



GRADO EN INGENIERÍA INFORMÁTICA

TRABAJO FINAL DE GRADO

FlatCrew: Pagos en grupo

Autor:
Alejandro LUJÁN LÓPEZ

Supervisor:
Jordi NEBOT CARDA
Tutor académico:
Carlos Antonio HERNÁNDEZ
ESPINOSA

Fecha de lectura: 14 de Noviembre de 2014
Curso académico 2013/2014

Resumen

El potente cambio en los hábitos de consumo de la sociedad y el rápido surgimiento y crecimiento de aplicaciones de pago móvil y comercio electrónico, lleva a algunas empresas del sector informático a tomar ventaja de estas tendencias y generar ideas para abarcar nuevos huecos de mercado. Una de estas ideas es FlatCrew, una aplicación de pago en grupo y gestión de tareas derivadas de la convivencia, enfocada a un sector joven de la sociedad, que integra con otras pasarelas de pago seguras. En este documento se describen los procesos de análisis, planificación y diseño que se han llevado a cabo para poner en marcha dicha idea, desarrollando un servicio REST que dota de funcionalidad a una aplicación móvil para dispositivos Android, a cargo de otro equipo de desarrolladores, como parte cliente. Debido al fuerte carácter financiero de la aplicación y al manejo de datos sensibles, se han de tener en cuenta algunos aspectos de seguridad, estudiándose el funcionamiento del protocolo OAuth 2.0 y de qué forma puede integrarse con el servicio implementado. Adicionalmente, se adoptan una serie de tecnologías y herramientas a lo largo del proceso de desarrollo del producto. Entre ellas, se encuentran Git, Jenkins y Gerrit, para control de versiones, integración continua y revisión de código, que se incorporan en la rutina de trabajo para asegurar una buena coordinación entre los miembros del equipo y así mejorar la productividad.

Palabras clave

RESTful, pagos en grupo, monedero, OAuth 2.0, Android

Keywords

RESTful, group payment, wallet, OAuth 2.0, Android

Índice general

1. Introducción	5
1.1. Contexto y motivación	5
1.2. Objetivos	6
1.2.1. Objetivos generales de producción	6
1.2.2. Objetivos específicos de producción	7
1.2.3. Objetivos de aprendizaje	7
2. Descripción del proyecto	9
3. Análisis de requisitos	11
3.1. Mockups	11
3.2. Casos de uso	19
3.3. Requisitos de datos	30
3.4. Plataforma tecnológica	35
4. Restricciones	37
4.1. Tiempo	37
4.2. Tecnología	37
4.3. Experiencia	37
5. Planificación del proyecto	39
5.1. Metodología y definición de tareas	39
5.2. Planificación temporal de las tareas	41
5.3. Estimación de recursos del proyecto	44
6. Diseño del sistema	47
6.1. Diseño del modelo de datos	47
6.2. Arquitectura del servicio	51
6.3. Validación de requisitos	58
7. Implementación y pruebas	61
7.1. Detalles de implementación	61
7.1.1. Transacciones	61
7.1.2. CakePHP: Componentes	62
7.1.3. Implementación OAuth 2.0	63
7.2. Variaciones en la planificación	67
7.3. Validación y pruebas	68
8. Conclusiones	69
9. Agradecimientos	71

1 Introducción

Este documento constituye la memoria del proyecto FlatCrew desarrollado durante la estancia en prácticas en la empresa PaynoPain Solutions S.L [13] , una empresa innovadora en los servicios financieros ubicada en el parque científico, tecnológico y empresarial de la Universitat Jaume I (Español II). Esta empresa está especializada en las aplicaciones y métodos de pago alternativos a través de telefonía, Internet u otras redes de comunicación. PaynoPain proporciona tanto servicios de pago a medida para otras empresas o entidades como aplicaciones enfocadas directamente al usuario final, ofreciendo una pasarela de pago segura, por ejemplo. Además, la empresa colabora con otras empresas de diseño y seguridad informática.

1.1. Contexto y motivación

A lo largo de su trayectoria empresarial PaynoPain ha creado y comercializado una serie de productos para cubrir las necesidades existentes en el mercado tecnológico de soluciones financieras. Uno de sus productos clave a la hora de contextualizar este proyecto es PangoPay [12]. El propósito principal de este producto es proporcionar una aplicación para *smartphones* que convierta el dispositivo del usuario en un monedero electrónico y que permita:

- Controlar las operaciones que se llevan a cabo en tu cuenta.
- Enviar y recibir pagos de tus amigos.
- Retirar efectivo desde cualquier cajero automático en cualquier parte del mundo.
- Realizar compras online o en comercios sin necesidad de dinero efectivo o tarjetas de crédito.
- Enviar dinero a cualquier parte del mundo.
- Compartir gastos y realizar pagos en grupo.

De todas estas funcionalidades, cabe destacar la de compartir gastos y realizar pagos en grupo. Los pagos en grupo consisten en completar una transacción asociada a un gasto repartido entre varios usuarios. La transacción sólo puede ser completada cuando todos los usuarios han saldado su deuda, de forma que la cantidad total quede cubierta.

Partiendo de esta premisa, surge la idea de un proyecto para poder dirigir los servicios de PaynoPain a un nuevo grupo de usuarios y explotar una necesidad potencial de mercado. El

nuevo producto contará con una parte que se integrará con los servicios ofrecidos por PangoPay para llevar a cabo pagos en grupo, aprovechando así tanto el software como la infraestructura que la empresa ya tiene operativos.

Esta idea se perfila gracias al equipo de diseño de la empresa, de forma que se garantiza la usabilidad proporcionando al equipo de desarrollo de la empresa, haciendo uso de *mockups* (propuesta del diseño de pantallas), una visión global de las actividades que realiza el usuario en la interacción con la aplicación móvil. Así pues, se genera la idea de FlatCrew [8] pero no comienza su desarrollo hasta el momento en que el alumno comienza la estancia en prácticas y le es asignado el proyecto.

Finalmente, cabe destacar la presencia de aplicaciones que proponen un conjunto de funcionalidades similares a FlatCrew y que ya se encuentra en el mercado de aplicaciones móviles. Por tanto, también se pretende identificar los puntos fuertes y débiles de éstas para conseguir mejorarlas en FlatCrew. A continuación se listan:

Fairshare [7] Esta aplicación permite gestionar y repartir las tareas de una casa para poder después llevar un seguimiento de quién las ha realizado y quién no. Además, también permite gestionar gastos y compras. En cuanto a la comunicación entre los usuarios que comparten piso, cuenta con un muro donde cada uno puede dejar una publicación para que todos sean notificados y la vean.

Splitwise [14] Esta aplicación se centra especialmente en el aspecto financiero de las gestiones de un piso compartido, dejando un poco de lado el reparto de tareas. Proporciona a los usuarios categorías específicas para cada tipo de gasto y permite realizar un seguimiento de cada uno de ellos de forma que sepan en cada momento qué deudas ya están saldadas y quién falta por pagar.

1.2. Objetivos

En esta sección se exponen los objetivos asociados con el proyecto y con la estancia del estudiante en la empresa. En primer lugar, se describen los objetivos de producción, es decir, qué resultados esperan obtenerse en cuanto al producto empresarial se refiere y en qué estado podrá este comenzar a considerarse un producto comercializable. En segundo lugar, se enumeran los diferentes objetivos de aprendizaje que se han previsto para el alumno y en qué forma se llevan a cabo dentro del entorno empresarial.

1.2.1. Objetivos generales de producción

A través de este proyecto, la empresa pretende lanzar al mercado una aplicación para dispositivos móviles que ponga a disposición de los usuarios una serie de funcionalidades que les permitan gestionar aspectos derivados de la convivencia, como realizar pagos en grupo y repartir tareas domésticas. A largo plazo, se espera conseguir también una unión e interrelación de los diferentes servicios que ésta ofrece. De esta forma, se crea un vínculo entre las funcionalidades

de los productos, favoreciendo su aceptación en el mercado. Así lo recoge *How Consumers Assess Product's Features?: A Case Study of Product Features of Smartphone* (Wee-Kheng Tan, Yi-DerYeh, Shin-Jia Chen, Yu-Cheng Lin, y Chia-Yu Kuo, 2012) [16] que analiza cómo afecta la interrelación entre funcionalidades nuevas y aquellas ya conocidas, por la misma u otras aplicaciones, a la aceptación por parte de los usuarios.

Por otra parte, se desea obtener una nueva vía de acceso al mercado de las aplicaciones móviles y los métodos de pago alternativos centrandolo en un sector joven de la población, favoreciendo la aceptación y el uso de este tipo de soluciones informáticas en la sociedad.

1.2.2. Objetivos específicos de producción

Al término de la estancia en prácticas se espera que el alumno haya conseguido planificar, diseñar y desarrollar un servicio que satisfaga las necesidades de la aplicación FlatCrew, encargandose de tareas como la autenticación de usuarios y la persistencia de datos entre otras funcionalidades que serán concretadas posteriormente en este documento, en el apartado de requisitos (Capítulo 3, pag. 11).

La meta más ambiciosa supone conseguir, además de la aplicación Android y del servicio que desarrollará el alumno funcionando perfectamente de forma conjunta, integrar también el servicio de pagos PangoPay para poder hacer efectivos los gastos compartidos que la aplicación gestionará. Sin embargo, tiene una mayor prioridad para la empresa lanzar la aplicación a *Google Play Store* en cuanto se hayan conseguido implementar todas las funcionalidades básicas, de forma que los usuarios comiencen a conocer las funcionalidades que les ofrece y se familiaricen con la interfaz.

1.2.3. Objetivos de aprendizaje

En cuanto al aprendizaje del alumno, se pueden diferenciar dos grandes bloques. En primer lugar, es importante que el alumno llegue a conocer los lenguajes de programación y los entornos de desarrollo que son empleados por los desarrolladores en la empresa para que la colaboración con el equipo pueda llevarse a cabo con fluidez y sin dificultades. A continuación se listan:

- Lenguaje de programación: PHP
- *Framework* de desarrollo: CakePHP
- Servidor HTTP: Apache
- Herramienta de gestión de proyectos: Redmine
- Control de versiones: Git
- Editor/Herramienta de desarrollo: PHPStorm

- Sistema de bases de datos: MySQL

Para familiarizarse con este primer bloque de herramientas, el alumno recibirá unas pautas por parte del equipo de sistemas de la empresa para encaminar el autoaprendizaje y conseguir así una rápida integración en el equipo.

En segundo lugar, la empresa tiene previsto incluir una serie de nuevas herramientas de integración y revisión de código al flujo habitual de trabajo del equipo. Sin embargo, no es plausible interrumpir la producción para que todo el equipo de desarrolladores investiguen dichas herramientas y comiencen a hacer uso de ellas, con las dificultades iniciales que ello puede suponer. Contrariamente, sí resulta permisible que el alumno dedique cierto tiempo, junto con dos empleados más, a conocer, instalar, configurar y comenzar a emplearlas con regularidad de forma que, posteriormente, sea más sencillo asistir al resto de desarrolladores y conseguir una rápida inclusión sin grandes retenciones en la producción. Concretamente, estas herramientas son:

Gerrit [9]Es una herramienta gratuita de revisión de código implementada sobre la herramienta de control de versiones Git. Permite que una revisión del *software* no pase directamente al repositorio principal, proporcionando un repositorio 'virtual' donde los cambios son únicamente propuestos y se mantienen a la espera de que los revisores designados los acepten. Sólo entonces, se unirán dichos cambios al repositorio principal. Un aspecto interesante de esta herramienta es que podemos designar a Jenkins como uno de los revisores, por tanto, si no supera las pruebas de integración de Jenkins, no es ni siquiera necesario que el resto de revisores se molesten en verificar los cambios. Además, existen complementos para editores y herramientas de desarrollo como PHPStorm, facilitando su uso.

Jenkins [10]Es una herramienta *open-source* escrita en Java que proporciona servicios de integración continua para el desarrollo *software*. El test de integración puede ser lanzado a través de varios mecanismos, sin embargo, uno de los más interesantes para el caso que planteamos en este documento, es mediante la herramienta de control de versiones. Así, cada vez que un desarrollador confirma una nueva revisión (commit), Jenkins comenzará las pruebas de integración. Habitualmente se lanza una batería de tests, si estos no se superan satisfactoriamente, se envía un mensaje al administrador.

2 Descripción del proyecto

El trabajo descrito en este documento corresponde concretamente a la parte servidor del proyecto FlatCrew. Sin embargo, a lo largo de las diferentes etapas de diseño e implementación, será necesario también considerar la integración con la parte cliente para poder acomodar al máximo sus necesidades. Así pues, a partir de este momento se hará referencia a ambas partes como *FlatCrew Server* y *FlatCrew Client* respectivamente.

Por otra parte, la lógica del proyecto en conjunto puede subdividirse en varias partes o unidades funcionales de acuerdo a las características que éstas aportan a la aplicación:

Amigos Es el eje principal de la aplicación, proporciona conexión e interacción de unos usuarios con otros. El mecanismo que se plantea para dar soporte a esta característica es similar al de otras aplicaciones de alta popularidad en el mercado como pueden ser WhatsApp, Telegram o Line. Un usuario podrá interactuar con aquellos usuarios que tenga como contacto en la agenda del dispositivo y que también hayan instalado y accedido a la aplicación. Además, se proporcionará un mecanismo para invitar a usar la aplicación a aquellos contactos que todavía no se hayan registrado.

Crews Es la segunda característica esencial del sistema, prácticamente todas las acciones que se pueden desencadenar, dependen de los grupos, de ahora en adelante *crews* para seguir con la nomenclatura utilizada durante el desarrollo. Un usuario agrupa a un conjunto de amigos creando una *crew* y puede asociar a esta, tareas y gastos. Uno de los miembros de la *crew* será el que tendrá derechos de administrador, esto le permitirá cambiar el nombre, la descripción o añadir nuevos amigos. Los derechos de administrador pueden pasar de un miembro a otro al abandonar la *crew*.

Gastos Cuando un usuario crea un gasto, éste queda asociado a una *crew* y el importe total establecido se reparte, uniformemente o no, entre los miembros de esa *crew*. Para realizar el reparto de forma heterogénea, el creador del gasto puede editar individualmente la cantidad que pagará cada uno de los miembros del grupo, sin embargo, la suma total de las deudas generadas habrá de ser igual al importe total del gasto. Mas allá, el gasto irá acompañado de una descripción y una fecha límite para cuando todas las deudas deben haber sido saldadas. Además, la aplicación proporcionará un resumen del estado de los gastos y deudas accediendo a una pestaña dedicada para ello.

Deudas Tras haber creado un gasto, los usuarios que participen en él, tendrán deudas pendientes en el sistema de forma automática. Para el caso de uso de FlatCrew, las deudas pendientes tendrán como beneficiario al responsable del gasto cuya cuenta será la que efectuará la transacción final con la cantidad total una vez se hayan saldado todas las deudas. Esto permite, a su vez, que los usuarios que no deseen asociar una cuenta PangoPay a FlatCrew puedan seguir gestionando los gastos compartidos aunque no se lleve a cabo el

pago de forma telemática. Esto repercute también en la forma en que se marcan las deudas como resueltas. El usuario, si no tiene asociada una cuenta PangoPay, cuando marque una deuda como saldada, asume que ha hecho llegar esa cantidad de dinero al beneficiario sin hacer uso de la pasarela de pago. Sin embargo, si tiene configurado el monedero de PangoPay, suponiendo que existen fondos suficientes, la transferencia se llevará a cabo a través de éste. Ambas partes son avisadas automáticamente mediante notificaciones de la aplicación de los cambios referentes a las deudas.

Listas Recogen una serie de artículos acompañados de un indicador que permite a los usuarios marcar cada uno de ellos como “completado” o “pendiente”. Pueden ser creadas sin necesidad de compartir con ningún amigo, por tanto, pueden llevar a cabo el papel del listas privadas de elementos pendientes. Por otra parte, si se comparten, pueden resultar muy útiles a la hora de organizar listas de compra entre amigos y llevar un seguimiento de lo que otros ya han completado. Por último, cada lista se identifica mediante un nombre que se introduce en el momento de la creación, el mismo nombre que verán los amigos con los que se comparta.

Tareas Es una de las funcionalidades que va asociada con más fuerza al escenario de los pisos compartidos. Las tareas que pueden crearse en la aplicación están siempre asociadas a una zona o elemento del hogar. Hay una larga lista de zonas provistas por la aplicación, sin embargo, también ofrece una zona comodín para suplir la necesidad de crear tareas para zonas no listadas. Todas las tareas se asociarán a una *crew* pero no por ello se repartirá entre todos los miembros de ella, sólo entre aquellos que el creador especifique. Finalmente, cada tarea tendrá un periodo de validez y una frecuencia de repetición.

Notificaciones En el mercado de las aplicaciones móviles, suponen una parte indispensable, aún más en aplicaciones con una alta interacción entre usuarios como la que se propone para este proyecto. Las notificaciones han pasado a formar parte del día a día de los usuarios de dispositivos móviles gracias a aplicaciones populares de mensajería instantánea como WhatsApp o Telegram. Además, en el contexto de escritorio, los usuarios también han incluido las notificaciones en su rutina habitual gracias a redes sociales como Twitter o Facebook. En este caso, las notificaciones son muy convenientes para alertar a los usuarios de los diferentes cambios y eventos que se pueden producir en la aplicación FlatCrew. Por una parte, las notificaciones *push* en la bandeja de sistema del dispositivo informarán brevemente de los eventos, permitiendo también acceder a la actividad implicada en la aplicación haciendo click en ellas. Por otra parte, la aplicación contará con una pestaña donde poder consultar de nuevo todas las notificaciones recibidas, ordenadas por categoría.

Todas estas características descritas anteriormente, necesitarán una implementación en *FlatCrew Server* que será el proveedor de la información que *FlatCrew Client* requerirá para mostrar al usuario. No sólo eso, *FlatCrew Server* también habrá de completar las acciones correspondientes a las peticiones POST que generará *FlatCrew Client*, desencadenadas por la interacción del usuario con la aplicación, realizando los cambios necesarios en la base de datos y devolviendo la información correspondiente, actualizada, si procede. Este tipo de operaciones pueden ser, por ejemplo, saldar una deuda, editar el nombre de una *crew* o añadir un nuevo miembro a esta.

3 Análisis de requisitos

Para comenzar a definir los requisitos de *FlatCrew Server*, se recurre a los *mockups* de la aplicación Android que, aunque forman parte del prototipo base de *FlatCrew Client*, resultarán de gran ayuda a la hora de definir y analizar las interacciones entre ambas partes y los requisitos asociados a éstas. Sin embargo, no basta únicamente con estos recursos de prototipo sino que es necesario tanto el apoyo y comunicación de los diseñadores de los *mockups*, para resolver dudas referentes al flujo de actividad de la aplicación, como el de los desarrolladores de *FlatCrew Client* para determinar con exactitud cuáles serán las necesidades de éste respecto a *FlatCrew Server*. Más tarde, en la sección 6.3 se validarán los métodos de *FlatCrew Server* gracias a los requisitos definidos en este apartado.

3.1. Mockups

En esta sección se muestran los diseños de pantallas realizados por el equipo de diseño de la empresa para ilustrar la idea de FlatCrew y establecer una base sobre la que poder apoyarse a la hora de definir los requisitos y, concretamente, los casos de uso que se verán en la siguiente sección. A continuación, se listan las diferentes figuras que componen dicho diseño de pantallas junto con una breve indicación para comprender mejor qué bloque funcional de la aplicación ilustran:

- **Pantalla de acceso y registro** Figura 3.1, ilustra las pantallas con las que interactúa el usuario en el momento de acceder a la aplicación si no está autenticado.
- **Pantalla principal y menú:** Figura 3.2, muestra cómo debe ser el menú *drawer* y qué información mostrar en la pantalla principal cuando el usuario ya haya introducido sus credenciales para acceder a la aplicación; será la primera pantalla que se mostrará.
- **Pantalla de deudas y *crews* asociados:** Figura 3.3, muestra las deudas asociadas a un usuario. Se accede tras haber hecho click en uno en el listado de la pantalla principal. También se puede navegar, como se muestra, para ver el detalle del *crew* asociado.
- **Pantallas de creación de entidades:** Figura 3.4, presenta las diferentes pantallas para crear las entidades básicas sobre las que recae gran parte de la interacción con la aplicación, detallando los campos que se muestran al usuario. Este tipo de diseño de pantallas es esencial para ayudar a determinar los requisitos de datos. Entre estas entidades básicas, se encuentran sin duda los usuarios, por tanto, se proporciona una pantalla para que el usuario invite a sus contactos a usar la aplicación.

- **Pantallas de monedero y amigos:** Figura 3.5, en este diseño se plasman los datos que deberán mostrarse al usuario cuando este acceda al monedero. Por otro lado, también se muestra un diseño de pantalla donde el usuario puede ver una lista con los contactos de su dispositivo que también utilicen FaltCrew.
- **Pantallas de gestión de listas:** Figura 3.6, en estas pantallas se muestran las opciones de manipulación y creación de listas que se ofrecerán al usuario. Proporcionan, indirectamente, indicaciones que serán útiles a la hora de diseñar el modelo de datos, puesto que las listas también constituyen una serie de entidades importantes en el conjunto de la aplicación.
- **Pantallas de gestión de tareas:** Figura 3.7, muestra las interfaces mediante las cuales el usuario será capaz de crear y repartir tareas, visualizar aquellas que le hayan sido asignadas e incluso modificar algunas de ellas. Al igual que en los anteriores diseños de pantallas de las entidades esenciales de la aplicación, este diseño aporta algunas pistas sobre qué elementos conviene tener muy en cuenta a la hora de organizar el diseño de datos.

Una vez estudiados los diseños de pantalla, prestando especial atención a los datos que se muestran y a las posibles operaciones que se entienden de los menús y botones de interacción, en la próxima sección se presentarán los casos de uso para la aplicación. Cabe destacar que para generar dichos casos de uso resulta esencial la colaboración con el equipo de desarrollo Android y con los expertos en diseño para llegar a un consenso en cuanto a las interacciones y funcionalidades que han de ofrecerse.

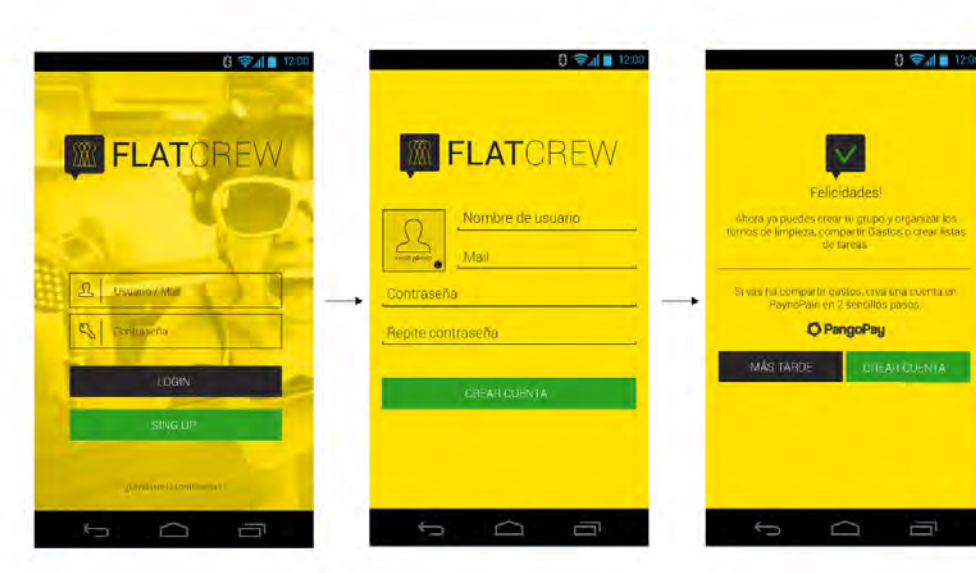


Figura 3.1: Pantalla de acceso y registro

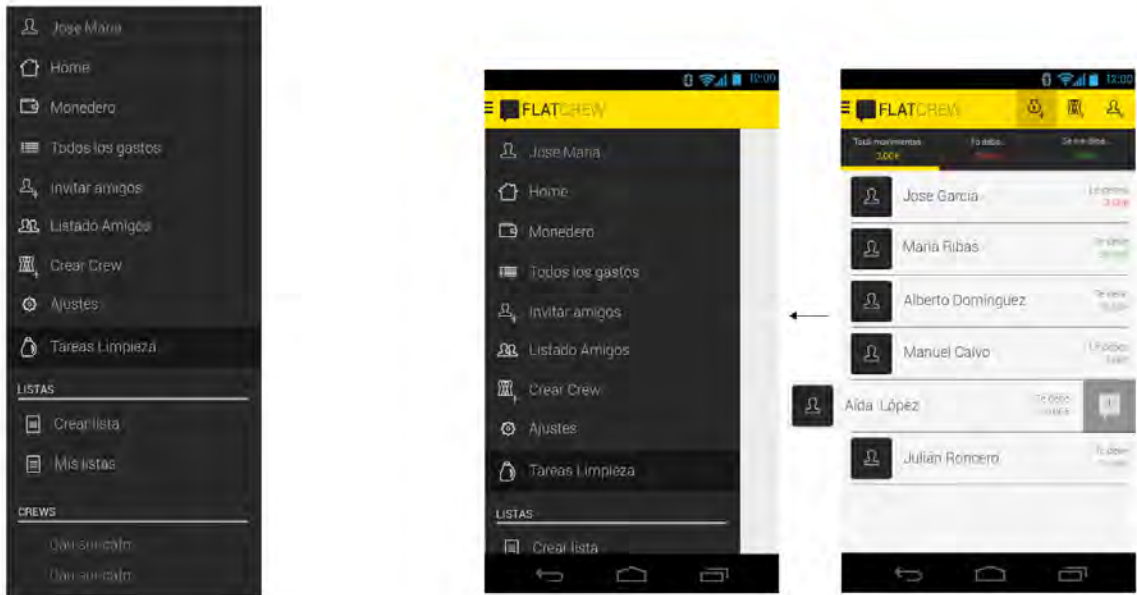


Figura 3.2: Pantalla principal y menú

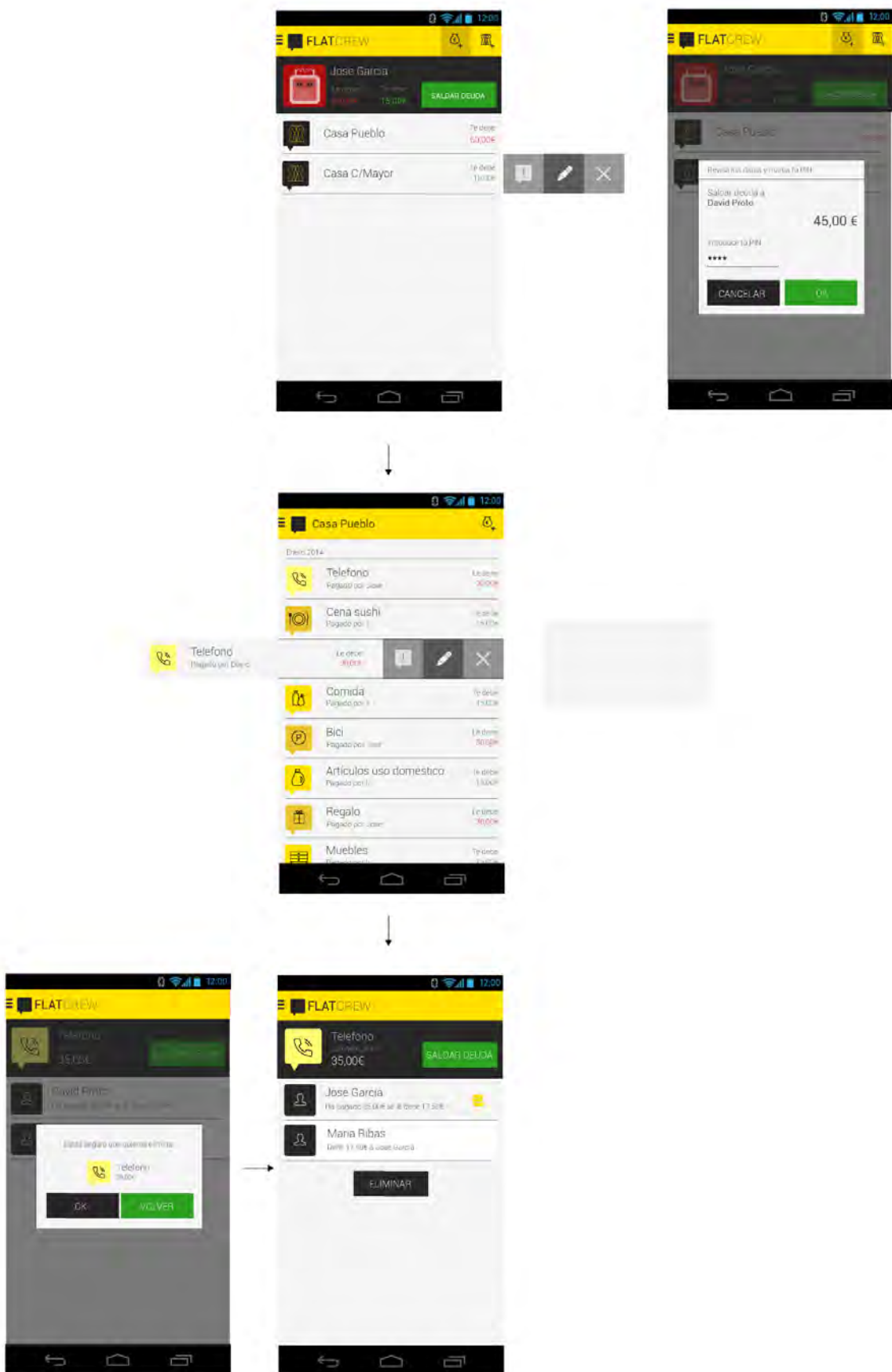


Figura 3.3: Detalle de deudas y crews asociados

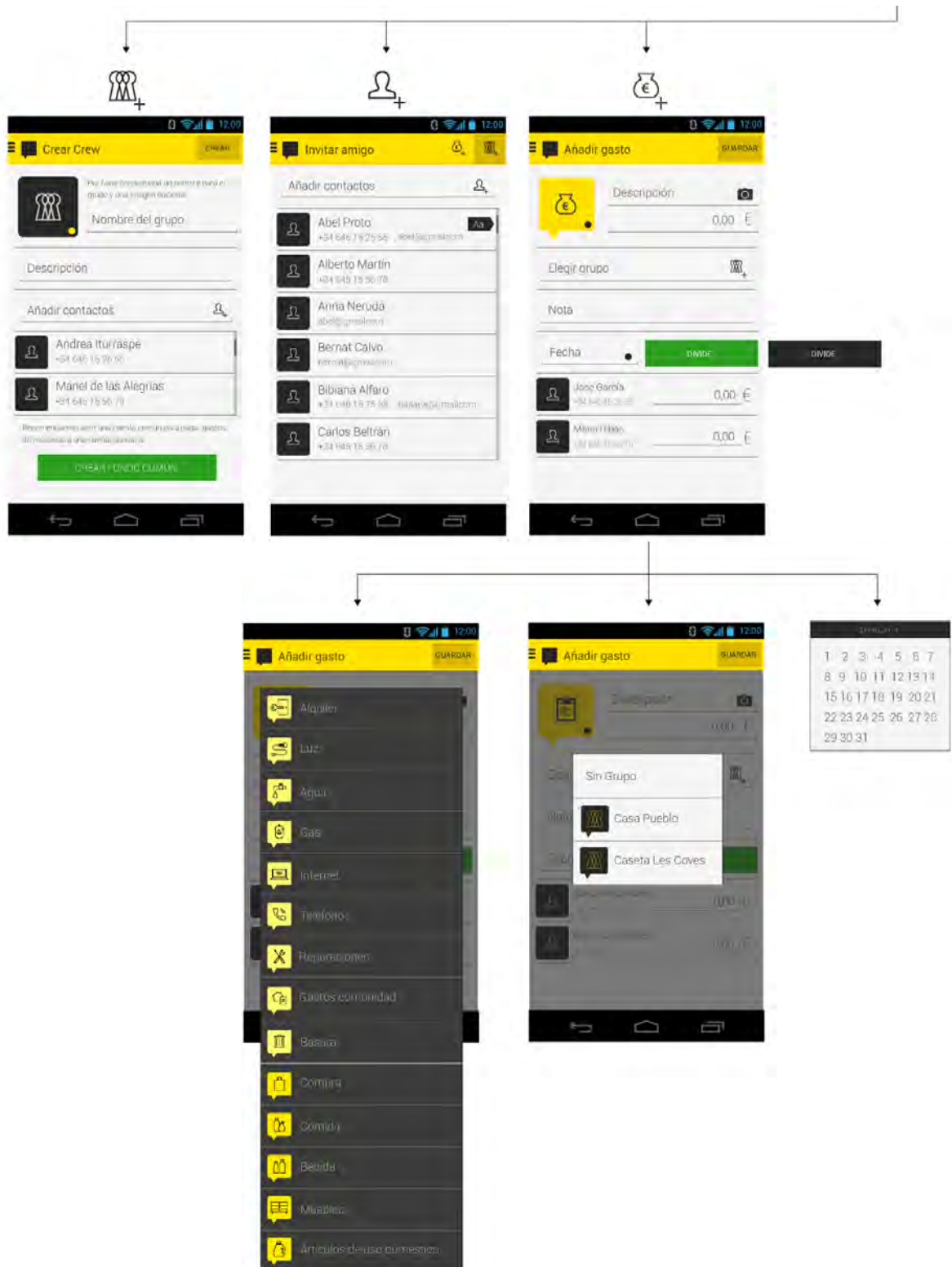
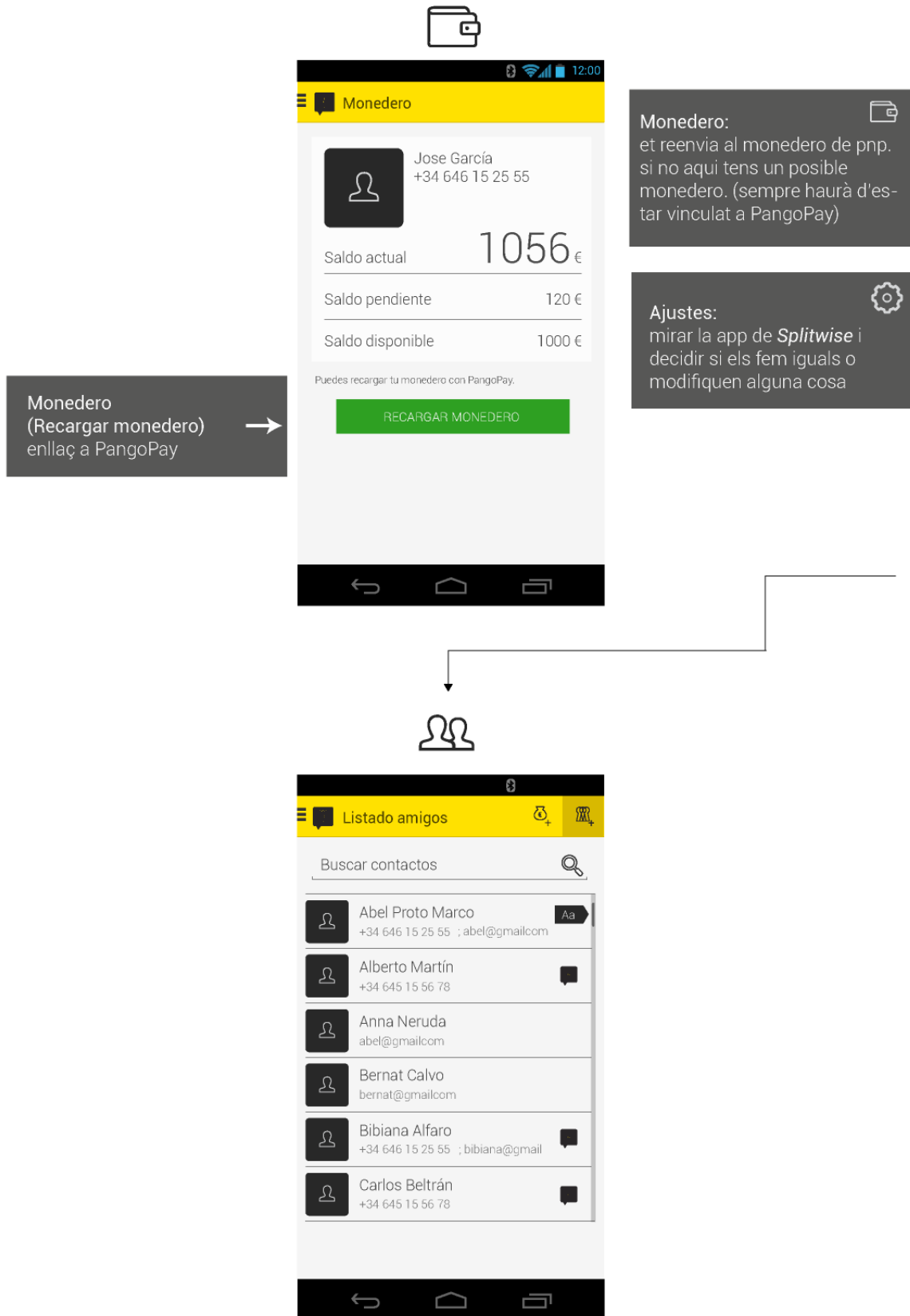


Figura 3.4: Pantallas de creación de entidades



Listas



Figura 3.6: Pantallas de gestión de listas

Repartir tareas

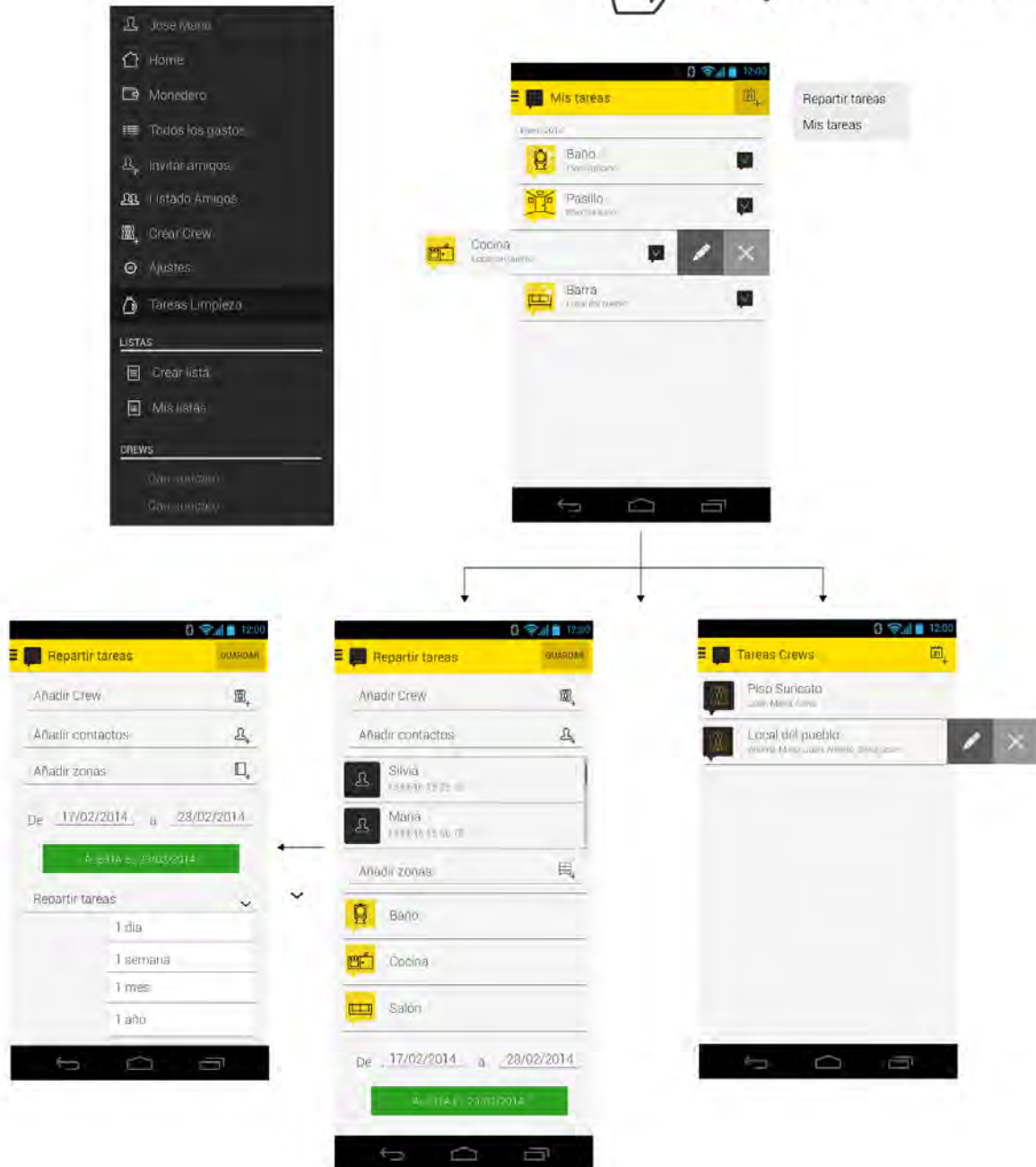


Figura 3.7: Pantallas de gestión de listas

3.2. Casos de uso

Para contribuir a la definición de requisitos de *FlatCrew Server*, en esta sección se analizan los casos de uso de todo el sistema FlatCrew. En este caso, sólo se distingue un actor que representa al usuario que utilizará la aplicación cliente desde su terminal Android. En la Figura 3.8 se muestra el diagrama de casos de uso. A continuación, se describen brevemente las diferentes interacciones que el usuario puede llevar a cabo al utilizar FlatCrew y que se completarán con las tablas correspondientes en este mismo apartado:

Gestionar grupos El usuario crea un *crew* haciendo click en el dialogo correspondiente del menú *drawer* de la aplicación y rellenando los detalles asociados además de elegir los contactos que pertenecerán a éste. Posteriormente, podrá editar los datos si es el administrador del grupo. En caso de que cualquier usuario quiera abandonar el *crew*, no habrá de tener ninguna deuda pendiente asociada a él. Ver Tabla 3.1.

Organizar tareas El usuario creará tareas de forma similar a los *crews*. Las peculiaridad de las tareas es que han de asociarse a un *crew* que ya exista y, además, se establecerá una frecuencia de reparto de la tarea así como una alerta para avisar al usuario de que todavía tiene esa tarea pendiente. Cabe destacar que no todos los miembros del *crew* asociado a la tarea han de participar en ella, pueden elegirse en el momento de la creación. Para modificar una tarea, el usuario habrá de ser el administrador de la misma. También puede abandonar la tarea desde el listado de tareas. Ver Tabla 3.2.

Compartir gastos El usuario desea crear gastos, compartirlos con otros usuarios y eliminarlos. Para crearlos y modificarlos, el proceso es similar al resto, en cambio, para eliminar un gasto, o ninguna de las deudas debe haber sido saldada todavía o todas ellas deben estar saldadas. Además, en relación con los gastos, el usuario puede querer alertar a un usuario para recordarle que tiene una deuda pendiente con él, esto se puede hacer interactuando con los elementos de la lista de deudas. Ver Tabla 3.3.

Gestionar listas El usuario crea listas donde añadir y eliminar elementos que pueden ser marcados como completados o pendientes. Estas listas pueden compartirse con otros usuarios que también podrán añadir nuevos elementos o eliminar antiguos, así como marcarlos como finalizados. Cualquier usuario que haya sido invitado a la lista puede abandonarla, sin embargo, sólo el creador puede editar algunos detalles como la descripción. Ver Tabla 3.4.

Saldar deudas El usuario, a través del listado del resumen de todos los gastos, puede seleccionar ver las deudas que tiene pendiente con otros usuarios y saldarlas. Puede saldarlas individualmente o en bloque, por ejemplo, saldar todas las deudas pendientes con un usuario en concreto. Para ello, no es necesario tener activado el monedero de PangoPay. Ver Tabla 3.5.

Recargar monedero El usuario recarga el monedero de PangoPay si tiene la aplicación correspondiente instalada en su dispositivo. FlatCrew solo ofrece la posibilidad de ver el saldo disponible y proporciona un acceso rápido para lanzar la aplicación PangoPay y así recargar el monedero. Ver Tabla 3.6.

Invitar contactos El usuario invita a sus contactos a usar la aplicación, bien a través de un

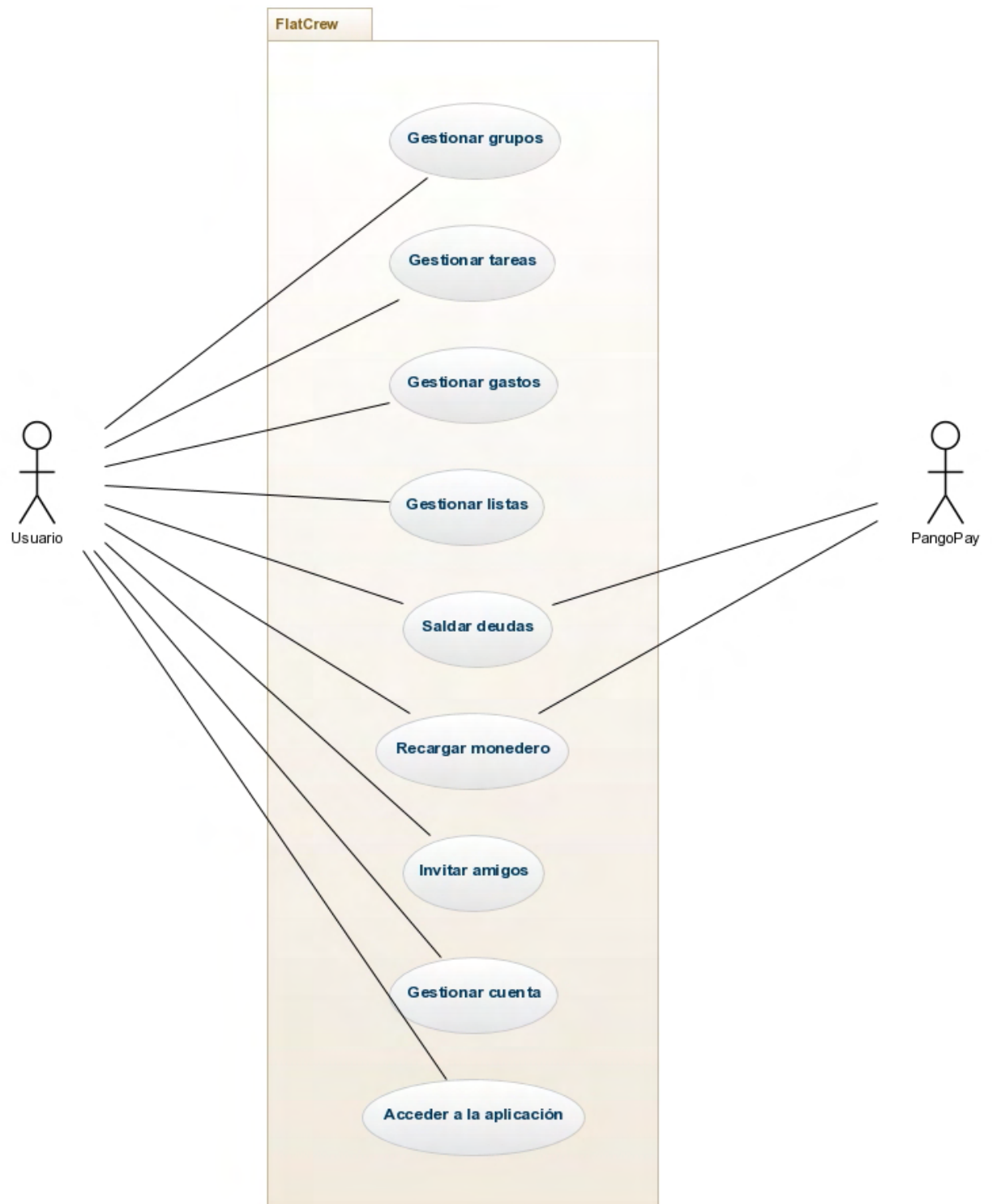


Figura 3.8: Diagrama de casos de uso

mensaje de texto (SMS) o a través del correo electrónico. Para ello sólo necesita acceder a la sección “Amigos” y seleccionar los contactos de la agenda a los que desea enviar la invitación. Ver Tabla 3.7.

Gestionar la cuenta de usuario El usuario modifica sus datos accediendo a la pantalla de ajustes de la aplicación, editando los campos correspondientes y guardando los cambios. Además, puede elegir eliminar su cuenta si desea que todos sus datos sean eliminados del sistema. Ver Tabla 3.8.

Registrarse y acceder a la aplicación El usuario se registra rellenando sus datos personales y de contacto en el formulario de registro. Una vez recibido el código vía SMS y confirmado el registro, el usuario entra en la aplicación haciendo uso de sus credenciales: dirección de correo electrónico y contraseña. En el caso de que haya olvidado la contraseña, la reestablecerá a través de un mensaje que se enviará al correo electrónico indicado e introduciendo una nueva contraseña haciendo uso del código de validación recibido. Ver Tabla 3.9.

Gestionar grupos	
Identificador: UCFC01	Nombre: Gestionar grupos
Descripción: Este caso de uso comprende las tareas de crear, abandonar y modificar grupos.	
Secuencia:	
<ul style="list-style-type: none"> ■ Crear grupo: <ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar “Crear Crew”. 2. Rellenar el nombre y descripción del Crew. 3. Seleccionar contactos que formarán parte del Crew. 4. Hacer click en “Crear” en la barra de menu para finalizar. ■ Abandonar grupo: <ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar un <i>crew</i>. 2. Hacer click en “Abandonar Crew” en la pantalla de resumen del Crew. 3. Responder al mensaje de confirmación. ■ Editar grupo: <ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar un <i>crew</i>. 2. Hacer click en los campos editables para modificar los atributos del Crew. 3. Pulsar el botón “Guardar”. 	
Condiciones Previas: El usuario debe haberse registrado y accedido con sus credenciales en la aplicación en su dispositivo Android. También deberá disponer de conexión a Internet para que los cambios sean efectivos. Adicionalmente, en el caso de abandonar un <i>crew</i> , será necesario que el usuario tenga todas sus deudas saldadas. Sólo el administrador puede editar los datos del <i>crew</i> .	

Tabla 3.1: Caso de uso Gestionar grupos

Gestionar tareas	
Identificador: UCFC02	Nombre: Gestionar tareas
Descripción: Este caso de uso comprende las funcionalidades de crear, modificar y eliminar tareas.	
Secuencia:	
<ul style="list-style-type: none"> ▪ Crear tarea: <ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar “Tareas de Limpieza”. 2. En el listado de tareas activas, hacer click en el icono de la interfaz para programar una nueva tarea. 3. Seleccionar el <i>crew</i>, los contactos de éste y las zonas o escenarios relacionados con la tarea, interaccionando con los campos editables. 4. Establecer la frecuencia de reparto y la alerta. 5. Pulsar el botón “Guardar”. ▪ Modificar tarea: <ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar “Tareas de Limpieza”. 2. Deslizar la tarea que se desee modificar. 3. Hacer click en el icono de edición que aparece tras deslizar. 4. Hacer click en los campos editables para modificar los atributos de la tarea. 5. Pulsar el botón “Guardar”. ▪ Eliminar tarea: <ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar “Tareas de Limpieza”. 2. Deslizar la tarea que se desee eliminar. 3. Hacer click en el icono de eliminar que aparece tras deslizar. 4. Responder al mensaje de confirmación. 	
Condiciones Previas: El usuario debe haberse registrado y accedido con sus credenciales en la aplicación en su dispositivo Android. También deberá disponer de conexión a Internet para que los cambios sean efectivos. Para modificar una tarea, el usuario que lleva a cabo la acción, ha de ser administrador del <i>crew</i> al que esté asociado la tarea.	

Tabla 3.2: Caso de uso Gestionar tareas

Gestionar gastos	
Identificador: UCFC03	Nombre: Gestionar gastos
Descripción: Este caso de uso comprende las funcionalidades de crear, modificar, eliminar gastos y enviar una alerta al deudor.	
Secuencia:	
<ul style="list-style-type: none"> ■ Crear gasto: <ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar “Añadir gastos”. 2. Rellenar los campos correspondientes con la descripción, la cantidad, el Crew asociado, el responsable y la fecha. 3. Editar cantidades en la misma pantalla para reparto heterogeneo. 4. Pulsar el botón “Guardar” en la barra de navegación. ■ Modificar gasto: <ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar “Todos los gastos”. 2. Hacer click en la deuda asociada. 3. Navegar a través del interfaz hasta el listado de gastos. 4. Deslizar el gasto que se desee modificar. 5. Hacer click en el icono de edición que aparece tras deslizar. 6. Modificar los atributos del gasto. 7. Pulsar el botón “Guardar” en la barra de navegación. ■ Eliminar gasto: <ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar “Todos los gastos”. 2. Hacer click en la deuda asociada. 3. Navegar a través del interfaz hasta el listado de gastos. 4. Seleccionar el gasto a eliminar. 5. Hacer click en el botón “Eliminar”. ■ Alertar deudor: <ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar “Todos los gastos”. 2. Deslizar la deuda asociada para hacer visibles las opciones. 3. Pulsar el icono de alerta que aparece tras deslizar. 4. Confirmar el envío del recordatorio (notificación push). 	
Condiciones Previas: El usuario debe haberse registrado y accedido con sus credenciales en la aplicación en su dispositivo Android. También deberá disponer de conexión a Internet para que los cambios sean efectivos. Para modificar o eliminar el gasto, el usuario que lleva a cabo la acción ha de ser el creador del gasto. Además, un gasto sólo se puede eliminar cuando todas o ninguna de las deudas han sido saldadas.	

Gestionar listas	
Identificador: UCFC04	Nombre: Gestionar listas
Descripción: Este caso de uso comprende las funcionalidades de crear, modificar, cambiar líneas y eliminar listas.	
Secuencia: <ul style="list-style-type: none"> ■ Crear lista: <ol style="list-style-type: none"> 1. Navegar a la pantalla principal. 2. Hacer click en el icono correspondiente de la barra de navegación para crear lista. 3. Rellenar los campos correspondientes con el nombre y los contactos. 4. Pulsar el botón “Guardar” en la barra de navegación. ■ Modificar lista: <ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar “Mis listas”. 2. Deslizar la lista que se desee modificar. 3. Hacer click en el icono de edición que aparece tras deslizar. 4. Modificar los atributos de la lista. 5. Pulsar el botón “Guardar” en la barra de navegación. ■ Abandonar lista: <ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar “Mis listas”. 2. Deslizar la lista que se desee abandonar. 3. Hacer click en el icono de eliminar que aparece tras deslizar. 4. Confirmar la operación en el mensaje de diálogo. ■ Cambiar líneas: <ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar “Mis listas”. 2. Seleccionar la lista deseada para ver las líneas que contiene. 3. Escribir en el cuadro de texto y pulsar en el icono de añadir. 4. Interactuar con los botones de cada línea para marcar como completada o eliminarla. 	
Condiciones Previas: El usuario debe haberse registrado y accedido con sus credenciales en la aplicación en su dispositivo Android. También deberá disponer de conexión a Internet para que los cambios sean efectivos. Sólo el creador de la lista puede modificarla pero todos pueden abandonarla o cambiar líneas.	

Tabla 3.4: Caso de uso Gestionar listas

Saldar deudas	
Identificador: UCFC05	Nombre: Saldar deudas
Descripción: Este caso de uso describe la secuencia que el usuario deberá seguir para saldar deudas con otros usuarios.	
Secuencia:	
<ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar “Todos los gastos”. 2. Seleccionar la pestaña “Yo debo”. 3. Pulsar sobre el cuadro con el nombre del usuario adeudado y navegar hasta ver la deuda. 4. Pulsar el botón “Saldar deuda”. 	
Condiciones Previas: El usuario debe haberse registrado y accedido con sus credenciales en la aplicación en su dispositivo Android. También deberá disponer de conexión a Internet para que los cambios sean efectivos.	

Tabla 3.5: Caso de uso Saldar deudas

Recargar monedero	
Identificador: UCFC06	Nombre: Recargar monedero
Descripción: Este caso de uso describe la secuencia que el usuario deberá seguir para recargar el monedero.	
Secuencia:	
<ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar “Monedero”. 2. Pulsar el botón “Recargar monedero”. 3. Tras lanzarse la aplicación PangoPay, seguir las acciones propias de dicha actividad para completar la recarga. 	
Condiciones Previas: El usuario debe haberse registrado y accedido con sus credenciales en la aplicación en su dispositivo Android. También deberá disponer de conexión a Internet para que los cambios sean efectivos. Adicionalmente, el usuario deberá tener instalada la aplicación PangoPay en el dispositivo.	

Tabla 3.6: Caso de uso Recargar monedero

Invitar usuarios	
Identificador: UCFC07	Nombre: Invitar usuarios
Descripción: Este caso de uso comprende la funcionalidad que permite al usuario invitar a sus contactos de la agenda a que comiencen a usar la aplicación.	
Secuencia:	
<ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar “Amigos”. 2. Hacer click en el icono “compartir” de la barra de navegación para enviar una invitación a través de las aplicaciones Android sugeridas. 3. Seleccionar una aplicación para completar la acción. 4. Efectuar el envío de la invitación siguiendo la secuencia adecuada a la aplicación seleccionada. 	
Condiciones Previas: El usuario debe haberse registrado y accedido con sus credenciales en la aplicación en su dispositivo Android. También deberá disponer de conexión a Internet para que los cambios sean efectivos.	

Tabla 3.7: Caso de uso Invitar usuarios

Gestionar cuentas	
Identificador: UCFC08	Nombre: Gestionar cuentas
Descripción: Este caso de uso comprende las funcionalidades que permiten al usuario modificar y eliminar su cuenta en FlatCrew.	
Secuencia: <ul style="list-style-type: none"> ■ Modificar cuenta: <ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar “Ajustes”. 2. Seleccionar la pestaña “Datos personales”. 3. Editar los cuadros de texto con la información referente al usuario. 4. Pulsar el botón “Guardar”. ■ Eliminar cuenta: <ol style="list-style-type: none"> 1. Hacer click en el elemento de la interfaz para abrir el menú <i>drawer</i> y seleccionar “Ajustes”. 2. Seleccionar la pestaña “Datos personales”. 3. Pulsar el botón “Eliminar cuenta”. 	
Condiciones Previas: El usuario debe haberse registrado y accedido con sus credenciales en la aplicación en su dispositivo Android. También deberá disponer de conexión a Internet para que los cambios sean efectivos.	

Tabla 3.8: Caso de uso Gestionar cuentas

Acceder a la aplicación	
Identificador: UCFC08	Nombre: Acceder a la aplicación
Descripción: Este caso de uso comprende las funcionalidades que permiten al usuario registrarse, acceder con sus credenciales en FlatCrew y restablecerlos en caso de pérdida.	
Secuencia:	
<ul style="list-style-type: none"> ■ Registrarse en FlatCrew: <ol style="list-style-type: none"> 1. Abrir la aplicación en el dispositivo. 2. Pulsar el botón “Registrarse”. 3. Rellenar el formulario con los datos correspondientes. 4. Pulsar el botón “Crear cuenta”. 5. Si la aplicación se mantiene abierta <ol style="list-style-type: none"> a) Detecta automáticamente la llegada del código de confirmación vía SMS y se habrá completado el registro. 6. En otro caso <ol style="list-style-type: none"> a) Abrir de nuevo la aplicación. b) Hacer click en “Introducir código de activación”. c) Introducir el número de teléfono móvil. d) Introducir código de activación. e) Pulsar “Aceptar”. ■ Acceder a FlatCrew: <ol style="list-style-type: none"> 1. Abrir la aplicación en el dispositivo. 2. Introducir el correo electrónico y la contraseña. 3. Pulsar el botón “Entrar”. ■ Restablecer contraseña: <ol style="list-style-type: none"> 1. Abrir la aplicación en el dispositivo. 2. Hacer click en el “¿Olvidaste la contraseña?”. 3. Introducir una dirección de correo electrónico. 4. Pulsar el botón “Enviar”. 5. Introducir el código recibido a través del correo electrónico. 6. Escribir la nueva contraseña. 7. Pulsar “Guardar”. 	
Condiciones Previas: Para acceder a la aplicación y restablecer la contraseña es necesario haber creado previamente una cuenta de usuario a través del registro. También deberá disponer de conexión a Internet para que los cambios sean efectivos.	

Tabla 3.9: Caso de uso Acceder a la aplicación

3.3. Requisitos de datos

En esta sección se analizan y describen aquellos datos que será necesario almacenar y gestionar para que la aplicación pueda funcionar correctamente.

En primer lugar, se analizan los requisitos de datos correspondientes a la funcionalidad central de la aplicación, ver Tablas 3.10, 3.11, 3.12, 3.13 y 3.14.

Usuarios	
Identificador: DRFC01	Nombre: User
Tipo: mantener datos	
Datos a mantener: nombre, apellidos, correo electrónico, contraseña y número de teléfono	
Detalles Contraseña: por motivos de seguridad la contraseña no puede almacenarse sin cifrar, se aplicará un algoritmo de <i>hash</i>	
Comentarios: Los datos del usuario son el requisito de datos más importante de la aplicación	

Tabla 3.10: Requisitos de datos para usuarios

Grupos	
Identificador: DRFC02	Nombre: Crew
Tipo: mantener datos	
Datos a mantener: nombre, descripción y creador	
Detalles Descripción: un Crew puede no tener ninguna descripción Creador: el creador es necesario porque actúa como el administrador del Crew	
Comentarios: Cuando el creador abandona el grupo, debe escogerse uno nuevo para que actúe de administrador, si éste no es el último en abandonar	

Tabla 3.11: Requisitos de datos para grupos

En segundo lugar, se tendrán en cuenta los requisitos de datos básicos para alcanzar el funcionamiento de OAuth 2.0. Como se verá más adelante, ver Sección 6.1, es posible que esta definición previa de requisitos se vea afectada por las restricciones correspondientes a las herramientas y complementos empleados para esta parte. También, en la sección 7.1, se describirá con más detalle otros requisitos que impone el uso del protocolo OAuth 2.0, aunque a priori, por la estructura de la parte servidor, sólo necesitaríamos mantener códigos de acceso y de actualización, ver Tablas 3.15 y 3.16.

Para finalizar con la definición de los requisitos de datos, en las Tablas 3.17, 3.18, 3.19, 3.20, se muestran las variables relevantes para dar soporte a la gestión de cuentas de usuario y notificaciones.

Listas	
Identificador: DRFC03	Nombre: Listing
Tipo: mantener datos	
Datos a mantener: nombre, creado y líneas de artículos	
Detalles	
Líneas: cada línea puede estar marcada como “finalizada” o “pendiente”	
Creador: el creador actúa como el administrador del Listing	
Comentarios:	
Cuando el creador abandona la lista, debe escogerse uno nuevo para que actúe de administrador, si éste no es el último en abandonar	

Tabla 3.12: Requisitos de datos para listas

Tareas	
Identificador: DRFC04	Nombre: Task
Tipo: mantener datos	
Datos a mantener: fecha de inicio, fecha de fin, aviso, Crew asociado y escenario	
Detalles	
Aviso: se puede asociar una alarma a la tarea para que alerte al usuario si la tarea no se ha llevado a cabo llegado a un momento determinado	
Escenario: la aplicación proporciona una lista variada de escenarios comunes en un entorno de piso compartido donde deben desempeñarse tareas; tareas predefinidas	
Comentarios:	
Sólo el administrador del crew asociado puede modificar las tareas y reasignarlas	

Tabla 3.13: Requisitos de datos para tareas

Gastos	
Identificador: DRFC05	Nombre: Expense
Tipo: mantener datos	
Datos a mantener: cantidad, descripción, fecha límite, nota, responsable y creador	
Detalles Cantidad: deberá repartirse entre los usuarios que comparten el gasto Creador: desempeña el papel de administrador al igual que en el caso de los grupos y listas Responsable: es aquella persona que hará efectivo el pago de la cantidad total, es decir, el resto le abonan su parte al responsable y cuando se han completado todas las partes, éste efectúa el pago Nota: es opcional, proporciona al administrador una herramienta para describir con más detalle el motivo u objetivo del gasto	
Comentarios: Posiblemente, al integrar el pago de los gastos con PangoPay, el rol del responsable cambie ligeramente pero seguirá siendo relevante mantener sus datos	

Tabla 3.14: Requisitos de datos para gastos

Access Tokens	
Identificador: DRFC06	Nombre: Access Token
Tipo: mantener datos	
Datos a mantener: código de acceso, usuario asociado y caducidad	
Detalles Caducidad: cuando el código de acceso caduque, deberá solicitarse un código nuevo empleando el código de actualización	
Comentarios: Los <i>tokens</i> caducados deberán ser eliminados de la base de datos	

Tabla 3.15: Requisitos de datos para *tokens* de acceso

Refresh Tokens	
Identificador: DRFC07	Nombre: Refresh Token
Tipo: mantener datos	
Datos a mantener: código de actualización, usuario asociado y caducidad	
Detalles Caducidad: cuando el código de actualización caduque, deberá solicitarse un código nuevo empleando los credenciales de usuario	
Comentarios: Los <i>tokens</i> caducados deberán ser eliminados de la base de datos	

Tabla 3.16: Requisitos de datos para *tokens* de actualización

Registros	
Identificador: DRFC08	Nombre: Registration
Tipo: mantener y limpiar datos	
Datos a mantener: código de registro, caducidad, datos de usuario	
Detalles Caducidad: cuando los datos de registro caduquen, el servidor deberá eliminarlos de la base de datos automáticamente para evitar congestión Datos de usuario: contiene la información que se añadirá como un nuevo usuario cuando se complete el registro	
Comentarios: Cuando se completa el registro y se añade el nuevo usuario al sistema, los datos de registro son eliminados automáticamente, no se espera a la caducidad de estos	

Tabla 3.17: Requisitos de datos para registros

Recuperaciones de contraseña	
Identificador: DRFC09	Nombre: Recovery
Tipo: mantener y limpiar datos	
Datos a mantener: código de recuperación, caducidad, usuario asociado	
Detalles Caducidad: cuando los datos de recuperación caduquen, el servidor deberá eliminarlos de la base de datos automáticamente para evitar congestión	
Comentarios: Cuando se completa el reestablecimiento de contraseña, los datos de recuperación son eliminados automáticamente, no se espera a la caducidad de estos	

Tabla 3.18: Requisitos de datos para recuperación de contraseñas

Notificaciones	
Identificador: DRFC10	Nombre: Notification
Tipo: mantener datos	
Datos a mantener: mensaje, acción asociada, creador, fecha de creación, destinatario	
Detalles Acción asociada: se dispondrá de una lista de etiquetas que permitirán categorizar las notificaciones y así poder filtrarlas	
Comentarios: Muchos de los datos enumerados son relevantes porque la aplicación cliente mostrará una pestaña con las novedades de los eventos ocurridos en la aplicación a partir de las notificaciones	

Tabla 3.19: Requisitos de datos para notificaciones

Dispositivos	
Identificador: DRFC11	Nombre: Device
Tipo: mantener datos	
Datos a mantener: identificador del dispositivo, tipo, usuario asociado	
Detalles Identificador del dispositivo: para utilizar las API y enviar notificaciones <i>push</i> es necesario almacenar el identificador que esta otorga a cada terminal Tipo: aunque no se contempla en el alcance inicial del proyecto, en un futuro pueden desarrollarse aplicaciones móvil para otras plataformas, por tanto, es necesario poder filtrar el tipo de plataforma	
Comentarios: No hay comentarios	

Tabla 3.20: Requisitos de datos para dispositivos

3.4. Plataforma tecnológica

No corresponde al alumno seleccionar o estudiar la plataforma tecnológica donde se desplegará el servicio en el momento de la puesta en producción. La empresa cuenta con sus propias máquinas que utiliza para desplegar los servidores en producción y que son mantenidos por el equipo de sistemas informáticos de la empresa.

En cuanto a la plataforma de desarrollo, se empleó un ordenador con sistema operativo Linux. El entorno para desarrollar el servicio y realizar las pruebas pertinentes se configuró en base al arquetipo LAMP (Linux Apache MySQL PHP). Existen herramientas para sistemas Linux que instalan automáticamente los paquetes necesarios para satisfacer los requisitos de este tipo de infraestructuras. Después de esto sólo fue necesario configurar algunas de las herramientas, por ejemplo, crear un *virtual host* para el servidor Apache o instalar algunos módulos complementarios para PHP.