Fíchalo: Gestión de bonos de servicios

Emilio Jiménez del Moral
Máster universitario de Desarrollo de aplicaciones para dispositivos móviles
Trabajo final de master

Nombre consultor: Eduard Martín Lineros
Profesor responsable de la asignatura: Carles Garrigues Olivella

Junio 2019

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Fíchalo: Gestión de bonos de servicios*
Nombre del autor:	*Emilio Jesús Jiménez del Moral*
Nombre del consultor/a:	*Eduard Martín Lineros*
Nombre del PRA:	*Carles Garrigues Olivella*
Fecha de entrega (mm/aaaa):	06/2019
Titulación::	*Máster universitario de Desarrollo de aplicaciones para dispositivos móviles*
Área del Trabajo Final:	*Trabajo final de master*
Idioma del trabajo:	*Español*
Palabras clave	*Bonos, Servicios, Fichar*

Resumen del Trabajo*:*

Este proyecto tiene por nombre **"Fíchalo: Gestión de bonos de servicios"**. Se trata de una aplicación Android que pretende ayudar a empresas a llevar la gestión de los bonos de servicios que venden a los clientes.

Existen muchas empresas como por ejemplo gimnasios, academias de inglés, escuelas de yoga, entrenadores personales, etc. que ofrecen bonos de usos de servicios a sus clientes para que puedan usarlos durante un tiempo determinado. En este tiempo la empresa necesita controlar cuántos usos tienen disponibles los clientes antes de proporcionar el servicio, y los clientes necesitan saber cuántos usos han consumido y cuantos le quedan por consumir.

El resultado que se quiere obtener, es facilitar la gestión de los bonos, hacer que la entrada al servicio por parte del cliente sea rápida y que ambas partes (cliente y empresa) sean debidamente informadas. De esta forma el cliente sabe de forma sencilla cuántos usos tiene disponible, y la empresa tiene la seguridad que está proporcionando justo los servicios que ha vendido.

Abstract:

This project has the name "Fíchalo: Service bonus management". It is an Android application that aims to help companies to take the management of the services that they sell to customers.

There are many companies such as gyms, English academies, yoga schools, personal trainers, etc. They offer bonuses of service uses to their clients so they can use them for a certain time. During this time, the company needs to control how many uses customers have available before providing the service, and customers need to know how many uses they have consumed and how many they have left to consume.

The objective is to facilitate the management of the bonds, make entry into the service by the client fast and that both the client and the company are duly informed. In this way the client knows in a simple way how many uses are available, and the company has the security that is providing just the services it has sold.

Índice

Lista de figuras

1. Introducción

1.1 Contexto y justificación del Trabajo

Existen muchas empresas como por ejemplo gimnasios, academias de inglés, escuelas de yoga, entrenadores personales, etc. que ofrecen bonos de usos de servicios a sus clientes para que puedan usarlos durante un tiempo determinado. En este tiempo la empresa necesita controlar cuántos usos tienen disponibles los clientes antes de proporcionar el servicio, y los clientes necesitan saber cuántos usos han consumido y cuantos le quedan por consumir.

Es un tema muy relevante porque la aplicación cubre una necesidad que afecta a un gran número de empresas de servicios independientemente del tamaño de las mismas y de los tipos de servicios que ofrezcan.

Actualmente las empresas que no disponen de un sistema informatizado, resuelven el problema mediante bonos (tarjetas físicas) que venden a sus clientes. Cuando el cliente hace uso del bono, la empresa de alguna manera lo marca, ya sea recortando parte del bono, apuntando la fecha del uso o marcándolo con un sello.

La gestión de los bonos mediante tarjetas físicas tiene los siguientes inconvenientes:

- Dependiendo del método del fichaje se podrían formar colas a la entrada del servicio.
- Si la empresa no está respaldando de alguna manera la información de cuántos bonos ha vendido, a quien y cuántos usos ha consumido cada cliente; en caso de que el cliente pierda la tarjeta, podría perder los usos que tenía disponibles.
- Es difícil gestionar las fechas de expiración de los bonos.
- Si el fichaje es complicado puede producir que algunos bonos no se hayan marcado correctamente y se pierdan usos.

El resultado que se quiere obtener es facilitar la gestión de los bonos y hacer que la entrada al servicio por parte del cliente sea rápida y ambas partes (cliente y empresa) sean debidamente informadas. De esta manera el cliente sabe de forma sencilla (y sin riesgo de pérdida del bono) cuántos usos tiene disponible, y la empresa tiene la seguridad que está dando justo los servicios que ha vendido.

Tras intentar realizar una búsqueda de aplicaciones similares usando palabras como como bonos, cupones, paquete de clases, paquete de servicios, fichajes, tarjetas de clases, gestión de clases, etc., no consigo encontrar aplicaciones enfocadas en exclusiva a resolver este problema.

Bobclass es una de las aplicaciones más parecida a lo que se necesita desarrollar, pero es una aplicación compleja con muchas funcionalidades, donde la gestión de bonos es sólo una pequeña parte de la aplicación. Está disponible sólo para la plataforma IOS y es necesario pagar una cuota mensual.

Hay soluciones integrales multiplataforma como ISMYGYM, que permite la gestión de clientes en gimnasios o centros deportivos. Este tipo de aplicaciones están hechas a medida para actividades muy concretas, y no casan con la idea de cubrir la necesidad a distintos tipos de empresas.

Existen aplicaciones como EventBrite y Ticketea que incluyen una parte de gestión de fichajes de personas, pero están más orientadas a la promoción y venta de eventos. Estas aplicaciones aunque tienen componentes parecidos, se alejan del objetivo principal de la aplicación que se quiere desarrollar, ya que normalmente trabajan con entradas de un solo uso. Otras aplicaciones como TimeTree, permiten unirse a eventos, pero es una aplicación tipo calendario que no cubre los fichajes.

Tras el análisis inicial de aplicaciones similares, parece que existe un hueco a cubrir que permitiría a las empresas gestionar sus servicios sin tener que contratar desarrollos a medida.

1.2 Objetivos del Trabajo

El objetivo principal del trabajo es realizar una aplicación móvil que gestione los bonos que la empresa vende a sus clientes y los usos que se han consumido.

Los objetivos de alto nivel son:

- Los usuarios podrán registrarse y autenticarse. Un usuario nuevo podrá darse de alta como empresa o como cliente que va a consumir bonos. Si el usuario ya existe podrá entrar en la aplicación con su cuenta.
- Si el usuario es una empresa, podrá:
  - Crear distintos tipos de bonos, especificando número de usos, precio y fecha de caducidad.
  - Enviar un bono a un usuario cliente de la aplicación. El pago se realizará fuera de la aplicación y una vez realizado, la empresa enviará el bono al usuario cliente ya registrado.
  - Ver los bonos que tienen sus clientes y el estado en el que se encuentran, es decir, cuántos usos han consumido y cuántos usos tienen disponibles.
  - Realizar el fichaje de los clientes, es decir descontar un uso del bono de los clientes que realicen la actividad.
  - Deshacer los usos en los bonos. Esto permitirá corregir errores de fichaje.
  - Ver el histórico de fichajes.
- Si el usuario es un cliente, podrá:
  - Ver la lista de bonos que tiene disponibles. De cada bono podrá ver los usos que le quedan.
  - Ver el histórico de fichajes.

Objetivos no funcionales:

- La plataforma para la que se va a desarrollar la aplicación es Android.
- Los datos deben estar almacenados en la nube, de modo que si una empresa o cliente pierde el móvil, no pierda las información sobre los bonos.
- Las interfaces deben ser limpias y sencillas, evitando elementos innecesarios que compliquen el uso de la aplicación.
- La seguridad de la aplicación debe evitar que los usuarios (empresas o clientes), vean información de otros usuarios.

1.3 Enfoque y método seguido

La estrategia a seguir es desarrollar un producto desde cero basado en una aplicación Android en base a los siguientes criterios:

- Se pretende desarrollar un producto muy específico difícil de encontrar. Algunas aplicaciones similares están muy vinculadas a la actividad que se quiere vender (por ejemplo gimnasios) y son poco versátiles a la hora de abarcar distintos tipos de servicios. Otras aplicaciones son tan extensas y con tantas funcionalidades, que se distancian del problema concreto que se quiere resolver: la gestión de bonos de usos de distintos tipos de servicios.

- La plataforma escogida es Android por las siguientes razones:
  - Android es usado casi por el 90% de los smartphones y por lo tanto es líder en el mercado llegando a más usuarios.

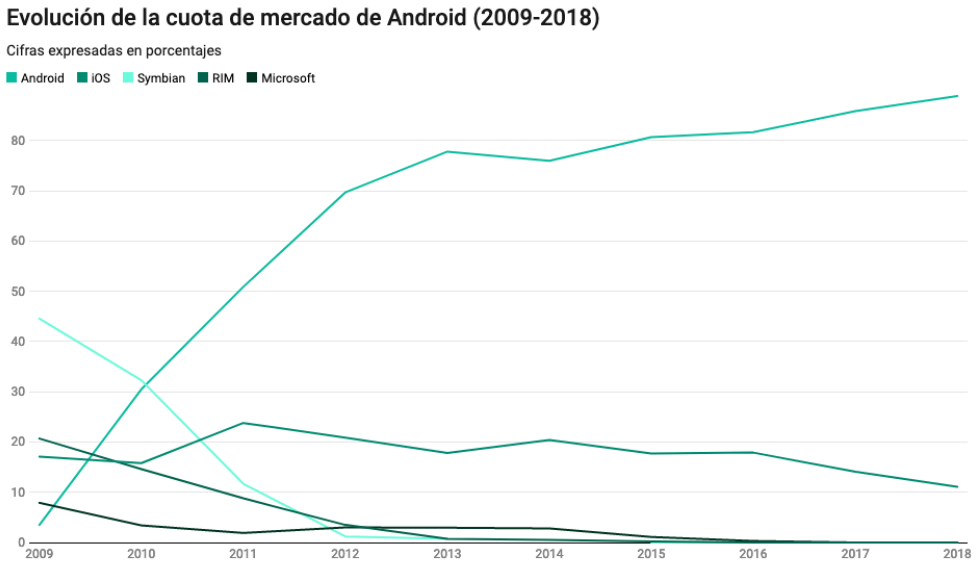


Ilustración 1: Los datos comienzan a partir del cuatrimestre de 2009. Chart: Xataka Móvil (1) Fuente: Statista (2)

  - Existen productos como firebase que agilizan y facilitan enormemente el desarrollo de funcionalidades horizontales (almacenamiento de base de datos en la nube, autenticación, mensajería) permitiendo a la aplicación escalar de una forma segura. Aunque esta tecnología también es integrable con otras plataformas, al ser una plataforma desarrollada por Google la facilidad en la integración con Android es notable.

- Desde el punto de vista de aprendizaje es una buena opción desarrollar los trabajos desde cero.

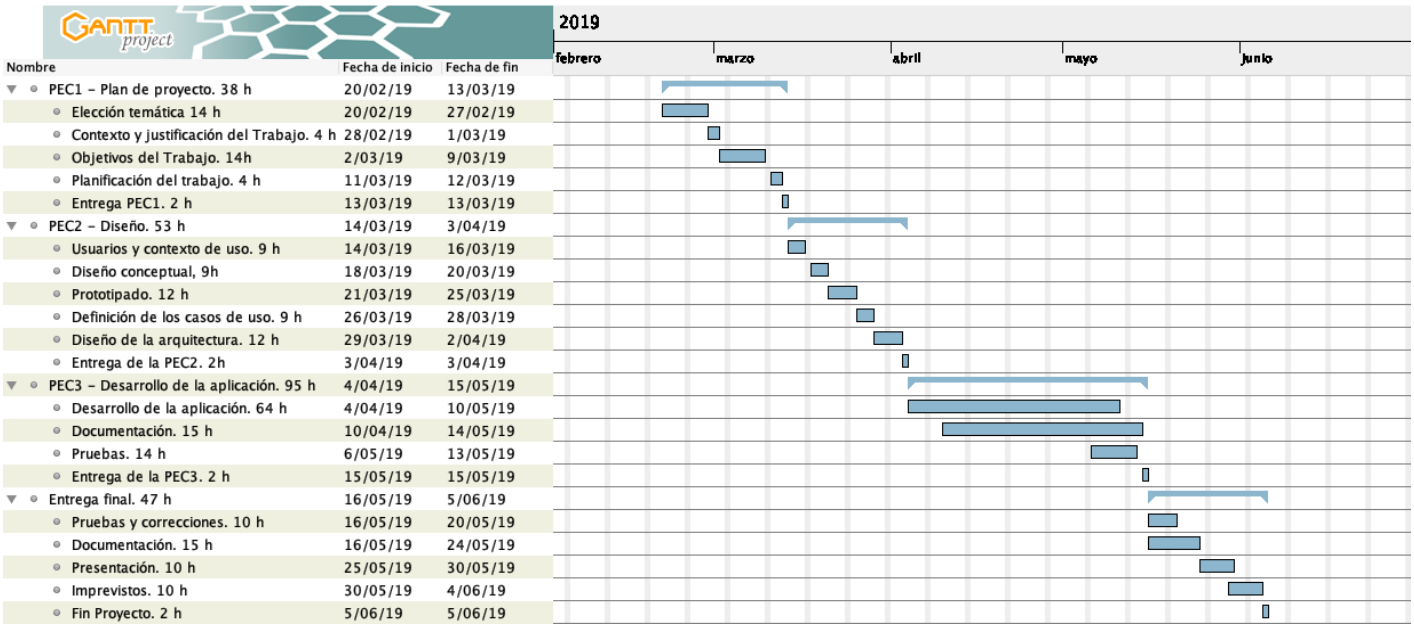
1.4 Planificación del Trabajo

El horario de trabajo durante el desarrollo del proyecto será de lunes a sábado un máximo de 3 horas al día.

Se considerarán los siguientes hitos a la hora de realizar la planificación de las tareas:

- PEC 1 – Plan de trabajo. Día 13 de marzo de 2019.
- PEC 2 – Diseño. Día 3 de abril de 2019.
- PEC 3 – Desarrollo de la aplicación . Día 15 de mayo de 2019.
- Entrega final. Día 5 de junio de 2019.

Teniendo en cuenta todo lo anterior, se ha realizado el siguiente diagrama de Gantt donde se plasma la planificación de las tareas (indicando la duración en horas) y las actividades que se llevarán a cabo (aunque durante el desarrollo del proyecto puede sufrir algún cambio).



Nombre	Fecha de inicio	Fecha de fin
PEC1 – Plan de proyecto. 38 h	20/02/19	13/03/19
Elección temática 14 h	20/02/19	27/02/19
Contexto y justificación del Trabajo. 4 h	28/02/19	1/03/19
Objetivos del Trabajo. 14h	2/03/19	9/03/19
Planificación del trabajo. 4 h	11/03/19	12/03/19
Entrega PEC1. 2 h	13/03/19	13/03/19
PEC2 – Diseño. 53 h	14/03/19	3/04/19
Usuarios y contexto de uso. 9 h	14/03/19	16/03/19
Diseño conceptual, 9h	18/03/19	20/03/19
Prototipado. 12 h	21/03/19	25/03/19
Definición de los casos de uso. 9 h	26/03/19	28/03/19
Diseño de la arquitectura. 12 h	29/03/19	2/04/19
Entrega de la PEC2. 2h	3/04/19	3/04/19
PEC3 – Desarrollo de la aplicación. 95 h	4/04/19	15/05/19
Desarrollo de la aplicación. 64 h	4/04/19	10/05/19
Documentación. 15 h	10/04/19	14/05/19
Pruebas. 14 h	6/05/19	13/05/19
Entrega de la PEC3. 2 h	15/05/19	15/05/19
Entrega final. 47 h	16/05/19	5/06/19
Pruebas y correcciones. 10 h	16/05/19	20/05/19
Documentación. 15 h	16/05/19	24/05/19
Presentación. 10 h	25/05/19	30/05/19
Imprevistos. 10 h	30/05/19	4/06/19
Fin Proyecto. 2 h	5/06/19	5/06/19

1.5 Breve sumario de productos obtenidos

1.5.1 PEC1 – Plan de trabajo

En el actual documento se definen las características del proyecto a desarrollar. Se establece el título, una descripción general del proyecto y los objetivos que se pretenden alcanzar.

En base a todo lo anterior, se definirá también una planificación temporal con las metas a conseguir para cada uno de los hitos o puntos de revisión establecidos.

1.5.2 PEC2 – Diseño.

En este documento se realizará un análisis de los requisitos, un diseño centrado en el usuario y un diseño técnico de la aplicación.

1.5.3 PEC3 – Implementación

En esta entrega se pretende tener prácticamente terminado el desarrollo de la aplicación, además de definir el conjunto de pruebas.

1.5.4 Entrega Final

En esta fase se entregará la memoria final del proyecto, la aplicación completamente desarrollada, una manual de usuario y una presentación.

1.6 Breve descripción de los otros capítulos de la memoria

El presente documento contiene la memoria del trabajo final de máster y se estructura de la siguiente forma:

- Capítulo 1: Contiene la introducción al trabajo, la necesidad que se desea cubrir y porqué es necesaria cubrirla, el resultado que se desea obtener describiendo los objetivos que se pretenden alcanzar, la estrategia escogida para llevar a cabo el trabajo y una planificación temporal.
- Capítulo 2: Expone el diseño de la aplicación, dividido en:
  - Diseño centrado en el usuarios
  - Catálogo de requisitos
  - El diseño técnico de la aplicación
- Capítulo 3: En este capítulo, se justifican todas las decisiones tomadas durante el proceso de desarrollo y se define un plan de pruebas.
- Capítulo 4: Contiene las conclusiones del desarrollo del proyecto, y las futuras mejoras sobre la aplicación.
- Capítulo 5: Glosario de términos.
- Capítulo 6: Bibliografía.
- Capítulo 7: Anexos.

2.Diseño

2.1.Diseño centrado en el usuario

El diseño centrado en el usuario sitúa al usuario en el centro de todo el proceso, de forma que lo involucra en todas las fases desarrollo evitando desarrollar algo que nadie necesita y promoviendo cubrir todas las necesidades reales que el usuario tiene.

Este término fue popularizado por Donald A. Norman. Según Norman (3), el diseño tiene que:

- Dejar muy claro al usuario qué puede hacer en cada momento.
- Ser transparente, es decir, poner a la vista tanto las posibles acciones como sus resultados.
- Facilitar la evaluación del estado del sistema en todo momento.

2.1.1 Usuarios y contexto de uso

La aplicación a desarrollar en este trabajo, va dirigida principalmente a usuarios que trabajan en una empresa de servicios y que necesitan gestionar los bonos de usos que venden a sus clientes, pero también está dirigida a los usuarios clientes que quieren llevar el control de los usos de servicios que tienen para usar.

Métodos de indagación

Las técnicas de indagación nos permiten conocer de primera mano el comportamiento de los usuarios y su interacción con el producto o servicio.

Observación

La observación es uno de los métodos de indagación realizados para conocer las características de los usuarios, sus necesidades, objetivos y contexto de uso. Con este método se ha realizado un seguimiento del comportamiento de distintas personas durante el proceso de venta de un bono, su posterior uso y el marcado del mismo.

En concreto se ha observado como fichan el uso de clases en dos empresas distintas: una academia de inglés y un rocódromo. Aunque en ambas empresas hay diferencias en el proceso, la base es muy similar.

En el caso de la academia de inglés, cuando un estudiante compra un bono de 10 usos, una persona autorizada que trabaja en la academia le proporciona una especie de cartulina con 10 filas. Después de cada clase el personal de la academia apunta la fecha en la que recibió la clase en una de las filas libres. Cuando un estudiante ha perdido la cartulina y pregunta en la academia cuantas clases le quedan, la forma que tiene el personal de calcularlo es ir a un calendario que tienen en Google y contar todas las clases que ha dado el usuario desde la venta del bono. Cuando finalmente el personal de la academia calcula los usos, le proporcionan al estudiante otra cartulina con tantas filas tachadas como clases haya dado hasta el momento.

Los estudiantes en las academias suelen estar en un gran rango de edad, aunque los mayoritarios son niños, niñas y adolescentes, también hay bastantes estudiantes de entre 18 y 45 años y algunos entre los 45 y los 60. Tanto los niños, como algunas de las personas más mayores no serán usuarios candidatos de la aplicación y seguirán usando el método tradicional.

Los usuarios clientes de las empresas necesitarán registrarse en la aplicación mediante un correo electrónico, y como la edad mínima para tener una cuenta gmail es de 13 años, los niños menores de esta edad quedarán excluidos en el uso de la aplicación.

En el caso del rocódromo el procedimiento es similar, pero en lugar de apuntar la fecha del uso, la tarjeta tiene tantos cuadrados como usos se hayan vendido y el personal del rocódromo va recortando con unas tijeras a la entrada o a la salida del escalador uno de ellos. Si el escalador pierde la tarjeta, el gerente del rocódromo revisa una hoja de cálculo de Google donde va apuntando los bonos que tiene cada persona y los usos que ha consumido, por lo que en este caso puede recuperar un poco más rápido la cantidad de usos que le quedan al escalador. Además las tarjetas tienen fecha de expiración, por lo que el personal del rocódromo suele avisar al escalador que le queda poco tiempo para poder hacer uso del bono.
Otro detalle interesante observado es que la misma persona que se encarga de recortar un uso del bono, también es entrenador, por lo tanto a veces tiene que cortar el entrenamiento para ir corriendo al mostrador y fichar a unos cuantos escaladores, para luego seguir con la clase.

Se ha observado que la media de los escaladores suelen ser jóvenes, también hay personas mayores entre 50 y 60 años. En general todos los escaladores están muy acostumbrados al uso del móvil ya que usan aplicaciones de tiempo e información de escuelas de escalada para hacer salidas al campo. Además ya usan otra aplicación llamada pasosBulder usada para hacer bloques en el rocódromo.

Entrevistas

Para aplicar este método de indagación se ha realizado una entrevista al gerente del rocódromo, de la cual se ha obtenido información valiosa sobre la experiencia y necesidades de los usuarios.

La entrevista se realizó en un ambiente distendido en el propio rocódromo y durante los descansos de un entrenamiento. De este modo se pudo observar en el mismo momento las diferentes problemáticas a las que se enfrenta en el día a día el gerente de la empresa que al mismo tiempo es entrenador.

Preguntas y respuestas realizadas:

1. ¿Cual crees que puede ser la parte más complicada del proceso de fichaje de un cliente?
Repuesta: tal vez no sea más complicada, pero si tardo bastante tiempo recortar el bono y apuntarlo en la hoja de cálculo, sobre todo cuando hay más de un cliente esperando.

2. Dime algo que te desagrade al realizar fichaje.
Respuesta: cuando estoy concentrado impartiendo una clase de escalada, o un entrenamiento, tengo que parar para ir al mostrador, y realizar los fichajes de la gente que no fichó a la entrada y tienen que fichar a la salida. De algún modo me gustaría

poder realizar el fichaje de estas personas más tarde o incluso en el mismo momento sin moverme del sitio donde estoy.

3. ¿Te gustaría que los clientes pudiesen fichar ellos mismos?
Respuesta: me gustaría ser yo el responsable de fichar para tener un mayor control.

4. ¿A parte de los bonos, hay otros tipos de productos que te cueste llevar un control?
Respuesta: hay bonos que no son por usos sino por meses, trimestres o por un año y cuando un cliente lo compra por ejemplo a mitad de mes, tengo que descontarle del precio la parte proporcional. Simplemente me gustaría venderlo por el precio real y poder controlar el día que expira el bono.

Como conclusiones podemos obtener que el fichaje de los bonos de los clientes es lento, no es flexible y puede detener otras actividades importantes de la empresa. Que hay productos con fechas de expiración difíciles de seguir un control, y que al empresario le gusta tener el control de las personas a las que está fichando.

También se ha podido observar que existen dos tipos diferenciados de perfiles de usuarios, por un lado tenemos usuarios que son los gerentes o el personal de la empresa y por otro lado el usuario cliente que consume lo servicios de las empresas.

2.1.2 Diseño conceptual

Una vez procesada toda la información obtenida como resultado de los métodos de indagación, mediante las técnicas de personas y escenarios se describe cómo los usuarios van a usar la aplicación en un contexto concreto.

Una vez definidos los escenarios, estos serán de utilidad para conceptualizar la estructura de la aplicación y los flujos de interacción.

Personas

Esta técnica, popularizada por Alan Cooper e inspirada en el método utilizado por los actores para desarrollar un personaje y dotarlo de verosimilitud, ayudará al desarrollador a tener siempre en mente a un usuario concreto, con unas características y objetivos bien definidos, mientras realiza su trabajo.

Lucía Jiménez Urbano

Ilustración 2: Foto perfil lucía Jiménez

Nombre: Lucía Jiménez Urbano
Edad: 23
Nivel de estudios: Estudiante de enfermería.

Lucía es soltera, tiene un gran grupo de amigos, vive en Córdoba con sus padres y es estudiante de enfermería. Asiste a clases de 8 a 15:00.

Tiene internet en casa y un ordenador portátil que usa casi exclusivamente para las tareas de la universidad. Suele usar el móvil para leer las noticias, hablar con sus amigos mediante WhatsApp y de vez en cuando revisa la actividad en FaceBook. El uso que realiza del móvil es limitado a unas pocas aplicaciones, y además no está muy al día de las novedades ya que se deja guiar por su grupo de amigos. Su móvil tiene sistema operativo Android. Es aficionada a la lectura, los idiomas y la música.

Descripción del escenario:

Lucía tiene hoy clases de Inglés en una academia al lado de su casa que se llama Kairos Town. Debido a que la universidad le demanda mucho tiempo, decidió comprar un bono de 10 clases para consumir durante los siguientes meses, ya que no se puede comprometer a ir todas las semanas.

El día que Lucía fue a la academia a informarse y compró el bono, María la gerente de la academia le ofreció la posibilidad de que se instalase la aplicación de bonos para llevar la cuenta de las clases desde el móvil. Para ello María le pide a Lucía el número de teléfono o correo electrónico para enviarle un enlace de instalación de la aplicación. Lucía recibe el enlace de la aplicación, la instala y se registra. A continuación María agrega un bono de 10 clases al usuario de Lucía en la aplicación. A partir de ese momento María puede ver en su aplicación una lista de bonos donde aparece un bono nuevo de kairos Town de diez usos. Si entra en el bono puede ver el detalle de mismo: empresa que lo ha vendido, el número de usos total, usados y disponibles, y en caso de que exista la fecha de expiración. Inicialmente tiene los 10 usos disponibles.

Con el paso del tiempo, Lucía asiste a varias clases, pero no sabe con exactitud cuantas clases tiene disponibles, por lo que abre la aplicación, entra en el bono y observa que todavía tiene disponibles dos clases. Entra en la academia, saluda a María que se encuentra en el mostrador, y como hay más gente esperando, María le indica que pase a la clase que mas tarde fichará el bono.

Lucía se encuentra dando clase, le vibra el móvil y en un descanso observa que ha recibido una notificación de la aplicación de bonos indicando que se ha consumido una clase de inglés. Al revisar el bono puede ver que ya sólo le queda una clase, además puede revisar en el detalle del bono las fechas de los fichajes, es decir, de las distintas clases recibidas.

Pedro Sharma

Ilustración 3: Foto perfil Pedro Sharma

Nombre: Pedro Sharma
Edad: 35
Nivel de estudios: Técnico Superior en Radioterapia y Dosimetría
Profesión: Entrenador y escalador.

Pedro está casado y tiene un niño de 3 años. Vive en Córdoba y pasa gran parte del tiempo en su negocio, un rocódromo en el que trabaja a horario partido de 11:00 a 14:00 y de 17:00 a 23:00.

Suele usar un portátil para gestionar el negocio y el control de los bonos que vende. Es muy activo con el móvil ya que tiene que gestionar las redes sociales y está continuamente en contacto con los clientes rocódromo para responder dudas sobre el horario, el entrenamiento, salidas al campo, etc.
Está al día de las novedades tanto en aplicaciones de redes sociales como en música, además puede considerarse un usuario avanzado.

Descripción del escenario:

Un día llega a oídos de Pedro que existe una aplicación para gestionar los bonos de uso de los clientes, como está agobiado con este tema porque no acaba de encontrar una buena fórmula decide instalar la aplicación.

Pedro se registra en la aplicación con la cuenta de correo electrónico del rocódromo indicando en el mismo registro que es un usuario de tipo empresa. Una vez registrado observa que puede dar de alta distintos tipos de bonos y comienza a registrar sus productos:

- Bono climbing fit de 5 usos y fecha de expiración 3 meses: 25 euros.
- Bono climbing fit de 10 usos y fecha de expiración 6 meses: 45 euros y fecha de expiración 3 meses.
- Bono uso libre de 5 usos y fecha de expiración 3 meses: 21 euros.
- Bono uso libre de 10 usos y fecha de expiración 6 meses: 40 euros.
- Bono clases de escalada 5 usos y fecha de expiración 12 meses: 25 euros.
- Bono clases de escalada 10 usos y fecha de expiración 12 meses: 45 euros.

Pedro abre el rocódromo, pone música, prepara algunas cosas, y comienzan a entrar los escaladores y escaladoras. Algunos van a clases grupales, otros pagan el uso ese día y otros usan bonos de uso que necesita que Pedro les fiche a la entrada o a la salida.

Entran tres personas, Victor tiene un bono de 5 clases de climbing fit (entrenamiento dirigido) y otro de uso de libre de 5 clases, Rafa tiene un bono de uso libre de 10 usos y Juan ya no le quedan usos disponibles y quiere uno de 5 de uso libre. Pedro revisa el perfil de Victor en la aplicación y observa que tiene dos bonos, le pregunta que quiere hacer hoy (si uso libre o climbing fit) para fichar en el bono adecuado, y realiza el fichaje. Rafa solo tienen un bono por lo que Pedro realiza el fichaje sobre el bono disponible, y a Juan le cobra el bono de 5 y a continuación le asigna el bono desde la aplicación. En unos 3 minutos ha podido despachar a tres personas y comienza a orientar unas personas que empiezan ese día a escalar.

Una vez que Pedro da algunas indicaciones a los nuevos escaladores, comienza a entrenar de forma dirigida a Victor, pero cuando está en medio del entrenamiento, hay algunas personas que se tienen que ir del rocódromo, no ficharon a la entrada y le preguntan a pedro si les puede fichar. Pedro sin moverse del sitio ficha a 4 personas que han estado escalando en uso libre y puede continuar con el entrenamiento.

Flujos de interacción

A partir de los escenarios de las personas podemos detallar el flujo de interacción de la aplicación, teniendo en cuenta dos perfiles muy diferenciados de usuarios, empresas y clientes.

El usuario ya sea una empresa o un cliente podrá registrarse en la aplicación de una forma fácil usando una de las cuantas google que tenga dadas de alta en el móvil. El usuario al entrar por primera vez pulsará en el inicio de sesión y la aplicación le mostrará un listado de cuentas para que escoja una de ellas. Además se le mostrará un aviso indicando que Google compartirá el nombre, dirección de correo electrónico y foto del perfil con la aplicación. Estos datos los verá la empresa y le facilitará localizar a sus clientes, y a su vez los clientes podrán ver esta información de las empresas. La aplicación también preguntará que tipo de usuario es, si empresa o cliente.

A partir de este momento el usuario ya está registrado, autenticado y podrá comenzar a usar la aplicación.

La pantalla de inicio después de autenticarse mostrará distintas opciones dependiendo del perfil del usuario.

Empresas

En la pantalla de inicio el usuario verá varias opciones:

- Fichar
- Enviar bonos
- Histórico de fichajes
- Clientes
- Tipos de bonos

La opción “Fichar” será una de las más usada. Cuando el usuario realiza esta acción, la aplicación mostrará un listado de clientes que actualmente tienen bonos activos. Tras seleccionar al cliente al que fichar, si solo tiene un bono el fichaje se hará automáticamente previa confirmación, y si tiene más de un bono, elegirá el tipo de bono y finalmente confirmará la operación. Esto provocará que el cliente reciba en el móvil una notificación de que ha sido fichado en un bono determinado.

Con la opción “Histórico de fichajes” podrá ver un listado de fichajes agrupados por día. Cada elemento del listado mostrará el usuario, la hora del fichaje y el tipo de bono consumido.

Con la opción “Clientes” el usuario accede a una pantalla que muestra un listado de usuarios que tienen o han tenido bonos con la empresa. Al pulsar en un usuario, se muestra una pantalla detalle del mismo, donde se pueden ver los bonos activos y los ya gastados. De cada bono activo se puede revisar cuantos usos hay consumidos y cuantos quedan por consumir, así como la fecha de compra y de expiración. Además se podrán revertir usos de los bonos ya que es posible que se comentan errores.

Si selecciona la opción “Enviar bonos” la aplicación mostrará una pantalla con un listado de usuarios que tienen o han tenido bonos, a continuación selecciona el cliente, selecciona el tipo de bono a enviar y pulsa aceptar en un diálogo de confirmación. Si el cliente es nuevo y no aparece en el listado, existirá una opción de búsqueda para encontrar nuevos usuarios.

Si accede a “Tipos de bonos” la aplicación mostrará una pantalla con un listado de tipos de bonos que ya ha dado de alta. A partir de esta pantalla, el usuario puede acceder a otras dos pantallas, una que muestra el detalle de un tipo de bono, o otra para dar de alta uno tipo nuevo especificando nombre, número de usos, precio y fecha de caducidad.

A continuación se muestra el árbol de navegación de un usuario de tipo empresa, representando las pantallas mediante rectángulos. La navegación entre las pantallas se representa mediante flechas unidireccionales y bidireccionales que indican la navegabilidad hacia atrás.

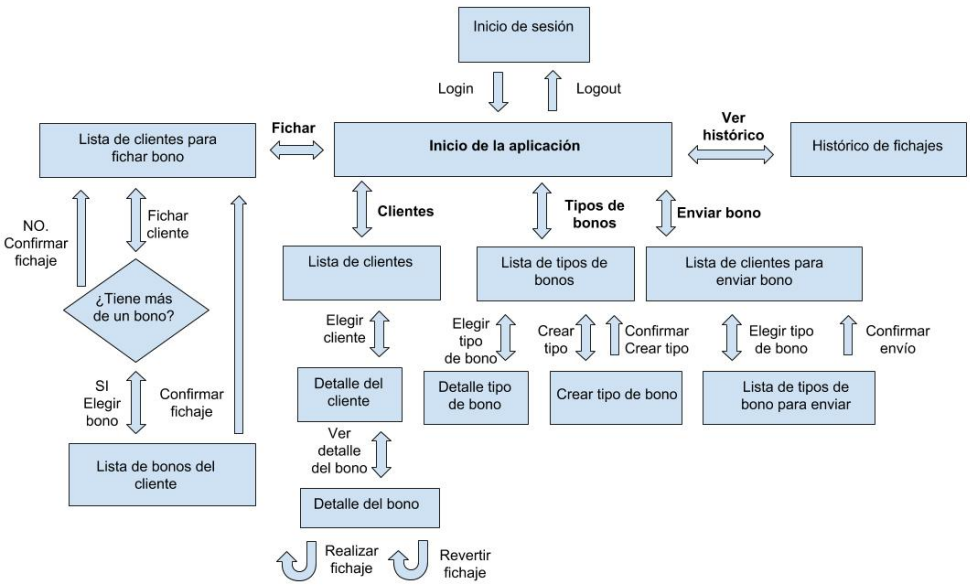


Ilustración 4: Árbol de navegación de un usuario de tipo empresa

Clientes

En la pantalla de inicio el usuario verá las siguientes opciones:

- Mis bonos.
- Histórico de fichajes.

Al entrar en “Mis bonos”, la aplicación mostrará un listado de bonos tanto activos como ya gastados. De cada bono se mostrará los usos disponibles y los consumidos, así como la fecha de expiración.

Con la opción “Histórico de fichajes” podrá ver un listado de fichajes agrupados por día. Cada elemento del listado mostrará la hora del fichaje y el tipo de bono consumido.

A continuación se muestra el árbol de navegación de un usuario de tipo cliente.

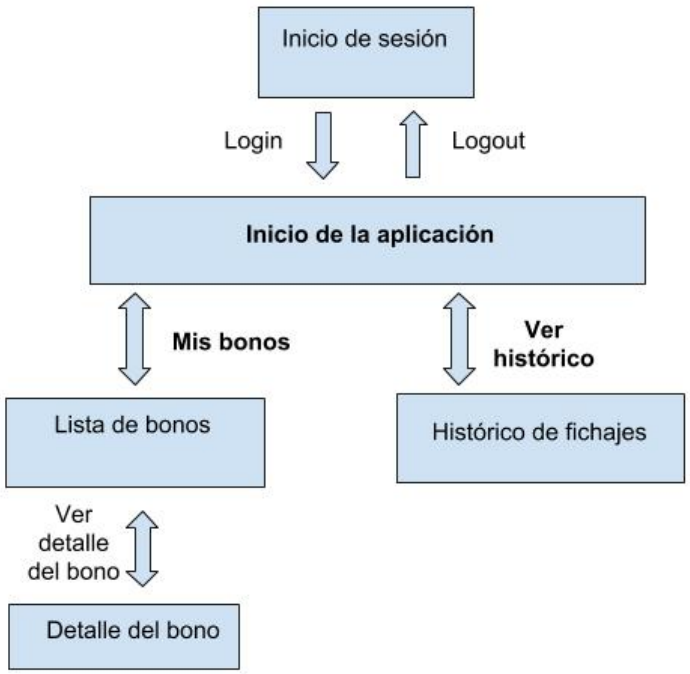


Ilustración 5: Árbol de navegación de un usuario de tipo cliente.

2.1.3 Prototipado

Tomando los flujos de interacción definidos en la fase anterior, se realizará un prototipo de la aplicación.

Un prototipo es una representación de la aplicación que permite mostrar decisiones de diseño y que éstas sean evaluadas antes de desarrollar el producto final. Su versatilidad hace que sea sencillo y económico introducir modificaciones en el diseño e iterar incorporando mejoras fruto de la discusión con los miembros del equipo o de los resultados obtenidos en la evaluación.

Prototipo de baja fidelidad

En la primera fase se desarrollará un prototipo de baja fidelidad realizando bocetos a mano alzada en el que obtendrá una aproximación de las pantallas y funcionalidades que se necesitan implementar

Pantalla de inicio de sesión e inicio de aplicación

Como se puede ver en la figura, la primera pantalla es el login del usuario. Cuando el usuario pulse el botón “Iniciar sesión”, se desplegará en la misma pantalla un listado que permite al usuario iniciar la sesión con una de las cuentas google que tenga dadas de alta en el dispositivo. Si es la primera vez que entra con la cuenta seleccionada, en la misma pantalla podrá elegir si es un perfil de tipo empresa.

La opción “Usar otra cuenta” es acceso directo para crear otra cuenta Google en el dispositivo.

Dependiendo del tipo de perfil (empresa o cliente), se muestra una pantalla de inicio de aplicación con distintas opciones. Además esta pantalla tiene un menú desplegable arriba a la izquierda que visualiza la cuenta de usuario en uso, y permite salir de la sesión.

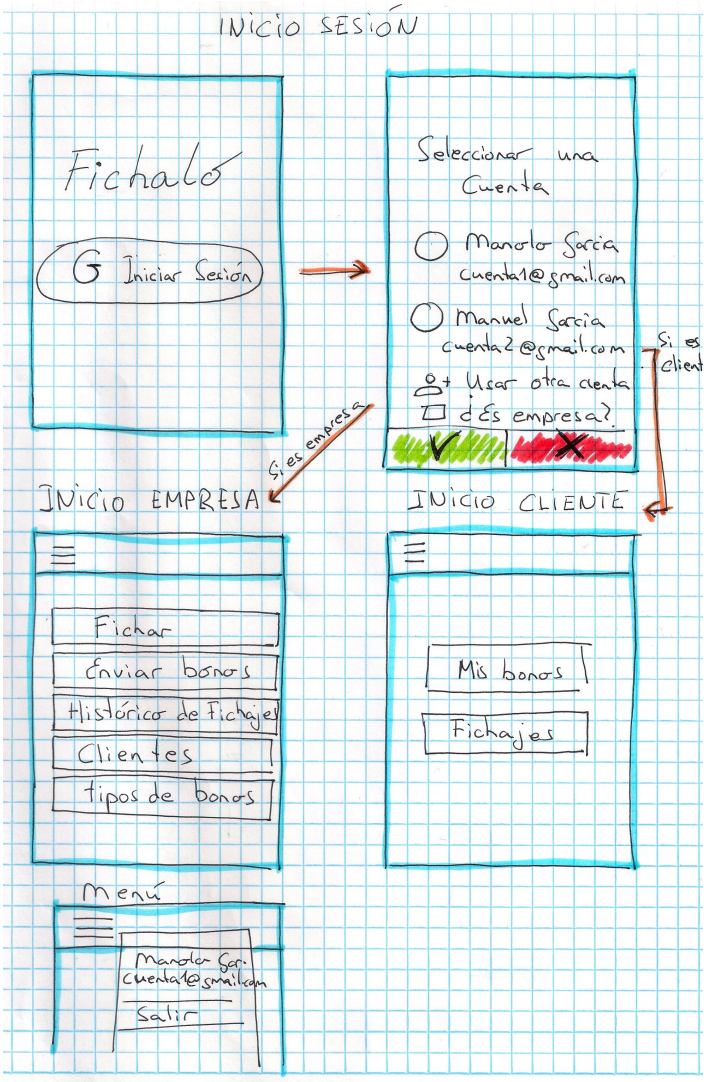


Ilustración 6: Boceto inicio de sesión e inicio de aplicación

Fichar clientes

En la siguiente figura se muestran las pantallas para fichar un cliente. La primera pantalla muestra un listado de clientes con información detalle sobre cada uno de ellos (su foto, el nombre, un listado de bonos con el número de usos, etc.) También se puede buscar un usuario en concreto usando la lupa.

Cuando la empresa pulsa en uno de los clientes, y esté tiene solo un bono, el fichaje se realizará automáticamente (previa confirmación que tendrá que aceptar o cancelar la empresa). Si el cliente tiene más de un bono, se muestra una segunda pantalla con un listado de bonos para elegir en que bono fichar.

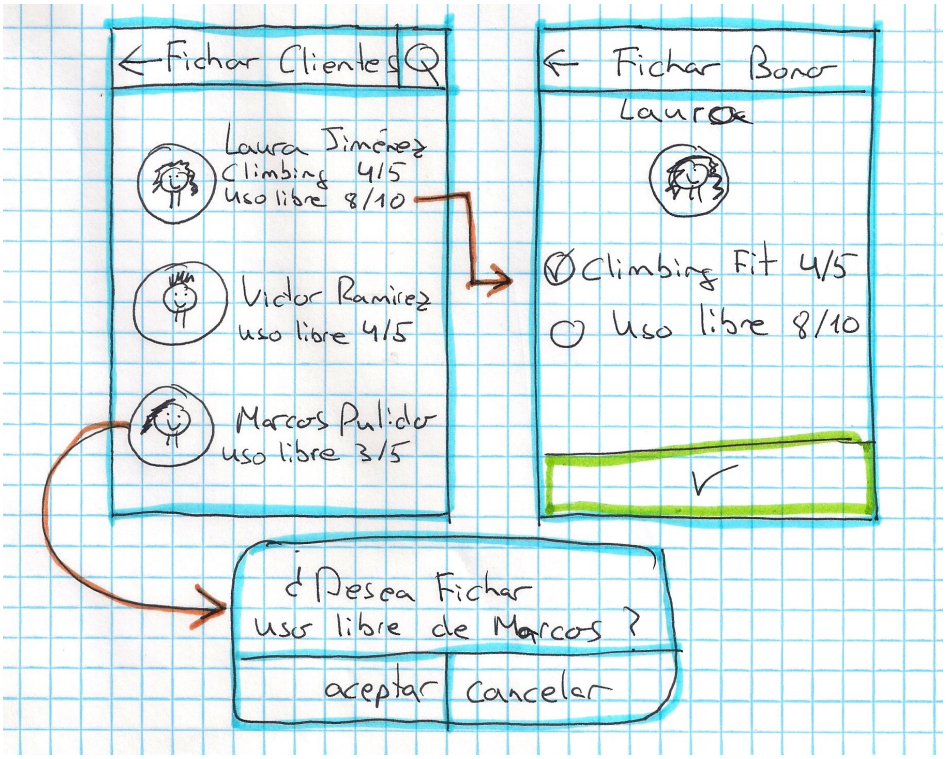


Ilustración 7: Boceto fichar clientes.

Histórico de fichajes para empresa

En esta pantalla, la empresa puede ver agrupados por días un listado de usuarios que han fichado. En cada fichaje se muestra la foto del cliente, el nombre, el bono con el que fichó y la hora del fichaje.

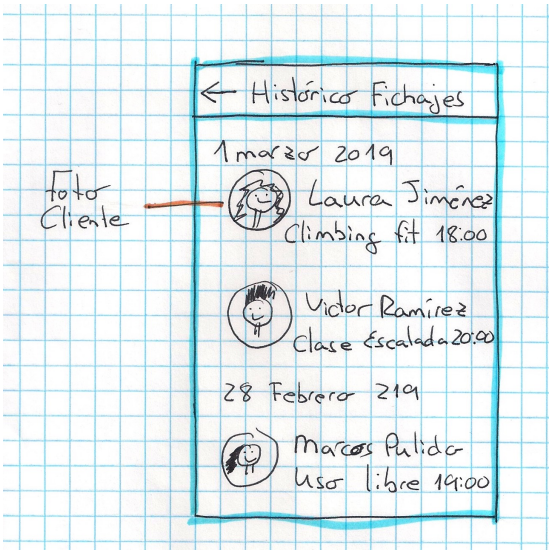


Ilustración 8: Boceto histórico de fichajes empresa

Enviar bonos

Cuando la empresa pulsa esta opción la aplicación muestra una pantalla con un listado de clientes (con la foto y el nombre de cada uno de ellos). Además usando la lupa podría filtrar rápidamente y buscar un usuario determinado. Cuando pulsa sobre un cliente, aparece una segunda pantalla con un listado de tipos de bonos a enviar, realiza la selección del bono o bonos y finalmente envía el bono pulsando el botón de confirmación.

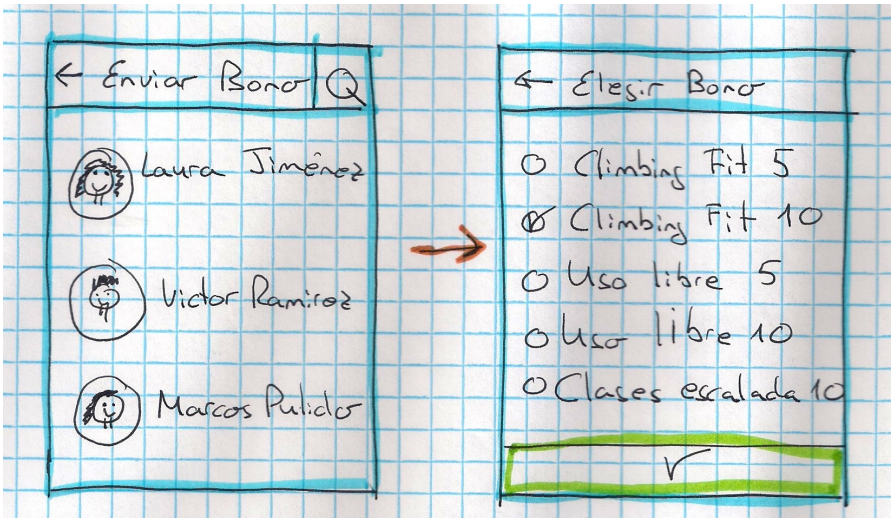


Ilustración 9: Boceto enviar bono

Tipos de bonos

A través de esta pantalla la empresa puede ver un listado con los tipos de bonos dados de alta. Para cada bono se muestra el nombre, el número de usos, el precio y el plazo máximo para consumir el bono.

Al pulsar en el icono “+” se abre una pantalla formulario para crear un bono nuevo.

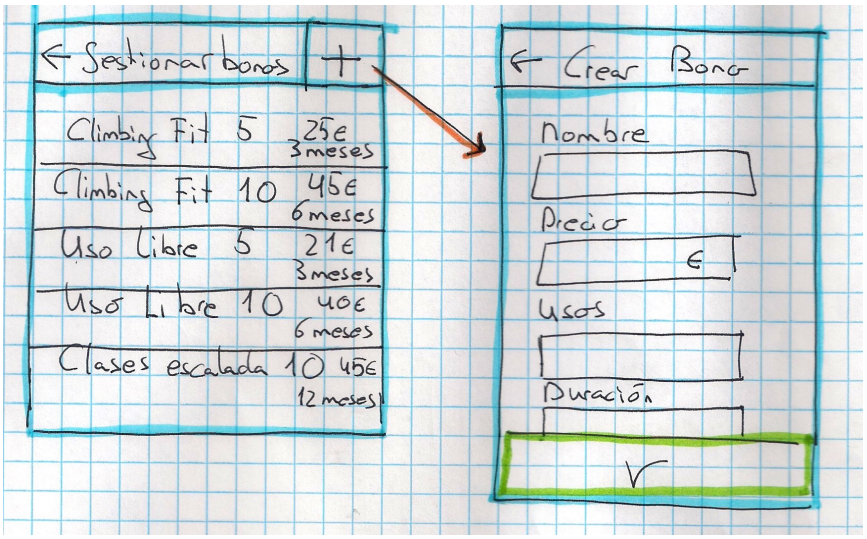


Ilustración 10: Boceto gestionar bonos

Clientes

A través de la pantalla de clientes, la empresa puede ver un listado de todos su clientes mostrando de cada uno de ellos la foto y el nombre.

Si se pulsa sobre uno de los clientes, se abre una pantalla con el detalle del cliente elegido, donde se muestra: la foto, el nombre, el correo y el listado de bonos activos que tiene.

Desde esta misma pantalla de detalle, a través de unos iconos situados al lado de cada bono, se puede fichar y des fichar (por si ha habido algún error) al cliente en un bono determinado.

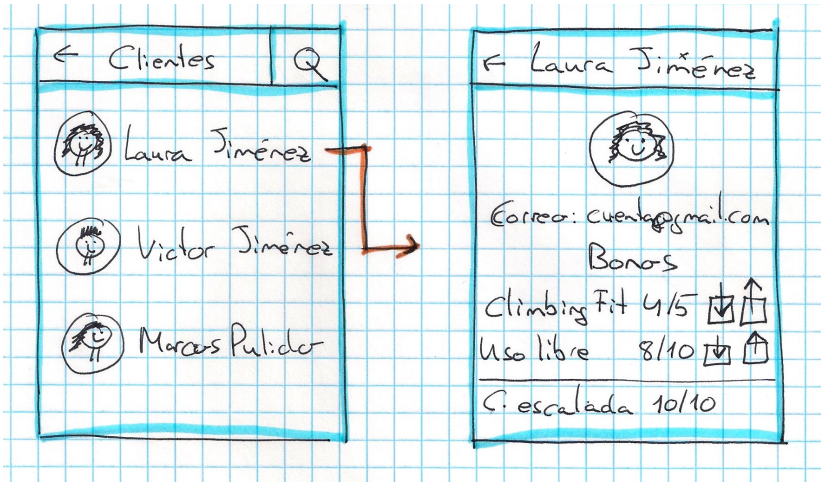


Ilustración 11: Boceto clientes

Mis bonos

Esta pantalla es exclusiva para los clientes, y muestra un listado de bonos con información sobre el consumo de los usos y la fecha de expiración. Desde esta misma pantalla se puede acceder a los bonos ya gastados pulsando el botón “Terminados”

Si se pulsa sobre uno de los bonos, aparece otra pantalla que muestra un histórico de fichajes de ese bono, la empresa que vendió el bono y la fecha en la que lo vendió.

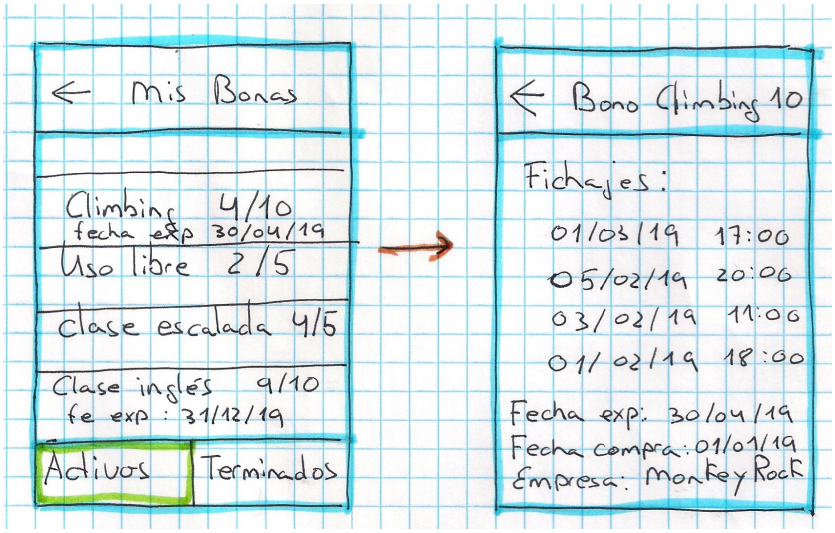


Ilustración 12: Boceto mis bonos

Histórico de fichajes para el cliente

Esta pantalla es exclusiva para los clientes, y muestra un listado de fichajes realizados.

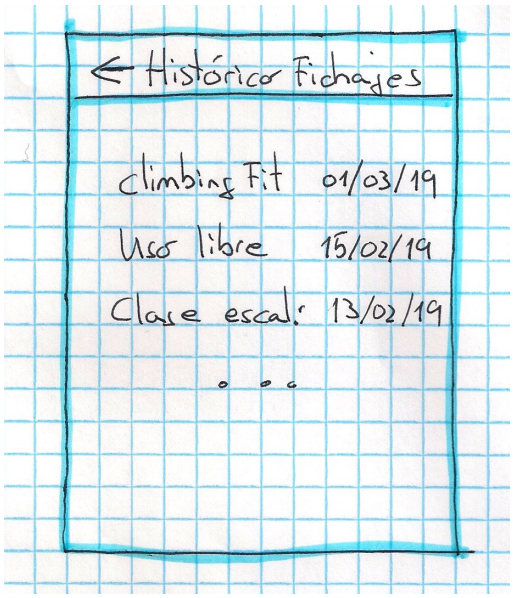


Ilustración 13: Boceto histórico de fichajes del cliente

Prototipo de alta fidelidad

A partir del prototipo de baja fidelidad y de la funcionalidad recogida, en esta fase se pretende alcanzar una representación visual más cercana al producto.

Pantalla de inicio de sesión

Como se ha explicado anteriormente, si el usuario no está identificado, la primera pantalla que muestra la aplicación es el de inicio de sesión con cuentas Google.

Al pulsar iniciar sesión se muestra un listado de cuentas a elegir y la pregunta al usuario si es cliente o empresa.

Ilustración 15: Inicio sesión

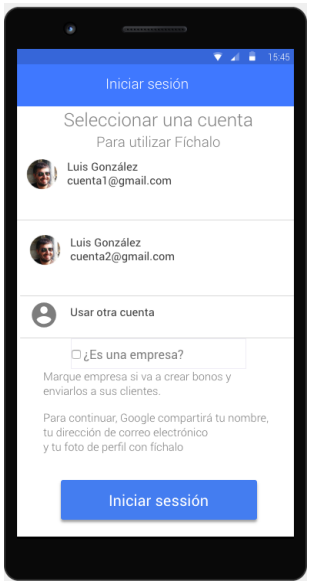


Ilustración 14: Elegir cuenta

Inicio de aplicación

Dependiendo del perfil del usuario se mostrarán las opciones de empresa o de cliente. Una vez identificado, el usuario pulsando el menú arriba a la izquierda, puede seleccionar la opción salir que le permite cerrar la sesión actual.

Ilustración 17: Inicio app empresa

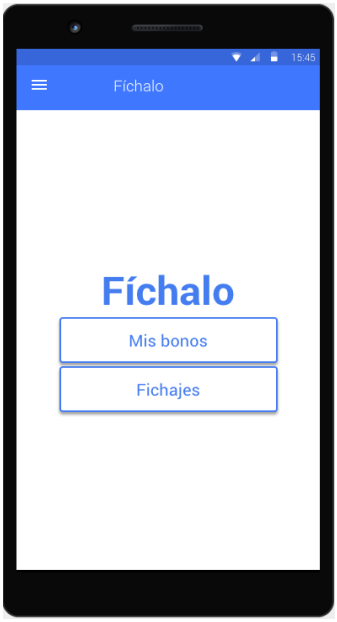


Ilustración 16: Inicio app cliente

Fichar clientes

La acción fichar clientes muestra un listado de clientes a seleccionar. Cuando el cliente seleccionado tiene un solo bono, el fichaje se realiza en el único bono que tiene disponible de forma automática (previa confirmación).

Si el cliente tiene más de un bono, se muestra una pantalla con información del cliente y un listado de bonos para elegir cual fichar.

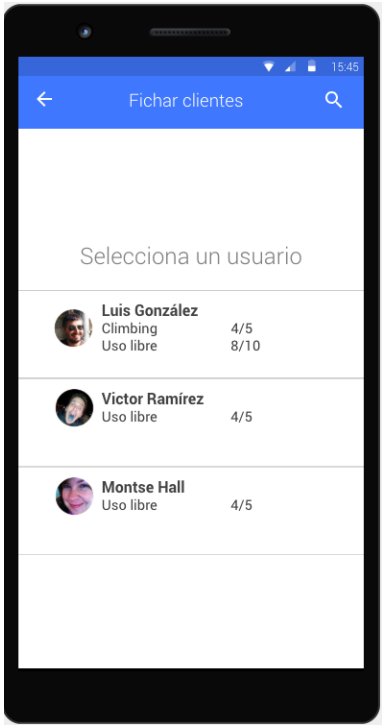


Ilustración 18: Seleccionar usuario para fichar

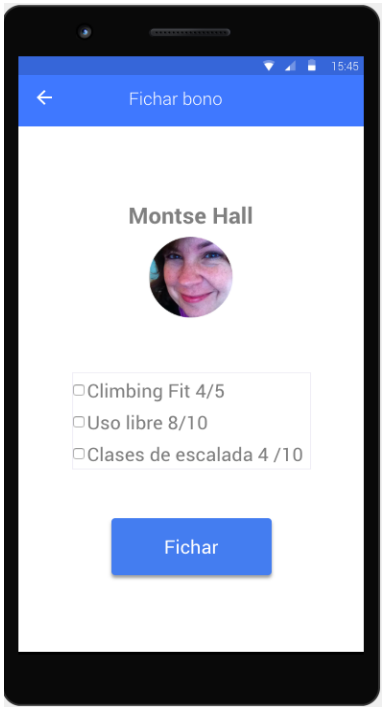


Ilustración 19: Fichar bono

Histórico de fichajes para empresa

Esta pantalla muestra agrupados por día un listado de los fichajes que se han ido sucediendo. De cada fichaje se muestra información del usuario que ha fichado, el bono y la hora.

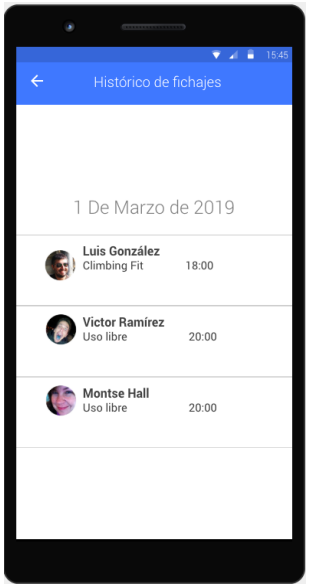


Ilustración 20: Histórico fichajes empresa

Enviar bonos

Continuando con las acciones sobre los clientes, la primera pantalla es un listado de clientes para elegir a cual se le envía el bono. Una vez elegido al cliente, la siguiente pantalla muestra el tipo de bono a enviar.

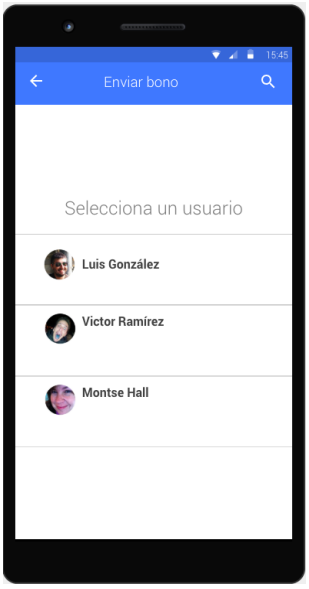


Ilustración 21: Seleccionar usuario para enviar bono

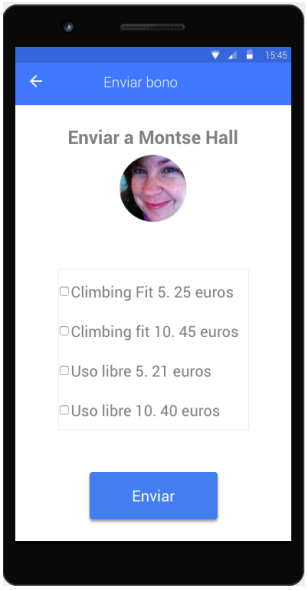


Ilustración 22: Enviar bono

Tipos de bonos

Esta pantalla permite a la empresa ver un listado de tipos de bonos que tiene dados de alta. Además pulsando sobre el icono de la barra superior “+”, accede a otra pantalla para crear un tipo de bono nuevo.

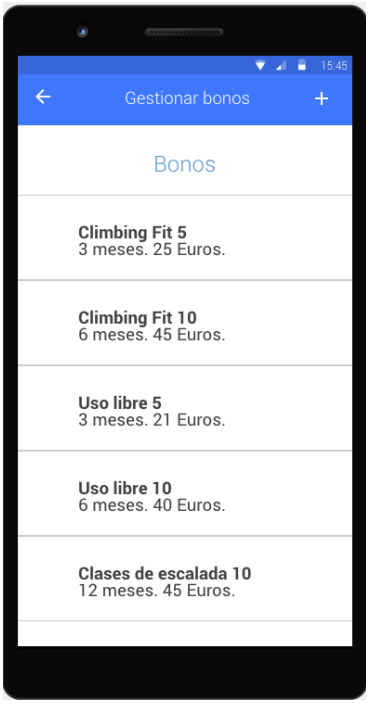


Ilustración 24: Ver tipos de bonos

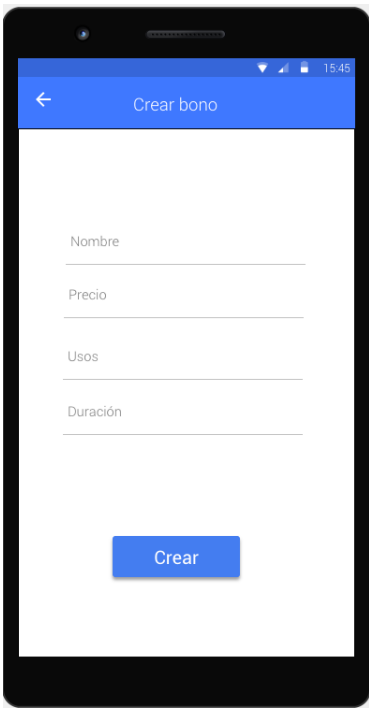


Ilustración 23: Crear tipo de bono

Clientes

Esta pantalla permite a la empresa consultar información sobre los clientes. En la primera pantalla puede ver un listado de clientes, y pulsando sobre uno de ellos accede a otra pantalla que muestra el detalle. Esta última pantalla no es solo de consulta, ya que además de ver el listado de bonos que posee el cliente, puede en cada uno de ellos realizar un fichaje o des fichar en caso de error.

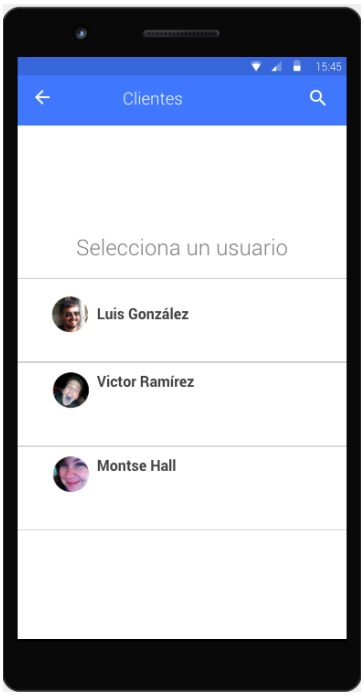


Ilustración 25: Listado de clientes

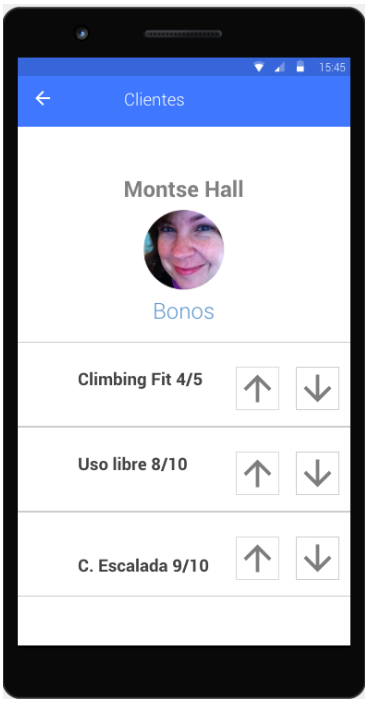


Ilustración 26: Ver

Mis bonos

Esta pantalla muestra a un cliente los bonos que tiene y el estado de cada uno de ellos. Si pulsa sobre uno de ellos, accederá a otra pantalla que muestra en detalle toda la información del bono: usos disponibles, empresa vendedora, fecha de compra, fecha de expiración y fichajes.

Ilustración 28: Ver mis bonos

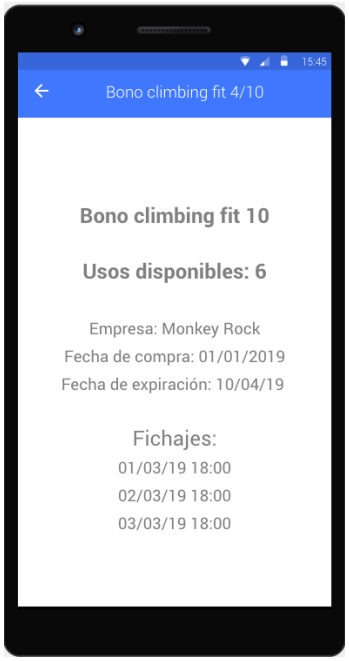


Ilustración 27: Ver detalle de bono

Histórico de fichajes para el cliente

Esta pantalla le permite al cliente consultar el uso realizado de los bonos, ya que puede ver un histórico de los bonos fichados, el día y la hora.

Ilustración 29: Histórico de fichajes

2.1.4 Evaluación

Esta fase, como su nombre indica, se trata de realizar una serie de técnicas de evaluación para detectar rápidamente errores o puntos de mejora en el diseño inicial.

El método escogido es un procedimiento iterativo, donde se modifica el diseño y se evalúa una y otra vez hasta obtener el mejor resultado posible.

Aunque el proceso es iterativo, durante el diseño centrado en el usuario se ha realizado una primera evaluación mientras se creaban los prototipos de baja y alta fidelidad. Esta evaluación ha consistido en hacer partícipe a varios usuarios tanto empresas como clientes en la elaboración de los prototipos.

Los usuarios de tipo empresas han coincidido que lo primero que desean ver cuando realizan las acciones como fichar y enviar bonos, es una lista de clientes para elegir sobre el que desean realizar la acción. Además necesitan que estas acciones sean sencillas, rápidas y que no requieran muchos pasos para llevarlas a cabo.

En un principio, la sección clientes se había concebido solo para mostrar información, pero los usuarios empresas han coincidido que estaría bien que desde esa pantalla también se pudiesen realizar acciones como fichar y des fichar. Por lo tanto las acciones fichar y enviar bonos quedarían como una especie de accesos directos, y el área de clientes sería como una zona donde se podrían realizar todas las acciones disponibles sobre un cliente.

Por otro lado, los usuarios de tipo clientes dan mucha importancia a conocer claramente cuando va a expirar el bono y cuantos usos tienen disponibles. También es relevante para ellos poder acceder a la información del histórico del uso de los bonos.

El proceso de evaluación a seguir durante la implementación consta de las siguientes fases:

1. Mostrar a varios usuarios (mediante un prototipo de alta fidelidad interactivo) varias pantallas de una funcionalidad específica.
2. Observar al usuario durante el manejo y anotar sus dudas y observaciones. Realizar también preguntas sobre posibles alternativas de mejora.
3. Rediseñar las pantallas con la información recogida en el punto anterior.
4. Implementar las pantallas rediseñadas del prototipo de alta fidelidad.
5. Hacer que el usuario use en la aplicación las pantallas ya implementadas. Con la información obtenida de este uso, volver a iterar en el proceso de rediseño si es necesario.
6. Comenzar todo el proceso con otro conjunto de pantallas que implementan otra funcionalidad.

2.2.Diseño técnico

2.2.1 Casos de uso

A continuación se detalla el catálogo de casos de uso que nos permite identificar el comportamiento de la aplicación dependiendo del tipo de usuario.

Caso de uso inicio de sesión

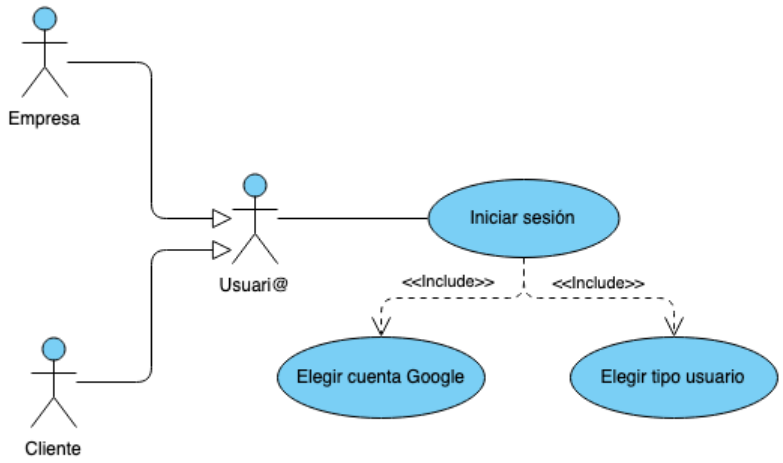


Ilustración 30: Caso de uso iniciar sesión

Identificador	CU-001
Nombre	**Iniciar sesión**
Actor	Usuario (empresa y cliente)
Descripción	El usuario inicia la sesión en la aplicación mediante una cuenta google.
Precondiciones	1. Tener la aplicación instalada. 2. Poseer al menos una cuenta google dada de alta en el dispositivo.
Escenario de éxito principal	1. Abrir la aplicación. 2. Pulsar el botón “Iniciar sesión” 3. Elegir una cuenta google y tipo de usuario (empresa o cliente) 4. Pulsar botón “Iniciar sesión”
Post condiciones	Se da de alta al usuario con la cuenta google elegida. El usuario accede a la aplicación y puede comenzar a usarla.

Caso de uso fichar cliente

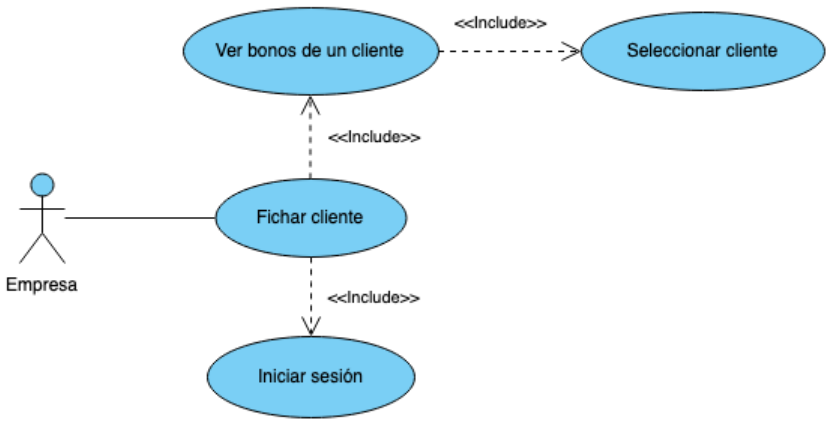


Ilustración 31: Caso de uso fichar cliente

Identificador	**CU-002**
Nombre	**Fichar cliente**
Actor	Usuario (empresa)
Descripción	La empresa ficha un bono determinado de un cliente.
Precondiciones	1. El usuario ha iniciado la sesión como empresa, y se encuentra en la pantalla principal. 2. El cliente al que se va a fichar tiene al menos un bono que la empresa le ha enviado, y este bono tiene que tener usos disponibles.
Escenario de éxito principal	1. Pulsar el botón “Fichar” 2. Seleccionar un cliente. 3. Si el cliente tiene más de un bono, seleccionar el bono que se quiere fichar.
Post condiciones	Se restará un uso en el bono del cliente elegido. El cliente verá el bono actualizado.

Caso de uso ver histórico de fichajes de empresa

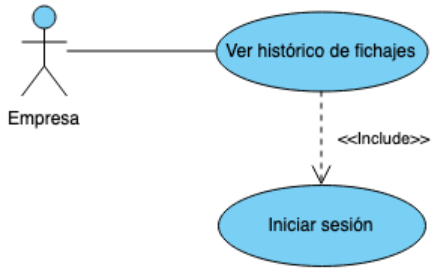


Ilustración 32: Caso de uso ver histórico de fichajes

Identificador	CU-003
Nombre	**Ver histórico de fichajes**
Actor	Usuario (empresa)
Descripción	La empresa ve un histórico de fichajes de sus clientes.
Precondiciones	1. El usuario ha iniciado la sesión como empresa y se encuentra en la pantalla principal. 2. La empresa a realizado fichajes a sus clientes.
Escenario de éxito principal	1. Pulsa el botón “Fichajes” 2. El usuario empresa puede ver un listado de fichajes, donde puede ver información del cliente, del bono y la hora del fichaje.

Caso de uso enviar bonos

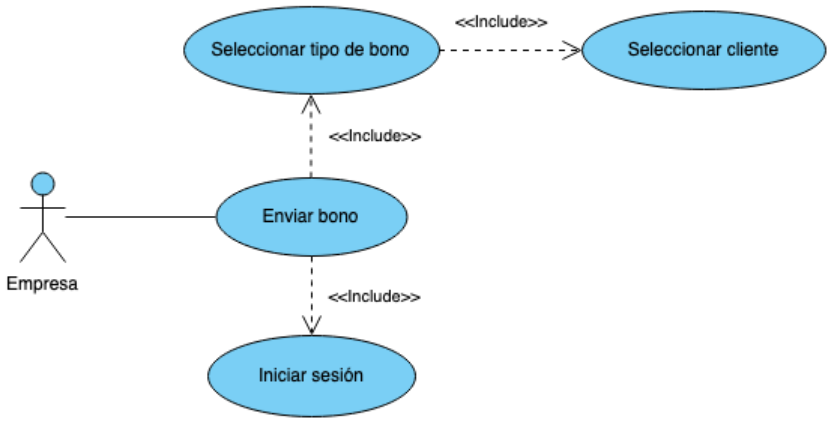


Ilustración 33: Caso de uso enviar bono

Identificador	CU-004
Nombre	**Enviar bono**
Actor	Usuario (empresa)
Descripción	La empresa envía un bono a un cliente.
Precondiciones	1. El usuario ha iniciado la sesión como empresa, y se encuentra en la pantalla principal. 2. La empresa ha creado al menos un tipo de bono.
Escenario de éxito principal	1. Pulsar el botón “Enviar bonos” 2. Seleccionar un cliente. 3. Seleccionar un tipo de bono. 4. Pulsar el botón “Enviar”
Post condiciones	1. El cliente tendrá disponible el bono enviado. 2. El cliente recibirá una notificación.

Caso de uso crear bono

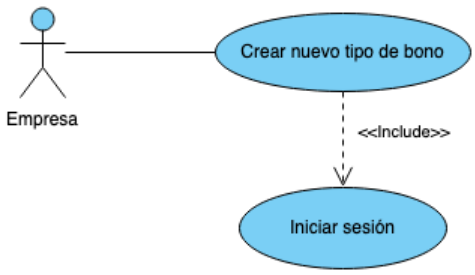


Ilustración 34: Caso de uso crear bono

Identificador	CU-005
Nombre	**Crear bono**
Actor	Usuario (empresa)
Descripción	La empresa crea un tipo de bono nuevo.
Precondiciones	1. El usuario ha iniciado la sesión como empresa, y se encuentra en la pantalla principal.
Escenario de éxito principal	1. Pulsar el botón “Bonos” 2. Pulsar el botón “+” 3. Rellenar el formulario con los datos del nuevo bono. 4. Pulsar el botón “Crear”
Post condiciones	Se creará un tipo de bono nuevo disponible para su envío.

Caso de ver cliente

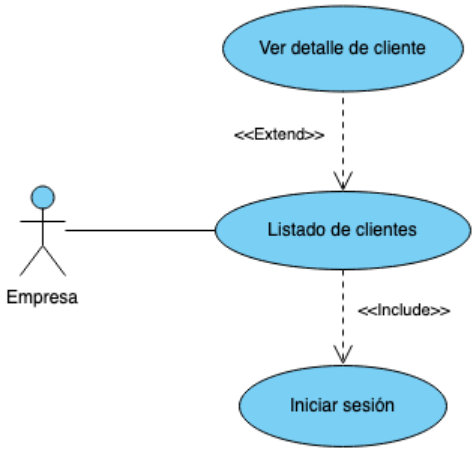


Ilustración 35: Caso de uso ver cliente

Identificador	**CU-006**
Nombre	**Ver cliente**
Actor	Usuario (empresa)
Descripción	La empresa ve información detallada sobre un cliente.
Precondiciones	1. El usuario ha iniciado la sesión como empresa, y se encuentra en la pantalla principal. 2. La empresa ha enviado anteriormente a un usuario al menos un bono.
Escenario de éxito principal	1. Pulsar el botón “Clientes” 2. Pulsar en uno de los clientes mostrados en el listado. 3. Visualizar información detallada del cliente.

Caso de uso ver mis bonos

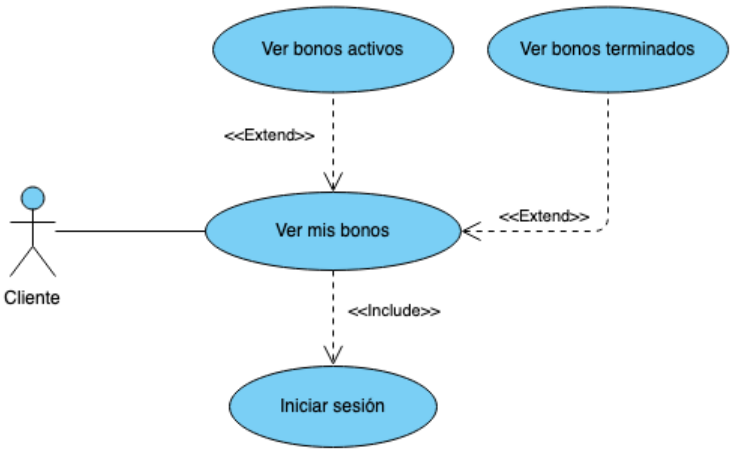


Ilustración 36: Caso de uso ver mis bonos

Identificador	**CU-007**
Nombre	**Ver mis bonos**
Actor	Usuario (cliente)
Descripción	El cliente ve el listado de bonos que tiene disponible.
Precondiciones	3. El usuario ha iniciado la sesión y se encuentra en la pantalla principal. 4. El cliente tiene o ha tenido bonos.
Escenario de éxito principal	1. Pulsar el botón “Mis bonos” 2. El usuario cliente puede ver un listado de bonos disponibles, pudiendo filtrar entre activos y terminados. 3. El usuario puede ver información detallada de cada bono: usos disponibles, fecha de compra, fecha de caducidad, etc.

Caso de uso ver mis fichajes

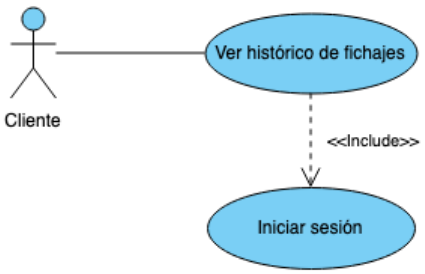


Ilustración 37: Caso de uso ver histórico de fichajes

Identificador	CU-008
Nombre	**Ver histórico de fichajes**
Actor	Usuario (cliente)
Descripción	El cliente ve un histórico de fichajes que ha realizado.
Precondiciones	5. El usuario ha iniciado la sesión y se encuentra en la pantalla principal. 6. El cliente tiene fichajes realizados.
Escenario de éxito principal	1. Pulsar el botón “Fichajes” 2. El usuario cliente puede ver un listado de fichajes, donde puede ver información del bono y la hora del fichaje.

2.2.2 Arquitectura del sistema

El objetivo de este apartado es el de definir la arquitectura del sistema, identificando las entidades que se representarán en la base de datos, las clases y la arquitectura del sistema.

Clases del modelo de datos

La aplicación tendrá las siguientes clases que representan el modelo de datos:

Usuario: Representa un usuario en la aplicación y puede ser de tipo empresa o cliente. Esta clase tiene los siguientes campos:

- Nombre.
- Foto de perfil.
- Correo electrónico del usuario.

- Tipo de usuario: cliente o empresa.
- Bonos: si es un usuario de tipo cliente, tendrá un listado de bonos.
- Tipos de bonos: si es un usuario de tipo empresa, tendrá un listado de tipos de bonos que ha creado.

Tipo de bono: representa un tipo de bono creado por un usuario empresa. Esta clase tiene los siguientes campos:

- **Nombre:** nombre del bono.
- **Usos:** número de usos del bono.
- **Duración:** el tiempo en meses que un usuario puede disfrutar del bono.
- **Precio:** el precio del bono.
- **Empresa**: usuario empresa al que pertenece el bono.

Bono: representa un bono comprado por un usuario y tiene los siguientes campos:

- **Tipo de bono:** tipo de bono al que pertenece este bono.
- **Usos consumidos:** usos consumidos por el cliente.
- **Usuario:** usuario al que pertenece el bono.
- **Fecha de caducidad:** representa la fecha en la que caduca el bono y que ya no podrá ser usado.

La información referente a los usos disponibles, podría ser calcula con un método que consulte el total de usos del tipo de bono y le reste los usos consumidos. Para facilitar el cálculo de los usos disponibles, el bono podría tener un campo total de usos cuyo valor sea copia del valor del tipo del bono.

Igualmente el nombre del bono se podría obtener con un método que consulte el nombre del tipo del bono, o podría ser un atributo con una copia del nombre del tipo del bono.

Fichaje: representa un fichaje de un bono de un cliente y tiene los siguientes campos:

- **Fecha y hora del fichaje.**
- **Bono:** bono donde se ha realizado el fichaje.

Esta clase tendrá varios métodos que pueden obtener (a través del bono del fichaje), el nombre del bono y el usuario del fichaje.

A continuación se muestra un diagrama de clases y la relación entre ellas.

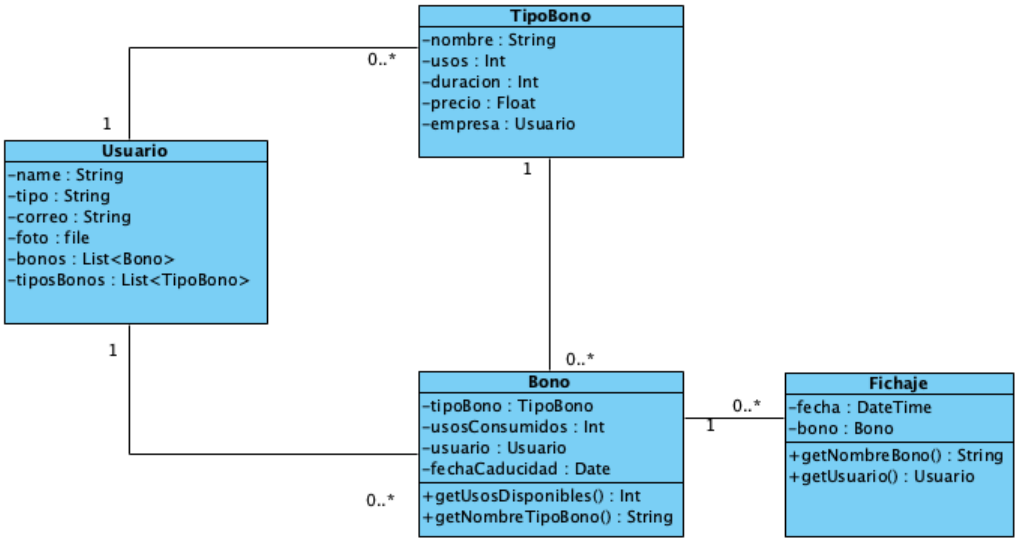


Ilustración 38: Diagrama de clases del modelo de datos.

En el diagrama se puede observar las siguientes relaciones:

- Un usuario empresa puede tener cero o más tipos de bonos, pero un tipo de bono solo puede pertenecer a un usuario empresa.
- Un usuario cliente puede tener cero o más bonos, pero un bono solo puede pertenecer a un cliente.
- Pueden existir cero o más bonos de un tipo de bono, pero un bono solo puede pertenecer a un tipo.
- Un bono puede tener cero o más fichajes, pero un fichaje solo puede pertenecer a un bono.

Base de datos

La base de datos guardará toda la información necesaria para que la aplicación funcione correctamente. Esta información será almacenada en una base de datos local, que a su vez será sincronizada con una base de datos en la nube.

El modelo entidad relación diseñado se ha representado en el apartado anterior mediante el diagrama de clases del modelo de datos, mostrando las relaciones entre las distintas clases. Estas clases se mapearán a tablas en la base de datos.

Diagrama de arquitectura del sistema

Para el desarrollo de la App se va a utilizar el patrón MVC (modelo-vista-controlador).

El patrón de software MVC es un patrón de arquitectura que tiene como misión separar los datos y la lógica de negocio de una aplicación de la vista o interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones.

Descripción de los componentes MVC:

- La vista, es la encargada de mostrar las interfaces de usuario, con o sin datos, y dejar que el usuario interactúe con ellas.
- El controlador controla todo lo que se puede realizar en la aplicación. Responde a las acciones de los usuarios en la vista. Además, es el elemento que hace de vínculo entre la vista y el modelo de datos.
- El modelo es el encargado de trabajar con los datos. Esta capa puede trabajar con una base de datos local que puede ser sincronizada en bases de datos en la nube como por ejemplo Firebase.

Algunas ventajas de utilizar un modelo MVC son:

- La implementación se realiza de forma modular, esto permite que las aplicaciones sean más escalables y facilitan su mantenimiento.
- La independencia de las vistas con los datos hace que no sea necesario cambiar el modelo de datos cuando se necesite cambiar la representación de los mismos.
- Permite a desarrolladores especializados encargarse de una capa determinada, por lo que puede aumentar la productividad.
- Patrón muy estandarizado y adoptado en el mundo de desarrollo.

A continuación se muestra un diagrama la arquitectura MVC.

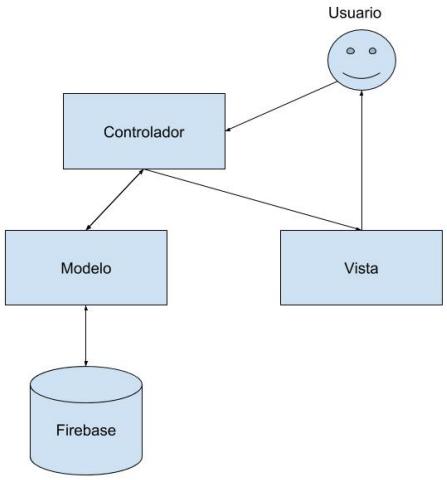


Ilustración 39: Arquitectura modelo-vista-controlador

Diagrama de paquetes

En este apartado se presenta el diseño de los distintos componentes de la aplicación y la dependencia entre los mismos, mediante un diagrama de paquetes. Igualmente, se muestra la dependencia con los componentes externos necesarios para la autenticación y el almacenamiento de los datos en la nube.

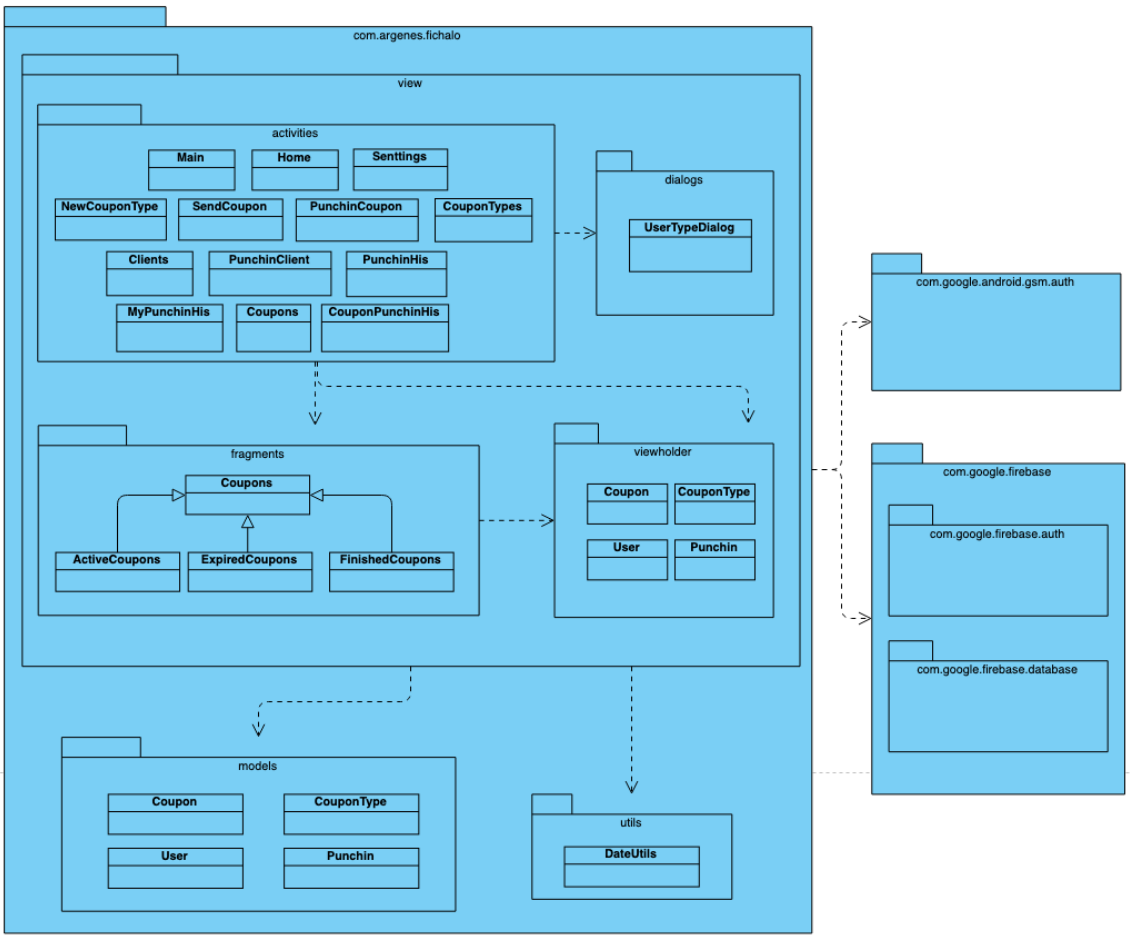


Ilustración 40: Arquitectura de componentes

com.argenes.fichalo es el paquete raíz donde vamos a englobar todos los componentes de la aplicación. En él encontraremos los siguientes sub-paquetes:

- **view:** este paquete incluye todas las clases necesarias para implementar tanto la vista como el controlador de la aplicación. En él se pueden encontrar los siguientes sub-paquetes:
  - **activities:** engloba todos los elementos Activity de la app. En la mayoría de los casos cada Activity corresponde a una vista (layout), aunque no es requisito obligatorio vincular la parte visual a un Activity.
  - **dialogs:** agrupa los elementos necesarios para preguntar al usuario por una determinada configuración, como por ejemplo el tipo de usuario.
  - **viewholder:** agrupa los clases que permiten definir la información y el comportamiento de los elementos dentro de los distintos listados: usuarios, bonos, tipos de bonos y fichajes.

    - **fragments:** agrupa fragmentos de código reutilizables usados en las activities, como por ejemplo los listado de bonos agrupados por estados.
- **model:** agrupa las clases que representan el modelo de datos**:** usuarios, bonos, tipos de bonos y fichajes.
- **utils:** paquete de utilidades que almacena las clases necesarias para el tratamiento de fechas.

Las principales clases Activity son:

- **Main**: es la actividad principal que comprobará si el usuario ha iniciado o no la sesión. Si el usuario ha iniciado la sesión redirigirá a la activity home y si no, mostrará el botón de inicio de sesión con una cuenta Google.
- **Home**: muestra el menú dependiendo del tipo de usuario.
- **Settings:** gestiona la configuración del tipo de usuario.
- **NewCouponType:** gestiona la creación de un nuevo tipo de bono.
- **SendCoupon:** envía un bono a un cliente.
- **PunchinClient:** presenta un listado de usuarios para escoger uno a fichar.
- **PunchinCoupon:** presenta un listado de bonos para fichar uno de ellos.
- **CouponTypes:** muestra los tipos de bonos de una empresa.
- **Clients:** muestra un listado de clientes de una empresa.
- **PunchinHis:** muestra un listado de fichajes producidos en bonos de una empresa.
- **Coupons:** muestra un listado de bonos de un cliente de un usuario determinado.
- **MyPunchinHis:** muestra el listado de fichajes de un cliente.
- **CouponPunchinHis:** muestra el listado de fichajes de un bono.

El paquete raíz de la aplicación depende de los componentes externos com.google.firebase (para la autenticación y sincronización de los datos de la aplicación en la nube), y de com.google.android.gsm.auth (necesario para la autenticación con las cuentas Google).

El proceso de autenticación con cuentas Google sigue los siguientes pasos:

- El usuario pulsa el botón de inicio de sesión con Google, que abre la pantalla de inicio de sesión de Google.
- El usuario inicia la sesión, y sus credenciales se devuelven a la aplicación.
- Se usan las credenciales de google para obtener las credenciales de autenticación de Firebase que se envían a Firebase para autenticar al usuario.
- Una vez las credenciales del usuario haya autenticado con éxito en Firebase se ejecuta una función callback para finalizar el proceso de inicio de sesión.

A continuación se muestra un diagrama del proceso:

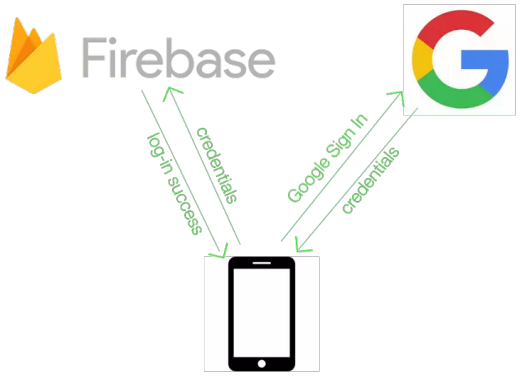


Ilustración 41: Autenticación con Google y Firebase

3. Implementación

Una vez concluida la fase de diseño, se procede al desarrollo de la aplicación, aprovechando los frameworks, librerías y APIs que ofrece el SDK oficial de Android.

3.1 Entorno de desarrollo

El proyecto se ha desarrollado en un MackBook Air con sistema operativo MacOS Mojave, haciendo uso de los siguientes componentes software:

- **Android Studio:** entorno de desarrollo oficial para la plataforma Android. He elegido el lenguaje de programación Java frente a Kotlin principalmente por que tengo una mayor experiencia.
- **Git:** sistema de control de versiones usando el cliente Source Tree que permite la creación de ramas features haciendo uso de git flow.

3.2 Desarrollo de la aplicación

En este apartado se justifican todas las decisiones tomadas durante el proceso de desarrollo.

Una vez listo el entorno de desarrollo, se comienza la creación del proyecto mediante el asistente de Android Studio.

3.2.1 Dependencias

Versión mínima del SDK

En uno de los pasos de la creación del proyecto, hay que especificar cual será la versión mínima de la API en la que se podrá ejecutar la aplicación. Se escoge la que cubre prácticamente el 100% de los dispositivos que es la API 14, es decir, la aplicación se podría ejecutar en dispositivos a partir de la versión Android 4.0 (IceCreamSandwich). Finalmente, debido a la dependencia de firebase-ui-database,

es obligatorio subir la versión mínima a la API 16 que cubrirá el 99,6%, cifra que sigue siendo muy elevada.

Dependencias de la aplicación

Las dependencias más importantes de la aplicación son referentes a la comunicación con el sistema de autenticación y base de datos en la nube:

- *'com.google.firebase:firebase-core:16.0.8':* necesaria para el uso de los distintos servicios Firebase.
- *'com.google.firebase:firebase-auth:16.2.1':* permite crear un sistema de autenticación con posibilidad de elegir distintos modos: usuario y contraseña, email, proveedores como google, twitter, etc.
- *'com.google.firebase:firebase-database:16.1.0':* permite la lectura y escritura de base de datos de google en la nube.
- *'com.firebaseui:firebase-ui-database:4.3.2':* facilita la integración de los servicios Firebase con la interfaz de la aplicación, como por ejemplo mostrar un listado a partir de una consulta a Firebase.
- 'com.squareup.picasso:picasso:2.71828': usada para mostrar las fotos de los usuarios a partir de la url de la imagen. Al introducir esta librería se produce un conflicto al estar presente la librería *"exifinterface"* en su versión 27 y 28. Este conflicto se ha solucionado excluyendo como dependencia de picasso la librería *"exifinterface"*
- El resto de las dependencias son las oficiales de google para el desarrollo de las interfaces, como por ejemplo cardview, recyclerview, constraint-layout, etc.

3.2.2 Integración con Firebase

Para realizar la integración con Firebase, he seguido minuciosamente la documentación oficial de Firebase [4].

Creación del proyecto Firebase

El primer paso para poder usar los servicios de autenticación y base de datos, es agregar Firebase al proyecto Android. Aquí he tenido algunos problemas en el paso final donde se valida la comunicación entre el proyecto android y el proyecto Firebase creado. Estos problemas eran debido a que el fichero google-service.json aunque estaba dentro de la app, el proyecto no conseguía hacer uso de él. Se solucionó reiniciando Android Studio y seleccionando la opción "Sync with filesystem"

Autenticación

Una vez conectada la aplicación al proyecto Firebase, es necesario decidir que tipo de autenticación se va a usar en la aplicación. Entre los distintos sistemas he elegido "Acceso con Google".

Al haber realizado la aplicación para dispositivos Android, todos los usuarios ya tienen una cuenta google con la que realizar el acceso. Esto permite a los usuarios hacer login sin necesidad de registrarse o introducir usuario y contraseña, sólo necesitan pulsar un botón y elegir la cuenta Google con la que quieren acceder a la aplicación.

Entre otros pasos, ha sido necesario habilitar el método de acceso con Google en la consola Firebase en la sección Auth.

La actividad principal “MainActivity”, se encarga de realizar la autenticación haciendo uso de la librería “firebase-auth”. Si la autenticación tiene éxito, como se ha explicado en el diseño, se inicia la actividad del menú “HomeActivity”, se actualizan los datos del usuario autenticado en la interfaz y se muestra el menú principal de opciones.

Base de datos

El siguiente paso es la integración de la aplicación con la base de datos de Firebase, que nos permite escribir y leer datos en la nube.

Firebase proporciona dos alternativas:

- *Realtime Database:* es una base de datos NoSQL alojada en la nube. Los datos se almacenan en formato JSON y se sincronizan en tiempo real con cada cliente conectado.
- *Cloud Firestore:* es la nueva versión de base de datos de Firebase. Actualmente está en fase beta y mejora a la anterior en un modelo más intuitivo de datos, consultas más ricas y mejor escalabilidad.

Aunque en principio Cloud Firestore pueda tener más ventajas, finalmente la alternativa elegida fue Realtime Database. Las principales razones son:

- Real Database es un producto estable y Cloud Firestore está en fase beta. Con un producto en fase beta tienes que asumir algunos riesgos de comportamiento, por lo que he elegido la seguridad de un producto estable.
- No he usado nunca Cloud Firestore, sin embargo si he realizado alguna prueba con Real Time.

Como se explica en el siguiente apartado, la elección de Realtime Database ha provocado una serie de cambios importantes el diseño técnico inicial.

3.2.3 Desarrollo con Firebase Real Time

En las fases previas al desarrollo, se diseñó una solución que incluía una base de datos relacional local que se sincronizaría en ambas direcciones con la base de datos en la nube.

La razón principal de esta decisión, fue tener experiencia en la gestión de base de datos relacionales SQLite en aplicaciones android y en el tratamiento de la información para mostrar los datos a través de los componentes de android: adapters, recyclerview, etc.

Durante el proceso de estudio de desarrollo con Real Time, llegué a la conclusión de deshacer la base de datos local SQlite, y usar directamente Real Time. Los motivos fueron:

- Al eliminar un intermediario (base de datos local) evitamos posibles problemas de sincronización entre ambas fuentes.
- El código usando sólo Real Time es más simple y menos propenso a errores.
- Evito la necesidad de realizar conversiones de base de datos relacional a no relacional.

- Real Time tiene la posibilidad de trabajar en modo sin conexión de forma transparente, ya que almacena localmente los datos y los actualiza cuando se recupera la conexión. Para activar esta opción sólo es necesario escribir una línea de código:

```
FirebaseDatabase.getInstance().setPersistenceEnabled(true)
;
```

La decisión de usar sólo Real Time, también ha tenido contrapartidas:

- Deja de tener sentido el diseño inicial de base de datos relacional, ya que los datos en Realtime Database se almacenan como objetos JSON. La base de datos puede conceptualizarse como un árbol JSON alojado en la nube. A diferencia de una base de datos de SQL, no hay tablas ni registros.
- Las consultas en Real Time son muy limitadas, por ejemplo, no se pueden realizar ordenaciones en orden inverso y no se pueden realizar consultas por más de un criterio al mismo tiempo. Esto provoca que tengas que cambiar la estructura de los datos en función de las consultas que necesites. Parece que esto se está resolviendo mejor en Cloud Firestore con consultas más ricas.

Estructura de la base de datos

Para realizar la nueva estructura de datos, he seguido la guía oficial de Google “Estructura datos” [5].

A diferencia de una base de datos de SQL, no hay tablas ni registros. Cuando le agregas datos al árbol JSON, estos se convierten en un nodo de la estructura JSON existente con una clave asociada.

Algunas recomendaciones seguidas en la estructuración de los datos:

- **Evitar la anidación de datos:** cuando obtienes datos de una ubicación de la base de datos, también se recuperan todos los nodos secundarios, por lo que el anidamiento no es una buena opción ya que estarías recuperando más información de la que en realidad necesitas.
- **Compactar las estructuras de datos:** al dividir la información en rutas de acceso independientes, que también se conocen como no normalizadas, se pueden descargar de manera eficaz en llamadas separadas, según sea necesario. Esto implica duplicar información en distintas ramas de la base de datos.
- **Crea datos escalables:** crear índices para por ejemplo poder relacionar los clientes con las empresas, y así recuperar los datos de una forma óptima.

En este apartado detallan los distintos nodos de datos que la aplicación crea.

Nodo users

Almacena una lista de nodos usuarios de la aplicación. Para cada nodo se almacena:

- username: nombre del usuario.
- email: correo electrónico.
- PhotoUrl: url de la foto del usuario.
- UserId: id que identifica al usuario en el sistema de autenticación.
- Users: es una lista de empresas de la que el usuario es cliente. La forma de representarlo es mediante una lista clave-valor de índices, donde la clave es el id del usuario empresa y el valor true. Se ha llamado users y no companies, por si en un futuro la aplicación se dirige más a uso colaborativo entre usuarios además de clientes y empresas.

Mediante el nodo de índices users, se puede realizar consultas como la lista de clientes de una empresa:

```
databaseReference.child("users").orderByChild("users/"+ idEmpre-
sa).equalTo(true);
```

Nodo users-coupon-types

Este nodo almacena los tipos de bonos que crea una empresa, y contiene una lista de nodos primarios con el id del usuario que crea el bono, y debajo de estos, cuelga una lista de nodos secundarios que representan los tipos de bonos donde se almacena el nombre, la duración, el id del usuario propietario, el precio y los usos.

Cuando una empresa crea un tipo de bono, se crea un nodo en esta rama de la base de datos.

Con esta estructura se pueden obtener los bonos de una empresa determinada:

```
databaseReference.child("user-coupon-
types").child(idEmpresa).orderByChild("name");
```

Nodo user-coupon

Este nodo almacena información de los bonos que la empresa envía a sus clientes y el estado actual de los mismos. Este nodo lo usan los clientes para consultar la información sobre sus bonos, por lo que el primer nivel del árbol indica el usuario cliente que tiene el bono, y el segundo nivel se divide en dos ramas, una rama “pending” del que cuelgan los bonos activos y una rama “archived” de la que cuelgan los bonos terminados.

De cada bono se almacena la duración, la fecha de expiración calculada, el nombre del bono, la fecha de compra, el id de la empresa que vendió el bono, el id del usuario que tiene el bono, el precio, los usos totales y los usos realizados.

Cuando una empresa envía un bono a un cliente se crea un nodo en esta rama de la base de datos.

Con esta estructura se puede obtener un listado de bonos pendientes de usos y no expirados de un cliente realizando la siguiente consulta:

```
databaseReference.child("user-coupon").
child(idCliente).child("pending").orderByChild("expirationDate")
.startAt(DateUtils.getCurrentDate());
```

Nodo user-coupon-owner

Es una rama gemela a la anterior, pero se usa para almacenar los bonos de un cliente de una empresa determinada. Este nodo es usado para que las empresas puedan ver información de los bonos de sus clientes, por lo que tienen un nivel más en el árbol que especifica el id del usuario empresa.

Cuando una empresa envía un bono a un cliente se crea un nodo en esta rama de la base de datos.

Por ejemplo para obtener un listado de bonos finalizados de un cliente de una empresa determinada, se necesita la siguiente consulta:

```
databaseReference.child("user-coupon-owner")
```

```
                    .child(idEmpresa).child(idCliente).child("ar
chived");
```

Nodos de fichaje

Cada vez que se realiza un fichaje o un des fichaje en un bono, se crean nodos de fichaje en tres ramas distintas:

- Nodo user-punchin: rama que almacena los fichajes a nivel de cliente. La usa el cliente para ver sus fichajes.
- Nodo coupon-punchin: rama que almacena los fichajes a nivel de bono. La usa tanto la empresa como el cliente cuando quiere visualizar los fichajes de un bono determinado.
- Nodo owner-punchin: rama que almacena los fichajes a nivel de empresa, es decir, los fichajes de todos los clientes de una empresa. Es usado por la empresa para visualizar un listado de todos los fichajes de sus clientes.

Por ejemplo para obtener los fichajes de un bono se realizaría la siguiente consulta:

```
databaseReference.child("coupon-punchin")
                .child(idBono).orderByChild("punchinTime");
```

Configuración de la base de datos

Para facilitar el desarrollo y las pruebas, las reglas de seguridad se han definido como públicas. En un futuro, antes de publicar la aplicación habrá que limitarlas.

También se ha decidido que la aplicación no trabaje en modo sin conexión. El motivo es evitar posibles conflictos de fichajes y de la información que visualizan los clientes de sus bonos cuando los usuarios trabajan sin conexión. Aún así se ha dejado preparada la aplicación para cambiar este funcionamiento fácilmente, para ello se ha creado la clase FichaloApplication en la que habría que des comentar la línea:

```
//FirebaseDatabase.getInstance().setPersistenceEnabled(true);
```

Clases para el manejo de datos

Para recoger los datos de Firebase Real Time y facilitar el tratamiento de los mismos se usan las siguientes clases Java:

- **CouponType:** gestiona la información de un tipo de bono creado por una empresa.
- **Coupon:** gestiona la información de un bono de un cliente donde se lleva la cuenta de los usos consumidos.
- **Punchin:** gestiona la información de un fichaje, ya sea a nivel de empresa, de cliente o de bono.
- **User:** gestiona la información de un usuario.

Cada clase tiene los mismos atributos que los nodos explicados en el apartado anterior de estructura de base de datos.

Gestión de los listados

La presentación de todos los listados en la aplicación siguen el mismo procedimiento:

- Creamos una clase actividad que se encarga de mostrar un listado concreto, por ejemplo los tipos bonos de una empresa: CouponTypeActivity
- En dicha clase creamos un adaptador de Firebase (clase FirebaseRecyclerAdapter de la librería firebase-ui) que va unido a una consulta Firebase (normalmente un listado), y a un ViewHolder que representa el elemento dentro del listado, por ejemplo CouponViewHolder
- En el ViewHolder se definen los campos que se van a visualizar, y las acciones que se pueden realizar sobre el elemento del listado dependiendo del tipo de usuario.

Usar FirebaseRecyclerAdapter tiene la ventaja que hace fácil actualizar automáticamente los elementos del listado cuando la base de datos cambia, pero es muy limitado a la hora filtrar elementos para excluir parte de los datos recibidos de Firebase

Cambios en el diseño de la aplicación

Durante el proceso de desarrollo se han producido una serie de cambios en el diseño centrado en el usuario, algunos por restricciones tecnológicas y otros para mejorar la usabilidad.

Según el diseño inicial, en la pantalla de elección de la cuenta Google se podía elegir el perfil del usuario. Esta elección se ha tenido que realizar después de la selección de la cuenta, ya que el encargado de dicha selección es un Intent de Google.

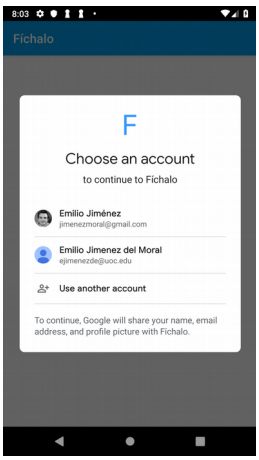


Ilustración 43: Selección de cuenta Google

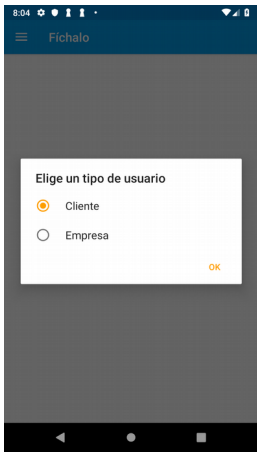


Ilustración 42: Selección tipo de usuario

Como norma general, en los listados se han usado CardView con el fin de mostrar más información y evitar el exceso de navegación a pantallas detalle. Esto ha permitido eliminar pantallas detalle innecesarias.

Para el perfil de empresa, se han unificado las pantallas “Tipos de Bonos” y “Enviar bono”. De esta manera en una única pantalla la empresa puede ver sus tipos de bonos, enviarlos o crear nuevos, reduciendo así la navegación necesaria para realizar este tipo de acciones.

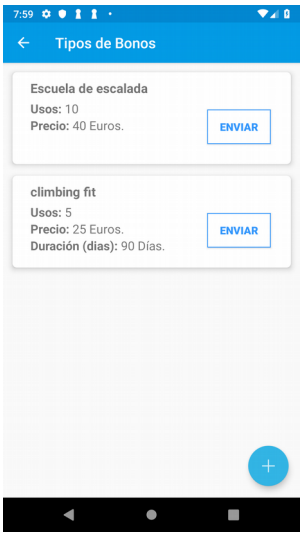


Ilustración 44: Pantalla unificada bonos de empresa

En la opción “Fichar” se ha decidido que independientemente de si el usuario tiene solo un bono o más, te debe llevar a una pantalla de bonos para elegir cual fichar. Así el listado de usuario no tiene tanta información, es más fácil encontrar el usuario elegido y además la pantalla del bono puede mostrar más información.

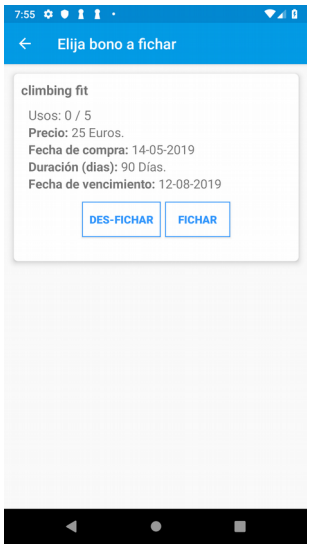


Ilustración 45: Fichar el bono de un cliente

En el listado de bonos que ve el cliente, y el listado de bonos que ve la empresa en la zona de clientes, se han añadido tres secciones:

- Activos: muestra los bonos no expirados que tienen usos disponibles.
- Expirados: muestra los bonos con usos disponibles pero que están expirados.

- Terminados: muestra los bonos que tienen todos los usos consumidos.

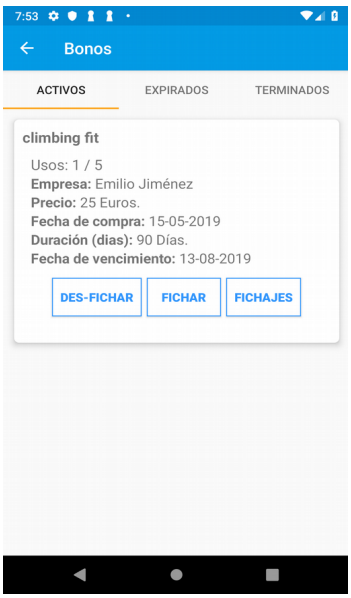


Ilustración 46: Listado de bonos por estado

Estado actual del proyecto

El proyecto ha sufrido retrasos que han hecho tener que aumentar el número de horas inicialmente previstas.

El motivo principal de los retrasos es el cambio de paradigma de base de datos relacional a no relacional, ya que al trabajar con nodos en lugar de tablas requiere un esfuerzo extra pensar en la des normalización y la replicación de la información.

La curva de aprendizaje de Firebase Real Time ha sido más pronunciada de lo esperado, ya que la escritura y lectura se hacen más complejas que las típicas consultas relacionales.

He encontrado limitaciones a la hora de leer la información. En Firebase Real Time no es posible realizar consultas complejas que se compongan de varios filtros de atributos principales y atributos de nodos secundarios al mismo tiempo, por lo que, es necesario replicar la información con unos criterios determinados en otros nodos.

Por los motivos explicados, no se ha podido cumplir al cien por cien lo planificado. La utilidad que se ha descartado y que no influye para que la aplicación cumpla su misión principal, son las notificaciones de fichajes. En realidad las notificaciones son un extra de información, ya que con Real Time, cuando la empresa realiza un fichaje, el cliente ve en la aplicación en tiempo real el fichaje sin necesidad de recargar la pantalla. También ha faltado introducir el filtro de búsqueda en el listado de usuarios.

Teniendo en cuenta el tiempo de desarrollo estimado para implementar las notificaciones, introducir el filtro de búsqueda en el listado de usuarios y que la maquetación de las distintas pantallas se puede mejorar, encuentro que el desarrollo de la aplicación está al 95% de lo inicialmente planificado.

3.3 Pruebas

Para realizar las pruebas se han usado dos cuentas Goolge distintas (una como cliente y otra como empresa), con la finalidad de ver como se actualiza la información de un lado y del otro.

A continuación se listan las distintas pruebas realizadas.

Nombre	Descripción
Autenticación con cuenta Google	La aplicación autentica correctamente con la cuenta Google escogida, y muestra la información del usuario en el menú lateral izquierdo.
Cerra sesión	Con el usuario autenticado, se pulsa sobre el botón de menú, y luego en cerrar sesión. La aplicación vuelve a la pantalla inicial para volver a iniciar la sesión.
Selección de tipo de usuario	Cuando el usuario se autentica, elige el tipo de usuario y se muestra el menú adecuado dependiendo si es empresa o cliente.
Cambio de tipo de usuario	Se elige la opción ajustes del menú lateral izquierdo, se pulsa sobre la opción “General”, se cambia el tipo de usuario. Cuando volvemos a la pantalla inicial de la aplicación el menú ha cambiado para mostrar las opciones del tipo de usuario elegido.
Crear tipo de bono	El usuario empresa pulsa la opción “Enviar bono”, luego pulsa el botón “+”, rellena el formulario y finalmente pulsa el botón de confirmación. La aplicación muestra el listado de tipo de bonos donde aparece el nuevo creado.
Enviar bono	El usuario empresa pulsa la opción “Enviar bono”, pulsa en uno de los bonos en el botón “Enviar” y elige el usuario al que quiere enviar el bono. El bono ha sido enviado y se comprueba entrando en la parte de clientes, eligiendo al cliente, y viendo los bonos activos que tiene. También si cierras la sesión y entras como el usuario cliente, se puede ver el bono pulsando en la opción “Mis bonos”
Fichar y des fichar bono	El usuario empresa pulsa en la opción “Fichar”, elige la persona y luego en el bono pulsa el botón “Fichar”. Se comprueba que aumenta el número de usos del bono. Si pulsa el botón “Des-Fichar” disminuye el número de usos del bono. Se comprueba que los fichajes realizados aparecen en el listado de fichajes de empresa. Se comprueba autenticando como cliente que sus bonos tienen los mismos usos que ve la empresa, y que además aparecen los fichajes en su listado de fichajes.

	Además si el bono se completa, es decir ya no le quedan más usos, se comprueba que este bono se mueve a la pestaña terminados en el área del cliente. Igualmente el cliente ve su bono finalizado.
Ve fichajes de empresa	El usuario empresa pulsa la opción “Fichajes” y puede ver un listado de todos los fichajes de sus clientes mostrando información de fecha, hora y bono fichado.
Ver clientes	El usuario empresa pulsa en la opción “Clientes” elige un cliente, y puede ver los bonos activos, expirados y terminados. De cada bono puede acceder a un listado de fichajes. También puede fichar y des-fichar en los bonos activos y expirados.
Ver mis bonos	El usuario cliente pulsa en la opción “Mis bonos” y puede navegar entre las distintas pestañas de activos, expirados y terminados para ver la información y los fichajes de los distintos bonos.

Todas las pruebas han sido realizadas manualmente con éxito.

4. Conclusiones

El trabajo final de master me ha brindado la oportunidad de crear un producto desde cero a partir de una idea, y me ha aportado una gran cantidad de conocimientos metodológicos de diseño y de desarrollo en tecnologías que desconocía.

Ha supuesto un gran reto poder cumplir la planificación debido al cambio de paradigma de base de datos relacional a no relacional, y a la curva de aprendizaje para gestionar bases de datos en la nube. Los imprevistos y la falta de tiempo, han supuesto un aprendizaje para ser flexible a los cambios y tomar decisiones prácticas.

Debido al sobre coste del tiempo de aprendizaje de nuevas tecnologías, no se ha podido implementar la totalidad de los objetivos, descartando las notificaciones que es un sistema extra de información que no impide las funciones principales de la aplicación.

Ha sido muy enriquecedor trabajar de forma iterativa sobre el diseño de la aplicación, para ir evolucionando el producto hacia las necesidades de los usuarios y aprender de los distintos puntos de vista de los mismos.

Creo que el resultado final puede considerarse un producto mínimo viable sobre el que se pueden realizar distintas mejoras. Algunas de ellas pueden ser:

- Mejorar la maquetación de las distintas pantallas incluyendo elementos gráficos que haga la aplicación más usable y vistosa profesionalmente.
- Incluir filtros en los listados para facilitar la búsqueda de los elementos.
- Incluir un sistema de notificaciones de fichajes.
- Permitir que los clientes puedan ficharse ellos mismos con el previo consentimiento de la empresa.
- Reconocer la localización del cliente (previo consentimiento de permiso) para facilitar el fichaje cuando el cliente se acerque a la localización de la empresa.
- Introducir módulo de reserva de clases.
- Permitir a los usuarios realizar la compra de los bonos desde la misma aplicación móvil.

Además de las mejoras de la aplicación, es muy importante como trabajo futuro elaborar un modelo de negocio que pueda rentabilizar el proyecto.

5. Glosario

- **Android**: es un sistema operativo móvil desarrollado por Google, basado en el Kernel de Linux y otros software de código abierto.

- **Bono:** Tarjeta de abono que da derecho a la utilización de un servicio durante cierto tiempo o un determinado número de veces.
- **Climbing Fit:** entrenamiento dirigido usando la escalada como medio.
- **Firebase:** Plataforma creada por Google para alojar bases de datos de las aplicaciones en la nube.
- **Realm:** sistema de gestión de base de datos locales en las aplicaciones.

6. Bibliografía

[1] Web. statista.com. Global mobile OS market share in sales to end users from 1st quarter 2009 to 2nd quarter 2018 https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/
(Consultado el 27/04/2019)

[2] Web. xatakamovil.com. Así es como Android se ha comido el mercado en diez años.
https://www.xatakamovil.com/sistemas-operativos/asi-como-android-se-ha-comido-mercado-diez-anos
(Consultado el 27/04/2019)

[3] Libro: Donald A. Norman, The design of everyday things, Basic Books, New York, 1988.

[4] Web. firebase.google.com. Documentación oficial de firebase
https://firebase.google.com/docs/android/setup
(Consultado el 25/03/2019)

[5] Web. Estructurar datos Firebase Real Time
(Consultado el 10/04/2019)
https://firebase.google.com/docs/database/android/structure-data

7. Anexos

7.1 Manual de usuario

En este apartado se desarrolla el manual de usuario que expone las distintas opciones que tienen tanto los clientes como las empresas.

7.1.1 Inicio de sesión

Si el usuario no está autenticado, la primera pantalla que muestra la aplicación es el de inicio de sesión con cuentas Google.

Al pulsar iniciar sesión se muestra un listado de cuentas Google a elegir para el inicio de la sesión. El usuario también pueda pulsar la opción “Usar otra cuenta” que iniciará un proceso externo a la aplicación para crear una nueva cuenta en el dispositivo.

Una vez el usuario haya seleccionado una cuenta e iniciado la sesión, la aplicación le preguntará el tipo de usuario y deberá escoger o cliente o empresa.

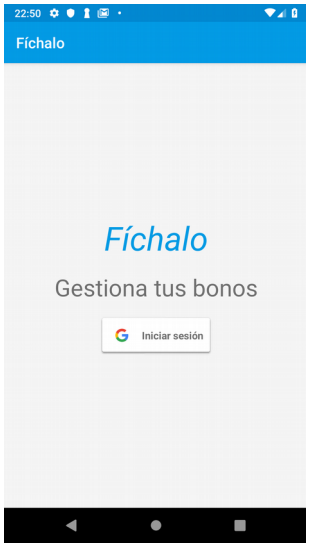


Ilustración 48: Inicio de sesión

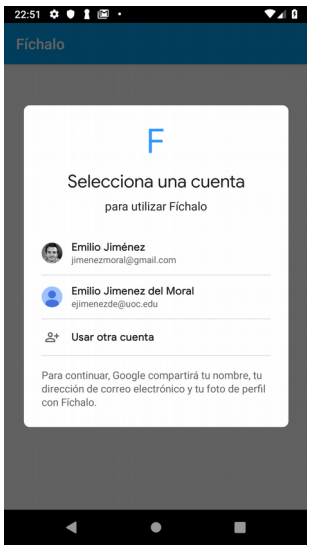


Ilustración 49: Seleccionar cuenta

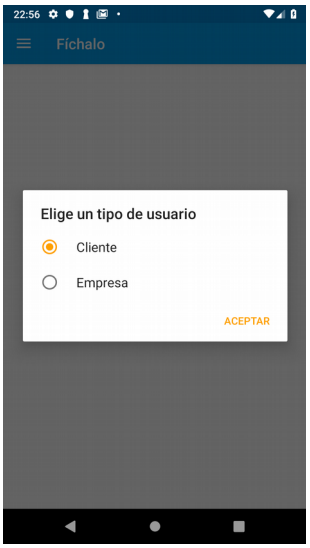


Ilustración 47: Elegir tipo de cliente

Dependiendo del tipo de usuario escogido, la aplicación mostrará un menú distinto con opciones distintas para la empresa y para el cliente.

7.1.2 Empresas

A continuación se muestra el menú para empresas.

Ilustración 50: Menú empresa

Pulsando opción “Enviar bonos”, la empresa puede ver los tipos bonos actualmente dados de alta, enviarlos a los clientes pulsando el botón “Enviar” y crear nuevos tipos de bonos pulsando el botón flotante “+”.

Si pulsamos el botón “Enviar” aparecerá un listado de usuarios para elegir a quien se le va a enviar el bono.

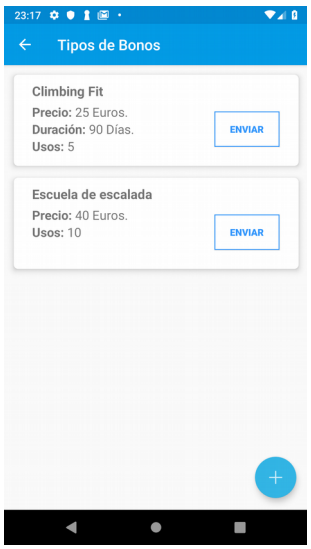


Ilustración 51: Listado de tipos de bonos

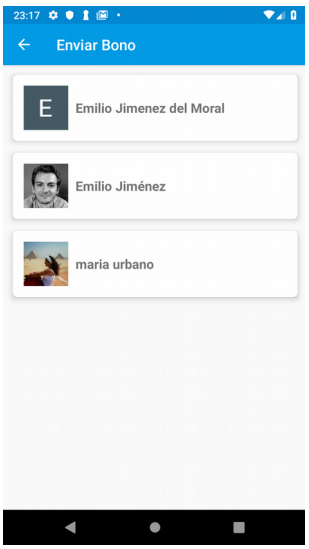


Ilustración 52: Listado de clientes para enviar bono

Si se pulsa el botón “+” se abrirá un formulario para crear un nuevo tipo de bono.

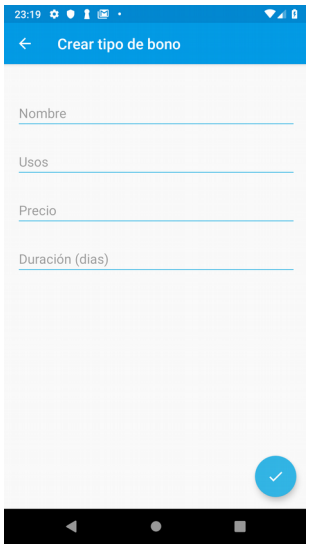


Ilustración 53: Formulario nuevo tipo de bono

La opción “Fichar” muestra una lista de clientes a fichar, y al pulsar en uno de ellos se muestra el listado de bonos del cliente escogido en el que se puede fichar y des-fichar.

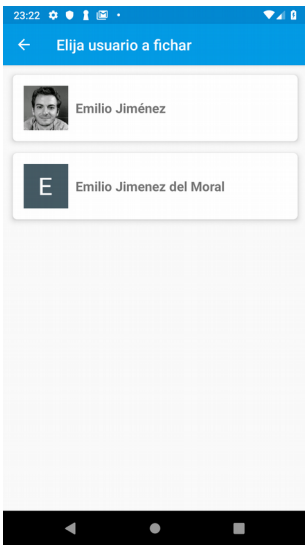


Ilustración 54: Usuarios a fichar

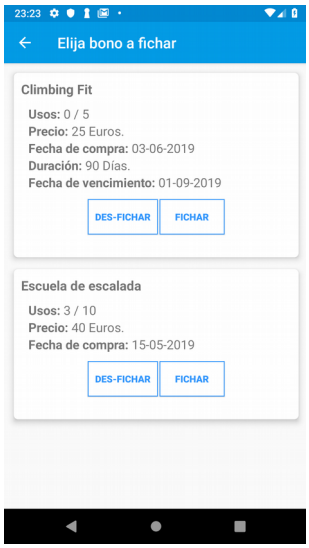


Ilustración 55: Bonos a fichar

La opción “Fichajes” muestra un listado de fichajes de todos los clientes.

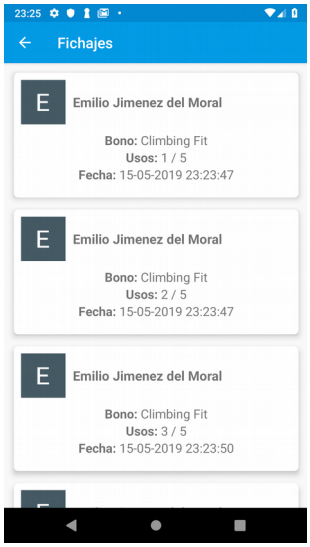


Ilustración 56: Fichajes de empresa

La opción “Clientes” tras elegir a uno de los clientes muestra información detallada sobre sus bonos y los estados de los mismos. Además desde esta pantalla se puede fichar, des-fichar y ver información sobre los fichajes de cada bono.

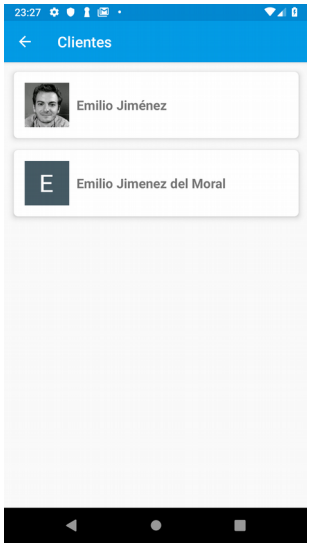


Ilustración 58: Listado de clientes

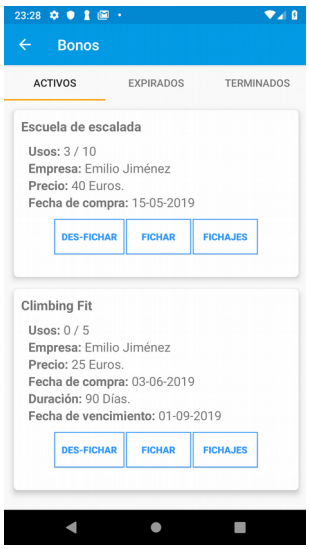


Ilustración 59: Listado de bonos de un cliente

Ilustración 57: Fichajes de un bono

7.1.3 Clientes

El menú para clientes tiene dos opciones, una para ver el estado de los bonos y otra para ver el listado de fichajes.

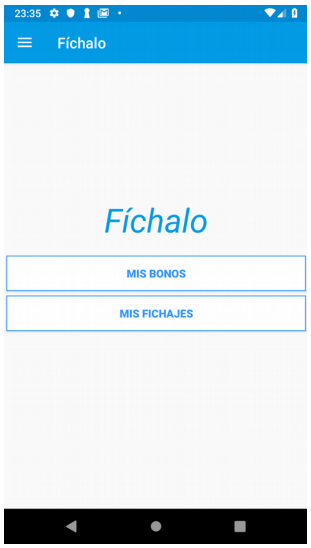


Ilustración 62: Menú cliente

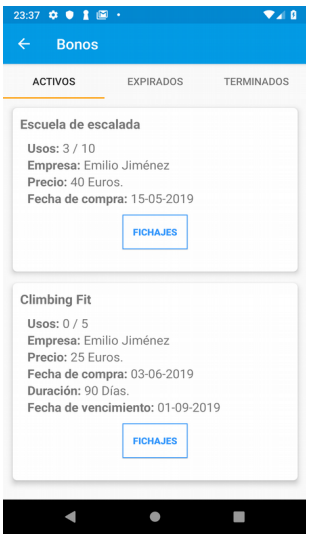


Ilustración 61: Mis bonos

Ilustración 60: Mis fichajes

7.2 Manual de despliegue

7.2.1 Instalar la aplicación

Para instalar la aplicación en tu dispositivo Android solo es necesario ejecutar el archivo .apk.

Para poder ejecutar la aplicación el dispositivo deberá tener como mínimo la versión Android 4.1 Jelly Bean (SDK 16).

7.2.2 Acceder al proyecto y generar Apk

Abrimos el proyecto con Android Studio y pulsamos en la opción de menú “Build” → “Build Apk”.

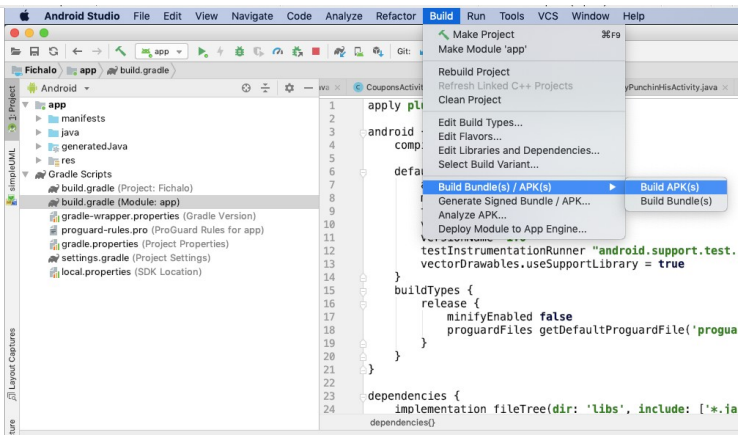


Ilustración 63: Generar apk