Split & Share Bill: App para dividir y compartir la cuenta entre amigos

Antonio Ureba Basallote
Desarrollo de aplicaciones para dispositivos móviles
Trabajo final del Master DADM
Universitat Oberta de Catalunya

Pau Dominkovics Coll
Carles Garrigues Olivella

Junio de 2019

A toda mi familia, por su apoyo constante, sobre todo a mis hijas,
por ser mi fuente de inspiración y el núcleo de mi vida.
También a las personas que me brindan su confianza y que contribuyen
en mi progreso, tanto laboral como personal.

FICHA DEL TRABAJO FINAL

Título del trabajo:	Split & Share Bill: App para dividir y compartir la cuenta entre amigos
Nombre del autor:	Antonio Ureba Basallote
Nombre del consultor/a:	Pau Dominkovics Coll
Nombre del PRA:	Carles Garrigues Olivella
Fecha de entrega (mm/aaaa):	05/2019
Titulación:	Desarrollo de aplicaciones para dispositivos móviles
Área del Trabajo Final:	Trabajo final de master DADM
Idioma del trabajo:	Castellano
Palabras claves:	dividir, compartir, cuenta, factura, ticket, amigos, comer, restaurante, gasto, OCR

Resumen del Trabajo (máximo 250 palabras):

Se trata de una aplicación para iPhone e iPad con la que se puede dividir el gasto de la factura de un restaurante, cafetería o cualquier sitio en el que se genere una factura a pagar entre varias personas, asignando a cada persona lo que le hayan servido.

Se pueden capturar los elementos de la factura mediante la cámara del dispositivo y asignar cada elemento a quien le corresponda pagarlo. Tras asignar cada producto se puede compartir la factura divida a través de la App o otras Apps de mensajería como SMS, correo electrónico o WhatsApp.

En la factura se podrán aplicar descuentos, el impuesto aplicado (si no lo toma automáticamente) y el porcentaje de propina que se desea dejar.
A través de la App podrás conectar con amigos y compartir la factura, e incluso se pueden formar grupos de amigos.

Todos los datos se guardan en la nube y se pueden consultar desde cualquier dispositivo a través de una API.

Abstract (in English, 250 words or less):

The present document contains the Final Project of the Master “Development for Mobile Devices” titled “Split & Share Bill: App to split and share the bill between friends”.
The principal objective of this project is the development and documentation of an application capable of recognizing a bill, capture the bill with the device camera and create a list with the elements that make up the bill, and be able to assign each item to each diner, paying each person for what he has taken.

In the current market of applications for iOS devices doesn't exist applications that can interpret a bill with OCR and can assign elements and calculate what each customer must pay when dividing the bill.

In the development of the application, development tools such as XCode have been used, with which the native app for iOS devices will be built, served by an API made with the framework “Codeigniter”.

ÍNDICE DE CONTENIDOS

1. Introducción 1
   1.1. Contexto y justificación del trabajo 1
   1.2. Objetivos del trabajo 2
   1.3. Enfoque y método seguido 3
   1.4. Planificación del trabajo 7
   1.5. Breve sumario de productos obtenidos 10
   1.6. Breve descripción de los otros capítulos de la memoria 10
2. Descripción general del proyecto 12
   2.1. Diseño centrado en el usuario (DCU) 12
      2.1.1. Usuarios y contexto de uso 12
         2.1.1.1. Fichas User-Persona 13
      2.1.2. Diseño conceptual 16
      2.1.3. Prototipo 16
         2.1.3.1. Navegabilidad de las pantallas 19
         2.1.3.2. Prototipo alta fidelidad 23
      2.1.4. Evaluación 32
   2.2. Diseño técnico de la aplicación 33
      2.2.1. Casos de uso 34
         2.2.1.1. Modelos de casos de uso 34
      2.2.2. Diseño de la arquitectura 47
         2.2.2.1. Arquitectura del software 47
         2.2.2.2. Diagrama Entidad-Relación (ER) 48
         2.2.2.3. Diseño lógico de la base de datos 53
         2.2.2.4. Diagrama de clases conceptuales 55
3. Fase de implementación 56
   3.1. Tecnologías utilizadas 57
      3.1.1. Lenguajes de programación 57
      3.1.2. Librerías y productos de terceros 57
      3.1.3. Sistema gestor de base de datos 58
      3.1.4. Servidor web 59

3.2. Análisis sobre lectura OCR de tickets 59
3.3. Implementación 60
3.3.1. Servidor 60
3.3.1.1. Estructura 60
3.3.1.2. Entornos de pruebas o Sandbox 61
3.3.2. App iOS 63
3.3.2.1. Estructura 64
3.4. Certificados 64
3.5. Pruebas 66
3.5.1. Plan de pruebas 66
3.5.2. Especificación del diseño de pruebas 66
3.5.2.1. Pruebas durante la implementación 66
3.5.2.2. Pruebas una vez finalizada la implementación 67
3.5.3. Especificación de los casos de prueba 67
3.5.3.1. Personas con bajos conocimientos sobre móviles 67
3.5.3.2. Personas con altos conocimientos sobre móviles 67
4. Conclusiones 69
4.1. Aspectos generales 69
4.2. Cumplimiento de la planificación y objetivos 70
4.3. Ampliaciones futuras 70
5. Glosario 71
6. Bibliografía y referencias 73
Libros y Artículos 73
Referencias web 73
Herramientas y software 74
6. Anexos 75
Anexo 1: Diagrama de Gantt a mayor escala 76
Anexo 2: Licencias de recursos utilizados 77
Anexo 3: Tickets reales para probar la App 81
Anexo 4: Manual de instalación del servidor 83

ÍNDICE DE FIGURAS

Diagrama de modelo en cascada 6
Diagrama de Gantt 9
Avenir® Next Font 17
Icon Set SS-Gizmo 17
Diseño de Logotipo de la aplicación 18
Gama de colores de la App 18
Navegabilidad de la aplicación (Parte 1/2) 21
Navegabilidad de la aplicación (Parte 2/2) 22
Pantalla Login y Pantalla Registro de usuario 23
Pantalla Selección de método para iniciar una nueva cuenta 24
Pantalla Capturar ticket mediante foto y Pantalla Recortar imagen 25
Pantalla Asignar elementos y Pantalla Añadir/modificar un elemento 26
Pantalla Elegir miembros para dividir y Pantalla Resumen de la cuenta dividida 27
Pantalla Historial de cuentas 28
Pantalla Amigos y Pantalla Grupos 29
Pantalla Añadir un amigo y Pantalla Asignar un amigo enlazado a uno ficticio 30
Pantalla Mi cuenta y Pantalla Mi perfil de usuario 31
Pantalla Asignar elementos con el tema oscuro y con el tema claro 32
Diagrama arquitectura Modelo-Vista-Controlador 47
Diagrama Entidad-Relación 49
Diagrama de clases conceptuales 55
Sandbox de pruebas y documentación de la API 62
Sandbox de pruebas - Login (Parte 1) 63
Sandbox de pruebas - Login (Parte 2) 63
Automatically manage signing Provisioning Profile XCode 65
Firebase Certificado APNS de desarrollo 65
Diagrama de Gantt a mayor escala 76

ÍNDICE DE TABLAS

Aplicaciones más relevantes encontradas en el estudio de mercado 4
Tabla Ficha User-Persona 1 14
Tabla Ficha User-Persona 2 15
Tabla de entidades 50
Tabla de atributos de la entidad Usuarios 51
Tabla de atributos de la entidad Cuentas 51
Tabla de atributos de la entidad Elementos 51
Tabla de atributos de la entidad ElementosUsuarios 51
Tabla de atributos de la entidad VisibilidadCuentas 52
Tabla de atributos de la entidad Grupos 52
Tabla de atributos de la entidad MiembrosGrupos 52
Tabla de atributos de la entidad VinculosUsuarios 53

1. Introducción

1.1. Contexto y justificación del trabajo

Es muy frecuente entre los grupos de amigos que compartan el pago de la factura de los sitios que visitan, ya sean restaurantes, cafeterías, pubs o cualquier sitio en los que hagan gastos con una factura en común.

Normalmente dividen el pago entre los asistentes a partes iguales porque es más cómodo y no tienen en cuenta el gasto que ha realizado cada uno. Esta práctica no siempre es justa y hay muchas personas que no están de acuerdo y prefieren pagar cada uno por lo que ha consumido.

Con esta aplicación se pretende agilizar esta práctica y hacerla más cómoda, de manera que echando una foto a la factura con la aplicación puedan asignarse cada elemento de la factura a quien la haya consumido de manera fácil e intuitiva. Luego se podrá compartir con el resto de amigos a través de la aplicación o por cualquier aplicación que permita compartir contenido, ya sea SMS, correo electrónico, WhatsApp o cualquier aplicación que lo permita.

1.2. Objetivos del trabajo

El Objetivo del trabajo es construir una aplicación que agilice la tarea de dividir la factura y que cada comensal pague por lo que le hayan servido.

Para la utilización de esta App no es necesario que todos los implicados, es decir, todas las personas que pagarán la cuenta, tenga la aplicación instalada. A la hora de crear usuarios podrán ser usuarios reales, que si tienen la App y podrán consultar la factura y se les guardará en su historial, y usuarios ficticios que no están ligados a ningún usuario pero que debe aparecer su figura como pagador. Más tarde estos usuarios ficticios podrán ser ligados a usuarios reales si deciden descargarse la App y crear una cuenta de usuario.

A continuación, se detallan los requerimientos funcionales y no funcionales de la aplicación.

Requerimientos funcionales:

- RF1: Iniciar sesión con tu usuario o a través de al menos dos redes sociales.
- RF2: Crear una cuenta de usuario a través del formulario de registro.
- RF3: Poder recordar la contraseña de usuario a través de correo electrónico.
- RF4: Se puede cambiar la foto de usuario eligiéndola de la galería o tomándola en el momento.
- RF5: Compartir tú código de usuario a través de apps de mensajería.
- RF6: Crear vínculos de amigos a través de un código.
- RF7: Crear/modificar usuarios ficticios[1] para poder dividir la factura con ellos.
- RF8: Asignar un nuevo amigo a un usuario ficticio que ya tenía.
- RF9: Crear un grupo de usuarios, con usuarios reales y ficticios.
- RF10: Se puede crear una nueva factura para dividir a través de una foto a la factura, subiendo una foto de la factura o añadiendo manualmente uno a uno los elementos que componen la factura.

[1] Un usuario ficticio es una persona que ha creado un usuario de la App para dividir con ella la cuenta sin necesidad de que se descargue la App.

- RF11: Se puede aplicar un porcentaje para la propina que se desee dar, la cual se incrementará en el precio de la factura.
- RF12: Se puede establecer el impuesto (por ejemplo IVA) a aplicar en la factura. Esto lo tomará automáticamente de la foto o se deberá establecer en el caso de que se haga manualmente.
- RF13: Se puede elegir/modificar los usuarios con los que se va a dividir la cuenta.
- RF14: En cualquier momento se puede modificar los elementos de la cuenta como el precio, la cantidad y la/s persona/s que lo ha/n tomado.
- RF15: Se puede compartir a través de cualquier app que permita compartir contenido.
- RF16: Cada factura la guarda en un histórico y se puede consultar en cualquier momento.
- RF17: En la configuración se puede establecer los valores por defecto, tales como el IVA, el porcentaje de propina y si desea recibir notificaciones.

Requerimientos no funcionales:

- RNF1: El dispositivo tiene que tener conexión a internet para poder iniciar sesión, compartir algo o guardar cualquier contenido en el servidor.
- RNF2: El dispositivo debe tener el sistema operativo iOS 9.0 como mínimo.
- RNF3: La conexión con el servidor se realiza de manera asíncrona para mayor fluidez y para que una mala calidad de la conexión no bloquee la interacción con el usuario.
- RNF4: La aplicación debe ser intuitiva y fácil de usar.

1.3. Enfoque y método seguido

Tras realizar un estudio de mercado en los markets de las plataformas más importantes, hemos encontrado que esta idea ya existe en el mercado, pero se puede mejorar y enfocar a otros conceptos.

Hay varias aplicaciones desarrolladas para Android y que podemos encontrar en Google Play con una idea similar a la propuesta en este documento, pero ninguna es tan completa.

En la App Store de Apple, encontramos menos variedad en aplicaciones de este tipo, y ninguna con la funcionalidad de leer facturas a través de la cámara del dispositivo y posterior lectura mediante OCR. La mayoría de estas aplicaciones se basan en dividir múltiples gastos que se introducen los importes manualmente.

A continuación, mostramos una lista de las aplicaciones más relevantes en encontradas en el estudio de mercado. En ellas podemos encontrar poco que mejorar porque el punto de vista de la aplicación es el compartir gastos múltiples y no la división de una factura. Nuestro contexto es diferente por lo que en el mercado iOS la hace especial ya que no podemos encontrar alguna App con estas funcionalidades.

Nombre	**Puntos fuertes y mejoras a considerar**	**Enlace**
Splid	- Creación de grupos de amigos. - Invitación por código.	Android e iOS: https://splid.app/
Splitwise	- Grupos de amigos. - Amistad por contactos del dispositivo.	Android e iOS: https://www.splitwise.com/
GroupXpense	- Historial desglosado de pagos.	iOS: https://itunes.apple.com/es/app/groupxpense/id406809188?mt=8
Divide la cuenta	- Lectura OCR de facturas - Grupos de amigos - Historial	Android: https://play.google.com/store/apps/details?id=com.astepanov.mobile.splitcheck

Aplicaciones más relevantes encontradas en el estudio de mercado

Tras analizar todas las aplicaciones existentes en el mercado o al menos las más relevantes, llegamos a la conclusión de que es posible realizar una aplicación multiplataforma con la mayoría de funcionalidades del resto de aplicaciones, centrándonos en el objetivo principal de la aplicación, que es el dividir una factura asignando cada elemento a la persona que lo ha tomado pagando cada uno lo que le hayan servido.

En este proyecto se va a desarrollar la aplicación para dispositivos iOS, tanto para iPhone como para iPad, ya que es el mercado que menos alternativas tiene y por tanto la que menos competencia. Además no existe ninguna aplicación que lea una factura mediante OCR, lo que la haría única en este aspecto dentro de su market.

La metodología de desarrollo seguida en este proyecto es la de desarrollo en cascada. “El desarrollo en cascada es el enfoque metodológico que ordena rigurosamente las etapas del proceso para el desarrollo de software, de tal forma que el inicio de cada etapa debe esperar a la finalización de la etapa anterior” [1].

Fases del proceso:

1. Análisis
2. Diseño
3. Implementación
4. Pruebas
5. Implantación
6. Mantenimiento

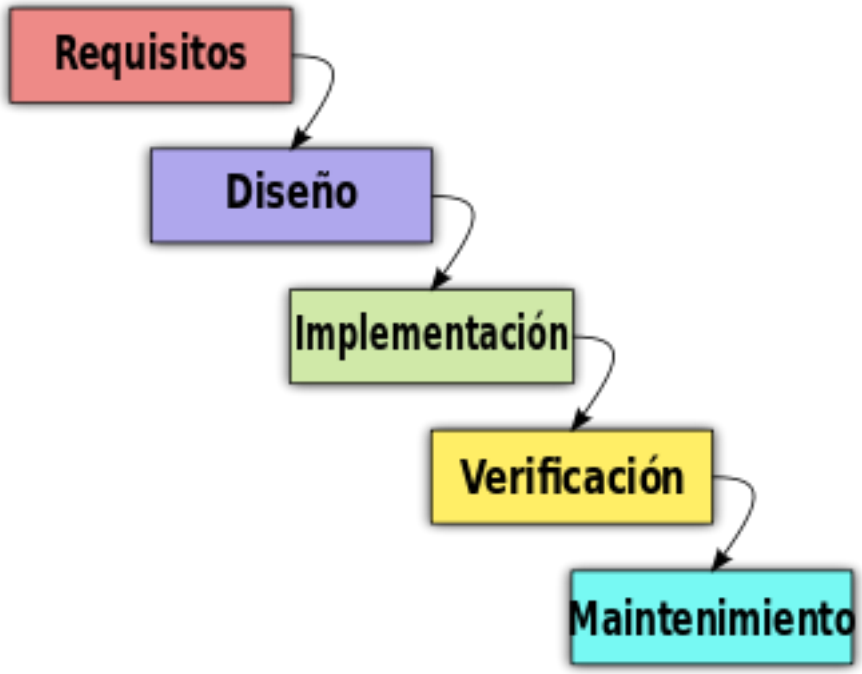


Diagrama de modelo en cascada

En la **fase de análisis** se analizan los necesidades de los usuarios con el fin de determinar el objetivo que se desea cumplir. Se analizan los requerimientos que conllevan cada objetivo y se identifica todos las funcionalidades mediante una especificación completa de lo que debe hacer el sistema.

En la **fase de diseño** se diseña el sistema con los datos recogidos en la etapa anterior. En esta fase se diseña el modelo de datos, el diseño de la interfaz a través de prototipos y su navegabilidad.

En la **fase de implementación** se implementa el código fuente, siguiendo los prototipos y el diseño de la etapa anterior. Durante la implementación se realizarán pruebas unitarias y ensayos para corregir errores durante su desarrollo.

En la **fase de pruebas**, una vez terminada la etapa anterior y con las pruebas unitarias ya superadas, se realizan las pruebas de integración con todo el sistema. Una vez culminada las pruebas de integración el software estará listo para su entrega y puesta en producción. Para aplicaciones móviles, se realizan pruebas en beta a través de los markets, TestFlight en el caso de iOS, probando todas las funcionalidades en varios dispositivos.

La **fase de implantación** es cuando el software se sube a producción. En el caso de las aplicaciones móviles cuando se suben a los markets y quedan disponibles en algún market para la descarga por parte de los usuarios finales.

La **fase de mantenimiento** se realiza una vez publicado el producto, y se trata de realizar un seguimiento sobre su buen funcionamiento. Un buen mantenimiento consiste en detectar posibles errores, o capturar estos para una posterior corrección. Para ello se utilizan herramientas como Crashlytics o Analytics[1] para un seguimiento del funcionamiento de la aplicación.

1.4. Planificación del trabajo

La planificación del proyecto se dividen en 4 hitos importantes, que corresponden a las entregas de cada PEC. Teniendo en cuenta el enfoque y el método elegido en el punto anterior, el primer hito, es decir, la primera entrega de las 4 PEC corresponde a la fase de análisis. En esta primera fase recabaremos información sobre el proyecto y haremos una planificación una vez obtenidos los requerimientos que se precisan para la construcción del producto.

El segundo hito corresponde a la segunda fase del proceso en la metodología elegida, que es la fase de diseño. Esta etapa recoge el diseño centrado en el usuario, donde se define el contexto de uso, el diseño conceptual y el prototipado. Además también incluye el diseño técnico de la aplicación, en el cual se definen los casos de uso y el diseño de la arquitectura de la aplicación.

El tercer hito corresponde a la implementación y pruebas, correspondientes a las fases 3 y 4 del modelo en cascada elegido. Durante el desarrollo, en primer lugar se desarrolla la parte del servidor, en la que se implementa el modelo de datos con la creación de la base de datos y la posterior API para la posterior conexión con la App. También se incluye un SandBox para la API para poder probarla y poder realizar las pruebas necesarias para su correcto funcionamiento.

Una vez desarrollada la API y probada, se desarrollará la aplicación. Durante la implementación se realizarán pruebas unitarias para comprobar el correcto funcionamiento de determinadas funcionalidades de la App. Una vez

[1] Crashlytics y Analytics son herramientas de Google Firebase que se integran en la App a través de su API.

finalizadas las fases de implementación y pruebas unitarias por separado, se realiza las pruebas de integración del sistema completo probando en diferentes dispositivos y por varios usuarios ajenos al desarrollo.

En el último hito y por tanto última entrega, se realizará la implantación de la App en producción, junto a la finalización de la memoria y creación de la presentación del proyecto.

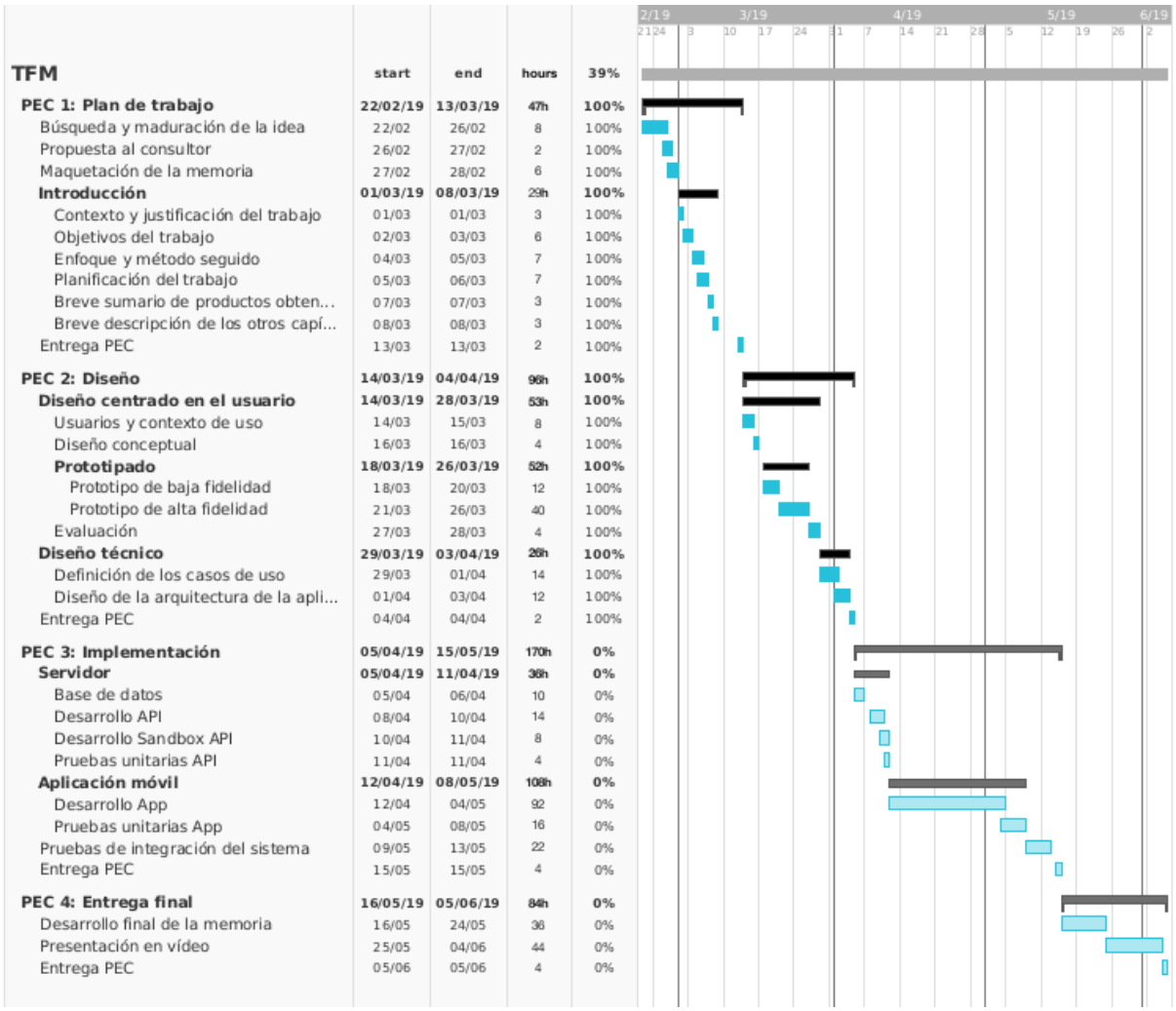
Todos los hitos, además conllevan la evolución de la memoria del TFM que se irá desarrollando en cada fase.

Para cada tarea se establecen x días de trabajo, y la horas dedicadas por días suelen variar los días entre semanas entre 3 y 5 horas, y los fines de semana entre 6 y 8 horas.

En el diagrama de Gantt que se muestra a continuación hay 4 grandes hitos, que son las entregas de las PEC junto con el nombre de las tareas, con su fecha de inicio y su fecha de finalización, y una estimación de las horas de trabajo. Las tareas también llevan asociado una porcentaje de realización que irá creciendo en los hitos conforme se vayan realizando las entregas.

Cada hito lleva una tarea de entrega, en la que se realizará el proceso de comprobación de la memoria, cumplimento de normativas de entrega del TFM y posterior subida al campus virtual para su entrega.

A continuación se muestra un diagrama de Gantt de toda la planificación:



TFM	start	end	hours	39%
PEC 1: Plan de trabajo	22/02/19	13/03/19	47h	100%
Búsqueda y maduración de la idea	22/02	26/02	8	100%
Propuesta al consultor	26/02	27/02	2	100%
Maquetación de la memoria	27/02	28/02	6	100%
Introducción	01/03/19	08/03/19	29h	100%
Contexto y justificación del trabajo	01/03	01/03	3	100%
Objetivos del trabajo	02/03	03/03	6	100%
Enfoque y método seguido	04/03	05/03	7	100%
Planificación del trabajo	05/03	06/03	7	100%
Breve sumario de productos obten...	07/03	07/03	3	100%
Breve descripción de los otros capí...	08/03	08/03	3	100%
Entrega PEC	13/03	13/03	2	100%
PEC 2: Diseño	14/03/19	04/04/19	96h	100%
Diseño centrado en el usuario	14/03/19	28/03/19	53h	100%
Usuarios y contexto de uso	14/03	15/03	8	100%
Diseño conceptual	16/03	16/03	4	100%
Prototipado	18/03/19	26/03/19	52h	100%
Prototipo de baja fidelidad	18/03	20/03	12	100%
Prototipo de alta fidelidad	21/03	26/03	40	100%
Evaluación	27/03	28/03	4	100%
Diseño técnico	29/03/19	03/04/19	26h	100%
Definición de los casos de uso	29/03	01/04	14	100%
Diseño de la arquitectura de la apli...	01/04	03/04	12	100%
Entrega PEC	04/04	04/04	2	100%
PEC 3: Implementación	05/04/19	15/05/19	170h	0%
Servidor	05/04/19	11/04/19	36h	0%
Base de datos	05/04	06/04	10	0%
Desarrollo API	08/04	10/04	14	0%
Desarrollo Sandbox API	10/04	11/04	8	0%
Pruebas unitarias API	11/04	11/04	4	0%
Aplicación móvil	12/04/19	08/05/19	108h	0%
Desarrollo App	12/04	04/05	92	0%
Pruebas unitarias App	04/05	08/05	16	0%
Pruebas de integración del sistema	09/05	13/05	22	0%
Entrega PEC	15/05	15/05	4	0%
PEC 4: Entrega final	16/05/19	05/06/19	84h	0%
Desarrollo final de la memoria	16/05	24/05	36	0%
Presentación en vídeo	25/05	04/06	44	0%
Entrega PEC	05/06	05/06	4	0%

Diagrama de Gantt

En la sección de anexos que podemos encontrar al final de la memoria se encuentra en el anexo 1 el mismo diagrama a mayor escala para una mejor visualización.

Tras realizar el punto “PEC 2: Diseño” realizamos una evaluación de la planificación anterior, y podemos confirmar que se ha cumplido estrictamente la planificación realizada respecto a los días de trabajo. Lo único que se ha incrementado son las horas de trabajo del prototipo de alta fidelidad, que se estimó 25 horas de trabajo y se han realizado 40 horas. Para el resto de tareas se han realizado siguiente la planificación.

1.5. Breve sumario de productos obtenidos

Los productos obtenidos con la realización de este proyecto son, en primer lugar, una aplicación nativa para iOS desarrollada con el lenguaje de programación Swift 4 a través del IDE XCode.

El proyecto también incluye una parte en el servidor que realizará los servicios de guardado de datos en la nube y por tanto poder utilizar tu cuenta de usuario en cualquier dispositivo. En la parte del servidor obtendremos una API para que la App pueda utilizarla, y además un SandBox de pruebas donde se podrá ejecutar la API desde la página simulando llamadas a la API y obteniendo la respuesta en formato JSON. Para el desarrollo de la parte del servidor utilizaremos un Framework PHP, como Codeigniter que se basa en el patrón modelo-vista-controlador.

Otro producto que se realizará durante el Trabajo de Fin de Master es la memoria del proyecto donde quedará documentado todo el proceso realizado durante el desarrollo del proyecto en todas sus fases.

Por último se obtendrá un vídeo de presentación donde se realizará una demostración del funcionamiento de la App para el tribunal del TFM.

1.6. Breve descripción de los otros capítulos de la memoria

El resto de capítulos corresponden a las fases de diseño, desarrollo, implementación y pruebas.

Una vez finalizada la introducción, en los siguientes capítulos profundizaremos en el análisis definiendo los casos de uso, realizándose para ello un diagrama UML donde se identificarán a los actores y flujo a seguir de manera gráfica, además de un listado con el detalle de cada caso de uso.

En la fase de diseño, se diseñará la arquitectura del sistema donde se realizarán los diferentes diagramas correspondientes al diseño de la base de datos, diagrama de clases, patrones de diseños y estructura de la API con la que conectará la aplicación.

En esta fase también se realizará el diseño conceptual, que se encargará de describir los escenarios desde el punto de vista de los usuarios.

Una vez realizado todo el diseño conceptual y lógica de la arquitectura se realizarán los prototipos a bajo y alto nivel de fidelidad. El prototipo debe ser navegable, y para ello se utilizará algún software de prototipado.

Terminada la fase de diseño, comenzaremos con la implementación del software. En primer lugar la parte del servidor, y luego la parte de la aplicación móvil. Durante el desarrollo se realizarán pruebas unitarias para comprobar el correcto funcionamiento de la App.

Una vez finalizada esta fase se realizarán pruebas de integración, que verifican que el producto se integra perfectamente con el servidor y totalmente funcional y sin errores.

Para la entrega final, se realizarán las conclusiones del proyecto y se incluirá en un anexo el manual de usuario de uso de la aplicación y del SandBox de pruebas de la Api.

2. Descripción general del proyecto

2.1. Diseño centrado en el usuario (DCU)

2.1.1. Usuarios y contexto de uso

Existen muchos tipos de personas, pero sin lugar a dudas las que más abundan son las personas que les gusta comer fuera de casa y acompañadas de amigos. En ese grupo de personas se encuentran los usuarios de esta aplicación.

Es muy común en estos encuentros pagar a medias la cuenta, es decir, dividir la cuenta entre las personas que forman el grupo y pagar todos lo mismo independientemente de lo que haya tomado cada uno. Por otra parte, están las personas que prefieren pagar por lo tomado desglosando la cuenta indicando lo que ha tomado cada miembro de la mesa. Estas son los principales usuarios que usarán la aplicación, ya que promueve el pago justo por lo tomado.

Tomando como contexto de uso la situación descrita anteriormente, se determinarán las características y objetivos que debe cumplir la aplicación, partiendo de las actuaciones que esta situación genera para satisfacer las necesidades de dichos usuarios.

2.1.1.1. Fichas User-Persona

Para el análisis de usuarios se le ha explicado a los usuarios la idea y luego se le ha mostrado un prototipo de baja fidelidad mostrando las funcionalidades en una primera versión.

Tras realizar el análisis, se cambió el diseño, se incluyeron nuevas funcionalidades y se modificaron otras porque tras analizarlo puede que no quedaran claras o fueran un poco confusas.

En este estudio participaron 5 personas (2 hombres y 3 mujeres), de las cuales solo mostraremos las fichas User-Persona de los usuarios que más influyeron en los cambios o que aportaron mejores ideas.

A continuación se muestran las fichas User-Persona para el análisis de usuario.

Ángel López

Edad: 30
Profesión: Programador
Familia: Casado sin hijos
Ciudad: Cádiz

¿Qué piensa de la App?
"Yo siempre he pagado a medias, y muchas veces me ha parecido injusto. Creo que esta App solucionará este problema y tendrá buena acogida".

¿Qué cree que la App puede aportar?
"La App va a facilitar un cambio de paradigma a la hora de pagar una cuenta a medías, ya que con solo una foto al ticket, puedes en pocos segundos, seleccionar lo que realmente debe pagar cada uno".

Puntos fuertes:
- Es una persona muy social.
- Le gusta la tecnología.
- Acoge bien los cambios.

Puntos débiles:
- No es una persona asertiva.
- Le cuesta reconocer errores.

Descripción del usuario:
Ángel es un usuario que casi todos los fines de semana come con sus hermanos y sus respectivas familias. Normalmente se reúnen al menos 15 personas y la cuenta la reparten entre las 5 familias (A sus padres les invitan entre todos).
Algunas familias están formadas por la pareja y 3 hijos ya adultos, y en otros caso la familia está formada por solo una pareja sin hijos.
Ángel siempre ha pensado que es injusto pagar a medias, pero por no hacer el cálculo de lo que ha tomado cada persona, pues se ha pagado a medias entre 5.

¿Como influyó su análisis en la aplicación?

Diseño: 5/10
Funcionalidad: 8/10
Navegabilidad: 6/10

Tabla Ficha User-Persona 1

Miriam Rodríguez

Edad: 22
Profesión: Estudiante
Familia: Soltera
Ciudad: Conil (Cádiz)

¿Qué piensa de la App?
"Es una buena idea, sobretodo para los que nos movemos con personas que tenemos distintos niveles adquisitivos".

¿Qué cree que la App puede aportar?
"Donde yo vivo lo tradicional es pagar todos por igual, y va a ser difícil cambiar esa costumbre social. Pero con Apps como esta, un cambio es posible ya que elimina la barrera de tener que estar haciendo cuentas".

Puntos fuertes:
- Es una persona muy positiva.
- Sincera.
- Ve la tecnología como un medio de vida.

Puntos débiles:
- Se apoya demasiado en la tecnología.
- Demasiado impulsiva.

Descripción del usuario:
Miriam es una estudiante de Medicina que frecuentemente sale a comer con sus amigos. Dentro del grupo de personas que frecuenta hay diferentes poderes adquisitivos, unos aún estudian, otros ya tienen un trabajo.
Cuando van a comer o tomar algo, los de mayor poder adquisitivo piden más platos o platos más caros que los de menor poder adquisitivo.
Luego como es costumbre, pagan la cuenta a medias.

¿Como influyó su análisis en la aplicación?

Diseño: 3/10
Funcionalidad: 10/10
Navegabilidad: 5/10

Tabla Ficha User-Persona 2

2.1.2. Diseño conceptual

A continuación se describen posibles escenarios de uso de la aplicación por parte del usuario:

- **Escenario 1**: Un grupo de amigos está almorzando en un restaurante y cada pide su plato y bebida. Hay personas que piden más bebida que otras. El precio de los platos varían mucho.
- **Escenario 2**: Un grupo de amigos se reúne a menudo en un pub para tomar copas. Hay personas que beben más que otras y el precio de las bebidas varían.
- **Escenario 3**: Un grupo de compañeros de piso realizan la compra en un supermercado. En la lista de la compra, hay artículos que utilizarán todos, es decir, artículos comunes, y además incluyen artículos que cada uno utilizará por separado.
- **Escenario 4**: A un grupo de amigos o a una persona en concreto le interesa guardar y llevar el control de los productos y del gasto que realiza en compras.

2.1.3. Prototipo

Antes de la fase de implementación se realiza el diseño final de la aplicación en dos niveles.

Primero se realiza el diseño en papel y lápiz a un nivel bajo de fidelidad para establecer una conexión con la aplicación. Una vez tomadas las decisiones que marcarán el diseño de la aplicación comenzamos con el diseño a alto nivel de fidelidad.

Antes de comenzar el diseño en alta definición, elegimos los posibles colores que probaremos, las diferentes fuentes y las imágenes que llevará. Es importante que el diseño tenga concordancia entre sus elementos, como por ejemplo que su iconografía sea del mismo tipo y no se vean iconos de diferentes terminaciones en la misma aplicación. Lo mismo pasa con los colores y con las fuentes.

Para el prototipo se han seleccionado los siguientes elementos que marcarán el diseño:

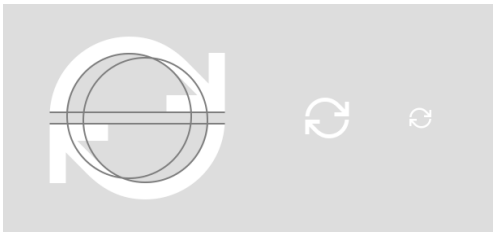
- **Fuente**: Para los texto se utilizará la tipografía Avenir Next con sus respectivos grosores para enmarcar la importancia de los textos (Avenir Next SemiBold, Avenir Next Medium y Avenir Next Regular).

  Se ha elegido esta fuente porque es bastante atractiva y ya viene integrada en XCode, pero podría utilizarse la fuente del sistema del dispositivo.

Avenir® Next

Avenir® Next Font

- **Iconos**: Para la iconografía hemos utilizado un Icon Set bastante completo como es SS-Gizmo, obteniendo iconos del mismo acabado para todo la aplicación. En los casos en los que no se ha encontrado un icono apropiado se ha realizado el icono con Illustrator CS6.



Icon Set SS-Gizmo

- **Imágenes**: El diseño incorpora imágenes de fondo con acabado borroso y con una capa oscura semitransparente para aportar opacidad y se pueda entender el contenido que está sobre él. También se han tomado imágenes de personas para simular los perfiles de los amigos. Estas imágenes se han tomado de bancos de fotos en su versión “vista preliminar” con marca de agua para el diseño.

  Además en las preferencias de la aplicación se incluirá una opción para cambiar el aspecto visual, para que el usuario pueda seleccionar el estilo que quiere en la aplicación ya sean temas claros o temas oscuros, así como diferentes opciones de imágenes de fondo.

- **Logotipo**: El diseño del logotipo de la aplicación representa un ticket con el nombre de la aplicación dentro. Se había valorado otras opciones, pero

está opción en particular, según usuarios encuestados les da sensación de que se asemeja a lo que podría ser una aplicación de un restaurante. Se compone de un fondo con los colores principales de la aplicación y en su interior un recuadro simulando una factura con el nombre de la aplicación dentro.

Diseño de Logotipo de la aplicación

- **Colores**: Se han establecido 5 colores para formar las 4 secciones de la aplicación de forma que cada sección tiene dos colores degradados con 135º, de manera que el último color de una sección es el primer color de la siguiente. Con este diseño se pretende que las acciones de cada sección se identifiquen con los colores que le corresponde dotando a cada sección de un identificación, lo cuál visualmente resulta más fácil de entender.

  Estas son las referencias de los colores que forman la gama:

  - #76FF03
  - #1976D2
  - #6200EA
  - #C51162
  - #DB9600

Gama de colores de la App

Dentro de la aplicación tenemos 4 secciones: Dividir cuenta, Historial, Amigos y Mi cuenta.

La combinación verde-azul es para la sección “Dividir cuenta” y todas las acciones relacionadas con esta sección se identificarán con estos colores, del mismo modo, la combinación azul-lila es para la sección “Historial”, la lila-rosa para la sección “Amigos” y la “rosa-naranja” para la sección “Mi cuenta”.

A continuación mostramos las pantallas del prototipo en alta fidelidad realizadas con los programas Photoshop CS6, Illustrator CS6 y Sketch App.

Medidas base a tener en cuenta:

Márgenes laterales: 20px.

Alto de campos de texto y botones: 50px.

Tamaño de las letras: 12 a 17 px.

Tamaño imágenes usuarios: 60px

2.1.3.1. Navegabilidad de las pantallas

A continuación se encuentra el mapa de navegabilidad de la App. En primer lugar se listarán las diferentes pantallas que forman la app identificándolas con un nombre y una descripción. En el mapa de navegación se utilizará dicho nombre para identificar cada pantalla.

Listado de pantallas

Launch: Pantalla inicial que se muestra mientras se carga la App.

Login: Pantalla para iniciar sesión.

Registro: Pantalla para registrar un nuevo usuario.

Recuperar contraseña: Pantalla para recuperar la contraseña olvidada de un usuario ya registrado.

Dashboard: Pantalla principal de un usuario logado. Contiene al resto de pantallas en adelante.

Dividir: Pantalla para seleccionar el método para añadir un nuevo ticket.

Dividir - Capturar ticket: Pantalla para capturar un nuevo ticket a través de la cámara.

Dividir - Seleccionar ticket: Pantalla para seleccionar de la galería de imágenes del dispositivo una imagen del ticket.

Dividir - Crop ticket: Pantalla para recortar/girar la imagen del ticket, dejando solo los elementos que forman la cuenta.

Dividir - Asignar: Pantalla donde se muestra todos los elementos del ticket capturados o introducidos manualmente donde se le asigna a los amigos con los que va a compartir la cuenta.

Dividir - Elemento: Pantalla para Añadir/Editar un elemento así como la cantidad que le corresponde a cada amigo.

Dividir - Divide con: Pantalla para seleccionar los amigos con los que se va a compartir la cuenta.

Dividir - Resumen: Pantalla con el resumen de la cuenta, indicando cuanto debe pagar cada amigo y un desglose de lo que le han servido.

Historial: Pantalla con el historial de tickets que se han guardado.

Amigos: Pantalla con los amigos (reales y ficticios) que tiene el usuario.

Amigos - Grupos: Pantalla con los grupos de amigos que tiene el usuario.

Amigos - Añadir amigo: Pantalla para añadir un amigo (real o ficticio). Para añadir un amigo real se introduce su código en el campo correspondiente y para crear un ficticio se le añade un nombre en el campo correspondiente.

Amigos - Enlazar amigo real con ficticio: Pantalla para unificar un amigo real que acabas de enlazar con él, con un usuario ficticio que ya tenías.

Amigos - Editar ficticio: Pantalla para editar los datos de un usuario ficticio.

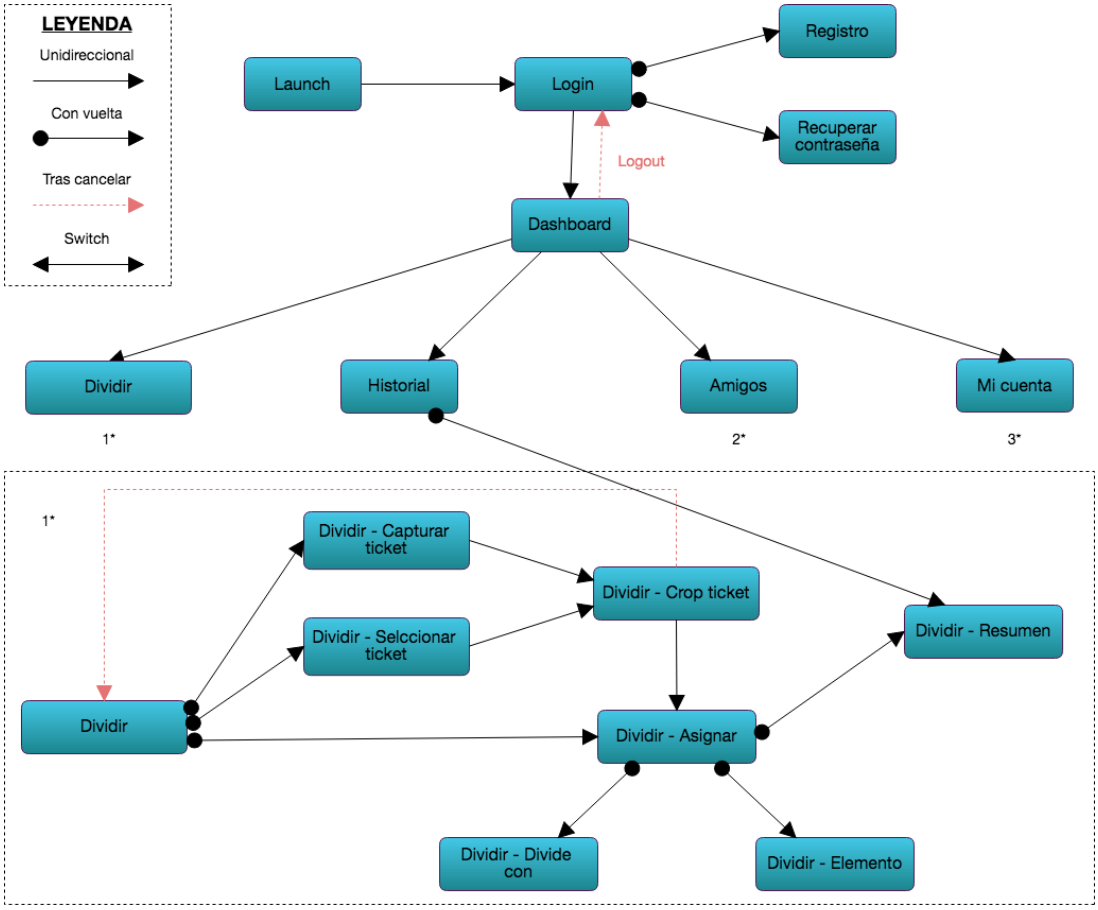
Amigos - Grupos - Añadir/Editar Grupo: Pantalla para crear o editar los datos y miembros de un grupo de amigos.

Mi cuenta: Pantalla menú de las siguientes pantallas.

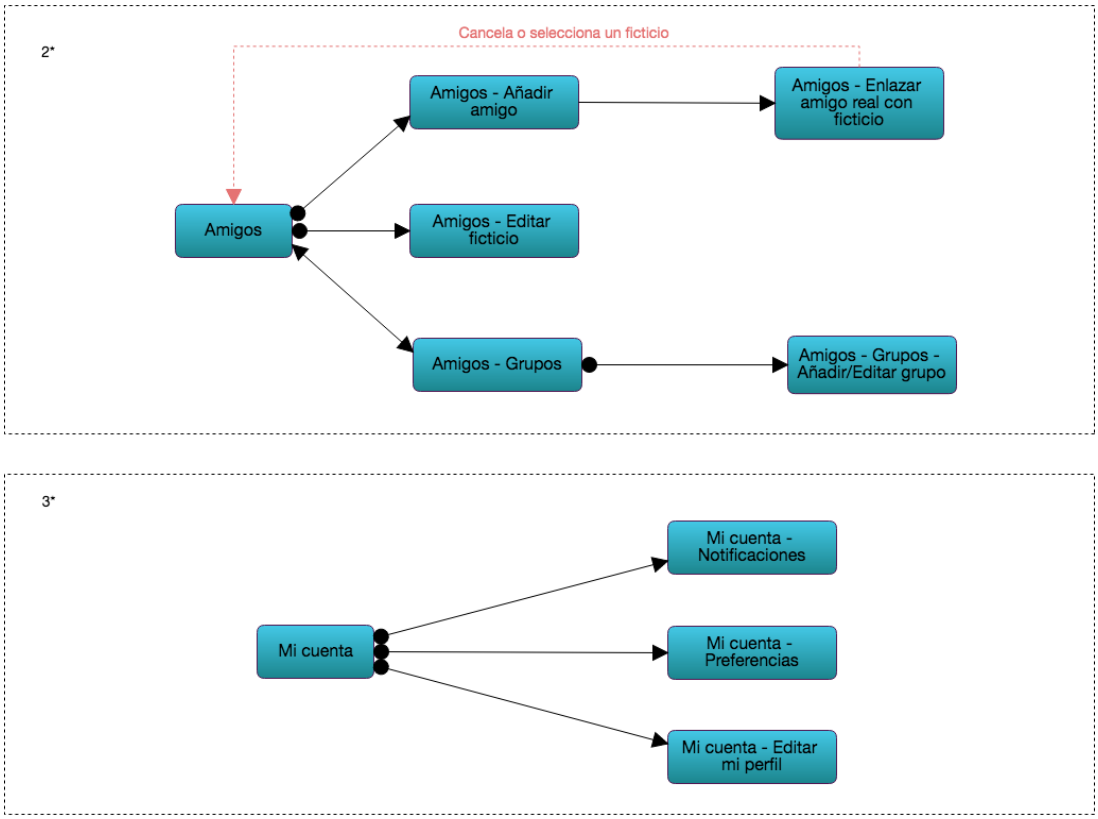
Mi cuenta - Notificaciones: Pantalla donde se muestra las notificaciones que ha recibido el usuario.

Mi cuenta - Preferencias: Pantalla para establecer las preferencias del usuario.

Mi cuenta - Editar mi perfil: Pantalla para editar los datos del perfil del usuario.



Navegabilidad de la aplicación (Parte 1/2)



Navegabilidad de la aplicación (Parte 2/2)

La aplicación comienza con una pantalla de login, el cual una vez logado permanecerá activo mientras que el usuario no decida cerrar la sesión. Por tanto cada vez que acceda a la aplicación esta accederá al Dashboard sin necesidad de volver a introducir correo y contraseña.

La navegación de pantallas dentro de la zona de usuarios logados sigue el sistema de tabs o pestañas en la zona inferior del dashboard, desde el cual se tiene acceso a los 4 grandes grupos de funcionalidades.

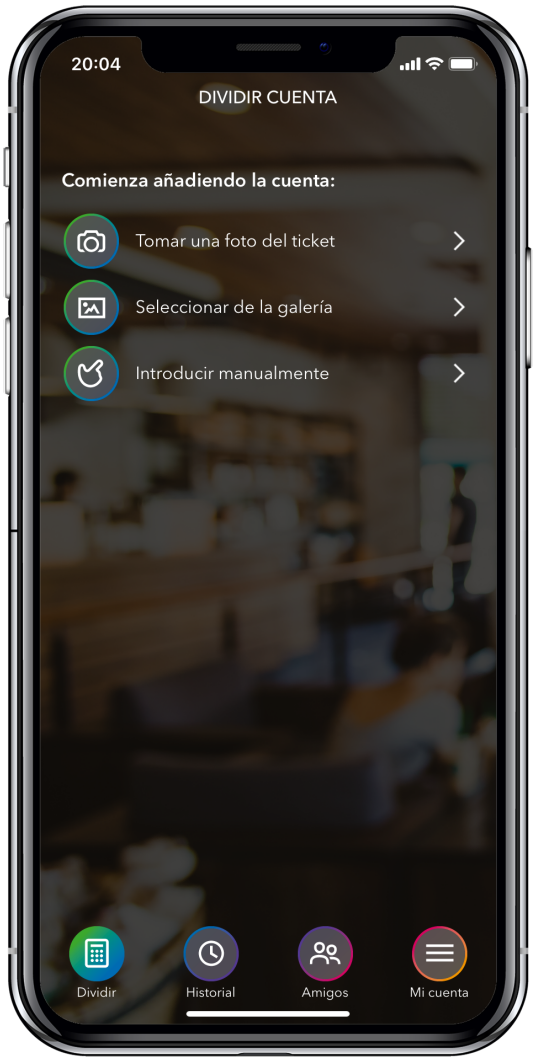
Se ha separado las funcionalidades en 4 grupos, y podemos acceder a ellas en cualquier momento, como por ejemplo, si el usuario está asignando los elementos de una cuenta, y necesita dar un usuario de alta, solo tendrá que pulsar “Amigos” en el tab inferior y podrá añadir un nuevo amigo sin perder lo que esté haciendo en la otra pestaña, así que una vez creado el usuario podrá volver a la pestaña “Dividir” y continuar con la asignación de elementos de la cuenta e incluso poder añadir el nuevo amigo creado.

2.1.3.2. Prototipo alta fidelidad

A continuación se muestra las pantallas diseñadas de la Aplicación. Puede consultar el mapa de navegabilidad del apartado anterior para visualizar una navegación completa de la Aplicación.

Debajo de cada pantalla puede encontrar la justificación del prototipado en el caso de que fuese necesaria su explicación.

Pantalla Login y Pantalla Registro de usuario



Pantalla Selección de método para iniciar una nueva cuenta

Esta es la pantalla principal de la aplicación y la que conduce a todas las funcionalidades principales de la App. En todo momento tendremos acceso a ella y podremos pasar de una pestaña a otra sin perder lo que hacemos en cada una, es decir, por ejemplo podremos añadir nuevos amigos mientras dividimos una cuenta.

Pantalla Capturar ticket mediante foto y Pantalla Recortar imagen

El prototipo de esta parte de la aplicación contiene las funcionalidades básicas que debe tener la captura y recorte del ticket. El diseño luego dependerá de la librería que se utilice.

Pantalla Asignar elementos y Pantalla Añadir/modificar un elemento

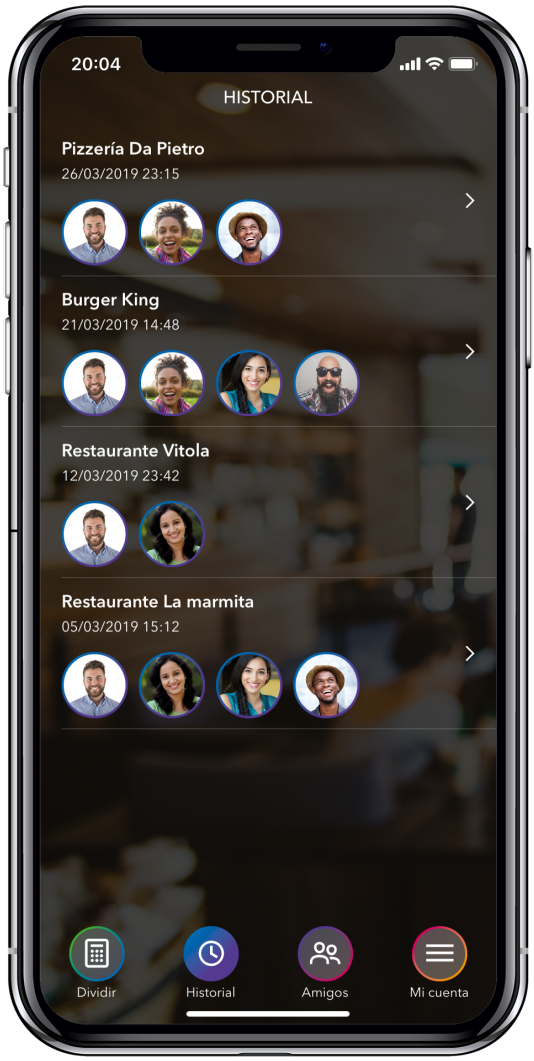
Esta pantalla es la que contiene la funcionalidad principal de la App, que es dividir la cuenta asignando los elementos de un ticket a los amigos con los que quieres compartir la cuenta. Está pensada para que en pocos toques se pueda realizar la división, por tanto por cada elemento de la lista se muestran los avatares de los amigos con los que quieres compartir, pulsando sobre ellos se pueden marcar o desmarcar, y si fuera necesario modificar el elemento o incrementar en más de uno la cantidad, tendríamos que editar el elemento pulsando sobre el icono del lápiz de la derecha.

Pantalla Elegir miembros para dividir y Pantalla Resumen de la cuenta dividida

La pantalla de la izquierda, es el paso siguiente a capturar el ticket, y es donde se eligen los amigos con los que vas a compartir la cuenta. Puedes añadir amigos o grupos completos.

En la pantalla de la derecha, se muestra el resumen de una cuenta cuando ya se han asignado todos los elementos. En cualquier momento puedes volver a la lista para editarlos, y además ya puedes compartir mediante las apps de mensajería el resumen de la cuenta.

En la zona de arriba se introduce el texto con el que se quiere guardar para luego más tarde consultarlo en el historial.



Pantalla Historial de cuentas

En esta pantalla, se muestra un historial ordenado descendentemente por fecha, con el nombre con el que lo hayamos guardado y las personas con las que se ha compartido.

Aquí aparecerán las cuentas que hayas creado tú o que te haya añadido un amigo.

Al pulsar sobre él, se abrirá la pantalla anterior con el resumen de la cuenta, en la cual podremos compartir vía mensajería y solo el creador de la cuenta podrá editar la lista de elementos del ticket.

Pantalla Amigos y Pantalla Grupos

Estas pantallas engloban la funcionalidad de gestionar los amigos y grupos de amigos, controlados por un switch que cambia a cada sección.

En la parte inferior está disponible tu código de usuario, el cual podrás compartir pulsando sobre él para enviarlo mediante Apps de mensajería a quien el usuario seleccione.

En la parte superior de cada sección podemos encontrar un botón para añadir un nuevo amigo o grupo.

Un amigo puede ser otro usuario de la aplicación que ha compartido su código contigo y tu enlazas con él a través de la App. Además puedes crear usuarios ficticios que gestiona el propio usuario para el caso de que quieras compartir una cuenta con alguien que no tenga o quiera descargar la App.

Pantalla Añadir un amigo y Pantalla Asignar un amigo enlazado a uno ficticio

En la primera pantalla podemos ver la pantalla de crear un nuevo usuario a través de un código que hayan compartido contigo o crear un amigo ficticio que gestionar el usuario que lo ha creado.

Cuando enlazamos con otro usuario a través de su código podemos luego fusionarlo con un amigo ficticio que tengamos en la lista, ya que es posible que lo hayamos creado anteriormente para compartir cuentas.

Pantalla Mi cuenta y Pantalla Mi perfil de usuario

A continuación se muestra las posibles alternativas al tema por defecto de la App. En ella se incluyen temas claros, temas oscuros y temas con otras imágenes de fondo. El usuario podrá seleccionar el tema desde las preferencias de usuarios, ya que cada usuario puede preferir un aspecto distinto. El aspecto solo se aplicará a la parte interna de la aplicación donde el usuario ya ha iniciado sesión.

Pantalla Asignar elementos con el tema oscuro y con el tema claro

2.1.4. Evaluación

El diseño de la aplicación se ha realizado en varias fases. En primer lugar se ha diseñado la app con un prototipo inicial de baja fidelidad en papel y lápiz, y luego las pantallas generadas han sido escaneadas y aplicadas en un software de prototipos navegable, en concreto invisionApp, para poder realizar una consulta en diferentes usuarios y así poder comprobar que el diseño sea fácil e intuitivo para el usuario.

Tras varias pruebas con diferentes usuarios y aplicados los cambios convenientes, se opta por realizar un diseño de alta definición y utilizar la misma técnica para probar el diseño real en los usuarios finales,

comprobando que la primera fase ha surgido efecto con los cambios que reportaban los usuarios.

Uno de los cambios, y donde más discordancia había en los usuarios era el tema. A unos les gustaba más un tema claro, a otros les gustaba que tuviese un modo oscuro, y a otros les gustaba una imagen de fondo. Así que se diseño de las 3 formas, y en la implementación se podrá seleccionar el diseño que más le guste al usuario.

Un tema muy importante en el diseño de la aplicación, tanto físico como técnico, ha sido el tener en cuenta la cantidad de los elementos de la cuenta, ya que sería conveniente validar la cantidad que se refleja en el ticket con la asignación a los amigos de ese elemento, validando así que coincide la cantidad con la asignación.

Este tema es muy delicado, ya que en el estudio realizado de los tickets que suelen dar en restaurantes, bares, etc… es muy difícil detectar ese dato como para luego hacer una validación. Más adelante, en la parte de implementación hablaremos sobre el análisis realizado a los tickets y los posibles problemas a la hora de leerlos mediante OCR.

2.2. Diseño técnico de la aplicación

La metodología de desarrollo elegida para la realización del proyecto se basa en el Rational Unified Process (RUP)[1], usando el Unified Modeling Language (UML)[2] como notación, ya que el conjunto de estas dos forman la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

Se ha decidido utilizar esta metodología ya que utilizaremos una metodología orientada a objetos y esta es la metodología más utilizada.

[1] RUP: Rational Unified Process, en español Proceso Unificado Racional, es un proceso de desarrollo de software orientado a objetos

[2] UML: Unified Modeling Language, en español Lenguaje Unificado de Modelado, es un lenguaje de modelado muy conocido y utilizado.

2.2.1. Casos de uso

2.2.1.1. Modelos de casos de uso

Caso de uso: Iniciar sesión con el correo y contraseña de usuario

- **Descripción**: Login de un usuario en la aplicación.
- **Precondición**: Estar registrado como usuario en la aplicación.
- **Postcondición**: El usuario accede al sistema.
- **Actores**: Usuario.
- **Resumen**: Un usuario desea iniciar sesión en la app, para ello introduce su correo y contraseña.
- **Escenario principal**:
    1. El caso de uso comienza cuando el usuario entra en la aplicación y quiere logarse.
    2. El usuario introduce su correo y contraseña en los campos correspondientes, y pulsa sobre el botón “Iniciar sesión”.
    3. El sistema comprueba que los datos de usuario son válidos (existe, el correo está validado y no ha sido baneado).
    4. El sistema le permite acceder a la zona privada de la aplicación.
- **Escenario alternativo**:

    0. El usuario puede cancelar en cualquier momento la operación.

    3a. Algunos de los campos están vacíos.

    1. Muestra el mensaje de advertencia y vuelve al paso 2.

    3b. El usuario no está registrado en el sistema.

    1. Muestra el mensaje de advertencia y vuelve al paso 2.

    3c. La contraseña no corresponde a la del usuario.

    1. Muestra el mensaje de advertencia y vuelve al paso 2.

    3d. El usuario ha sido baneado.

    1. Muestra el mensaje de advertencia y vuelve al paso 2.

    3e. El usuario no ha confirmado su cuenta de correo.

    1. El usuario inicia sesión y le muestra una pantalla con las indicaciones para validar el correo de usuario proporcionado.

Caso de uso: Iniciar sesión a través de una red social

- **Descripción**: Login/Registro de un usuario en la aplicación.
- **Precondición**: Disponer de una cuenta en Facebook, Twitter o Google.
- **Postcondición**: El usuario accede al sistema y se registra su usuario si no tenía una cuenta ya creada con los datos proporcionados por la red social.
- **Actores**: Usuario.
- **Resumen**: Un usuario desea iniciar sesión en la app, para ello pulsa en algunas de las redes sociales para hacer login/registrarse con el permiso que otorga la red social.
- **Escenario principal**:
    1. El caso de uso comienza cuando el usuario entra en la aplicación y quiere logarse.
    2. El usuario pulsa sobre el botón de alguna red social que hay en la pantalla de login para acceder a través del permiso y los datos que proporciona la red social.
    3. La aplicación abre en un navegador o la aplicación de la red social en caso de tenerla instalada, y pregunta al usuario si le concede permiso para proporcionar los datos (correo, nombre y avatar) a nuestra aplicación.
    4. El usuario acepta el permiso de proporcionar estos datos a la aplicación.
    5. El sistema comprueba que los datos proporcionados son válidos, es decir, le han proporcionado un correo y un nombre de usuario.
    6. El sistema comprueba que hay un usuario con esos datos.
    7. El sistema le permite acceder a la zona privada de la aplicación.
- **Escenario alternativo**:

    0. El usuario puede cancelar en cualquier momento la operación.

    3a. El usuario deniega el permiso en la aplicación de la red social para proporcionarnos los datos solicitados.

        1. Muestra el mensaje de advertencia y vuelve al paso 1.

    4a. El usuario acepta el permiso pero modifica los datos a proporcionar y no devuelve los datos solicitados.

1. Muestra el mensaje de advertencia y vuelve al paso 1.

6a. No hay ningún usuario en el sistema con los datos proporcionados.

1. El sistema crea el usuario con los datos.
2. El sistema valida el correo del usuario al venir ya validado de una Red social.
3. Continua por el paso 7.

6b. El usuario ha sido baneado.

1. Muestra el mensaje de advertencia y vuelve al paso 1.

Caso de uso: Crear una cuenta de usuario

- **Descripción**: Registro de una nueva cuenta de usuario.
- **Precondición**: Disponer de una cuenta de correo electrónico.
- **Postcondición**: Creación de una cuenta de usuario.
- **Actores**: Usuario.
- **Resumen**: Se registra la cuenta de usuario con los datos proporcionados y se envía un correo de validación para la verificación del correo proporcionado.
- **Escenario principal**:
  1. El caso de uso comienza cuando el usuario entra en la aplicación y pulsa sobre registrarse.
  2. El usuario introduce su nombre o alias, correo electrónico y la contraseña en los campos correspondientes, y pulsa sobre el botón “Registrarse”.
  3. El sistema comprueba que los datos de usuario son válidos (no existe el correo electrónico en el sistema, el correo y la contraseña son válidas).
  4. El sistema crea el usuario.
  5. El sistema envía un correo con un enlace para verificar el correo electrónico proporcionado.
- **Escenario alternativo**:
  0. El usuario puede cancelar en cualquier momento la operación.

3a. Algunos de los campos están vacíos.

1. Muestra el mensaje de advertencia y vuelve al paso 2.

3b. El usuario ya está registrado en el sistema.

1. Muestra el mensaje de advertencia y vuelve al paso 2.

3c. La correo no es válido.

1. Muestra el mensaje de advertencia y vuelve al paso 2.

3d. La contraseña no cumple con los requisitos exigidos.

1. Muestra el mensaje de advertencia y vuelve al paso 2.

Caso de uso: Recordar contraseña a través del correo electrónico

- **Descripción**: Recordar contraseña de usuario.
- **Precondición**: Disponer de una cuenta de usuario.
- **Postcondición**: Enviar un correo con un enlace para introducir una nueva contraseña.
- **Actores**: Usuario.
- **Resumen**: Se envía al usuario un correo electrónico con un enlace para que proporcione una nueva contraseña y pueda acceder al sistema.
- **Escenario principal**:
    1. El caso de uso comienza cuando el usuario no recuerda su contraseña y pulsa sobre “Olvidé mi contraseña”.
    2. El usuario introduce correo electrónico con el que se registró y pulsa sobre el botón “Recordar contraseña”.
    3. El sistema comprueba que existe un usuario con ese correo.
    4. El sistema crea un permiso temporal para que desde un enlace proporcionado el usuario pueda introducir una nueva contraseña.
    5. El sistema envía un correo con dicho enlace.
    6. El usuario pulsa en el enlace recibido por correo y se le abre un formulario en un navegador.
    7. El usuario introduce la nueva contraseña en el campo especificado.
    8. El sistema válida que la contraseña proporcionada sea válida.
    9. El sistema actualiza la nueva contraseña del usuario.

- **Escenario alternativo**:

  0. El usuario puede cancelar en cualquier momento la operación.

  3a. El campo 'correo electrónico' está vacío.

  1. Muestra el mensaje de advertencia y vuelve al paso 2.

  3b. El usuario no está registrado en el sistema.

  1. Muestra el mensaje de advertencia y vuelve al paso 2.

  8a. La contraseña no cumple con los requisitos exigidos.

  1. Muestra el mensaje de advertencia y vuelve al paso 7.

Caso de uso: Cambio de avatar del perfil de usuario

- **Descripción**: Modificar la imagen de avatar del perfil de usuario.
- **Precondición**: Haber iniciado sesión en la aplicación.
- **Postcondición**: El usuario modifica su avatar del perfil.
- **Actores**: Usuario.
- **Resumen**: Un usuario desea modificar la imagen de avatar.
- **Escenario principal**:
  1. El caso de uso solicita al sistema cambiar su foto de perfil.
  2. El usuario pulsa sobre la imagen de su avatar.
  3. La aplicación muestra las opciones para proporcionar la imagen (capturar foto o escoger de la galería).
  4. El usuario selecciona el método para proporcionar la imagen y la proporciona.
  5. El sistema comprueba la imagen y cambia la foto del usuario en su perfil.
- **Escenario alternativo**:

  0. El usuario puede cancelar en cualquier momento la operación.

  5a. El usuario no proporciona una imagen.

  1. Muestra el mensaje de advertencia y vuelve al paso 2.

Caso de uso: Compartir tu código de usuario

- **Descripción**: Compartir tú código de usuario a través de las apps de mensajería.
- **Precondición**: Haber iniciado sesión en la aplicación.
- **Postcondición**: El usuario comparte su código a través de la aplicación seleccionada para ello.
- **Actores**: Usuario.
- **Resumen**: Un usuario desea compartir su código con algún amigo a través de una app de mensajería o cualquier otra que permita compartir contenido.
- **Escenario principal**:
    1. El caso de uso comienza cuando accede a la sección de “amigos”.
    2. El usuario pulsa su código para compartirlo.
    3. El sistema muestra las diferentes aplicaciones por las que puede compartir el texto.
    4. El usuario selecciona la aplicación por la que compartirá el código.
    5. En la aplicación seleccionada se encargará de seleccionar con quién y cómo quiere compartirse.
- **Escenario alternativo**:
    0. El usuario puede cancelar en cualquier momento la operación.

Caso de uso: Crear vínculos de amistad a través de un código

- **Descripción**: Un amigo te ha compartido su código y quieres agregarlo como amigo.
- **Precondición**: Haber iniciado sesión en la aplicación.
- **Postcondición**: Se crea un vínculo de amistad entre los usuarios.
- **Actores**: Usuario 1 (comparte su código) y usuario 2 (vincula el usuario a través del código en la aplicación).
- **Resumen**: Un usuario desea agregar como amigo a otro usuario de la aplicación a través de su código de usuario.
- **Escenario principal**:
    1. El caso de uso comienza cuando el usuario solicita al sistema vincular a un amigo.

2. El usuario escribe el código que le ha compartido su amigo y pulsa sobre “Añadir amigo”.
3. El sistema comprueba que ese código pertenece a algún usuario.
4. El sistema vincula a ambos como amigos.

- **Escenario alternativo**:

0. El usuario puede cancelar en cualquier momento la operación.

3a. El campo ‘código’ está vacío.

1. Muestra el mensaje de advertencia y vuelve al paso 2.

3b. El código no pertenece a ningún usuario.

1. Muestra el mensaje de advertencia y vuelve al paso 2.

3c. El código proporcionado pertenece a un amigo que ya tenías vinculado.

1. Muestra el mensaje de advertencia y vuelve al paso 2.

Caso de uso: Crear amigos ficticios

- **Descripción**: Crear un usuario no real, con el que puedes realizar la división de cuentas sin necesidad de descargar la app.
- **Precondición**: Haber iniciado sesión en la aplicación.
- **Postcondición**: Se vincula un usuario ficticio al creador de dicho amigo.
- **Actores**: Usuario.
- **Resumen**: Un usuario desea crear un usuario que esté vinculado únicamente a él y poder realizar divisiones en cuentas.
- **Escenario principal**:

1. El caso de uso comienza cuando el usuario solicita al sistema crear un amigo ficticio.
2. El usuario escribe el nombre del amigo ficticio y pulsa sobre “Añadir amigo”.
3. El sistema comprueba de que no exista un usuario con el mismo nombre dentro de sus usuarios ficticios.
4. El sistema solicita una imagen de avatar para dicho amigo.
5. El usuario pulsa sobre la imagen por defecto del avatar del usuario ficticio.

6. La aplicación muestra las opciones para proporcionar la imagen (capturar foto o escoger de la galería).
7. El usuario selecciona el método para proporcionar la imagen y la proporciona.
8. El sistema comprueba la imagen y cambia la foto del avatar del usuario ficticio.
9. El sistema añade como amigo al usuario ficticio.

- **Escenario alternativo**:

0. El usuario puede cancelar en cualquier momento la operación.

3a. El campo 'nombre' está vacío.

1. Muestra el mensaje de advertencia y vuelve al paso 2.

3b. El nombre proporcionado ya existe como amigo ficticio del usuario.

1. Muestra el mensaje de advertencia y vuelve al paso 2.

8a. El usuario no proporciona una imagen.

1. Muestra el mensaje de advertencia y vuelve al paso 5.

Caso de uso: Asignar un amigo real a un usuario ficticio

- **Descripción**: Tras crear un vínculo de amistad con otro usuario cabe la posibilidad de que ya lo tuvieras como amigo ficticio.
- **Precondición**: Tener un amigo vinculado y un amigo ficticio.
- **Postcondición**: Se unifica el usuario amigo con el usuario ficticio heredando las cuentas divididas que tenía el ficticio el usuario real.
- **Actores**: Usuario 1 (creado del amigo ficticio) y usuario 2 (usuario que queda hereda el usuario ficticio).
- **Resumen**: Un usuario desea vincular un usuario ficticio con un usuario amigo real.
- **Escenario principal**:

1. El caso de uso comienza cuando el usuario solicita al sistema asignar un amigo real a uno ficticio.
2. El sistema muestra el listado de usuarios ficticios que tiene el usuario.
3. El usuario selecciona el usuario ficticio que quiere vincular al real.

4. El sistema pasa las cuentas dividas del usuario ficticio al usuario real.
5. El sistema elimina el usuario ficticio.

- **Escenario alternativo**:

0. El usuario puede cancelar en cualquier momento la operación.

2a. El listado de usuarios ficticio está vacío.

1. Muestra el mensaje de advertencia y cancela la operación.

Caso de uso: Crear un grupo de amigos

- **Descripción**: Un usuario crea un grupo de amigos de manera personal y transparente para los usuarios agregados al grupo.
- **Precondición**: Tener al menos un amigo real o ficticio vinculado como amigo para poder añadir al grupo.
- **Postcondición**: Se crea un grupo de amigos.
- **Actores**: Usuario.
- **Resumen**: Un usuario desea crear un grupo de amigos para añadirlos rápidamente en la división de cuentas.
- **Escenario principal**:

1. El caso de uso comienza cuando el usuario solicita al sistema crear un grupo de amigos.
2. El usuario establece la foto de avatar del grupo, rellena el campo nombre, los usuarios que forman el grupo y pulsa sobre “Crear grupo”.
3. El sistema comprueba los datos.
4. El sistema crea el grupo con los datos y usuarios seleccionados.

- **Escenario alternativo**:

0. El usuario puede cancelar en cualquier momento la operación.

3a. El campo ‘nombre’ está vacío.

1. Muestra el mensaje de advertencia y vuelve al paso 2.

3b. La imagen del grupo no se ha establecido.

1. Muestra el mensaje de advertencia y vuelve al paso 2.

3c. No se ha seleccionado ningún usuario.

1. Muestra el mensaje de advertencia y vuelve al paso 2.

Caso de uso: Crear una nueva cuenta/ticket para dividir entre amigos

- **Descripción**: El usuario introduce un nuevo ticket y se asignan el responsable de pagar cada elemento de la lista.
- **Precondición**: Ninguna.
- **Postcondición**: Se genera un resumen de los gastos asignados a cada miembro.
- **Actores**: Usuario y amigos (usuarios reales y ficticios) asignados.
- **Resumen**: Un usuario crea un nuevo ticket para dividir los gastos entre varios amigos.
- **Escenario principal**:
    1. El caso de uso comienza cuando el usuario solicita al sistema crear una nueva cuenta para compartir gastos.
    2. El usuario selecciona la forma en la que introduce el ticket (mediante un foto ya sea seleccionada de la galería o tomada con la cámara o introduciendo los elementos de la lista manualmente).
    3. El sistema procesa la imagen y obtiene el listado de elementos del ticket mediante OCR.
    4. El sistema muestra el listado de elementos obtenidos o muestra el listado vacío en el caso de seleccionar la introducción manual en el paso 2. Además muestra un campo para establecer la propina que se dividirá proporcionalmente y el porcentaje de impuesto a aplicar si es necesario.
    5. El usuario modifica los elementos de lista en los casos que requieran modificación o incluye nuevos elementos.
    6. El pulsa sobre el botón “Dividir con” para elegir a los amigos con los que compartir la cuenta.
    7. El sistema muestra el listado de amigos y grupos disponibles para incluir en la cuenta.
    8. El usuario selecciona a los usuarios con los que desea compartir la cuenta y pulsa sobre el botón “Asignar”.

9. El sistema muestra el listado cada elemento del ticket con los usuarios asignados en el paso anterior en un estado no seleccionado.
10. El usuario pulsa sobre los avatares de los usuarios en cada elemento para asignar a dichos usuarios como responsables de pagar ese elemento.
11. El usuario establece el porcentaje de propina y el porcentaje de impuesto a incluir en los precios desglosados en caso de estos fuesen sin impuesto incluido.
12. El usuario pulsa sobre “Validar”.
13. El sistema comprueba que todos los datos son correctos y que está el ticket está completamente asignado.
14. El sistema guarda los datos y muestra el resumen de la división de gastos.
15. El usuario puede establecer un nombre a la división de esta cuenta que por defecto será la fecha en la que se ha guardado.
16. El sistema guarda el nombre de esta división de cuenta.
17. El sistema notifica a los usuarios reales (que tengan activado en las preferencias que desean recibir notificaciones de este tipo) que hayan sido asignados a la división de esta cuenta.

- **Escenario alternativo**:

0. El usuario puede cancelar en cualquier momento la operación.

3a. El lector de OCR no lee ningún texto en el ticket o no es capaz de obtener el listado de elementos del ticket.

1. Muestra el mensaje de advertencia y vuelve al paso 2.

7a. El listado de amigos está vacío y no puede seleccionar ninguno.

1. Sigue con el siguiente paso y el gasto será completamente para el usuario que está realizando la división del ticket[1].

13a. El sistema comprueba de que existen elementos sin asignar.

1. Muestra el mensaje de advertencia y vuelve al paso 12.

16a. El campo “cuenta” está vacío

1. Muestra el mensaje de advertencia y vuelve al paso 15.

[1] Esta funcionalidad puede ser interesante cuando se desea guardar tickets solo para ti a modo de histórico de gastos.

Caso de uso: Compartir el resumen de una cuenta dividida a través de una app de mensajería

- **Descripción**: Un usuario tras crear una división de una cuenta o accediendo a ella desde el histórico puede compartir el texto resumen de lo que debe pagar cada miembro asignado.
- **Precondición**: Haber creado la división de una cuenta.
- **Postcondición**: Se envía un texto resumen de la división de gastos.
- **Actores**: Usuario.
- **Resumen**: Envío del texto resumen de una cuenta a través de una aplicación de mensajería.
- **Escenario principal**:
    1. El caso de uso comienza cuando el usuario solicita al sistema enviar el resumen de una cuenta dividida.
    2. El usuario accede al resumen de una cuenta dividida, ya sea tras terminar de crearla o desde el historial.
    3. El usuario pulsa sobre el botón “Compartir”.
    4. El sistema muestra las diferentes aplicaciones por las que puede compartir el texto.
    5. El usuario selecciona la aplicación por la que compartirá el código.
    6. En la aplicación seleccionada se encargará de seleccionar con quién o cómo quiere compartirse.
- **Escenario alternativo**:
    0. El usuario puede cancelar en cualquier momento la operación.

Caso de uso: Consultar el histórico de cuentas divididas

- **Descripción**: Listado del histórico de cuentas creadas por ti o por los usuarios que son amigos vinculados y te asignan a una de sus cuentas divididas.
- **Precondición**: Haber creado alguna cuenta o haber sido asignado por algún amigo a alguna cuenta dividida.
- **Postcondición**: Se muestra el listado histórico.

- **Actores**: Usuario.
- **Resumen**: Mostrar el listado histórico de cuentas divididas en las que estés asignado como miembro.
- **Escenario principal**:
    1. El caso de uso comienza cuando el usuario solicita al sistema ver el histórico de cuentas divididas.
    2. El sistema muestra el listado de cuentas divididas.
- **Escenario alternativo**:
    0. El usuario puede cancelar en cualquier momento la operación.

Caso de uso: Establecer/modificar las preferencias de usuario

- **Descripción**: Establecer las preferencias de usuario.
- **Precondición**: Haber iniciado sesión como usuario.
- **Postcondición**: Cambiar las preferencias de usuario.
- **Actores**: Usuario.
- **Resumen**: Cambiar las preferencias de usuario, que son, las notificaciones que deseas recibir, el valor de la propina y el tipo de impuesto que se establecerá por defecto.
- **Escenario principal**:
    1. El caso de uso comienza cuando el usuario solicita al sistema las preferencias de usuario.
    2. El sistema muestra las preferencias de usuario.
    3. El usuario modifica las preferencias individualmente.
    4. El sistema guarda los cambios al modificar la preferencia.
- **Escenario alternativo**:
    0. El usuario puede cancelar en cualquier momento la operación.

2.2.2. Diseño de la arquitectura

2.2.2.1. Arquitectura del software

La aplicación utiliza la arquitectura Modelo-Vista-Controlador tanto en el cliente como en el servidor, ambos basados en frameworks MVC.

La arquitectura **MVC** es un patrón de arquitectura de software, que separa los datos y la lógica de negocio, construyendo tres componentes o capas distintas, trabajando a varios niveles de abstracción y poder así separar la interfaz de la implementación. Además favorece el mantenimiento y la reutilización de código.

Los tres componentes o capas son:

- **Vista**: Es la capa con el nivel más alto de abstracción y por tanto la más cercana al usuario. Es la responsable de la interacción entre el usuario y el sistema.
- **Controlador**: Esta capa recibe todas las peticiones del usuario y es la encargada de implementar las reglas que deben cumplirse (funcionalidades del sistema).
- **Modelo**: Es la capa con el nivel más bajo de abstracción y es la encargada de interactuar con el SGBD.

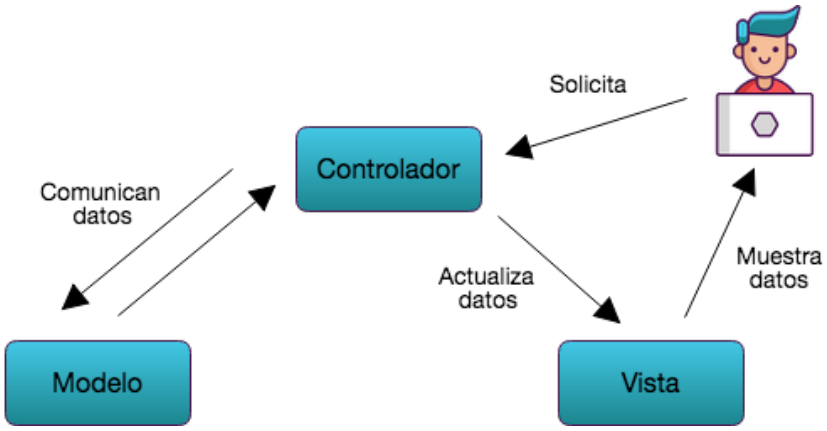


Diagrama arquitectura Modelo-Vista-Controlador

2.2.2.2. Diagrama Entidad-Relación (ER)

A continuación se muestra el diagrama Entidad-Relación. Para mayor comprensión y visibilidad del diagrama se han omitido los atributos de las entidades, ya que esto dificultaría la lectura del diagrama al ser un número considerado los atributos de las entidades. Para ver los atributos de cada entidad véase la sección “Tipo de entidades” a continuación del diagrama.

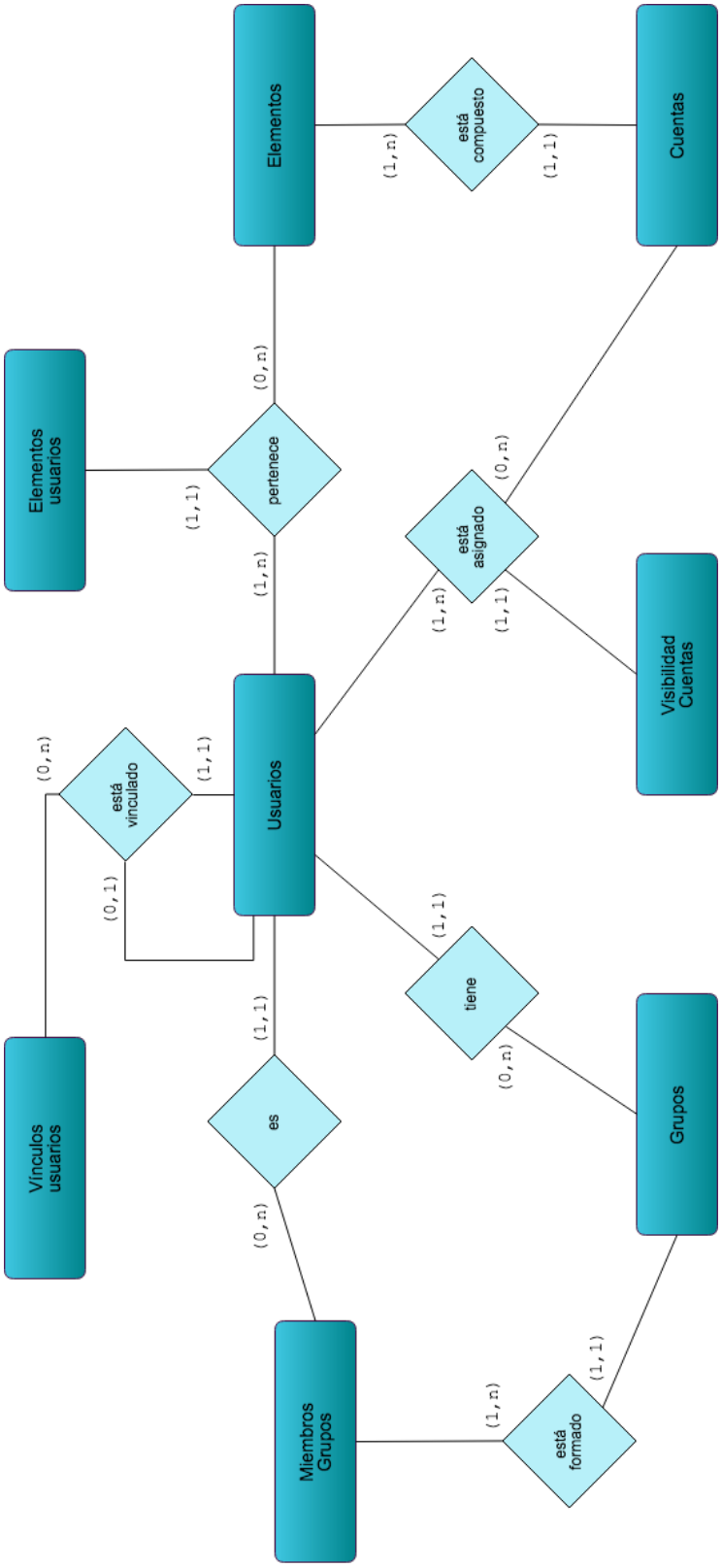


Diagrama Entidad-Relación

Tipos de entidades

En el siguiente cuadro se muestran todas las entidades, junto a sus atributos y el tipo de entidad.

Nombre	Atributos	Tipo
Usuarios	usuarioId, nombre, correo, password, validado, saneado, avatar, codigo, ficticio, pertenece	Fuerte
Cuentas	cuentaId, nombre, fecha, impuesto, propina	Fuerte
Elementos	elementoId, nombre, precio, cuentaId	Fuerte
ElementosUsuarios	elementousuarioId, elementoId, usuarioId, cantidad	Fuerte
VisibilidadCuentas	visibilidadcuentaId, cuentaId, usuarioId, creador	Fuerte
Grupo	grupoId, nombre, avatar, usuarioId	Fuerte
MiembrosGrupos	miembrogrupoId, grupoId, usuarioId	Fuerte
VinculosUsuarios	vinculousuarioId, usuarioId1, usuarioId2, fecha	Fuerte

Tabla de entidades

Atributos de las entidades

En los siguientes cuadros se muestran las entidades descritas en el apartado anterior con la descripción de cada uno de los atributos que forman cada entidad.

Nombre	Descripción	Tipo	Nulo
usuarioId	Identificador único del usuario	CP	No
nombre	Nombre del usuario	Simple	No
correo	Correo electrónico del usuario	Simple	Si
password	Contraseña del usuario	Simple	Si
validado	Estado de verificación del usuario	Simple	No
baneado	Estado de bloqueo del usuario	Simple	No
avatar	Ubicación de la imagen de avatar del usuario	Simple	No
código	Código de amigo del usuario	Simple	Si

ficticio	Es usuario ficticio	Simple	No
pertenece	Pertenece a un usuarioId como usuario ficticio	Simple	No

Tabla de atributos de la entidad Usuarios

Nombre	Descripción	Tipo	Nulo
cuentaId	Identificador único de la cuenta	CP	No
nombre	Nombre de la cuenta	Simple	No
fecha	Fecha de creación	Simple	No
impuesto	Porcentaje de impuesto aplicado a la cuenta	Simple	No
propina	Porcentaje de propina aplicada a la cuenta	Simple	No

Tabla de atributos de la entidad Cuentas

Nombre	Descripción	Tipo	Nulo
elementoId	Identificador único del elemento	CP	No
nombre	Nombre del elemento	Simple	No
precio	Precio del elemento	Simple	No
cuentaId	Identificador de la cuenta a la que pertenece	Simple	No

Tabla de atributos de la entidad Elementos

Nombre	Descripción	Tipo	Nulo
elementousuarioId	Identificador único del elemento del usuario	Simple	No
elementoId	Identificador del elemento	CP	No
usuarioId	Identificador del usuario asociado al elemento	CP	No
cantidad	Cantidad que se le aplica	Simple	No

Tabla de atributos de la entidad ElementosUsuarios

Nombre	Descripción	Tipo	Nulo
visibilidadcuentaId	Identificador único de la visibilidad de la cuenta	Simple	No
cuentaId	Identificador de la cuenta	CP	No
usuarioId	Identificador del usuario que ve la cuenta	CP	No
creador	Es el creador de la cuenta	Simple	No

Tabla de atributos de la entidad VisibilidadCuentas

Nombre	Descripción	Tipo	Nulo
grupoId	Identificador único del grupo	CP	No
nombre	Nombre del grupo	Simple	No
avatar	Ubicación de la imagen de avatar del grupo	Simple	No
usuarioId	Identificador del usuario al que pertenece	Simple	No

Tabla de atributos de la entidad Grupos

Nombre	Descripción	Tipo	Nulo
miembrogrupoId	Identificador único del miembro del grupo	Simple	No
grupoId	Identificador del grupo al que pertenece	CP	No
usuarioId	Identificador del usuario que pertenece al grupo	CP	No

Tabla de atributos de la entidad MiembrosGrupos

Nombre	Descripción	Tipo	Nulo
vinculousuarioId	Identificador único del vínculo entre usuarios	Simple	No
usuarioId1	Identificador de usuario que vincula al segundo usuario	CP	No
UsuarioId2	Identificador del usuario que ha sido vinculado al primer usuario	CP	No
fecha	Fecha de creación del vínculo	Simple	No

Tabla de atributos de la entidad VinculosUsuarios

Relaciones

Las relaciones que intervienen son:

- **Pertenece**: Relación que existe entre Usuarios y Elementos.
- **Está compuesto**: Relación que existe entre Cuentas y Elementos.
- **Está asignado**: Relación que existe entre Cuentas y Usuarios.
- **Tiene**: Relación que existe entre Usuarios y Grupos.
- **Está formado**: Relación que existe entre Grupos y MiembrosGrupos.
- **Es**: Relación que existe entre MiembrosGrupos y Usuarios.
- **Está vinculado**: Relación que existe entre Usuarios y la misma entidad.

2.2.2.3. Diseño lógico de la base de datos

Tablas de la base de datos después del estudio lógico.

- **Usuarios**: usuarioId, nombre, correo, password, validado, saneado, avatar, codigo, ficticio, pertenece
  - Clave primaria: usuarioId
- **Cuentas**: cuentaId, nombre, fecha, impuesto, propina
  - Clave primaria: cuentaId
- **Elementos**: elementoId, nombre, precio, cuentaId
  - Clave primaria: elementoId
  - Clave foránea: cuentaId a Cuentas.cuentaId
- **ElementosUsuarios**: elementousuarioId, elementoId, usuarioId, cantidad
  - Clave primaria: elementoId, usuarioId
  - Clave foránea: elementoId a Elementos.elementoId, usuarioId a Usuarios.usuarioId
- **VisibilidadCuentas**: visibilidadcuentaId, cuentaId, usuarioId, creador
  - Clave primaria: cuentaId, usuarioId
  - Clave foránea: cuentaId a Cuentas.cuentaId, usuarioId a Usuarios.usuarioId

- **Grupos**: grupoId, nombre, avatar, usuarioId
  - Clave primaria: grupoId
  - Clave foránea: usuarioId a Usuarios.usuarioId
- **MiembrosGrupos**: miembrogrupoId, grupoId, usuarioId
  - Clave primaria: miembrogrupoId
  - Clave foránea: grupoId a Grupos.grupoId, usuarioId a Usuarios.usuarioId
- **VinculosUsuarios**: vinculousuarioId, usuarioId1, usuarioId2, fecha
  - Clave primaria: vinculousuarioId
  - Clave foránea: usuarioId1 a Usuarios.usuarioId, usuarioId2 a Usuarios.usuarioId

2.2.2.4. Diagrama de clases conceptuales

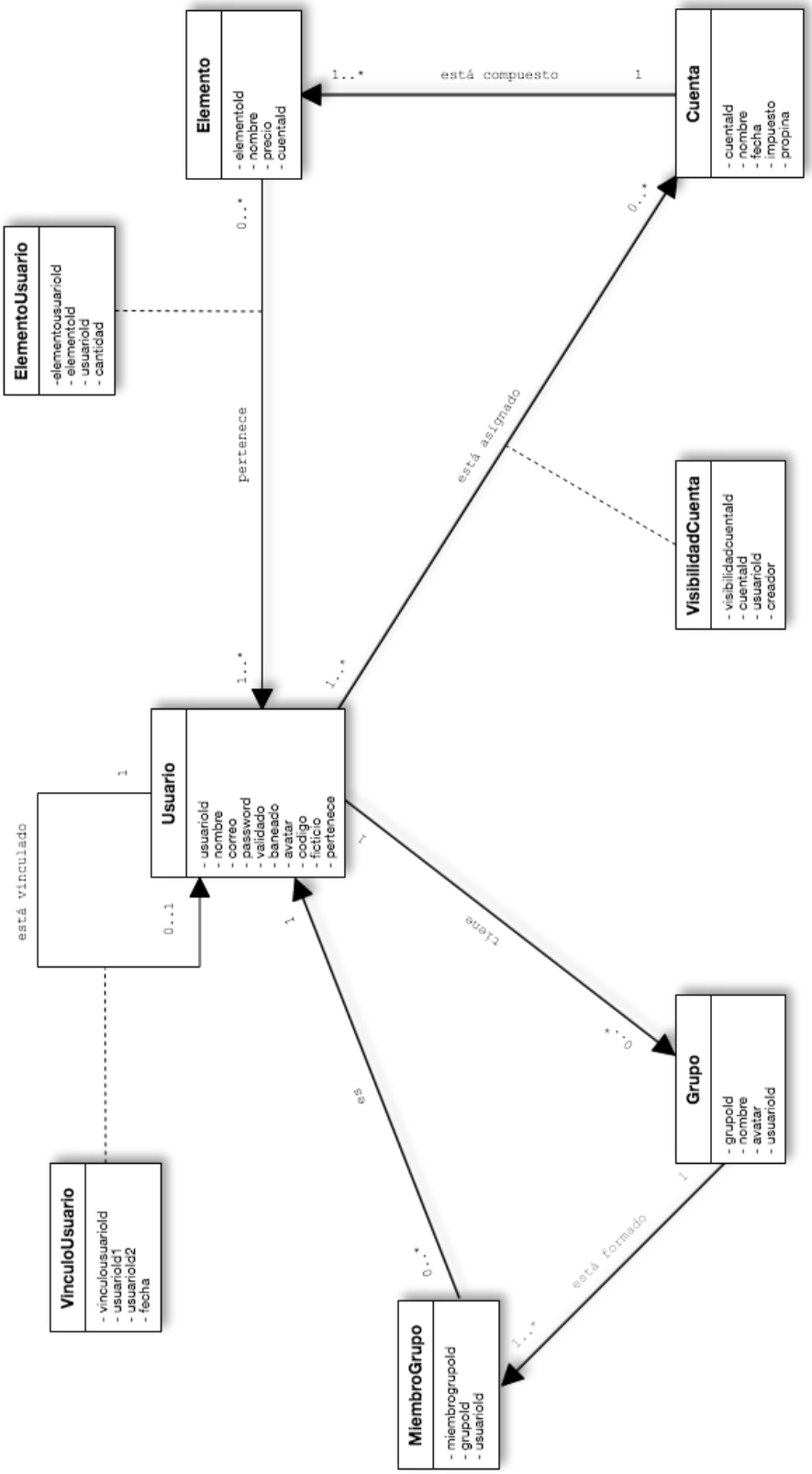


Diagrama de clases conceptuales

3. Fase de implementación

Una vez definido tanto el diseño técnico como el diseño conceptual, así como definir las principales características de la aplicación, pasamos a la fase de implantación del producto.

El código tanto de la Aplicación como del Servidor se entregan junto a la memoria. Debido al tamaño limitado de subida del campus virtual, en la entrega de código no irá incluido los Pods generados para el funcionamiento del proyecto, pero bastara con ejecutar “pod install” para la creación de las librerías mediante los Pods indicados en el fichero Podfile.

En caso de que alguno de los pods al descargarlo contenga la versión de Swift a 5, habría que cambiarlo manualmente en el proyecto a la versión 4.2 de Swift que es la que engloba el resto del proyecto.

3.1. Tecnologías utilizadas

A continuación, se definirá la tecnología utilizada para el desarrollo tanto en el lado del servidor para construir la API que provee de datos a la aplicación como la propia aplicación.

3.1.1. Lenguajes de programación

Para la codificación de la aplicación se ha elegido Swift, ya que el lenguaje que se utiliza para implementar aplicaciones para iOS. Existen otros lenguajes como objetive-c pero irán en desuso y finalmente Swift será el lenguaje principal para el desarrollo de iOS.

La versión de Swift es la 4.2, ya que aunque ya se encuentra disponible la versión 5.0 de Swift, cuando se empezó a desarrollar acababa de salir, y aún las librerías de terceros que se utilizan en el desarrollo aún no habían sido adaptadas a esta versión del lenguaje.

En la parte del servidor, hemos utilizado PHP 7 a través del framework CodeIgniter. Además utilizamos HTML5, CSS y javascript. También se utilizan algunos frameworks como bootstrap para CSS y jQuery para javascript.

3.1.2. Librerías y productos de terceros

Con el fin de ampliar las funcionalidades del sistema, se necesitó el uso de algunas librerías externas así como productos de terceros que haremos uso de ellas a través de sus APIs.

Librerías utilizadas en la App a través del repositorio de CocoaPods:

- AlamoFire: Es una librería de redes HTTP escrita en Swift que proporciona una elegante interfaz sobre peticiones HTTP que simplifica de manera efectiva una serie de tareas comunes en las comunicaciones de redes.
- PINRemoteImage: Es una librería para administrar descargas, procesar y almacenar en caché imágenes, descargan las imágenes desde una URL de manera asíncrona para que el rendimiento de la App no se vea afectado por una mala conexión.

- IGRPhotoTweaks: Es una librería que permite expandir una interfaz para recortar fotos. Permite al usuario arrastrar, rotar, escalar y recortar la imagen.
- HorizontalDial: Es una librería que ofrece un componente para mostrar una rueda con el fin de utilizarlo en la librería IGRPhotoTweaks para rotar las imágenes.
- DropDown: Es una librería que provee de listas desplegables con estilo material design.

Además se utiliza Firebase para determinadas funcionalidades como Notificaciones, lectura OCR, analítica y reportes de fallos. Para ello contamos con las siguientes librerías:

- Firebase/Core: Es una librería core de Firebase la cual debemos añadir para hacer uso de las demás. Además incluye Analytics.
- Crashlytics y Fabric: Es una librería para reportar errores en la aplicación de manera que cuando la aplicación se cierra inesperadamente por un fallo, esta librería nos reporta el fallo para poder subsanarlo.
- Firebase/Messaging: Es una librería de Firebase para recibir notificaciones push.
- Firebase/MLVision y Firebase/MLVisionTextModel: Es una librería de Firebase dentro del SDK móvil ML Kit, que brinda la experiencia del aprendizaje automático. En nuestro caso la utilizamos para el reconocimiento de texto en imágenes para la lectura OCR de los tickets.

3.1.3. Sistema gestor de base de datos

El sistema de gestión de base de datos elegido ha sido MySQL, ya que es uno de los más utilizados debido a su rapidez y facilidad de uso, además es libre.

Además de ser una base de datos muy rápida, fácil de usar y segura, como gestor de base de datos es una aplicación que se encarga de manejar los datos almacenados en la base de datos de una forma fácil y cómoda.

El motor de guardado elegido ha sido InnoDB ya que ofrece una fiabilidad y consistencia muy superior a MyISAM, además de guardar físicamente los

registros en el orden de la clave primaria, mientras que MyISAM los guarda en el orden en que fueron añadidos.

3.1.4. Servidor web

Durante el proceso de creación del proyecto utilizaremos un servidor apache, puesto que además de ser libre, se integra perfectamente con PHP y tiene varios módulos para extender su funcionalidad.

Para ello haremos uso de XAMPP, que aplicación que contiene todas las tecnologías necesarias.

XAMPP es un servidor que consiste en la base de datos MySQL, el servidor Web Apache, y los intérpretes para lenguajes PHP y Perl.

Su nombre proviene del acrónimo X (para cualquier sistema operativo) Apache, MySQL, PHP, Perl.

Para la puesta en producción se ha buscado un hosting gratuito que provea las funcionalidades básicas para la demostración del proyecto. Tras estudiar varias opciones y descartar otras varías por no proporcionar funcionalidades o rendimiento necesario para la demostración, se decidió utilizar el hosting https://x10hosting.com/ que cumple con los requisitos mínimos.

Este hosting nos ofrece un cPanel para administrar archivos, usuarios ftp, correo, base de datos y dominios, y nos ofrece gratuitamente un subdominio y una cuenta de correo con el que trabajar.

Más adelante, en la implantación de la parte del servidor se mostrarán más detalles.

3.2. Análisis sobre lectura OCR de tickets

Para obtener a ciencia cierta la capacidad de lectura de los tickets y como los posibles productos que tenemos a disposición son capaces de obtener datos de él, es necesario realizar un análisis exhaustivo sobre este tema, ya que no solo hay que estudiar la multitud de formatos que nos podemos encontrar si no la capacidad de las librerías actuales para obtener dichos datos.

En primer lugar se hizo un estudio sobre los tickets, intentando obtener una muestra significativa de los tickets que dan los restaurantes y estudiando la manera en que muestran los datos.

Los datos principales o mínimos a obtener son, el concepto, la cantidad y el precio total. Hay veces que la cantidad se integra con el concepto, o viene en una columna delante del concepto o detrás. Ahí no reside el problema. El problema reside cuando la cantidad se integra en el concepto o hay una cuarta columna que te dice el precio unitario.

Como realmente no tenemos certeza de obtener el dato, la parte de validar la cantidad con la asignación a los amigos que han tomado eso no va a ser posible, por tanto, los datos que nos importan realmente son el concepto (que es un dato llamativo y fácil de obtener porque suele ser una cadena, y el precio total de ese concepto, que casi siempre irá en la ultima columna.

Hay caso que la última columna no es el importe, pero si no es la última en la anterior por lo que se hace una comprobación que si dicha columna corresponde con un importe (00,00) la tomamos como importe.

3.3. Implementación

A continuación comentamos la descripción de la implementación en el lado del servidor y en la lado de la aplicación:

3.3.1. Servidor

Como se describe en el apartado de tecnologías utilizadas la parte del servidor está implementada con el framework CodeIgniter, que utiliza el patrón de diseño Modelo-Vista-Controlador.

3.3.1.1. Estructura

En el lado del servidor solo muestra una landing page describiendo la App, y contiene una parte donde se documenta todo la API de la aplicación las cuales son llamadas desde la App a través de la librería Alamofire.

A continuación se muestra la estructura básica de un proyecto CodeIgniter donde se ven las carpetas en la que se ha implementado código para realizar el proyecto en el servidor.

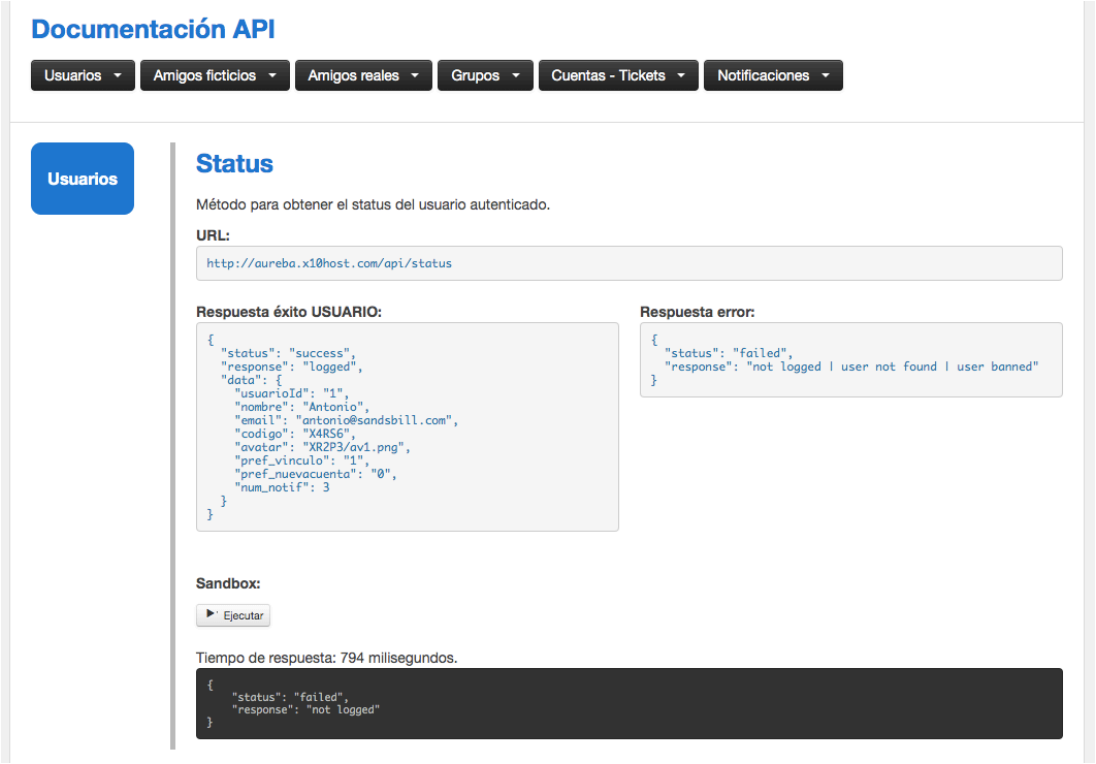
+ Application
    + controllers
    + libraries
    + models
    + views
+ Assets
    + css
    + fonts
    + images
    + js
    + uploads

3.3.1.2. Entornos de pruebas o Sandbox

Como se ha comentado en el apartado anterior, el servidor provee de una página de pruebas para la App con toda su documentación, nombre de las APIs, como llamarlas, que parámetros hay que enviarles, que respuesta de éxito y error podemos obtener, etc…

Además se pueden probar las llamadas, pintando el json resultante debajo de cada llamada.

A continuación, mostramos diferentes imágenes del sandbox:



Sandbox de pruebas y documentación de la API

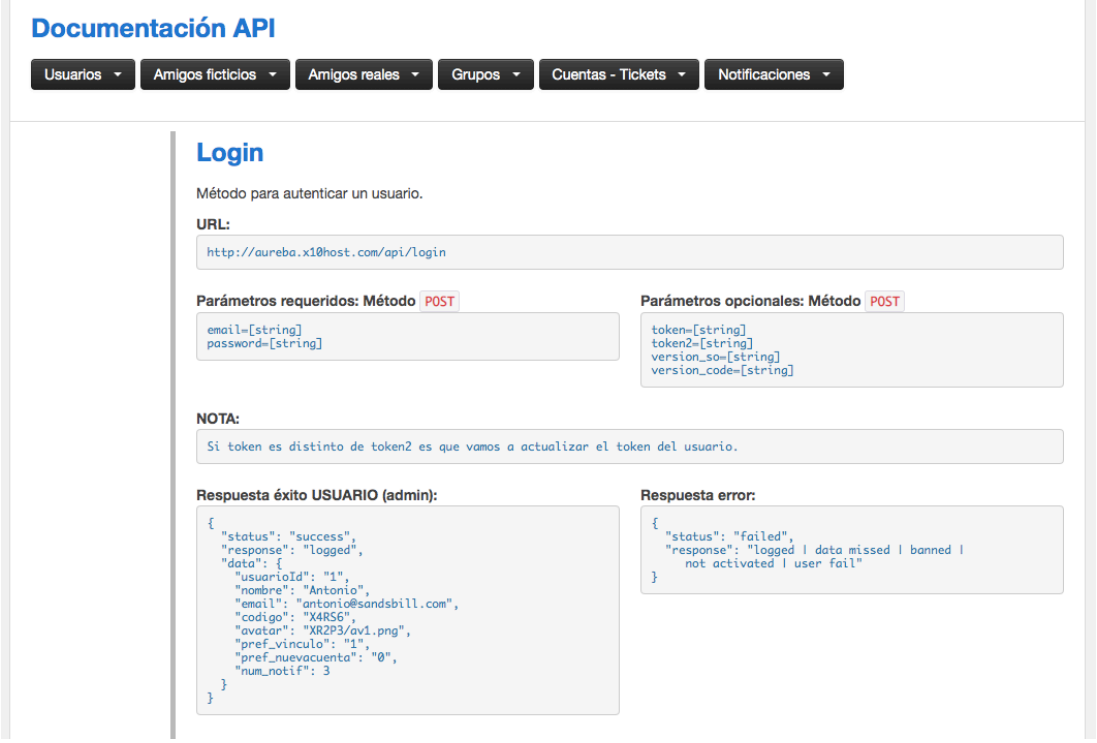
Como se muestra en la imagen, están definidas todas las llamadas que realiza la aplicación al servidor para que este provea de datos a la aplicación. Están agrupadas en 6 grupos definidas por funcionalidades.

Cada llamada puede recibir parámetros o no, y siempre obtiene una respuesta de éxito o de fracaso a través de la etiqueta “status”, y en caso de fracaso te dice el motivo a través de la etiqueta “response”.

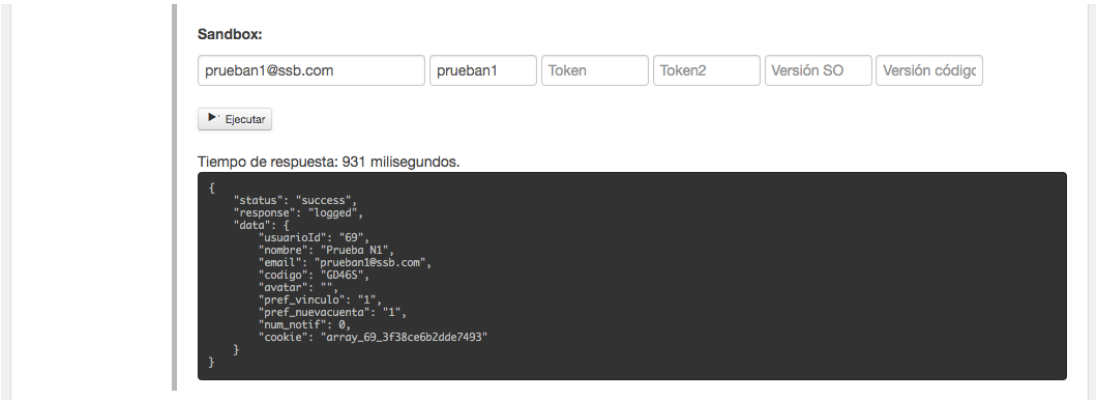
En el caso, en que además la llamada envíen datos lo hará a través de la etiqueta “data”.

Otro ejemplo más complejo y que envíe parámetros es la API de Login, que envía los parámetros correo, contraseña, los tokens para firebase, el sistema operativo, y la versión de la aplicación.

Como se ve en la imagen, los únicos datos obligatorios son el email y el password, y los demás son opciones.



Sandbox de pruebas - Login (Parte 1)



Sandbox de pruebas - Login (Parte 2)

Puede acceder al panel del Sandbox del proyecto desde la siguiente URL http://aureba.x10host.com/documentacion con la contraseña: doc.

3.3.2. App iOS

Para el desarrollo de la aplicación en iOS se ha implementado a través del IDE XCode 10.1.

La aplicación está disponible en dos idiomas, inglés y español, para versiones de iOS 11 o superior. La decisión de poner una versión del Sistema Operativo tan alta (la versión más actualizada es la 12), es porque todos los dispositivos que actualmente tienen soporte por parte de Apple puede actualizarse hasta iOS 12. Por tanto, no merece la pena bajar el soporte a versiones más bajas de iOS.

3.3.2.1. Estructura

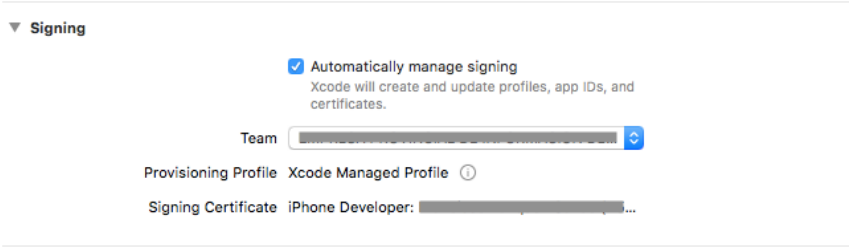
La aplicación no se va basa en un único StoryBoard, si no es archivos xib independientes por cada clase ViewController. Por una parte tenemos la carpeta configuración en la que se encuentra los archivos localizables para la internacionalización y la carpeta para xcassets que contiene las imágenes. La mayoría de las imágenes son vectoriales, pero otras contienen la imagen en diferentes formatos para las diferentes densidades de pantallas.

Por otro lado, tenemos la carpeta UI, que contienen las clases ViewController de cada pantalla junto a su xib. Se han estructurado por funcionalidades de la App para mantener concordancia con el funcionamiento de la App.

Además de los archivos por defecto de un proyecto XCode, podemos encontrar el proyecto de Pods, que se han generado a través de el repositorio de CocoaPods, y contiene las librerías que le hemos indicado en el fichero Podfile.

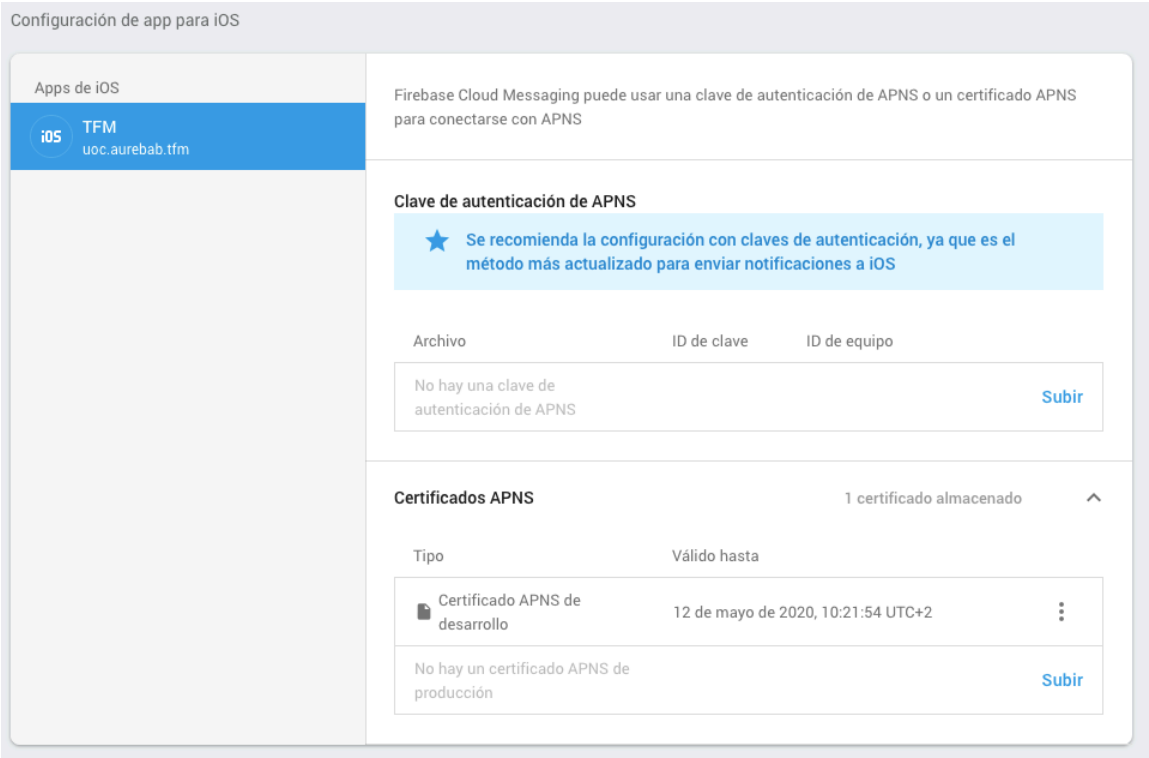
3.4. Certificados

Para el caso de pruebas en dispositivos reales y las pruebas de notificaciones push se ha utilizado una cuenta de desarrollador ajena al proyecto y se han exportado los certificados correspondientes para realizar las pruebas. El caso del provisioning profile de desarrollo ha sido el propio XCode el que lo ha creado y gestionado con la cuenta de desarrollador.



Automatically manage signing Provisioning Profile XCode

En cambio el certificado para las pruebas de notificaciones ha sido necesario exportar un fichero p12 creando el fichero pem desde la cuenta de desarrollador (https://developer.apple.com/), añadirlo al llavero del sistema operativo y generar posteriormente el certificado p12 exportándolo. Posteriormente, se ha incluido dicho fichero en Firebase para que pueda gestionar las notificaciones APNS de Apple.



Firebase Certificado APNS de desarrollo

3.5. Pruebas

3.5.1. Plan de pruebas

Para comprobar el correcto funcionamiento tanto de la aplicación como de la API se realizó un plan de pruebas para confirmar la calidad del software. El plan de pruebas consiste en:

Pruebas sobre los datos: Se realizó pruebas sobre los objetos que interactúan con la base de datos, de manera que al introducir unos datos en el sistema, se espere la salida correcta.

Pruebas sobre la interfaz: Se realizó pruebas sobre la manera de introducir los datos en el sistema, es decir, que los campos de los formularios no provocarán errores.

Pruebas sobre los subsistemas: Se realizó pruebas a los subsistemas individualmente, para comprobar su funcionamiento.

Pruebas sobre los sistemas: Se realizó pruebas a los subsistemas, es decir, al conjunto de estos que interactúan entre sí, pasándose información sin provocar errores.

El plan de pruebas llevado a cabo se ha realizado en 2 espacios temporales:

Durante la implementación: En este periodo se realizan sobretodo pruebas de clases y subsistemas.

Una vez finalizada la implementación: Una vez completa la aplicación y la API se realizan pruebas de todo tipo para comprobar que la aplicación funcione correctamente.

3.5.2. Especificación del diseño de pruebas

3.5.2.1. Pruebas durante la implementación

Las pruebas durante la implementación de la aplicación son esenciales, ya que así se evita arrastrar errores y corregirlos en el momento de su implementación. Se realizan pruebas sobre los datos de cada

modulo, es decir, el tratamiento de estos, la obtención y grabado en la base de datos, y la manera de ser interpretados por los subsistemas.

3.5.2.2. Pruebas una vez finalizada la implementación

Una vez finalizada la implementación había que comprobar que todo el trabajo realizado fuera correcto, y poder confirmar que hemos realizado una aplicación que posea una buena calidad. Para ello se empleó un proceso de pruebas que comprobara los más mínimos detalles.

Se empezó por comprobar el correcto funcionamiento de la aplicación ya fuera ejecutada desde cualquier dispositivo iOS, tanto en iPhone como en iPad para los sistemas operativos a los que la aplicación a soporte. Se prueba el correcto funcionamiento en los dispositivos elegidos y, aunque la visualización puede cambiar un poco por los diferentes tamaños de pantalla, el funcionamiento es correcto.

Luego se probó que toda la integración con la API fuera correcta, incluso simulando un red de datos lenta para ver como reaccionaba siendo los resultados buenos en redes de este tipo.

3.5.3. Especificación de los casos de prueba

Para la realización de estas pruebas la aplicación fue probada por dos grupos de personas:

3.5.3.1. Personas con bajos conocimientos sobre móviles

En este grupo las personas tienen conocimientos básicos sobre los dispositivos móviles, así como de las aplicaciones móviles. Se probó en un grupo reducido de personas que tienen móvil pero que no hacen un uso constante de sus aplicaciones, salvo algunas en concreto como WhatsApp. En este grupo no descubrió ningún error, pero sugirieron pequeños cambios para un mejor entendimiento de las opciones que ofrece la aplicación.

3.5.3.2. Personas con altos conocimientos sobre móviles

En este grupo se encuentran personas que son profesionales de la informática y usuarios que suelen utilizar un gran número de aplicaciones móviles. Ellos no descubrieron grandes errores, salvo algún que otro pequeño fallo que fue subsanado posteriormente. También sugirieron

algunas mejoras visuales y algunas opciones que se podían introducir para una mayor comprensión del sistema, como fue un CoachMark que aunque se encontraba en la hoja de ruta, no ha dado tiempo a implementarlo para la entrega.

4. Conclusiones

4.1. Aspectos generales

Este proyecto abarca todos los aspectos necesarios que se llevan a cabo en el desarrollo de una aplicación móvil. Ha sido un proyecto atractivo de realizar porque sus funcionalidades van ligadas a un mismo objetivo que es el dividir una cuenta entre amigos, por los que su recorrido durante todas las fases y sobre todo en la implementación hacen que cada porción de código que se escribe este más cerca de unir el puzzle que forma la aplicación.

De este proyecto me llevo aprendido dos grandes lecciones. Una, que es muy fácil plantear cosas que luego son complicadas de llevar a cabo, por lo que me hace reflexionar sobre la valoración de mi trabajo. Y otra, también como anécdota que es lo importante que es leer la documentación original de un producto, y no traducciones. Esto último tiene su explicación en la documentación de ML Kit de Firebase, donde su documentación en español no estaba actualizada y me dio varios quebraderos de cabeza hasta que vi la documentación en inglés y el software lo habían actualizado pero no la documentación en español.

4.2. Cumplimiento de la planificación y objetivos

He de decir que mi planificación inicial era bastante adaptada a la disponibilidad de tiempo que le iba a poder dedicar al proyecto debido a combinarla con la vida laboral y familiar, por tanto, he tenido que cumplir casi a raja tabla al menos la planificación de los días dedicados a cada tarea. Esto no quiere decir que las horas estimadas cada día fueran las que indiqué inicialmente, habiendo días que he tenido que incrementar las horas de trabajo para poder llegar a la planificación y con ello a la entrega.

Hay algunos aspectos de la implementación que no he podido implementar por falta de tiempo, como son los coach mark, que consistiría en un tutorial en las pantallas importantes de la App que expliquen al usuario como funciona la App.

También me hubiera gustado poder haber realizado test automatizados tanto a nivel de código como a nivel de interfaz.

4.3. Ampliaciones futuras

Las ampliaciones futuras serían los objetivos que no he podido cumplir en la fase de implementación y que detallamos en el punto anterior, y por supuesto implementar la aplicación en otras plataformas, principalmente en Android.

5. Glosario

- **RF**: Requerimiento o requisito funcional.
- **RNF**: Requerimiento o requisito no funcional.
- **OCR**: Optical Character Recognition o reconocimiento óptico de caracteres.
- **API**: Application Programming Interface o Interfaz de programación de aplicaciones.
- **UML**: Unified Modeling Language o Lenguaje Unificado de Modelado.
- **IDE**: Integrated Development Environment o Entorno de Desarrollo Integrado.
- **XCode**: IDE para macOS que contiene un conjunto de herramientas creadas por Apple destinadas al desarrollo de software para macOS, iOS, watchOS y tvOS.
- **Framework**: Conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

- **Swift**: Es un lenguaje de programación multiparadigma creado por Apple enfocado en el desarrollo de aplicaciones para iOS y macOS.
- **JSON**: JavaScript Object Notation o Notación de objeto de JavaScript. Es un formato de texto sencillo para el intercambio de datos.
- **SGBD**: Sistema gestor de base de datos.

6. Bibliografía y referencias

Libros y Artículos

- [Guilherme Siqueira Simões y Carlos Eduardo Vázquez, 2018]. Ingeniería de requisitos: Software orientado al negocio. Publicación independiente, 2018.
- [Roger S. Pressman, 2010]. Ingeniería del Software. Un enfoque práctico. 7ª Edición. McGraw-Hill, Mexico D.F, 2010.
- [Kendall Scott y Doug Rosenberg, 1999]. Use Case Driven Object Modeling with UML: A practical approach. 1ª Edición. Addison-Wesley, EEUU, 1999

Referencias web

- Metodología gestión de requerimientos. [Consulta: 04 de marzo de 2019].

https://sites.google.com/site/metodologiareq/home

- [1] Desarrollo en cascada. Wikipedia. [Consulta: 05 de marzo de 2019].

https://es.wikipedia.org/wiki/Desarrollo_en_cascada

- Pruebas de software. Wikipedia. [Consulta: 05 de marzo de 2019].

https://es.wikipedia.org/wiki/Pruebas_de_software

- Modelo-vista-controlador. Wikipedia. [Consulta: 30 de marzo de 2019].

https://es.wikipedia.org/wiki/Modelo-vista-controlador

Herramientas y software

- Desarrollo de la memoria. Pages para macOS.

  https://www.apple.com/es/pages/

- Diagrama de Gantt. TemGantt Online.

  https://www.teamgantt.com/

- Software de edición gráfica. Sketch, Adobe PhotoShop CS6 y Adobe Illustrator CS6.

  https://www.sketch.com/

  https://www.adobe.com/es/creativecloud.html

- Software de edición de diagramas. Cacoo.

  https://cacoo.com/

- Filezilla

  https://filezilla-project.org/

6. Anexos

Anexo 1: Diagrama de Gantt a mayor escala

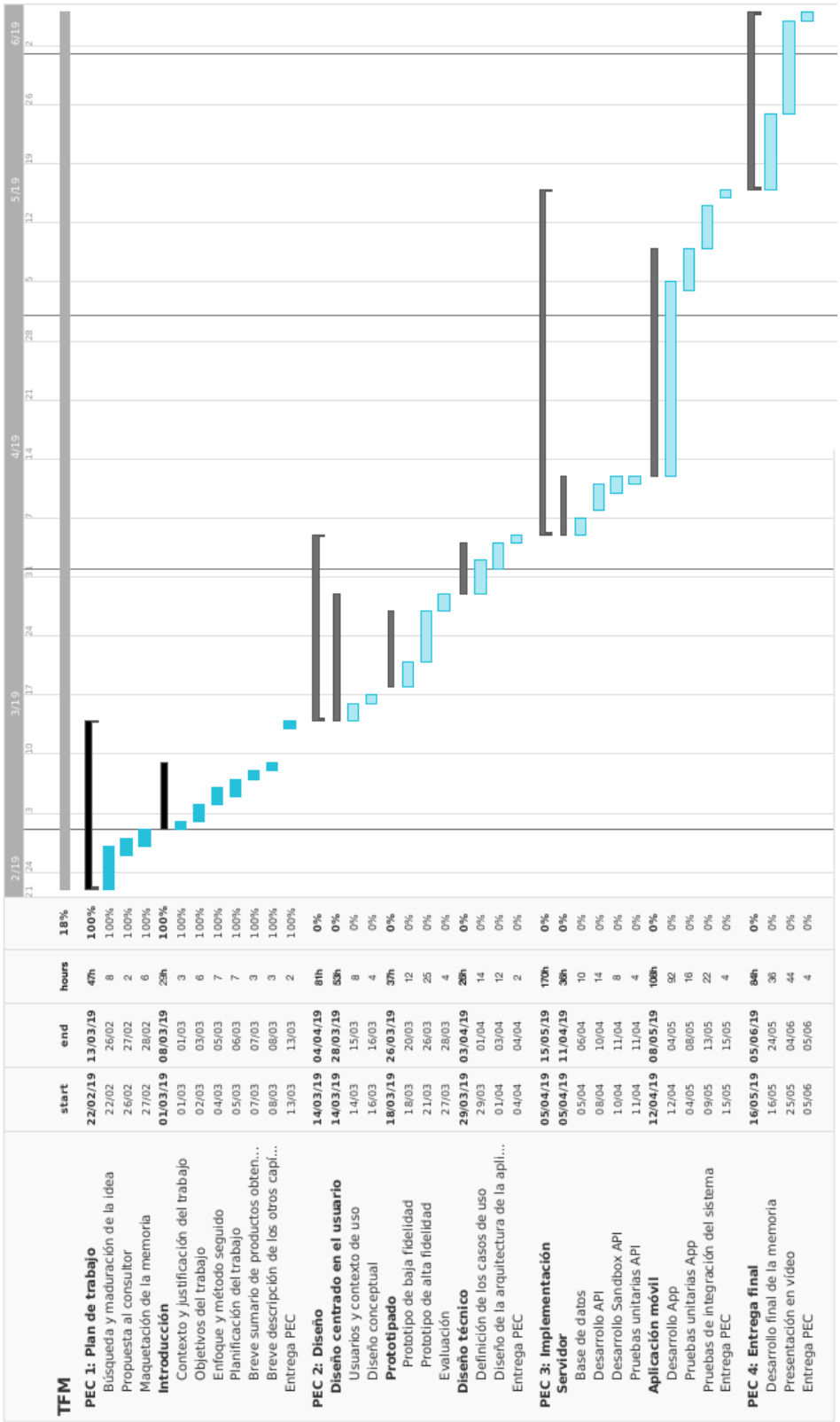


Diagrama de Gantt a mayor escala

Anexo 2: Licencias de recursos utilizados

- **Imágenes**:
  - Diagrama de modelo en cascada.

    https://es.wikipedia.org/wiki/Desarrollo_en_cascada#/media/File:El_modelo_de_desarrollo_en_cascada.svg

    Licencia: CC BY 3.0
  - Mockup iPhone para diseño del prototipo.

    https://www.freepik.com/free-vector/iphone-x-with-white-screen_2481092.htm

    Licencia: Licencia estándar freepik.
  - UI kits Apple Design Resources.

    https://developer.apple.com/design/resources/

    Licencia: Free for design.
  - Imagen grupo 1 de personas utilizadas en el diseño de la app.

    https://es.123rf.com/imagenes-de-archivo/personas.html?sti=moe2tlozc8ov89uigm|&mediapopup=45174925

    Licencia: Licencia estándar 123RF.
  - Imagen grupo 2 de personas utilizadas en el diseño de la app.

    https://es.123rf.com/imagenes-de-archivo/personas.html?alttext=1&start=100&sti=moe2tlozc8ov89uigm|&mediapopup=28507916

    Licencia: Licencia estándar 123RF.
  - Imagen hombre 1 de personas utilizadas en el diseño de la app.

    https://es.123rf.com/imagenes-de-archivo/personas.html?alttext=1&start=100&sti=moe2tlozc8ov89uigm|&mediapopup=36347666

    Licencia: Licencia estándar 123RF.
  - Imagen hombre 2 de personas utilizadas en el diseño de la app.

    https://es.123rf.com/imagenes-de-archivo/selfie.html?oriSearch=personas&sti=o3fi43qsvp2vqqh6ru|&mediapopup=30277328

Licencia: Licencia estándar 123RF.

- Imagen hombre 3 de personas utilizadas en el diseño de la app.

  https://es.123rf.com/imagenes-de-archivo/persona.html?oriSearch=selfie&sti=o8k0ar6ptxxwqx4q6r|&mediapopup=38665682

  Licencia: Licencia estándar 123RF.

- Imagen mujer 1 de personas utilizadas en el diseño de la app.

  https://es.123rf.com/imagenes-de-archivo/selfie.html?oriSearch=personas&sti=o3fi43qsvp2vqqh6ru|&mediapopup=40345149

  Licencia: Licencia estándar 123RF.

- Imagen mujer 2 de personas utilizadas en el diseño de la app.

  https://es.123rf.com/imagenes-de-archivo/persona.html?oriSearch=selfie&sti=o8k0ar6ptxxwqx4q6r|&mediapopup=45333973

  Licencia: Licencia estándar 123RF.

- Imagen fondo memoria, App y presentación.

  https://es.123rf.com/imagenes-de-archivo/persona.html?oriSearch=selfie&sti=o8k0ar6ptxxwqx4q6r|&mediapopup=45333973

  Licencia: Licencia estándar Freepik.

- Imagen fondo presentación.

  https://www.freepik.com/free-photo/creative-flat-lay-photo-workspace-desk_3520019.htm

  Licencia: Licencia estándar Freepik.

- Imagen Recurso prentación.

  https://www.freepik.com/free-vector/infographic-business-template-with-circular-elements_1269300.htm

  Licencia: Licencia estándar Freepik.

- Imagen fondo App.

  https://www.freepik.com/free-photo/closeup-italian-food-dinner_3277797.htm

  Licencia: Licencia estándar Freepik.

- Imagen fondo App.

https://www.freepik.com/free-photo/friends-having-beach-party-with-snacks_3741874.htm

Licencia: Licencia estándar Freepik.

- **Iconos**:
  - Icono diagrama Modelo-Vista-Controlador.

    https://www.flaticon.com/free-icon/programmer_1488581
  - Iconos fichas User-Persona

    https://www.flaticon.com/packs/young-avatar-collection

    Licencia: Free License (with attribution).
  - Icono Facebook Pantalla login App.

    https://www.flaticon.com/free-icon/facebook_145802

    Licencia: Free License (with attribution).
  - Icono Twitter Pantalla login App.

    https://www.flaticon.com/free-icon/twitter_145812

    Licencia: Free License (with attribution).
  - Icono Google Pantalla login App (Modificado).

    https://www.flaticon.com/free-icon/google-plus_145804

    Licencia: Free License (with attribution).
- **Fuentes**:
  - Avenir next.

    Fuente integrada en XCode. Su licencia depende del número de descargas de la App.

    Licencia: https://www.fonts.com/font/linotype/avenir-next/licenses
  - SS Gizmo.

    Fuente integrada en XCode. Su licencia depende del número de descargas de la App.

    Licencia: https://symbolset.com/license
- **Librerías**:
  - Librería Alamofire

Licencia: https://github.com/Alamofire/Alamofire

- Librería PINRemoteImage

  Licencia: https://github.com/pinterest/PINRemoteImage

- Librería IGRPhotoTweaks

  Licencia: https://github.com/IGRSoft/IGRPhotoTweaks

- Librería HorizontalDial

  Licencia: https://github.com/kciter/HorizontalDial

- Librería DropDown

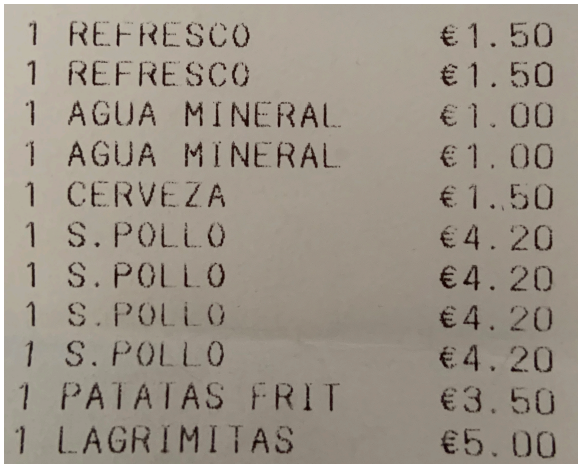
  Licencia: https://github.com/AssistoLab/DropDown

- Firebase y librerías asociadas

  Licencia: https://firebase.google.com/?hl=es-419

Anexo 3: Tickets reales para probar la App

A continuación se muestra 4 tickets para probar la App con diferentes formatos de ticket. El primer ticket es como el usuario a través de la pantalla de Crop debe dejar el ticket para un procesamiento de la imagen.



1	REFRESCO	€1.50
1	REFRESCO	€1.50
1	AGUA MINERAL	€1.00
1	AGUA MINERAL	€1.00
1	CERVEZA	€1.50
1	S.POLLO	€4.20
1	S.POLLO	€4.20
1	S.POLLO	€4.20
1	S.POLLO	€4.20
1	PATATAS FRIT	€3.50
1	LAGRIMITAS	€5.00

REST. PUERTA DE LA VICTORIAS.C
CALLE LA VICTORIA 7
11540 SANLUCAR DE BDA
NIF: J-72308562

Albarán: 1-1-1-21571
Fecha:01/03/2019 14:45
Mesa: Mesa 108

Uds	Descripción	Total
1	Z. PIÑA	1,40
1	POLLO EMPANADO	5,00
1	COCACOLA ZERO	1,40
2	PAN Y PICOS	0,80
1	T. CHOCOS	3,00
2	CLARA LIMαN	3,20
1	TARTAR DE SALMON	9,50
1	MILHOJA DE MANGO	10,00
1	QUESO PROVOLO AL HORNO	4,50
1	TAPA DE ARROZ	2,80
1	COCACOLA	1,40

F 43,00

GRACIAS POR SU VISITA
I.V.A. INCLUIDO

Bar Cafetería
El Recreo
HERMANOS "NEVA"

CRTA. TREBUJENA, 14 SANLUCAR DE BARRAMEDA
BAR CAFETERIA EL RECREO C.B.
NIF E-72224959
TELEFONOS 600 740 010 637 736 931

N. TICKET: TM COMP. FECHA: 24/02/2019
COMENSALES: MESA: 028

UNID.	DESCRIPCION	PRECIO	IMPORTE
2	CERVEZA 33 CL	1,00	2,00
2	KAS LIMON 20 CL	1,00	2,00
1	COCA COLA ZERO	1,00	1,00
2	PAPAS ARRIERAS	8,00	16,00
2	MED. LAGARTITO I	5,00	10,00
1	MED. CHOCOS FRIT	6,00	6,00
1	ROLLITO	2,50	2,50
1	T. CRUJIENTE	2,50	2,50
1	T. CHOCO PAN FRI	2,50	2,50
1	COLA TORO	3,00	3,00
1	coquina	8,00	8,00
3	TOCINO CIELO NA	2,50	7,50
1	CARAJILLO BAILE	1,50	1,50
1	RUAVIEJA COPA H	3,00	3,00
6	PAN PICOS	0,40	2,40

BASE	%IVA	IMP. IVA
63,55	10,00	6,35

TOTAL IMPORTE 69,90
PENDIENTE DE COBRO

IVA INCLUIDO 10,00%
GRACIAS POR SU VISITA

VENTA - R[illegible]RANTE
*** EL M[illegible]O ***
Ctra. Sanlúcar[illegible] Km. 4
11550 Chi[illegible]iz)

06/ene/2019 CAO [illegible]ra. B3/ 4821
Mesa 24 Cubiertos 1

Cant.	Plato		Importe
2,00	CERVEZA	1,50	3,00
1,00	Jarra Cerveza 1L		5,00
1,00	Cervezas SIN		1,50
1,00	COCA COLA		1,50
2,00	FANTA LIMON	1,50	3,00
1,00	ZUMO DE PIÑA		1,50
1,00	Sol. Ternera RETINTO		16,00
1,00	Fil. Pollo Empanado		8,00
1,00	parrillada carne		20,00
2,00	Chocos 1/2	8,00	16,00
1,00	CAFE CON LECHE		1,20
1,00	Tarta De Manzana		3,50
2,00	TARTA DE PIÑONE	3,50	7,00
1,00	tarta de la abuela		4,00
8,00	CUBIERTOS	0,75	6,00
2,00	Almejas Mariner	10,00	20,00

TOTAL 117,20

Gracias por su visita
Telf. 956 389406 - 956 389129
CIF: E11463536
I.V.A. Incluido

Anexo 4: Manual de instalación del servidor

Requisitos del servidor

- PHP 5.6
- Mysql 5.1

A continuación mostramos los pasos para configurar el servidor y la base de datos. En la carpeta servidor se encuentra las carpetas del proyecto Codeigniter que debemos alojar en el servidor. La carpeta raíz debe contener las carpetas:

- Application
- assets
- index.php
- system
- user_guide

Antes de pasar a la configuración debemos ejecutar el script SQL para crear la estructura de tablas de la base de datos. Este fichero se llama “database_inicial.sql” y puede ser importado desde PHPMyAdmin.

Una vez creada la base de datos pasamos a configurar el proyecto Codeigniter. Para ello debemos modificar los archivos:

- application/config/config.php:

  En este archivo debemos cambiar solo dos parámetros:

  - $base = ‘/home/aurebax1/public_html/'; donde se establece la ruta absoluta hasta la carpeta que contiene la carpeta raíz del proyecto codeigniter.
  - $base_url = 'http://aureba.x10host.com/'; donde se establece la url que accede a dicha carpeta.

- application/config/database.php:

  En este archivo debemos cambiar los parámetros de configuración de la base de datos:

  - hostname, username, password, database…

- application/config/email.php:

  En este archivo debemos cambiar los parámetros de configuración del correo que tenemos establecido como emisor de los correos para recuperación de contraseñas y validación de correo del usuario:

  - useragent, protocol, mailpath, smtp_host, smtp_user, smtp_pass, smtp_port…