *Application*; Aplicación en español.
- *Backend*: Parte menos visible de un sistema, se suele encargar principalmente de la gestión de los datos y el almacenamiento. El API REST puede ser una de las partes de este componente.
- Base de Datos: Software que permite el almacenamiento de datos de una manera organizada siguiente una estructura previamente definida.
- DCU: Diseño Centrado en el Usuario, una metodología de diseño de software que antepone el usuario al resto, construyendo el software con el usuario en el centro.
- Framework: Entorno de desarrollo que incluye diversas funciones para agilizar la creación de software.
- *Frontend*: Parte más cercana al usuario de un sistema, incluye tanto la interfaz gráfica como las interacciones con el mismo.
- Librería/Plugin: Código que puede ser añadido al software para incrementar las funcionalidades del mismo sin necesidad de ser desarrolladas por el programador.
- MVC: Modelo Vista Controlador, un patrón de desarrollo software que divide el desarrollo en tres capas, la de modelo (datos), la de vista (lo que se ve, las interacciones) y el controlador (la lógica interna del software).
- UX: *User eXperience*; Experiencia de Usuario en castellano.

7. Referencias Bibliográficas

Expansion.com. (12/06/2015). “Guía para compartir piso”. *expansion.com.* Disponible en: http://www.expansion.com/ahorro/2015/06/12/557ab8fb22601d19428b456d.html. Fecha de consulta: marzo de 2019

Fotocasa. (23/02/2016). “Compañeros de piso: principales problemas de convivencia”. *Fotocasa.* Disponible en: https://www.fotocasa.es/blog/alquiler/companeros-de-piso-cuales-son-los-principales-problemas-de-convivencia/. Fecha de consulta: marzo de 2019

Google Play. (27/02/2019A). “Splitwise - Cuentas y gastos” *play.google.com.* Disponible en: https://play.google.com/store/apps/details?id=com.Splitwise.SplitwiseMobile. Fecha de consulta: marzo de 2019

Google Play. (27/02/2019B). “Bring! Lista de compras” *play.google.com.* Disponible en: https://play.google.com/store/apps/details?id=ch.publisheria.bring. Fecha de consulta: marzo de 2019

Navarro, D. (07/01/2019). “Así es la generación ‘piso compartido’: Adriana se emancipó con 18 años y ya ha compartido piso seis veces”. *lasexta.com.* Disponible en: https://www.lasexta.com/tribus-ocultas/artes/asi-es-la-generacion-piso-compartido-adriana-se-emancipo-con-18-anos-y-ya-ha-compartido-piso-seis-veces_201901075c33bc480cf24fa4056e9ec0.html. Fecha de consulta: marzo de 2019

8. Bibliografía web consultada

Amazon. “¿Qué es Amazon EC2?”. Disponible en: https://docs.aws.amazon.com/es_es/AWSEC2/latest/UserGuide/concepts.html. Fecha de consulta: febrero a junio de 2019

Angular. “Introduction to the Angular Docs”. Disponible en: https://angular.io/docs. Fecha de consulta: febrero a junio de 2019

Apicurio. “Getting Started with Apicurio Studio”. Disponible en: https://apicurio-studio.readme.io/docs. Fecha de consulta: febrero a junio de 2019

Babich, N. (28/08/2018). “Best Practices For Mobile Form Design”. Disponible en: https://www.smashingmagazine.com/2018/08/best-practices-for-mobile-form-design/. Fecha de consulta: febrero a junio de 2019

Doctrine. “Welcome to Doctrine 2 ORM's documentation”. Disponible en: https://www.doctrine-project.org/projects/doctrine-orm/en/2.6/index.html#welcome-to-doctrine-2-orm-s-documentation. Fecha de consulta: febrero a junio de 2019

Hartikainen, J. (23/03/2016). “Sinon Tutorial: JavaScript Testing with Mocks, Spies & Stubs”. Disponible en: https://www.sitepoint.com/sinon-tutorial-javascript-testing-mocks-spies-stubs/. Fecha de consulta: febrero a junio de 2019

Ionic. “Ionic Framework”. Disponible en: https://ionicframework.com/docs. Fecha de consulta: febrero a junio de 2019

JustinMind “All-in-one prototyping tool for web and mobile”. Disponible en: https://www.justinmind.com/. Fecha de consulta: febrero a junio de 2019

Manca, F. (26/08/2018). “RESTful API Design: 13 Best Practices to Make Your Users Happy”. Disponible en: https://blog.florimond.dev/restful-api-design-13-best-practices-to-make-your-users-happy. Fecha de consulta: febrero a junio de 2019

McKechie, D. “Winwheel.js Tutorials & Documentation”. Disponible en: http://dougtesting.net/winwheel/docs. Fecha de consulta: febrero a junio de 2019

Molina, N. (12/11/2017). “Pruebas unitarias en Ionic: Providers”. Disponible en: https://blog.ng-classroom.com/blog/ionic2/uni-test-provider/. Fecha de consulta: febrero a junio de 2019

Ngx-translate. “ngx-translate”. Disponible en: https://github.com/ngx-translate/core. Fecha de consulta: febrero a junio de 2019

One Signal. “Ionic SDK Setup”. Disponible en: https://documentation.onesignal.com/docs/ionic-sdk-setup. Fecha de consulta: febrero a junio de 2019

One Signal. “Create notification: Sends notifications to your users”. Disponible en: https://documentation.onesignal.com/reference. Fecha de consulta: febrero a junio de 2019

Zend Framework. “Zend Framework 3”. Disponible en: https://framework.zend.com/learn. Fecha de consulta: febrero a junio de 2019

2muchcoffeecom. “ngx-restangular”. Disponible en: https://github.com/2muchcoffeecom/ngx-restangular. Fecha de consulta: febrero a junio de 2019

9. Anexos

Junto a esta memoria se proporciona información adicional en forma de anexos para complementar el trabajo.

- Anexo 1: Manual de usuario, con el cual un usuario puede aprender a utilizar la aplicación.
- Anexo 2: Manual de instalación, con el cual un desarrollador puede coger el proyecto, instalarlo en su ordenador y continuar con el desarrollo.
- Anexo 3: Resultados de la encuesta, donde se muestran los resultados obtenidos en la encuesta del apartado 2.1.2 de este trabajo.