



UNIVERSIDAD DE JAÉN

Escuela Politécnica Superior (Jaén)

Trabajo Fin de Grado

PROTOTIPO SOFTWARE DE SISTEMA DE GESTIÓN DE COMANDAS ON-LINE

Alumno: Sergio Muriana Cobo

Tutor: Prof. D. Pedro José Sánchez Sánchez

Dpto: Informática

Septiembre, 2015



Universidad de Jaén
Escuela Politécnica Superior de Jaén
Departamento de Informática

Don Pedro José Sánchez Sánchez, tutor del Proyecto Fin de Carrera titulado:
Prototipo software de sistema de gestión de comandas on-line, que presenta Sergio
Muriana Cobo, autoriza su presentación para defensa y evaluación en la Escuela
Politécnica Superior de Jaén.

Jaén, Junio de 2015

El alumno:

Los tutores:

Sergio Muriana Cobo

Pedro José Sánchez Sánchez

Agradecimientos

Me gustaría agradecer las facilidades que me ha dado Pedro José Sánchez Sánchez como tutor para el desarrollo de este proyecto.

Quiero agradecer a los compañeros de carrera que me han hecho disfrutar de esta etapa formándome junto a ellos en especial a David, Jose, Pancorbo, Juan, Imanol y Lolo con los cuales he pasado momentos duros de trabajo y estudio pero también grandes momentos de risas y fiestas, ya no os considero compañeros, sois amigos que tengo repartidos por la geografía Española.

No puedo olvidarme de agradecerle a mi amigo Carlos Morera toda la ayuda que me ha brindado desde el momento que decidí hacer la carrera hasta la misma realización de este proyecto, una persona que considero un referente como trabajador y como persona.

Durante la carrera he conocido a una persona que pasó de ser una extraña con acento de fuera a ser mi compañera de viaje. Gracias por estos 5 años a tu lado, por tu forma de ver la vida, con humor y trabajo duro, porque sin ti todavía estaría cursando asignaturas y porque has puesto la guinda del pastel en esta etapa. Gracias Teresa!

Aunque todo esto no tendría sentido sin mis pilares, los que me han ayudado a ser como soy y de los cuales sigo fijándome para ser mejor como persona, siempre han creído más en mi que yo mismo y siempre han apostado por mí, mis padres Jose Luis y Elvira y mi hermano Jose, de los cuales admiro profundamente y espero seguir compartiendo mis triunfos durante muchos años.

Gracias a mi tita Clementina, que más que una tita fue mi segunda abuela con la que pude compartir mis primeros años de carrera pero que por desgracia no he podido compartir mi graduación y finalización de estudios, sé que estas orgullosa de mí y que siempre nos acompañaras.

ÍNDICE DE ILUSTRACIONES	9
ÍNDICE DE TABLAS.....	11
1. INTRODUCCIÓN	13
1.1. INTRODUCCIÓN AL PROYECTO	13
1.2. MOTIVACIÓN	14
1.3. PROPÓSITOS Y OBJETIVOS	15
1.4. RESULTADOS ESPERADOS.....	16
1.5. PLANIFICACIÓN	16
1.5.1. Actividades.....	16
1.5.2. Esquema de división del trabajo	17
1.5.3. Planificación temporal	17
1.5.4. Cálculo de costes.....	20
2. TECNOLOGÍAS	24
2.1. SERVIDOR HTTP APACHE	24
2.2. BASES DE DATOS	24
2.2.1. MySQL	24
2.3. ANDROID	25
2.4. LENGUAJES DE PROGRAMACIÓN	26
2.4.1. PHP.....	26
2.4.2. HTML.....	26
2.4.3. CSS.....	27
2.4.4. JavaScript.....	27
2.4.5. Java.....	28
2.5. FRAMEWORK	28
2.5.1. Slim	28
2.6. LIBRERÍAS.....	29
2.6.1. Doctrine 2	29
2.6.2. HTTP Basic Authentication for Slim Framework	29
2.6.3. Password compat.....	29
2.6.4. PHP-JWT	29
2.6.5. ActiveAndroid	29
2.6.6. ZXing ("Zebra Crossing")	30
2.6.7. Retrofit.....	30
2.6.8. OkHttp	30
2.6.9. Dagger.....	30
2.6.10. Otto.....	31
2.6.11. Butterknife	31
2.6.12. Appcompat v7	31
2.6.13. Support v4	31
2.6.14. Support v13.....	31
2.6.15. Timber	31
2.6.16. Play Services.....	31
2.6.17. Picasso.....	32
2.6.18. Crouton.....	32
2.6.19. Android System Bar Tint	32
2.6.20. Magde	32
2.6.21. Scalpel.....	32

2.7. HERRAMIENTAS DE DESARROLLO.....	33
2.7.1. <i>Servidor</i>	33
2.7.2. <i>Cliente</i>	33
3. ANÁLISIS.....	34
3.1. DESCRIPCIÓN	34
3.2. ANÁLISIS DE REQUISITOS.....	34
3.3. REQUISITOS FUNCIONALES.....	35
3.4. REQUISITOS NO FUNCIONALES.....	37
3.5. PERFILES.....	39
3.5.1. <i>Empleado</i>	39
3.5.2. <i>Comensal</i>	39
3.6. CASOS DE USO	40
3.6.1. <i>Diagrama frontera del sistema</i>	40
3.6.2. <i>Diagramas de casos de uso</i>	41
3.6.3. <i>Descripción de los casos de uso</i>	44
4. DISEÑO.....	54
4.1. DESCRIPCIÓN	54
4.2. FASE DE ESTRUCTURA DEL SISTEMA	54
4.2.1. <i>Sistema</i>	54
4.2.2. <i>Servidor</i>	56
4.2.3. <i>Cliente Móvil</i>	60
4.3. DISEÑO DE DATOS.....	61
4.4. DISEÑO DE INTERFAZ	69
4.4.1. <i>Guía de estilo</i>	69
4.4.2. <i>Diseño de pantallas</i>	69
5. IMPLEMENTACIÓN.....	78
5.1. DESCRIPCIÓN	78
5.2. ARQUITECTURA DEL SISTEMA	78
5.2.1. <i>Arquitectura del cliente móvil</i>	79
5.3. FICHEROS.....	81
6. CONCLUSIONES	84
BIBLIOGRAFÍA	86
ANEXO I. MANUAL DE INSTALACIÓN DEL SERVICIO WEB.....	89
ANEXO II. MANUAL DE INSTALACIÓN DE LA APLICACIÓN ANDROID	97
ANEXO III. GUÍA DE USUARIO DE CLIENTE WEB	100
ANEXO IV. GUÍA DE USUARIO DE LA APLICACIÓN ANDROID	104

Índice de ilustraciones

Ilustración 1.1: Esquema de división del trabajo	17
Ilustración 1.2: Diagrama de Gantt	19
Ilustración 1.3: Logotipo Apache	24
Ilustración 1.4: Logotipo MySQL	24
Ilustración 1.5: Logotipo Android	25
Ilustración 1.6: Logotipo PHP	26
Ilustración 1.7: Logotipo HTML	26
Ilustración 1.8: Logotipo CSS3	27
Ilustración 1.9: Logotipo JavaScript	27
Ilustración 1.10: Logotipo Java	28
Ilustración 1.11: Logotipo Slim	28
Ilustración 1.12: Logotipo Doctrine	29
Ilustración 1.13: Logotipo ZXing	30
Ilustración 3.1: Diagrama frontera del sistema	40
Ilustración 3.2: Diagrama de caso de uso de Identificación	41
Ilustración 3.3: Diagrama de caso de uso de Ver Carta	41
Ilustración 3.4: Diagrama de caso de uso de Pedido	42
Ilustración 3.5: Diagrama de caso de uso de Preparar Pedido	42
Ilustración 3.6: Diagrama de caso de uso de Servir Pedido	43
Ilustración 4.1: Esquema Sistema de autenticación	54
Ilustración 4.2: Esquema Modelo Vista Controlador	56
Ilustración 4.3: Diagrama de clases del sistema	57
Ilustración 4.4: Diagrama de clases de resource	58
Ilustración 4.5: Diagrama de clases de Service	58
Ilustración 4.6: Diagrama de clases de Entity	59
Ilustración 4.7: Modelo Vista Presentador	60
Ilustración 4.8: Esquema conceptual	62
Ilustración 5.1: Esquema de la arquitectura general del sistema	78
Ilustración 5.2: Esquema de la arquitectura Android	80

Índice de tablas

Tabla 1: Planificación temporal	18
Tabla 2: Caso de uso de Registrarse	44
Tabla 3: Caso de uso de Identificarse	45
Tabla 4: Caso de uso de Escanear mesa	46
Tabla 5: Caso de uso de Ver carta	47
Tabla 6: Caso de uso de Añadir producto al pedido	48
Tabla 7: Caso de uso de Quitar producto del pedido	49
Tabla 8: Caso de uso de Realizar Pedido	50
Tabla 9: Caso de uso de Preparar Pedido	51
Tabla 10: Caso de uso de Servir Pedido	52
Tabla 11: Tabla de Cliente	63
Tabla 12: Tabla de Comanda	64
Tabla 13: Tabla de Notificación	64
Tabla 14: Tabla de ProductoComanda	65
Tabla 15: Tabla de Producto	65
Tabla 16: Tabla de CategoriaProducto	65
Tabla 17: Tabla de Categoría	66
Tabla 18: Tabla de Empleado	66
Tabla 19: Tabla de Mesa	67

1. INTRODUCCIÓN

1.1. Introducción al proyecto

En estos últimos años estamos presenciando la revolución digital, algo semejante a lo que ya ha ocurrido en otras épocas históricas como la revolución francesa o la revolución industrial: revoluciones económicas y tecnológicas que transforman por completo la sociedad. Esta revolución ha cambiado la manera en que la gente piensa, actúa, comunica, trabaja y gana su sustento. Ha forjado nuevas modalidades de crear conocimientos, educar a la población y transmitir información. Ha reestructurado la forma en que los países hacen negocios y rigen su economía, se gobiernan y comprometen políticamente. Ha proporcionado la entrega rápida de ayuda humanitaria y asistencia sanitaria, y una nueva visión de protección del medio ambiente. Y hasta ha creado nuevas formas de entretenimiento y ocio.

Un indicativo del momento en el que nos encontramos es que 4 de las 5 empresas más valiosas del mundo en el 2015 son de base tecnológica [1]. Las 2 más valiosas son las propietarias de los sistemas operativos móviles más utilizados a día de hoy.

Estas empresas son conscientes de la importancia de los dispositivos móviles y compiten por hacerse con parte del mercado, por ello invierten millones de dólares año tras año, intentando aunar el mayor número de usuarios posibles, saben que para captar usuarios, entre otras cosas, necesitan una gran variedad de aplicaciones de calidad y con actualizaciones periódicas. Por ello fomentan grandes comunidades de desarrolladores, aportan grandes cantidades de material didáctico para facilitar el aprendizaje, incluso desarrollan librerías y herramientas para hacer más fácil la vida de los desarrolladores.

Según José María Álvarez Pallette [2], consejero delegado de Telefónica “Nos encontramos ante un cambio profundo y el móvil es el elemento transformador que provoca una verdadera revolución en el comportamiento de los usuarios y su mentalidad”

La telefonía móvil en España, al igual que en su entorno europeo, es más avanzada que la del resto del mundo: La gran mayoría de los teléfonos que se venden son ya smartphones y entre los más jóvenes disponer de estos terminales se ha convertido en una necesidad, como indica el hecho de que el 86% de las personas entre 18 y 24 años posee un smartphone. El teléfono inteligente no es un dispositivo más, sino que permite incluso transformar el comportamiento de los usuarios.

Por otra parte nos encontramos sumidos en una crisis económica con una tasa de paro de la población del 23,78 %, llegando al 50,07 % la tasa de paro de la población menor de 25 años. En estos momentos se está incentivando el autoempleo como medida para paliar este drama.

Actualmente la economía española está sustentada en gran parte por el turismo, representando el 10,3% del PIB y generando el 11% del empleo en el país.

El sector de la hostelería es otro de los pilares de la economía española representando el 7% del PIB.

1.2. Motivación

Una noche de verano, tras media hora de espera para conseguir una mesa libre en la terraza de un bar abarrotado de comensales, empezó a crecer una idea en mi cabeza. Observando que un solo camarero tenía que tomar nota, servir, recoger y cobrar a más de 20 mesas, era evidente que el servicio desde el primer momento no iba a ser bueno, efectivamente tras esperar media hora que se quedase una mesa libre, tuvimos que esperar otros 20 minutos hasta que conseguimos que se acercará a la mesa para pedir algo de beber, esa misma noche charlando con los amigos se me ocurrió que podría ser interesante ofrecer alguna herramienta que ayude a este tipo de establecimientos a optimizar los tiempos para poder dar un servicio de mayor calidad. En ese momento surgió la idea de hacer un Prototipo software de sistema de gestión de comandas on-line para el Trabajo de Fin de Grado. Aprovechando mi espíritu emprendedor, decidí complementar mi formación en la Universidad realizando la mención en “Informática Empresarial”, lo que me ha permitido analizar esta idea como un proyecto empresarial, obteniendo

una visión más general del proyecto mediante un plan de empresa y animándome a seguir con esta idea hacia delante.

1.3. Propósitos y objetivos

Tras estudiar mucho la idea, he llegado a la conclusión de que el flujo de trabajo ideal de la aplicación consistiría en códigos QR que identificarían cada mesa unívocamente, desde los cuales le diría a la aplicación móvil en que restaurante y mesa se encuentra sentado el usuario mediante el uso de la cámara de fotos y una conexión a internet. La aplicación mostrará el menú con diferentes categorías organizando las diferentes productos que puede pedir el usuario. Una vez que el usuario haya añadido todos los productos que desea pedir, realizará el pedido y será notificado al camarero el pedido que ha realizado la mesa del usuario.

El propósito de este trabajo de fin de grado es diseñar e implementar un sistema de gestión de comandas mediante dispositivos móviles. Este sistema se compone de dos partes, por un lado está el cliente, el cual desde su teléfono móvil puede consultar el menú del restaurante y hacer pedidos en tiempo real. Por otra parte se encuentra el establecimiento hostelero, el cual recibe desde una página web las comandas que realiza cada mesa.

Para conseguir este propósito debemos de cumplir una serie de objetivos:

1. Búsqueda y revisión bibliográfica.
2. Estudio de los servidores más interesantes.
3. Estudio de la plataforma de desarrollo de Android.
4. Estudio de las tecnologías necesarias.
5. Objetivos y funcionalidad del prototipo
6. Desarrollo del prototipo
7. Evaluación del prototipo
8. Redactar la memoria del Trabajo Fin de Grado

1.4. Resultados esperados

Tras la realización del Trabajo de Fin de Grado se espera conseguir los siguientes resultados:

- Aplicación prototipo que permita a los usuarios realizar comandas
- Aplicación web donde pueda consultar el hostelero los pedidos que se están realizando
- Documentación: memoria y manuales tanto de usuario como de desarrollador para su correcta utilización en posteriores actualizaciones

1.5. Planificación

Para conseguir los resultados esperados hay que realizar una planificación, identificando las actividades principales, realizando un esquema para dividir el trabajo, asignando previsiones de tiempos a cada tarea y organizándolas en una línea temporal para poder realizar la estimación más precisa posible de cuando estará el proyecto finalizado.

1.5.1. Actividades

A continuación vamos a listar todas las actividades que necesitamos para completar a lo largo de la realización de este proyecto:

1. Inicio.
2. Búsqueda bibliográfica.
3. Estudio de las plataformas de desarrollo móvil.
4. Estudio de la tecnología Android y sus plataformas de desarrollo.
5. Estudio de las tecnologías para la comunicación cliente/servidor.
6. Estudio de las librerías de desarrollo disponibles para Android.
7. Construir el diagrama de interacción.
8. Construir el diagrama de clases.
9. Codificar.
10. Redactar la memoria del Trabajo Fin de Grado.
11. Fin.

1.5.2. Esquema de división del trabajo

Las actividades identificadas para la realización del proyecto pueden dividirse en 4 grupos.

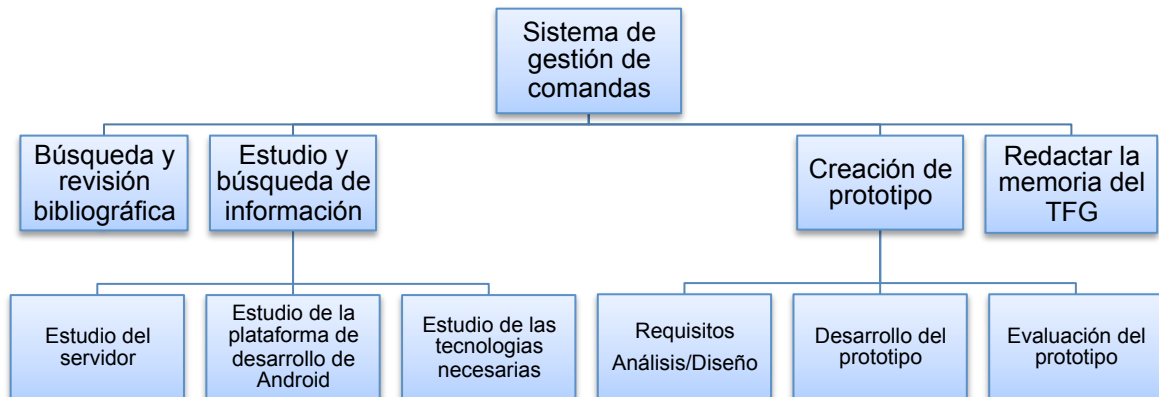


Ilustración 1.1: Esquema de división del trabajo

1.5.3. Planificación temporal

Una vez decididas las tareas a realizar tenemos que asignarles tiempos a las tareas, para ello vamos a desgranar las tareas para ser más precisos en la estimación. Cada día se dedicará 6 horas al proyecto.

Tarea	Duración	Inicio	Fin
Completar Memoria	85 días	2015-01-10	2015-05-05
Búsqueda bibliográfica	14 días	2015-01-10	2015-01-28
Estudio de la plataforma de desarrollo de Android	19 días	2015-01-30	2015-02-25
Estudio de las tecnologías necesarias	19 días	2015-01-30	2015-02-25
Estudio del servidor	19 días	2015-01-30	2015-02-25
Revisión bibliográfica	19 días	2015-01-30	2015-02-25
Análisis App Móvil	20 días	2015-02-26	2015-03-23
Análisis Servidor	20 días	2015-02-26	2015-03-23
Diseño App Móvil	14 días	2015-03-24	2015-04-12
Diseño Servidor	14 días	2015-03-24	2015-04-12
Desarrollo Prototipo	7 días	2015-04-13	2015-04-21
Desarrollo Servidor	7 días	2015-04-22	2015-05-01
Pruebas	2 días	2015-05-02	2015-05-04

Tabla 1: Planificación temporal

Una vez obtenidas todas las actividades, hemos de hacer una asignación temporal para saber que debemos de hacer en cada jornada de desarrollo del proyecto.

Un diagrama de Gantt representa las actividades que componen un proyecto junto con la duración de cada una de ellas. Cada nodo representa a una actividad y su longitud la duración de la misma.

Hemos estimado 85 días dedicando 6 horas diarias, sumando un total de 510 horas de dedicación al proyecto, un 70% más de lo requerido para el Trabajo de Fin de Grado debido al proceso búsqueda de información, estudio de las tecnologías necesarias y el tiempo necesario para el aprendizaje de la plataforma de desarrollo Android.

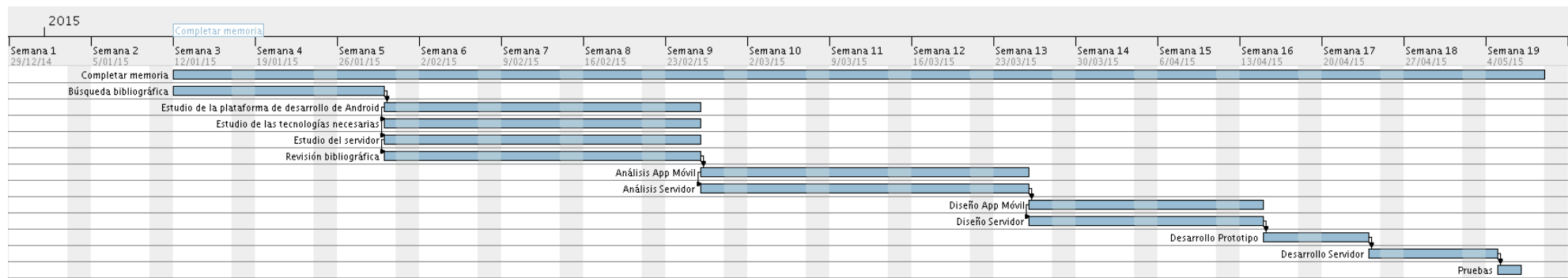


Ilustración 1.2: Diagrama de Gantt

1.5.4. *Cálculo de costes*

Una vez calculada las horas necesarias para llevar a cabo este proyecto podemos hacer una estimación de costes del desarrollo del proyecto, no vamos a calcular el coste de mantenimiento del proyecto ya que estamos desarrollando un prototipo y no un producto comercial. Tenemos que tener en cuenta los recursos humanos y los recursos hardware y software a utilizar para la realización del mismo.

Para el cálculo de costes que conlleva este proyecto se van a detallar los recursos necesarios para llevar a cabo su realización. La mayoría de recursos software no supondrán gasto alguno, gracias a las licencias ofrecidas por la Universidad de Jaén, aunque para ser lo más fiel posible a la realidad pondremos el coste de los mismos si no se contase con las licencias de la Universidad de Jaén. En el cómputo de costes asociados al hardware tenemos que contar con un ordenador Macbook Pro y un dispositivo móvil Samsung Galaxy S3. También necesitamos contar con un servidor pero no lo tenemos en cuenta como gasto porque para el desarrollo del proyecto ha sido facilitado de manera gratuita por parte de la Universidad de Jaén.

Software:

• PHPStorm	95 €
• Android Studio	0 €
• MySQL WorkBench	0 €
• xPlan	59 €
• Visual Paradigm Standar Single-seat license	270 €
• Pack Microsoft Office	119 €
• Licencia Android developer	30 €

Coste total software = 573 €

Hardware:

- Hosting (coste anual) 35,88 €

Este hosting presenta como principales características 25GB de espacio web, tráfico ilimitado, 10 cuentas de correo, un dominio gratis, una base de datos MySQL, soporte para el lenguaje de programación PHP y soporte técnico 24 horas, características suficiente para el proyecto.

$$\text{Coste por día} = \frac{\text{Coste por año}}{365} = \frac{35,88 \text{ €}}{365} = 0,09 \text{ €/día}$$

$$\text{Coste para el proyecto} = 0,09 \text{ €/día} * 85 \text{ días} = \underline{8,35 \text{ €}}$$

- Macbook Pro Pantalla de retina 13 1800€

El portátil cuenta con un procesador Intel Core i7 de 2,8 GHz, 16 GB 1600MHz DDR3 de memoria RAM y un disco duro SSD de 256 Gb, a parte del sistema operativo OS X Yosemite. Todas estas características nos permitirá desarrollar el proyecto de manera fluida aumentando la productividad.

$$\text{Coste hardware} = \frac{\text{Coste inicial}}{\text{Años amortización}} = \frac{1800 \text{ €}}{4 \text{ años}} = 450 \text{ €/año}$$

$$\text{Coste por día} = \frac{\text{Coste por año}}{365} = \frac{450 \text{ €}}{365} = 1,23 \text{ €/día}$$

$$\text{Coste para el proyecto} = 1,23 \text{ €/día} * 85 \text{ días} = \underline{104,79 \text{ €}}$$

- Samsung Galaxy S3 360€

El dispositivo móvil tiene una pantalla HD de 4.8 pulgadas, procesador Exynos 4 Quad de cuatro núcleos a 1.4GHz, 1GB de RAM, 16GB de memoria RAM y sistema operativo Andorid 4.0 Ice Cream Sandwich

$$\text{Coste hardware} = \frac{\text{Coste inicial}}{\text{Años amortización}} = \frac{360 \text{ €}}{2 \text{ años}} = 180 \text{ €/año}$$

$$\text{Coste por día} = \frac{\text{Coste por año}}{365} = \frac{180 \text{ €}}{365} = 0,49 \text{ €/día}$$

$$\text{Coste para el proyecto} = 0,49 \text{ €/día} * 85 \text{ días} = \underline{41,92 \text{ €}}$$

$$\text{Coste total hardware} = 155,06 \text{ €}$$

Recursos humanos:

Según el “XVI Convenio colectivo estatal de empresas de consultoría y estudios de mercados y de la opinión pública” del 2009 [3] dependiendo de las funciones realizadas variará el salario del personal, al ser el proyecto desarrollado por una sola persona se le asignará la clasificación profesional Analista-Programador, el cual según el convenio colectivo cobrará 1.468,54€ mensuales, siendo 14 pagas al año. Repartiendo el total que se cobraría al cabo de un año y dividiéndolo entre 365 días y 8 horas diarias nos da un coste de 7,04€ por hora.

- Sergio Muriana Cobo 7,04 € / hora

Total horas = 85 días * 6 horas = 510 horas

Coste total de personal = 510 horas * 7,04€ / horas = 3.590,88 €

Coste total del proyecto = 573 € + 155,06 € + 3.590,88 € = 4.319,34 €

Coste para una empresa:

Ya sabemos que coste tendría el proyecto sumando el software, hardware y las horas de trabajo necesarias, aunque este coste solo tendría sentido si se quedase en un trabajo académico, en el caso de que quisiéramos venderle el proyecto a una empresa habría que sumar más gastos como los impuestos y el mantenimiento.

Lo primero que tenemos que saber es el modelo de negocio que vamos a utilizar, en este caso creo que el modelo que mejor resultados puede dar es un modelo de pago por subscripción, es decir, el establecimiento que desee tener el servicio ofrecido en el proyecto pagará una cuota mensual donde se incluirá el mantenimiento, la amortización del proyecto y un porcentaje de beneficio.

Para calcular el precio de la cuota tenemos que estimar cuantos establecimientos tendremos adheridos en los próximos años, para ello vamos a centrarnos en el territorio andaluz, donde contamos con más de 50.000 establecimientos hosteleros [4], poniéndonos como objetivo llegar al 0,1% de los establecimientos estaríamos dándole servicio a 50 establecimientos.

Teniendo una estimación de establecimientos, ahora tenemos que calcular el coste del primer año, sumando los costes de desarrollo, servidores, conexión a internet, una línea de teléfono, mantenimiento y soporte.

Coste desarrollo proyecto = 4.319,34 €

Servidor (pago anual) = 35,88 €

Conexión a internet 50Mb + Teléfono móvil = 47 € * 12 meses = 564 €

Mantenimiento y soporte = 1.468,54 € * 14 pagas = 20559,56 €

Total = 25.478,78 € anuales

Para calcular los costes de internet y teléfono se ha tenido en cuenta el precio actual ofrecido por la empresa “Ono”, para calcular el mantenimiento y soporte se ha considerado que una persona a jornada completa es capaz de realizarlo y según convenio cobraría 1.468,54€ mensuales repartido en 14 pagas, la suma de todos los costes nos da un total de 25.478,78 €, sin impuestos. Para simplificar el proceso consideraremos que el único trabajador es menor de 30 años y no ha estado nunca dado de alta como autónomo, será la primera vez que se da de alta como autónomo, beneficiándose de la tarifa plana [5], teniendo un coste los primeros 6 meses de 55,78 € mensuales y los siguientes 6 meses 134,06 € mensuales.

Coste Autónomo durante un año = 55,78 * 6 meses + 134,07 * 6 meses = 1139,1 €

Coste total anual = 1139,1 € + 25.478,78 € = 26617,88 € anuales

Teniendo ya todos los costes durante un año podemos ver el coste que le supondría a los establecimientos mensualmente sin IVA.

Cuota mensual = 26617,88 € / 50 establecimientos / 12 meses = 44,36 €

Al vender un servicio hay que sumarle al precio el IVA del 21%.

$44,36 * 1,21 = 53,67\text{€}$ **Precio Final con IVA.**

2. TECNOLOGÍAS

Para poder realizar el desarrollo de la parte servidora, del cliente móvil y del cliente web se va a utilizar las siguientes tecnologías, los criterios de selección de las tecnologías serán que dispongan de una buena documentación, de una gran comunidad y siempre que se pueda que sean estándares dentro de su ámbito.

2.1. Servidor HTTP Apache



Ilustración 1.3: Logotipo Apache ¹

Se utilizará un servidor Apache para recibir, almacenar, procesar y enviar toda la información necesaria del sistema. Se ha elegido Apache porque es de código abierto, multiplataforma, modular, extensible y con una gran comunidad de usuarios [4]. Cabe destacar que los servidores Apache son los más utilizados a nivel mundial.

2.2. Bases de datos

2.2.1. MySQL



Ilustración 1.4: Logotipo MySQL ²

MySQL nos permitirá gestionar las bases de datos relacionales del sistema, [5] entre otras características cuenta con que es multihilo, multiusuario y extensamente usado.

¹ <http://www.muycomputer.com/wp-content/uploads/2013/08/Apache-HTTP-Server-630x450.jpg>

² <https://upload.wikimedia.org/wikipedia/en/thumb/6/62/MySQL.svg/1280px-MySQL.svg.png>

2.3. Android



Ilustración 1.5: Logotipo Android ³

Android es un sistema operativo basado en el kernel de Linux, diseñado principalmente para dispositivos móviles con pantalla táctil [6], aunque ya están trabajando en las versiones para automóviles y televisión. Es el sistema operativo para smartphones más utilizado en el mundo con más de mil millones de usuarios, lo que supone que el 80% de las personas con smartphone utilizan Android [7]. Cuenta con una de las mayores comunidades de desarrolladores dentro del sector de los smartphones, teniendo disponibles miles de librerías y tutoriales que facilitan el desarrollo de aplicaciones para este sistema operativo. Al ser Google el impulsor y propietario de Android se ha encargado de que todos los servicios que ofrece Google sean compatibles con el sistema operativo, enriqueciéndolo de fuentes de información como puede ser “Google Maps” para los mapas, “Analytics” para métricas de uso de la aplicación o “Google Places” para obtener detalles sobre los negocios y puntos de interés locales entre otros servicios que ofrece Google. Una de las claves para su éxito es que el sistema operativo es libre, por lo que cualquier persona o empresa puede usarlo, modificarlo y añadirlo a sus dispositivos, consiguiendo con esto que la mayoría de las empresas fabricantes de dispositivos móviles hayan adaptado Android a sus dispositivos y ofreciendo al usuario un gran abanico de dispositivos adaptándose a las necesidades y al poder adquisitivo de los usuarios.

³ <http://www.geekyblog.org/wp-content/uploads/2012/01/Android-Logo.png>

2.4. Lenguajes de programación

Existen cientos de lenguajes de programación, a la hora de elegir un lenguaje de programación es importante saber el objetivo que queremos conseguir, como entre nuestros objetivos está el desarrollo de una aplicación Android y el desarrollo de un servidor tendremos que elegir los lenguajes acorde con estas necesidades.

2.4.1. *PHP*



Ilustración 1.6: Logotipo PHP ⁴

PHP es el lenguaje que se utilizará en la parte del servidor, PHP es un acrónimo de “Hypertext Pre-processor” [7], se define como orientado a objetos, el código es interpretado por el servidor generando la página web resultante.

2.4.2. *HTML*



Ilustración 1.7: Logotipo HTML ⁵

HTML es uno de los lenguajes interpretados que se utilizará en la parte del cliente, con este lenguaje definiremos la estructura y el contenido de las páginas webs, HTML son las siglas de HyperText Markup Language (Lenguaje de marcas de hipertexto). Es un estándar a cargo de la W3C [8], organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación. Actualmente se recomienda usar la especificación de HTML5 ya que incluye nuevas características pensadas para ayudar a los creadores de aplicaciones Web.

⁴ <https://upload.wikimedia.org/wikipedia/commons/thumb/2/27/PHP-logo.svg/800px-PHP-logo.svg.png>

⁵ http://www.w3.org/html/logo/downloads/HTML5_Logo_256.png

2.4.3. CSS



Ilustración 1.8: Logotipo CSS3 ⁶

CSS es otro de los lenguajes interpretados que se utilizará en la parte del cliente para definir los aspectos visuales y formatos de las páginas webs, CSS son las siglas de Cascading Style Sheet (Hojas de Estilo en Cascada). Al igual que HTML, CSS es estándar de la W3C. Actualmente se recomienda utilizar la especificación de CSS3.

2.4.4. JavaScript



Ilustración 1.9: Logotipo JavaScript ⁷

JavaScript es el tercer y ultimo lenguaje que se usará interpretado en la parte del cliente, se define como orientado a objetos, basado en prototipos, débilmente tipado y dinámico [9]. Nos permite añadir dinamismo tanto al contenido como al aspecto de las páginas web. Normalmente es un lenguaje interpretado por parte del cliente aunque esta en auge la utilización en el lado del servidor.

⁶ <http://managementscience.biz/wp-content/uploads/2014/10/css3-png.png>

⁷ <http://www.hogoworks.com/images/4.jpg>

2.4.5. *Java*



Ilustración 1.10: Logotipo Java ⁸

Java es el lenguaje que se utilizará para desarrollar la aplicación móvil, es un lenguaje de programación orientado a objetos y multiplataforma entre otras muchas características [10]. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel como la manipulación directa de punteros o memoria.

Las aplicaciones Java están típicamente compiladas en un bytecode. En el tiempo de ejecución, el bytecode es normalmente interpretado o compilado a código nativo para la ejecución en la máquina virtual (JVM).

2.5. **Framework**

2.5.1. *Slim*



Ilustración 1.11: Logotipo Slim ⁹

Para crear el servicio API REST se ha optado por utilizar el framework Slim que nos proporciona la estructura básica para crear este tipo de servicios [11], sigue el patrón modelo-vista-controlador, es liviano, la sintaxis es clara y hace uso extensivo de todas las cualidades de PHP 5.3

⁸ http://kttv.images.worldnow.com/images/20568216_SA.jpg

⁹ http://kuikie.com/data/resources/images/sim-framework-for-php-logo_110.png

2.6. Librerías

2.6.1. *Doctrine 2*



Ilustración 1.12: Logotipo Doctrine ¹⁰

Doctrine es un proyecto que engloba varias librerías en lenguaje PHP, nosotros utilizaremos el paquete ORM que incluye las librerías Common y DBAL [12]. Este paquete contiene el mapeador objeto-relacional lo que nos permitirá hacer persistencia de objetos PHP a la base de datos de forma fácil e intuitiva.

2.6.2. *HTTP Basic Authentication for Slim Framework*

Es un middleware que implementa la autenticación http básica para el Framework Slim [13]. Esto nos permite comprobar si el usuario y contraseña son correctos comprobándolo en la base de datos.

2.6.3. *Password compat*

Esta librería tiene como objetivo proporcionar compatibilidad hacia delante con las funciones password_* que se incluyen en PHP 5.5 [14].

2.6.4. *PHP-JWT*

Esta librería sirve para codificar y decodificar Json Web Tokens (JWT) en PHP [15]. En el tema 4 se explica como funciona el sistema de Json Web Tokens.

2.6.5. *ActiveAndroid*

ActiveAndroid es un ORM rápido y sencillo de utilizar en Android, [16] permitiéndonos trabajar de forma más cómoda e intuitiva para el motor de base de datos SQLite.

¹⁰ <http://web.ontuts.com/wp-content/uploads/2010/07/doctrineorm.jpg>

2.6.6. ZXing ("Zebra Crossing")



Ilustración 1.13: Logotipo ZXing ¹¹

ZXing (que se pronuncia "zebra crossing") es una librería de procesamiento de códigos de barras en 1D y 2D, open source y multiformato implementada en Java [17]. Su objetivo es utilizar la cámara integrada de los teléfonos móviles para fotografiar y decodificar códigos de barras con el dispositivo. Actualmente soporta hasta 15 formatos diferentes incluyendo los códigos QR.

2.6.7. Retrofit

Retrofit es una librería que nos da muchas facilidades a la hora de trabajar con una API REST desde Android [18]. Nos convierte nuestra API REST en una interfaz Java, permitiendo conversión de objetos, llamadas asíncronas y muchas otras funcionalidades.

2.6.8. OkHttp

OkHttp es un cliente HTTP [19] que es eficiente en la agrupación de conexiones para reducir la latencia de solicitudes, GZIP transparente para el desarrollador y almacenamiento en caché de las respuestas para evitar repetición en las solicitudes entre otras características.

2.6.9. Dagger

Dagger es un inyector de dependencias eficiente para Android [20]. La inyección de dependencias es un patrón de diseño orientado a objetos, en el que se suministran objetos a una clase en lugar de ser la propia clase quien cree el objeto. Esto nos permite hacer un código modular y reutilizable.

¹¹ <https://avatars0.githubusercontent.com/u/1122572?v=3&s=400>

2.6.10. *Otto*

Otto es un bus de eventos diseñado para separar diferentes partes de la aplicación al mismo tiempo que les permite comunicarse de manera eficiente [21]. Descendiente de Guayaba, Otto agrega funcionalidad única de un bus de eventos ya refinado, así como especializada a la plataforma Android.

2.6.11. *Butterknife*

Butterknife nos permite inyectar Vistas mediante Anotaciones [22] lo que nos permite gestionar las dependencias de nuestra aplicación Android.

2.6.12. *Appcompat v7*

Appcompat v7 nos facilita el uso del Action Bar en Android con el estilo de material design para versiones anteriores a Android 5.0 [23]. Esta librería es compatible con dispositivos que tenga la versión 2.1 o superior y necesita de la librería "Support v4".

2.6.13. *Support v4*

Support v4 nos permite que se use componentes de la aplicación en versiones que de forma nativa no lo permitía [23]. Esta librería solo es compatible con dispositivos que tenga la versión 1.6 o superior

2.6.14. *Support v13*

Support v13 esta diseñada para utilizarse en dispositivos con la versión 3.2 o superior [23] y añade soporte para utilizar clases de los Fragments.

2.6.15. *Timber*

Timber es un logger [24] con una pequeña API extensible que proporciona utilidad en la parte superior de la clase Log de Android.

2.6.16. *Play Services*

Play Services nos permite utilizar funcionalidades de las aplicaciones de Google [25] dentro de nuestra aplicación.

2.6.17. *Picasso*

Picasso es una librería para Android que nos ayuda a mejorar la experiencia del usuario cuando se trabaja con imágenes [26], esto lo hace mediante creación de hilos para descargar imágenes, las almacena en caché y nos da la posibilidad de almacenar las imágenes en disco, todo esto de una forma sencilla para el desarrollador.

2.6.18. *Crouton*

Crouton nos permite mostrar notificaciones personalizadas en Android [27] de una manera no intrusiva y visualmente agradable.

2.6.19. *Android System Bar Tint*

Android 4.4 (KitKat) introdujo un estilo de interfaz de usuario del sistema transparente para barras de estado y de navegación, “Android System Bar Tint” [28] aplica un fondo teñido a la interfaz de usuario del sistema Android.

2.6.20. *Magde*

Magde es una herramienta de depuración [29] para determinar si los mapas de bits se están creando en su resolución nativa.

2.6.21. *Scalpel*

Scalpel es una herramienta de depuración [30] para visualizar las diferentes capas que hay debajo de la aplicación Android.

2.7. Herramientas de desarrollo

Se han decidido utilizar varias herramientas de desarrollo ya que al trabajar con diferentes lenguajes y tecnologías había herramientas que se ajustaban mejor para unas partes de desarrollo que para otras.

2.7.1. Servidor

Para el diseño y la creación de la base de datos se ha elegido utilizar MySQL WorkBench.

Al trabajar sobre el lenguaje PHP se ha decidido utilizar PHPStorm 8.0.1 ya que esta herramienta tiene como base IntelliJ IDEA.

Para depurar las llamadas de la API se ha optado por el plugin “Advanced REST client” disponible para el navegador Chrome.

2.7.2. Cliente

Para desarrollar sobre Android se ha optado por Android Studio, herramienta recomendada por Google, también esta basado sobre IntelliJ IDEA lo que permite trabajar indistintamente con los dos entornos de desarrollo sin problemas, ya que todos los atajos de teclado y el diseño de la interfaz son similares en ambos casos.

También se utilizará Gradle sobre Android Studio, esta herramienta nos permite construir el proyecto de forma automatizada permitiendo la automatización de dependencias.

3. ANÁLISIS

3.1. Descripción

El análisis o modelado de requerimientos es el primer paso para resolver un problema de ingeniería del software. Eso permite entender mejor el problema y proporciona una base para la solución (diseño). Los problemas complejos son difíciles de resolver por completo. Por esta razón, debe de usarse la estrategia de divide y vencerás. Un problema grande y complejo se divide en subproblemas hasta que cada uno de éstos sea relativamente fácil de entender.

3.2. Análisis de requisitos

El análisis de requisitos es una parte fundamental del desarrollo de un proyecto software, los mayores errores en el software se producen en esta etapa siendo también los errores más complicados y caros de reparar.

La especificación de requisitos de software brinda una descripción completa del comportamiento del sistema [31]. Consta de un conjunto de casos de uso que describe todas las interacciones que tendrán los usuarios con el software. Los casos de uso también son conocidos como requisitos funcionales. Además de los casos de uso, la especificación de requisitos de software también contiene requisitos no funcionales (o complementarios). Los requisitos no funcionales son requisitos que imponen restricciones en el diseño o la implementación.

3.3. Requisitos funcionales

Los requisitos funcionales definen las funcionalidades del sistema o sus componentes. A continuación se muestra el listado de funcionalidades identificadas para cumplir con los objetivos del proyecto:

- Registrarse como usuario
- Iniciar Sesión
- Consultar Pedidos
- Recibir Notificaciones
- Escanear QR de la mesa
- Consultar productos
- Llamar al camarero
- Realizar pedidos
- Pedir la cuenta

Una vez identificadas las funcionalidades necesarias es necesario discriminar que funcionalidades tendrá cada tipo de usuario del sistema, identificando como tipos de usuario al Empleado y al Comensal, siendo dos roles diferentes que pueden compartir funcionalidades pero que también tendrán funcionalidades exclusivas asociadas a su rol.

- Común a todos los usuarios

1. Registrarse como usuario

Es necesario que los usuario se registren en el sistema para poder identificar a que rol pertenecen y asociar las acciones realizadas a cada usuario.

2. Iniciar Sesión

Para poder utilizar las funcionalidades asociadas al rol del usuario es necesario que el sistema conozca que usuario.

- Rol Empleado

1. Consultar Pedidos

Los empleados tendrán acceso a los pedidos realizados por los clientes en tiempo real.

2. Recibir Notificaciones

Los empleados podrán recibir notificaciones por parte de los clientes para conocer si desean pagar la cuenta o simplemente necesita que vaya el camarero a la mesa.

- Rol Comensal

1. Escanear QR de la mesa

Los clientes podrán escanear el código QR asociado a la mesa desde sus dispositivos móviles para que el sistema pueda enviarles la información relacionada con el establecimiento.

2. Consultar productos

Los clientes podrán consultar la lista de productos disponibles en el establecimiento.

3. Llamar al camarero

Los clientes podrán llamar al camarero desde su dispositivo para que le atienda personalmente en la mesa.

4. Realizar pedidos

Los clientes podrán realizar pedidos de los productos disponibles en el establecimiento,

5. Pedir la cuenta

Los clientes podrán solicitar la cuenta para realizar el pago y cerrar la cuenta.

3.4. Requisitos no funcionales

Al utilizar una arquitectura Cliente/Servidor hay que diferenciar entre los equipos informáticos del servidor y los equipos informáticos del cliente

Requisitos no funcionales del servidor

- **Requisitos de Interfaz externa**

El sistema debe de utilizar una arquitectura API REST, este tipo de arquitectura permite que cualquier tipo de dispositivo pueda consultar al sistema.

- **Requisitos de Rendimiento**

El sistema debe de responder a las peticiones en el menor tiempo posible.

- **Requisitos de Seguridad**

La funcionalidad y la información del sistema debe estar protegida de accesos no autorizados.

- **Requisitos de Confiabilidad**

El sistema debe de ser tolerante a fallos y las operaciones deben de ser transaccionales.

- **Requisitos de Software**

El sistema debe ser multiplataforma, será desarrollado sobre el sistema operativo OS X Yosemite y se desplegará sobre Ubuntu 12.04.

- **Requisitos de Hardware**

- Capacidad de procesamiento:

El sistema debe de tener capacidad para atender a todas las peticiones que se reciban por parte de los usuarios del sistema de manera simultanea.

- Memoria principal:

El sistema necesita un mínimo de 2GB para su correcto funcionamiento, esta necesidad puede ir creciendo si se va incrementando el número de usuarios y establecimientos.

- Almacenamiento:

No necesita una gran capacidad, con 20GB es suficiente para que el sistema funcione de forma correcta.

- Conexión a internet:

Con una conexión de 10MB simétricos el sistema funcionará correctamente.

Requisitos no funcionales del cliente móvil

- **Requisitos de Software**

Debe de ser dispositivos sobre el sistema operativo Android con versión superior a 2.2

- **Requisitos de Seguridad**

La funcionalidad y la información debe estar protegida de accesos no autorizados.

- **Requisitos de Hardware**

Debe de disponer de cámara trasera y conexión a internet.

- **Requisitos de Interfaz de usuario**

Debe de seguir la guía de estilo de Android y cumplir los principios de usabilidad y accesibilidad.

3.5. Perfiles

Los perfiles de usuario se elaboran para identificar las necesidades que tienen los usuarios a nivel de interfaz gráfica. Para una correcta definición de los perfiles de usuario debemos de identificar la experiencia en torno al hardware y software [32], la experiencia en aplicaciones del mismo dominio, la experiencia en el trabajo y la frecuencia del uso del software y el grado de rotación del personal

3.5.1. Empleado

Experiencia en torno al hardware y software: Sabe utilizar smartphones y aplicaciones TPVs

Experiencia en aplicaciones del mismo dominio: Está acostumbrado a utilizar TPVs para añadir pedidos y cobrar.

Experiencia en el trabajo: Tiene experiencia de 2 años en su sector.

Frecuencia de uso del software y grado de rotación del personal: Es una herramienta esencial en su trabajo, la utiliza durante toda su jornada de trabajo, el grado de rotación es alto.

3.5.2. Comensal

Experiencia en torno al hardware y software: Tiene conocimientos en el uso de smartphone

Experiencia en aplicaciones del mismo dominio: Tiene conocimientos en compras por internet, es aficionado a mirar productos de su interés por internet.

Frecuencia de uso del software: Utiliza la aplicación cada vez que sale a comer a su restaurante favorito.

3.6. Casos de uso

Los casos de uso son una técnica para especificar el comportamiento de un sistema. Un caso de uso es un método para expresar como alguien o algo externo a un sistema lo utiliza.

3.6.1. Diagrama frontera del sistema

Antes de especificar cada casos de uso es conveniente mostrar un esquema que represente la funcionalidad general y completa del sistema y el conjunto de actores partícipes en el sistema.

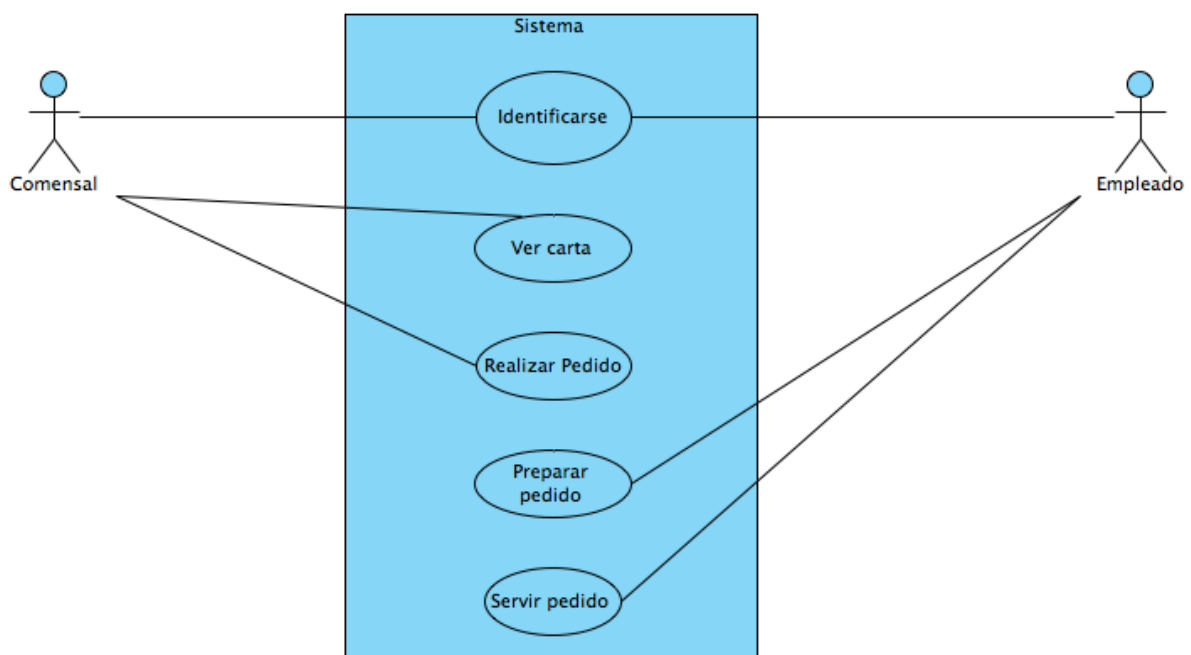


Ilustración 3.1: Diagrama frontera del sistema

3.6.2. Diagramas de casos de uso

- Identificación

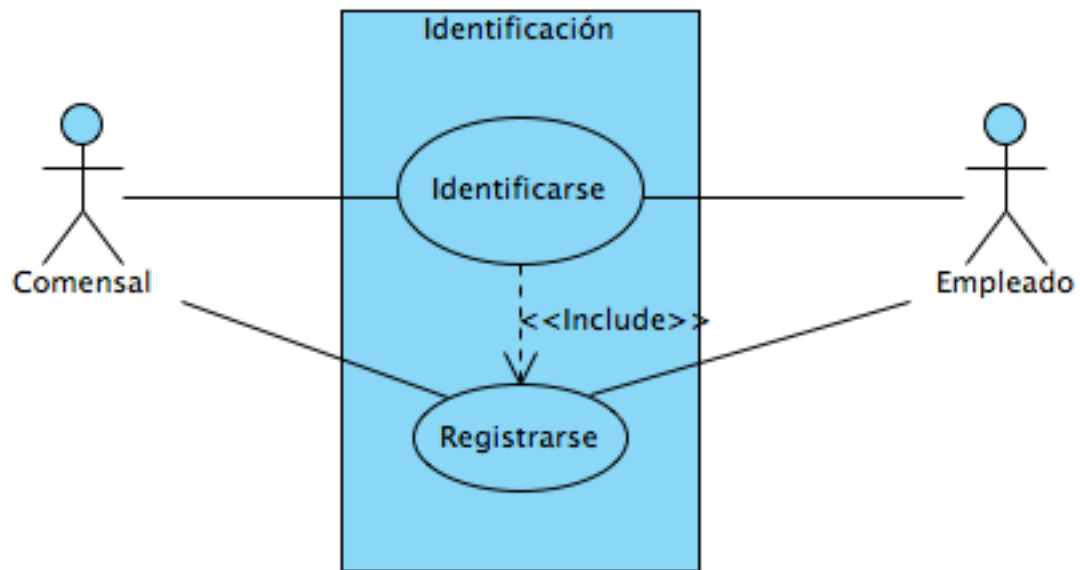


Ilustración 3.2: Diagrama de caso de uso de Identificación

- Ver Carta

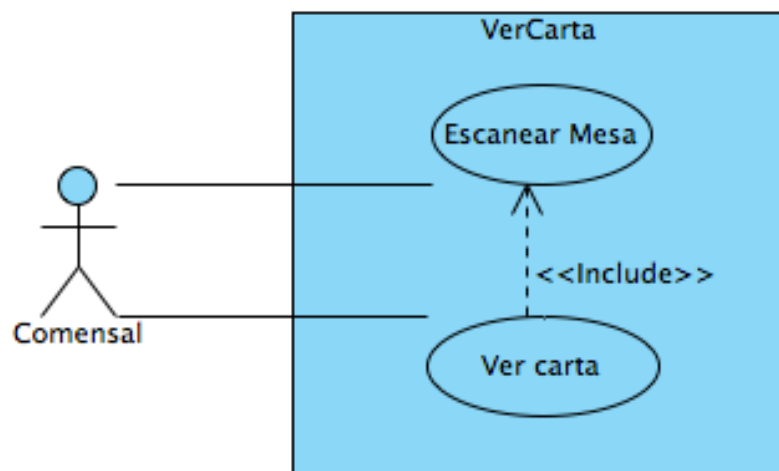


Ilustración 3.3: Diagrama de caso de uso de Ver Carta

- Realizar Pedido

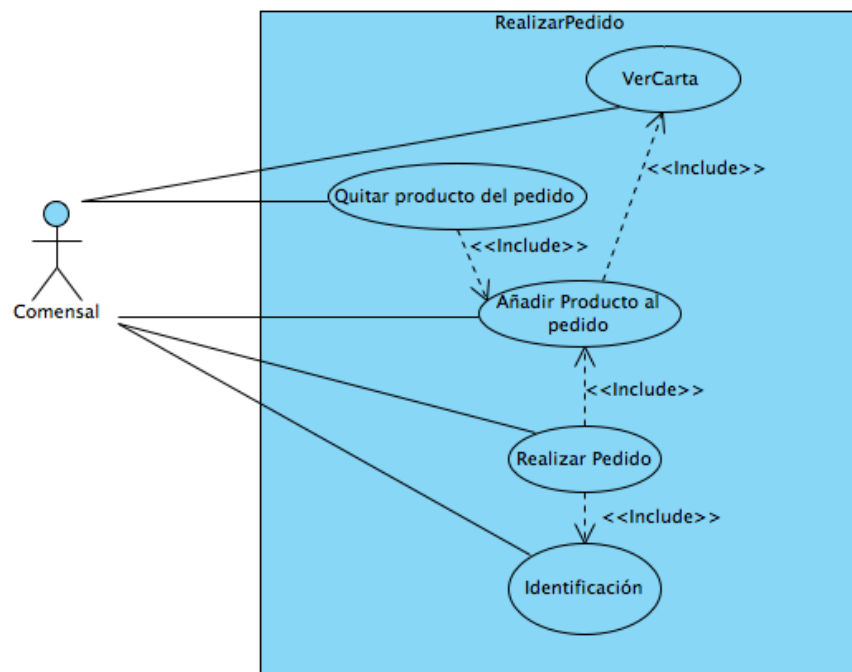


Ilustración 3.4: Diagrama de caso de uso de Pedido

- Preparar Pedido

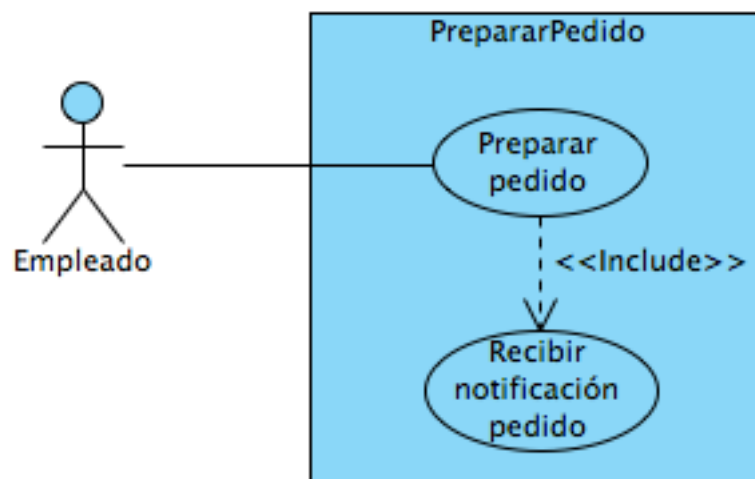


Ilustración 3.5: Diagrama de caso de uso de Preparar Pedido

- Servir Pedido

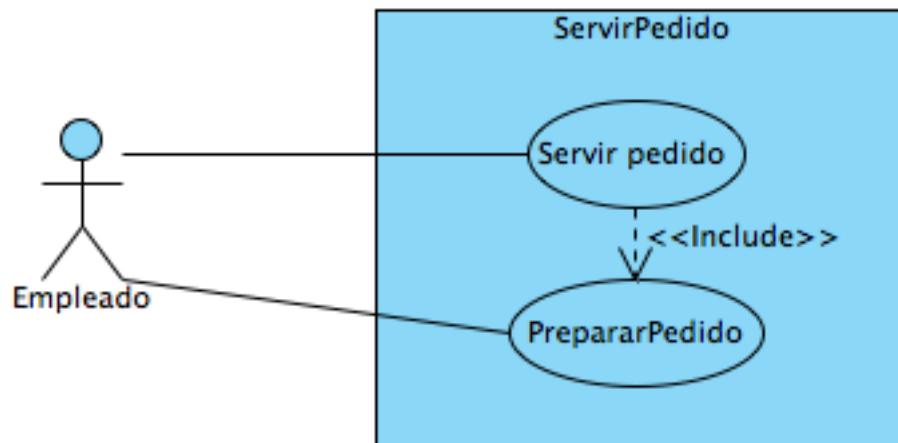


Ilustración 3.6: Diagrama de caso de uso de Servir Pedido

3.6.3. Descripción de los casos de uso

CASO DE USO	Registrarse	
Resumen	El usuario se registra en el sistema y puede acceder a las funcionalidades	
Actores	Comensal y empleado	
Precondiciones	El dispositivo móvil debe de disponer de conexión a Internet	
Escenario principal	Paso	Acción
	1	El usuario accede a la pantalla de registro
	2	El sistema muestra un formulario con los campos necesarios para el registro
	3	El usuario rellena los campos del formulario
	4	El usuario realiza el registro de la aplicación
	5	El sistema guarda los datos de registro y le muestra la pantalla principal
Postcondiciones	Los datos del usuario quedan guardados en el sistema	
Excepciones	Paso	Acción
	1	El usuario cancela el registro pulsando el botón del atrás del dispositivo
	2	El usuario no rellena un campo necesario
	2.1	El sistema muestra un mensaje de error
	3	El usuario utiliza un email ya registrado
	3.1	El sistema muestra un mensaje de error

Tabla 2: Caso de uso de Registrarse

CASO DE USO	Identificarse	
Resumen	El usuario se identifica en el sistema	
Actores	Comensal y empleado	
Precondiciones	El dispositivo móvil debe de disponer de conexión a Internet. El usuario debe de estar registrado en el sistema	
Escenario principal	Paso	Acción
	1	El usuario accede a la pantalla de identificación
	2	El sistema le muestra un formulario con los campos email y contraseña
	3	El usuario rellena el formulario
	4	El usuario inicia la sesión en la aplicación
	5	El sistema comprueba que los datos son correctos
	6	El sistema envía al usuario a la pantalla principal
Postcondiciones	El usuario accede a funcionalidades extras	
Excepciones	Paso	Acción
	1	El usuario cancela la identificación pulsando el botón del atrás del dispositivo
	2	El usuario no rellena alguno de los campos
	2.1	El sistema muestra un mensaje de error
	3	El usuario no pone correctamente algunos de los campos
	3.1	El sistema muestra un mensaje de error

Tabla 3: Caso de uso de Identificarse

CASO DE USO	Escanear mesa	
Resumen	El usuario apunta con la cámara del móvil al código QR que se encuentra en la mesa	
Actores	Comensal	
Precondiciones	El dispositivo móvil debe de cámara de fotos. El usuario debe de estar identificado en el sistema.	
Escenario principal	Paso	Acción
	1	El comensal accede al escaneo de mesa
	2	El sistema activa la cámara y espera a detectar un código QR
	3	El sistema lee el código QR
	4	El sistema identifica el establecimiento y la mesa
	5	El sistema muestra la carta del establecimiento
Postcondiciones	El comensal consulta la carta del establecimiento	
Excepciones	Paso	Acción
	1	El comensal no apunta con la cámara a un código QR
	2	El comensal apunta a un código QR invalido
	2.1	El sistema muestra un código de error

Tabla 4: Caso de uso de Escanear mesa

CASO DE USO	Ver carta	
Resumen	El comensal ve los productos del establecimiento	
Actores	Comensal	
Precondiciones	El comensal debe de haber escaneado el código QR de la mesa.	
Escenario principal	Paso	Acción
	1	El comensal selecciona una categoría
	2	El sistema muestra los productos de la categoría
	3	El comensal consulta los productos de la categoría.
Postcondiciones	El comensal añade productos al carrito	
Excepciones	Paso	Acción
	1	Los productos no están disponible
	1.1	El sistema muestra un mensaje de error

Tabla 5: Caso de uso de Ver carta

CASO DE USO	Añadir producto al pedido	
Resumen	El comensal añade los productos que desea comprar y se encuentran disponibles en la carta.	
Actores	Comensal	
Precondiciones	El usuario debe de haber escaneado el código QR de la mesa.	
Escenario principal	Paso	Acción
	1	El comensal añade un producto al pedido
	2	El sistema añade una unidad del producto al carrito
Postcondiciones	El comensal realiza el pedido	
Excepciones	Paso	Acción

Tabla 6: Caso de uso de Añadir producto al pedido

CASO DE USO	Quitar producto del pedido	
Resumen	El comensal elimina uno de los productos que tiene en el carrito antes de realizar el pedido	
Actores	Comensal	
Precondiciones	El comensal ha añadido el producto al pedido	
Escenario principal	Paso	Acción
	1	El comensal accede a su pedido
	2	El sistema muestra los productos añadidos al carrito
	3	El comensal decrementa una unidad de un producto
	4	El sistema decrementa en una unidad el producto
Postcondiciones	El comensal realiza un pedido	
Excepciones	Paso	Acción
	1	

Tabla 7: Caso de uso de Quitar producto del pedido

CASO DE USO	Realizar Pedido	
Resumen	El comensal realiza un pedido de productos	
Actores	Comensal	
Precondiciones	El comensal tiene que haber añadido productos al pedido	
Escenario principal	Paso	Acción
	1	El comensal realiza el pedido
	2	El sistema notifica el pedido al establecimiento
	3	El sistema pasa el pedido realizado a la cuenta
Postcondiciones	El comensal pide la factura	
Excepciones	Paso	Acción
	1	No se envía el pedido al servidor
	1.1	El sistema muestra un mensaje de error

Tabla 8: Caso de uso de Realizar Pedido

CASO DE USO	Preparar Pedido	
Resumen	Preparar el pedido del comensal	
Actores	Empleado	
Precondiciones	El comensal tiene que haber realizado un pedido.	
Escenario principal	Paso	Acción
	1	El sistema notifica el pedido al empleado
	2	El empleado prepara el pedido
Postcondiciones	El empleado sirve la comida	
Excepciones	Paso	Acción
	1	

Tabla 9: Caso de uso de Preparar Pedido

CASO DE USO	Servir Pedido	
Resumen	El empleado sirve el pedido al comensal	
Actores	Empleado	
Precondiciones	El pedido tiene que estar preparado	
Escenario principal	Paso	Acción
	1	El empleado sirve el pedido
	2	El empleado pulsa el botón "servido" en el pedido
	3	El sistema elimina el producto de la pantalla del empleado
Postcondiciones		
Excepciones	Paso	Acción
	1	

Tabla 10: Caso de uso de Servir Pedido

4. DISEÑO

4.1. Descripción

El diseño representa características del software que ayuda a los profesionales a elaborar el software con eficacia: arquitectura, interfaz de usuario y detalle en el nivel de componente.

4.2. Fase de estructura del sistema

4.2.1. Sistema

Sistema de autenticación

Se utilizará un sistema de autenticación sin estado con tokens usando el estándar JSON Web Tokens, a partir de ahora lo llamaremos JWT.

El funcionamiento será el siguiente [33]. El usuario se autentica en nuestra aplicación (desde la web como camarero o desde la aplicación móvil como comensal) con un usuario y contraseña. A partir de este momento cada petición HTTP que haga el usuario irá acompañada de un Token. Este Token se almacena en el cliente y es una firma cifrada que permite a nuestro servidor identificar al usuario.

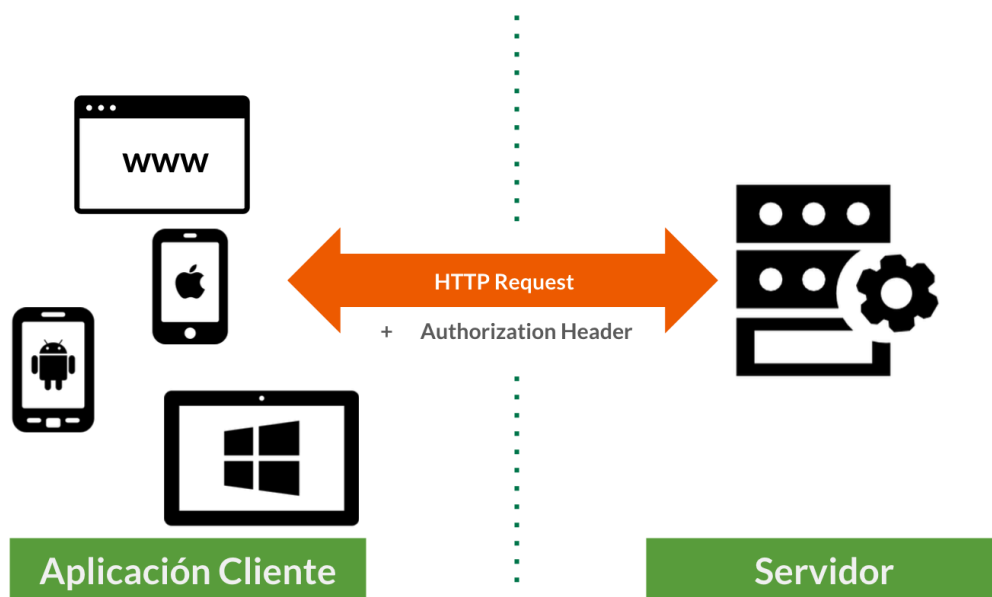


Ilustración 4.1: Esquema Sistema de autenticación ¹²

¹² <https://carlosazaustre.es/blog/content/images/2015/02/Screen-Shot-2015-02-18-at-00-43-59.png>

JWT utiliza el formato JSON y esta compuesto por 3 cadenas de texto separadas por puntos. El la primera cadena de texto es la cabecera del token que identifica el tipo y la codificación. La segunda cadena de texto contiene el identificador de usuario, la fecha de creación del token y la fecha de expiración del mismo. La tercera y ultima cadena de texto es la firma formada por las anteriores cadenas y cifradas en Base64 con una clave secreta, sirviendo de Hash para comprobar la integridad de los datos.

4.2.2. Servidor

Modelo vista controlador

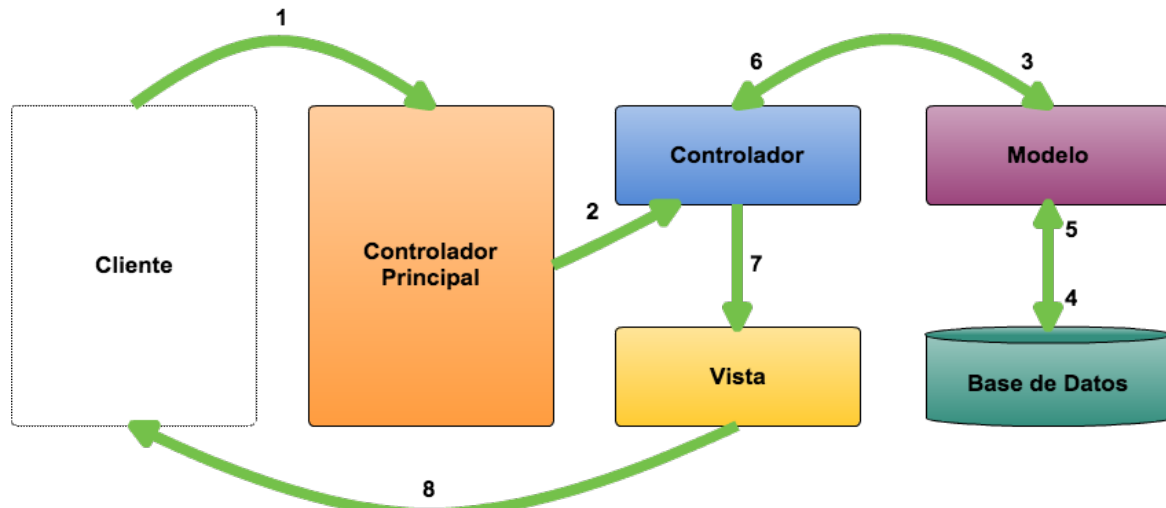


Ilustración 4.2: Esquema Modelo Vista Controlador

1. El controlador principal recibe una petición del cliente.
2. El controlador principal averigua la validez de los datos enviados y dirige la petición al controlador correspondiente.
3. El controlador correspondiente trata esta petición. Para ello, puede necesitar los datos de la base de datos y entonces tiene que referirse al modelo.
4. El modelo accede a la base de datos, para insertar, suprimir, actualizar o recuperar los datos.
5. El modelo recibe los datos de la base.
6. El modelo transmite al controlador el resultado del acceso a la base de datos.
7. Según el resultado recibido, el controlador escoge la vista a mostrar al cliente y le proporciona los elementos dinámicos.
8. El cliente recibe la vista escogida.

Diagrama UML de clases

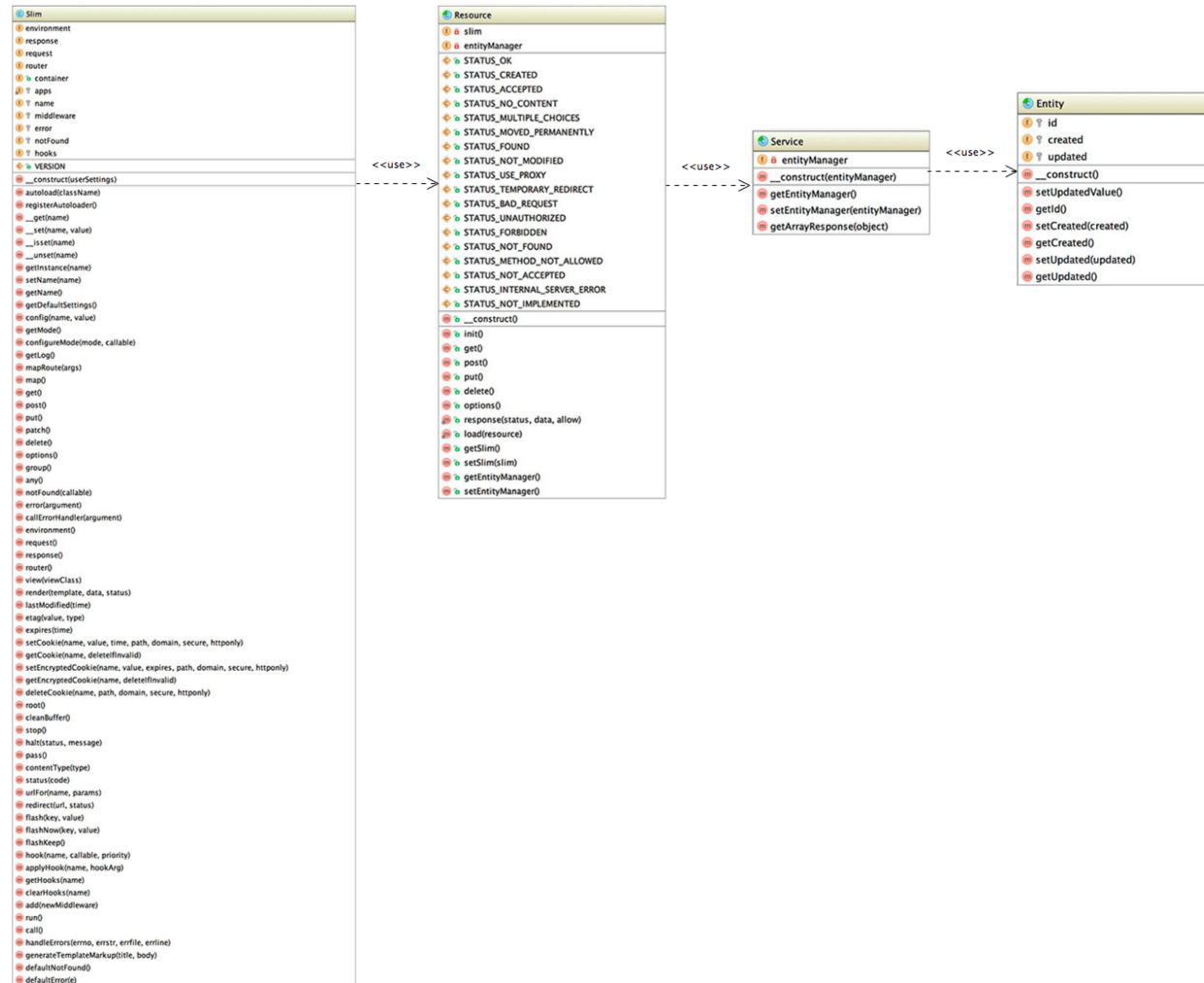


Ilustración 4.3: Diagrama de clases del sistema

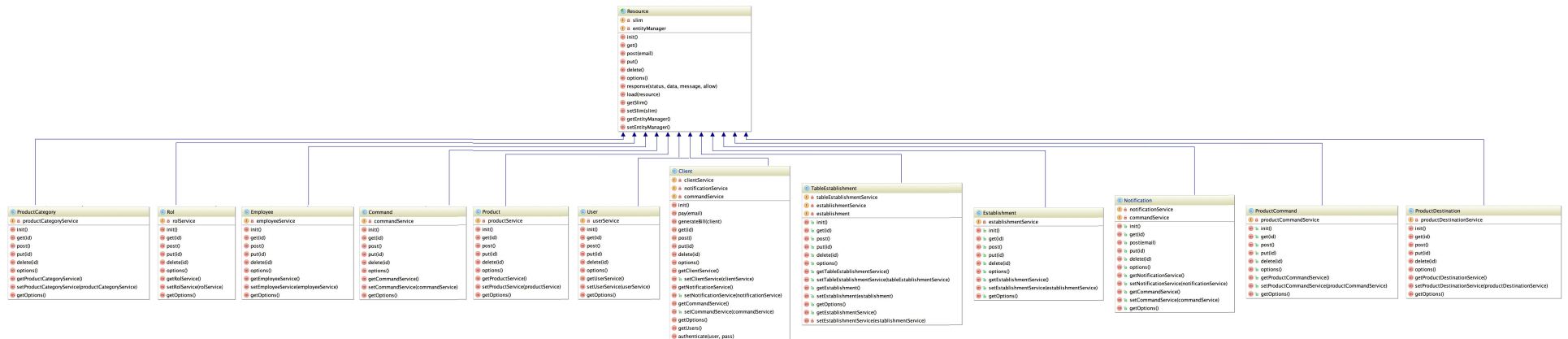


Ilustración 4.4: Diagrama de clases de resource

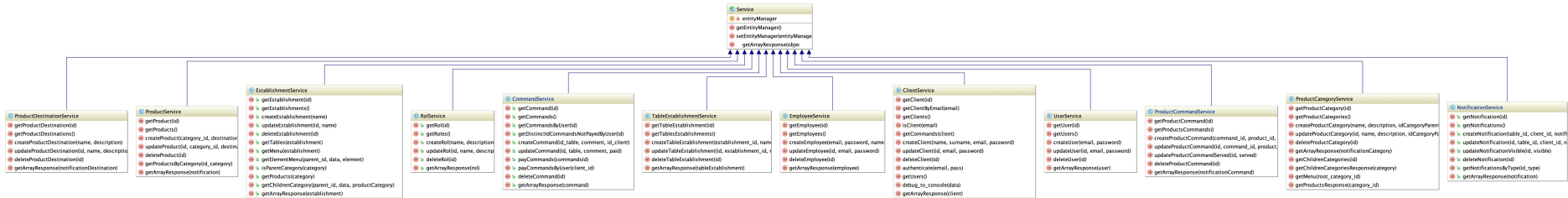


Ilustración 4.5: Diagrama de clases de Service

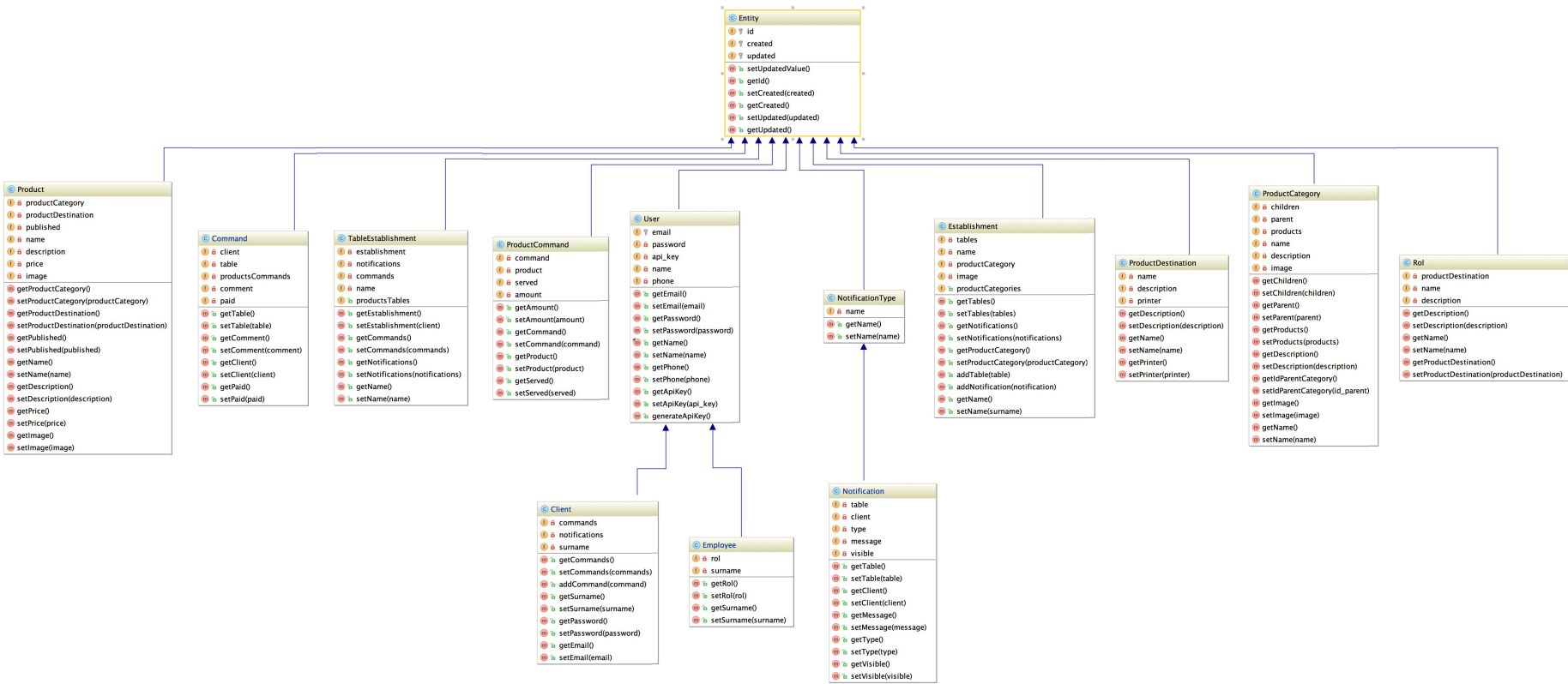


Ilustración 4.6: Diagrama de clases de Entity

4.2.3. Cliente Móvil

Modelo Vista Presentador (Vista pasiva)

El patrón MVP es una derivación del Modelo-Vista-Controlador separa cuidadosamente el Modelo de la Vista y se rompe la relación directa que hay entre ellos [34]. El núcleo del MVP es la interacción entre la Vista y el Presentador. La Vista expone un contrato a través del cual el Presentador interactúa con la Vista. Cuando los usuarios interactúan con la vista, ésta invoca un método en el Presentador y el Presentador realiza la tarea requerida en el Modelo y luego actualiza la vista mediante el Contrato.

Es importante aclarar que el Modelo representa las el modelo de objetos de la capa de negocio.

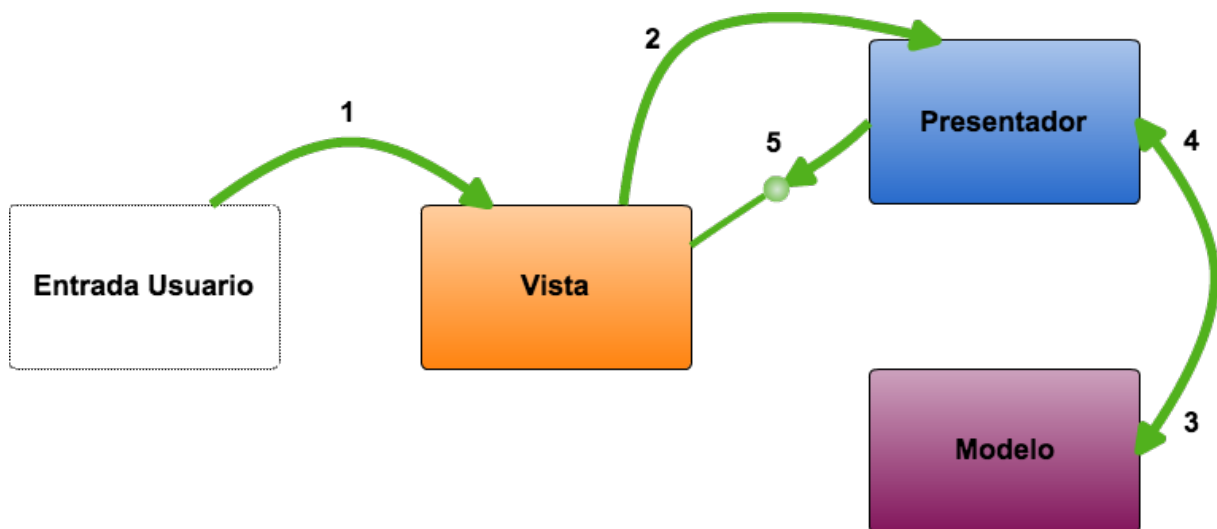


Ilustración 4.7: Modelo Vista Presentador

1. El usuario interactúa con la vista.
2. La vista notifica al presentador la acción realizada por el usuario.
3. El presentador realiza la tarea requerida en el Modelo.
4. El modelo realiza la lógica de la aplicación y notifica al Presentador los cambios a realizar.
5. El presentador actualiza la vista mediante el contrato que tienen entre ambos.

4.3. Diseño de datos

El diseño de datos nos permite concretar las estructuras de datos que se van a requerir para implementar el software.

En esta fase del diseño del software vamos a determinar la estructura de los datos sobre los que vamos a trabajar. Diferenciaremos varios elementos:

Los **clientes** tendrán los siguientes atributos: Nombre, apellidos, email y contraseña.

Los **empleados** estarán compuestos por nombre, apellidos, teléfono, email, contraseña y rol.

Las **comandas** estarán asociadas a un cliente y una mesa, si esta pagada la comanda y los productos asociados a la comanda con la cantidad que se haya pedido.

Los **productos** podrá pertenecer a varias comandas y tendrá nombre, precio y categoría del producto.

Las **categorías de productos** estará compuesto por productos, nombre de la categoría, una imagen y categoría padre.

Los **establecimientos** estarán compuestos por mesas y una categoría de productos padre, también tendrá un nombre del establecimiento.

Las **mesas** contendrá el identificador de la mesa y el establecimiento al que pertenece.

Las **notificaciones** estarán compuestas por el cliente, la mesa y el tipo de notificación.

Los **tipos de notificaciones** tendrán un nombre y estarán compuestos por varias notificaciones.

Esquema conceptual

Una vez identificados los componentes del sistema se deben de convertir a entidades o relaciones. Clientes (cliente), empleados (empleado), comandas (comanda), productos (producto), categorías de productos (categoría del producto), establecimientos (establecimiento), mesas (mesa), tipos de notificaciones (tipo de notificación) y notificaciones (notificación) serán entidades del esquema conceptual.

Las relaciones son:

Un establecimiento tiene asignado una categoría de productos y mesas.

Una categoría de productos tiene asignado productos y un producto puede estar asignado a una comanda.

Una comanda tiene asignada una mesa y un cliente.

Una notificación tiene asignado un cliente.

Podemos observar el esquema conceptual en la siguiente ilustración.

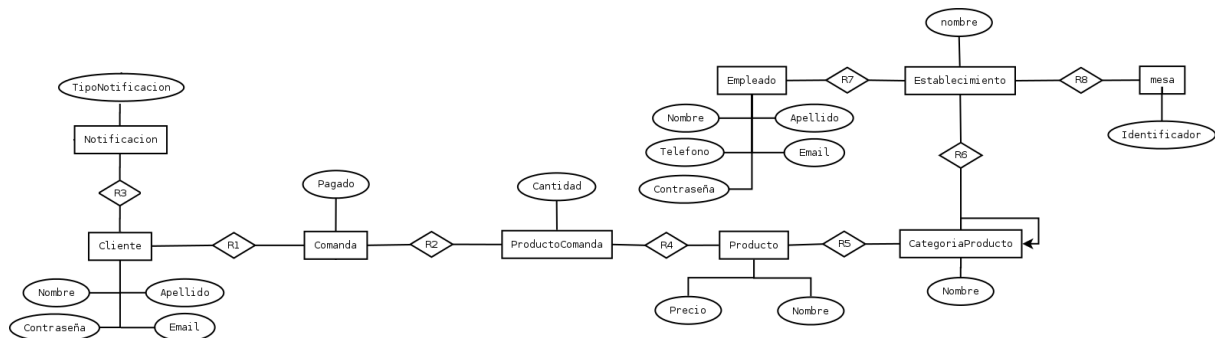


Ilustración 4.8: Esquema conceptual

De este esquema podemos sacar las siguientes relaciones:

- R1: tiene
- R2: compuesto de
- R3: realiza
- R4: contiene

- R5: pertenece
- R6: asignado a
- R7: contiene
- R8: contiene

Tablas de la Base de datos

Una vez definidos todos los componentes y sus atributos necesarios para almacenarlos como objetos tenemos que definir la estructura que tendrán las tablas en la base de datos, al igual que los tipos de los atributos de dichos objetos.

Del estudio anterior sacamos que necesitamos definir 9 tablas, estas serán: Cliente (client), comanda (command), notificación (notification), productoComanda (product_command), producto (product), categoriaProducto (product_category), establecimiento (establishment), empleado (employee) y mesa (table_establishment).

Campo	Tipo	Clave	Descripción
Id	INT	PRIMARIA	Identificador del cliente
Email	VARCHAR(255)	ÚNICO, NO NULO	Dirección de correo electrónico del cliente
Nombre	VARCHAR(80)		Nombre del cliente
Apellido	VARCHAR(80)		Apellidos del cliente
Contraseña	VARCHAR(80)		Contraseña del cliente

Tabla 11: Tabla de Cliente

Campo	Tipo	Clave	Descripción
Id	INT	PRIMARIA	Identificador de la comanda
Cliente_id	INT	FORANEA	Identificador del cliente que realiza la comanda
Mesa_id	INT	FORANEA	Identificador de la mesa desde la que se realiza la comanda
Pagado	TINYINT(1)		Variable de dos estado, una comanda puede estar pagada o no

Tabla 12: Tabla de Comanda

Campo	Tipo	Clave	Descripción
Id	INT	PRIMARIA	Identificador de la notificación
Mesa_id	INT	FORANEA	Mesa desde la que se realiza la notificación
Cliente_id	INT	FORANEA	Cliente que genera la notificación
Tipo_notificacion	INT		Tipo de notificación, 1 para llamar al camarero, 2 para pedir la cuenta

Tabla 13: Tabla de Notificación

Campo	Tipo	Clave	Descripción
Id	INT	PRIMARIA	Identificador de productoComanda
Comanda_id	INT	FORANEA	Identificador de la comanda
Producto_id	INT	FORANEA	Identificador del producto
Cantidad	INT		Unidades del producto pedidos

Tabla 14: Tabla de ProductoComanda

Campo	Tipo	Clave	Descripción
Id	INT	PRIMARIA	Identificador de producto
ProductoCategoria_id	INT	FORANEA	Identificador de la categoría del producto
Nombre	VARCHAR(80)		Nombre del producto
Precio	FLOAT		Precio unitario del producto

Tabla 15: Tabla de Producto

Campo	Tipo	Clave	Descripción
Id	INT	PRIMARIA	Identificador de CategoriaProducto
CategoriaPadre_id	INT	FORANEA	
Nombre	VARCHAR(80)		

Tabla 16: Tabla de CategoriaProducto

Campo	Tipo	Clave	Descripción
Id	INT	PRIMARIA	Identificador de categoría
CategoriaPadre_id	INT	FORANEA	Identificador de la categoría padre, si es la categoría raíz será -1
Nombre	VARCHAR(80)		Nombre de la categoría
Imagen	VARCHAR(300)		URL de la imagen asociada a la categoría

Tabla 17: Tabla de Categoría

Campo	Tipo	Clave	Descripción
Id	INT	PRIMARIA	Identificador de empleado
Nombre	VARCHAR(80)		Nombre del empleado
Apellidos	VARCHAR(80)		Apellidos del empleado
Teléfono	VARCHAR(11)		Teléfono del empleado
Email	VARCHAR(255)	ÚNICO, NO NULO	Dirección de correo electrónico del empleado
Contraseña	VARCHAR(80)		Contraseña del empleado

Tabla 18: Tabla de Empleado

Campo	Tipo	Clave	Descripción
Id	INT	PRIMARIA	Identificador de mesa
Establecimiento_id	INT	FORANEA	Identificador del establecimiento asociado
Nombre	VARCHAR(80)		Nombre de la mesa, útil para

Tabla 19: Tabla de Mesa

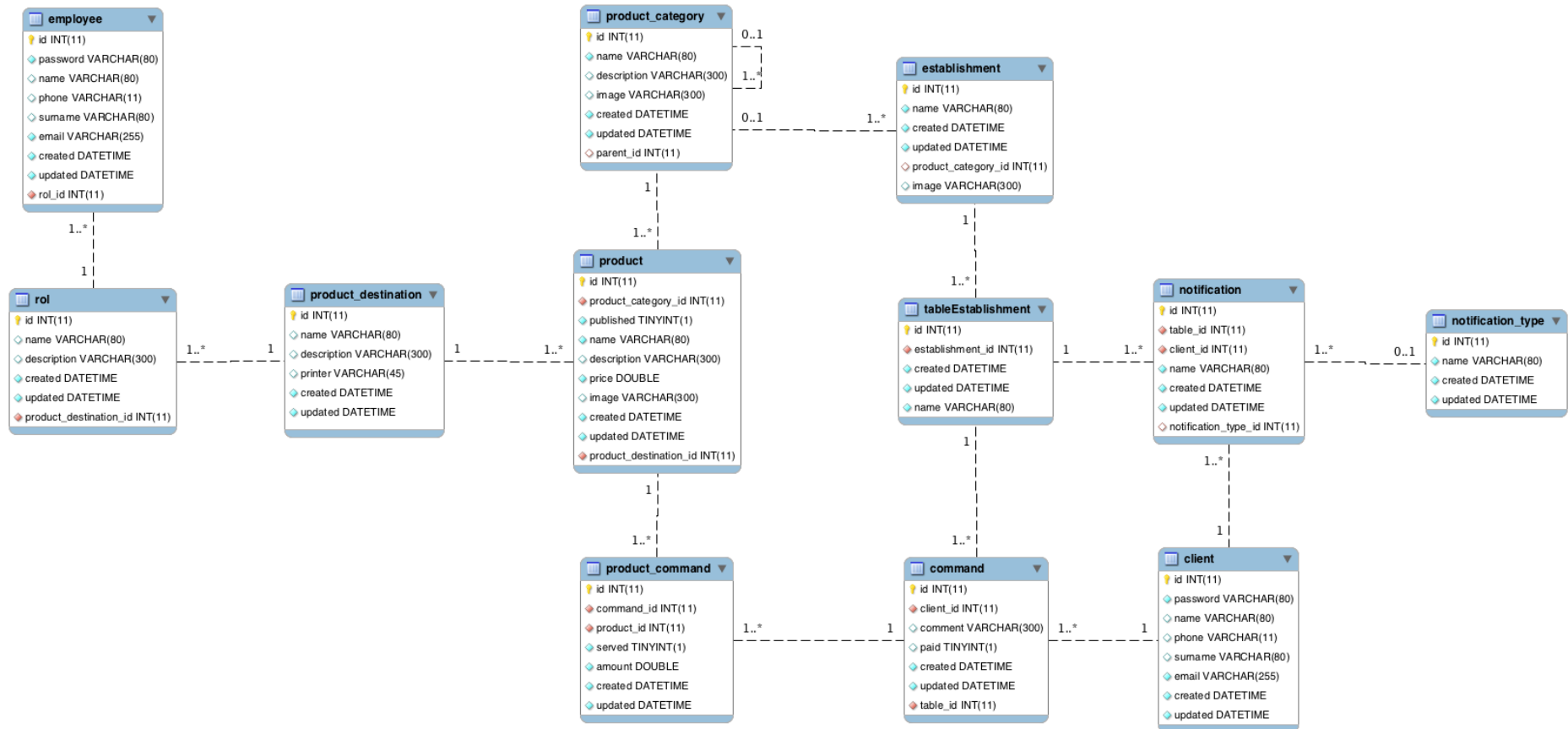


Ilustración 4.9: Esquema Entidad Relación

4.4. Diseño de interfaz

En esta fase del diseño del sistema se define cual va a ser la apariencia visual de la aplicación. El diseño de la interfaz tiene como finalidad que las aplicaciones sean más atractivas, además de hacer que la interacción con el usuario sea lo más intuitiva posible.

4.4.1. Guía de estilo

La guía de estilo a seguir es la oficial de Android para dispositivos Android 5.0 Lollipop conocido este tipo de diseño como Material Design, toda la documentación necesario está disponible en la web <https://developer.android.com/design/material> (Abril, 2015)

4.4.2. Diseño de pantallas

A continuación se enumerarán y describirán las pantallas de la aplicación Android al igual que se mostrará un diagrama de la navegación entre pantallas.

a) Lista de pantallas

Teniendo un conocimiento previo de la información que se quiere visualizar, podemos realizar una lista exhaustiva de las pantallas necesarias para permitir que los usuarios naveguen e interactúen con los datos.

Nuestra aplicación móvil queremos que permita al usuario registrarse, iniciar sesión, escanear códigos QR, consultar productos, realizar pedidos y pedir la cuenta. De aquí podemos extraer las pantallas necesarias para cumplir con estos requisitos.

1. Inicio

Pantalla principal desde la que se podrá acceder a las pantallas de registro de usuario en el caso de que no sea un usuario registrado, inicio de sesión si el usuario está registrado y cámara para capturar un código QR

2. Registro de usuario

Pantalla con un formulario de registro, el usuario podrá poner su nombre, apellido, email y contraseña para registrarse en el sistema

3. Inicio de sesión

Pantalla con un formulario de inicio de sesión, el usuario podrá poner su email y contraseña para identificarse en el sistema.

4. Cámara

Pantalla donde se mostrará la imagen que esté capturando la cámara frontal del dispositivo junto aun texto explicativo de cómo capturar el código QR.

5. Lista de categorías

Pantalla con una lista de categorías donde cada categoría mostrará su nombre y una foto que identifique la categoría.

6. Lista de productos

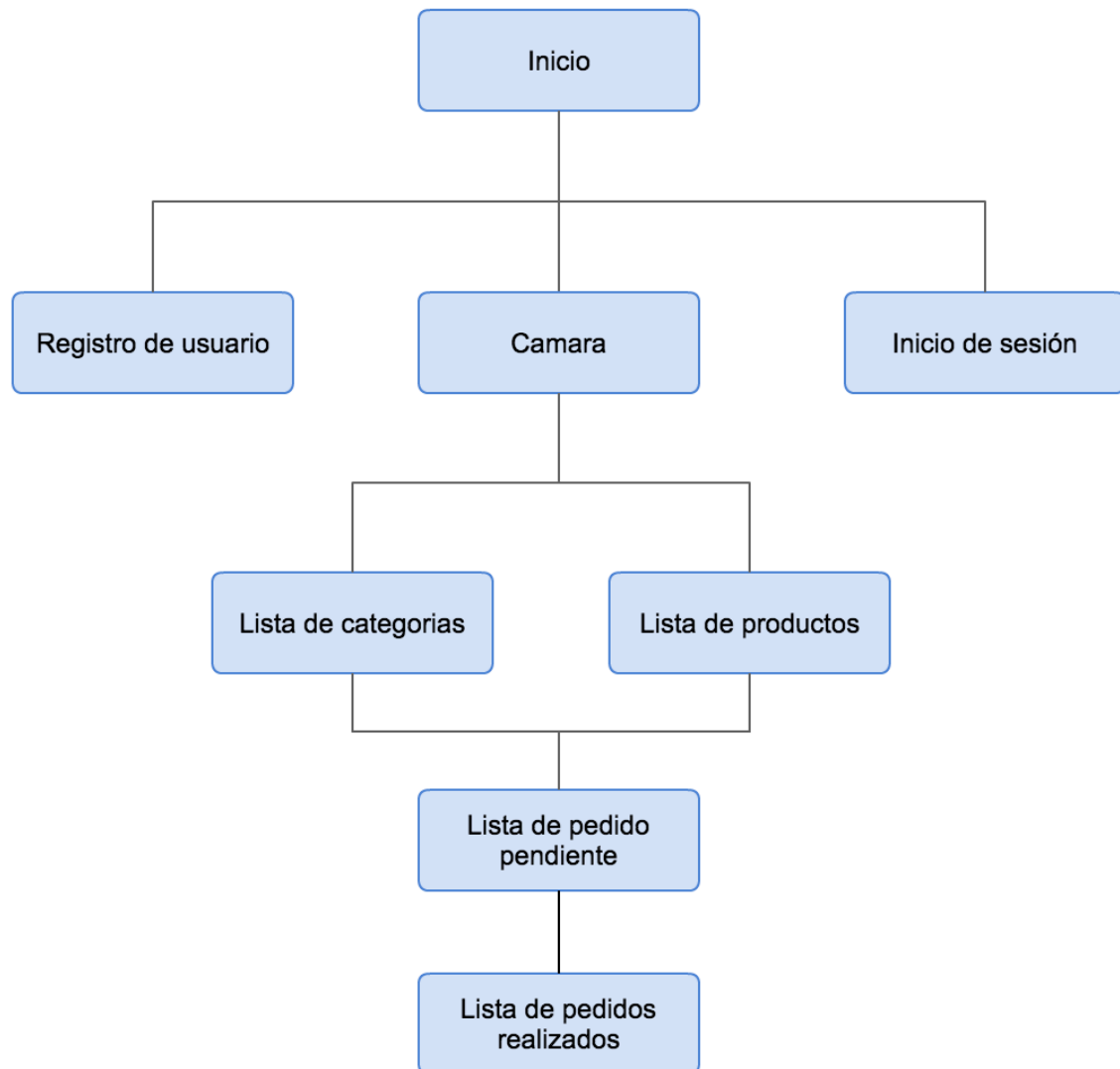
Pantalla con una lista de productos donde cada producto mostrará su nombre, el tamaño del producto, su precio y un botón para añadir una unida al carrito, esta pantalla es dinámica, ya que si añades un producto al carrito cambiará el aspecto de ese producto en la lista, apareciendo el número de unidades que tienes añadidas al carrito de ese producto, pudiendo añadir o quitar tantas unidades como se quieran.

7. Lista de pedido pendiente

Pantalla donde se podrá consultar y modificar la lista de productos pendientes de enviar al establecimiento.

8. Lista de pedidos realizados

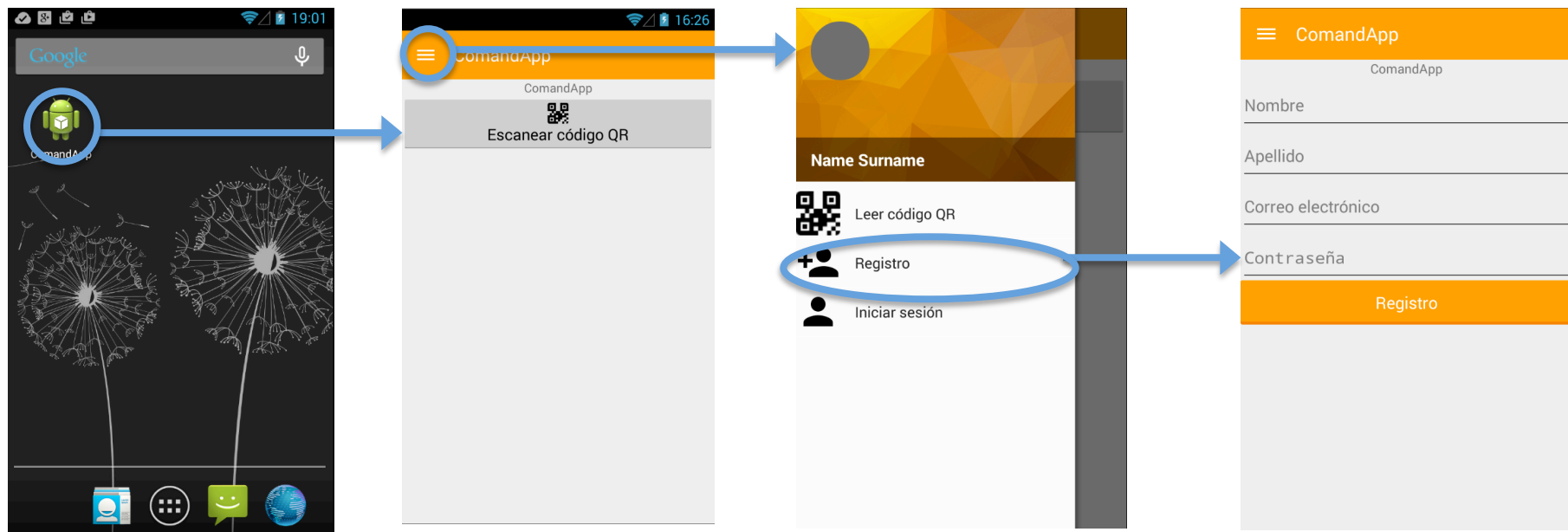
Pantalla donde se podrá consultar los productos pedidos al establecimiento y se podrá solicitar la cuenta de la mesa.

b) Diagrama de relación entre pantallas**Ilustración 4.10: Diagrama Pantallas Aplicación Móvil**

c) Storyboards

Los Storyboards nos permite ver la secuencia de pantallas de la interfaz de usuario, esta técnica nos permite comprender mejor la lógica entre pantallas y ver gráficamente como funcionaría cada caso de uso.

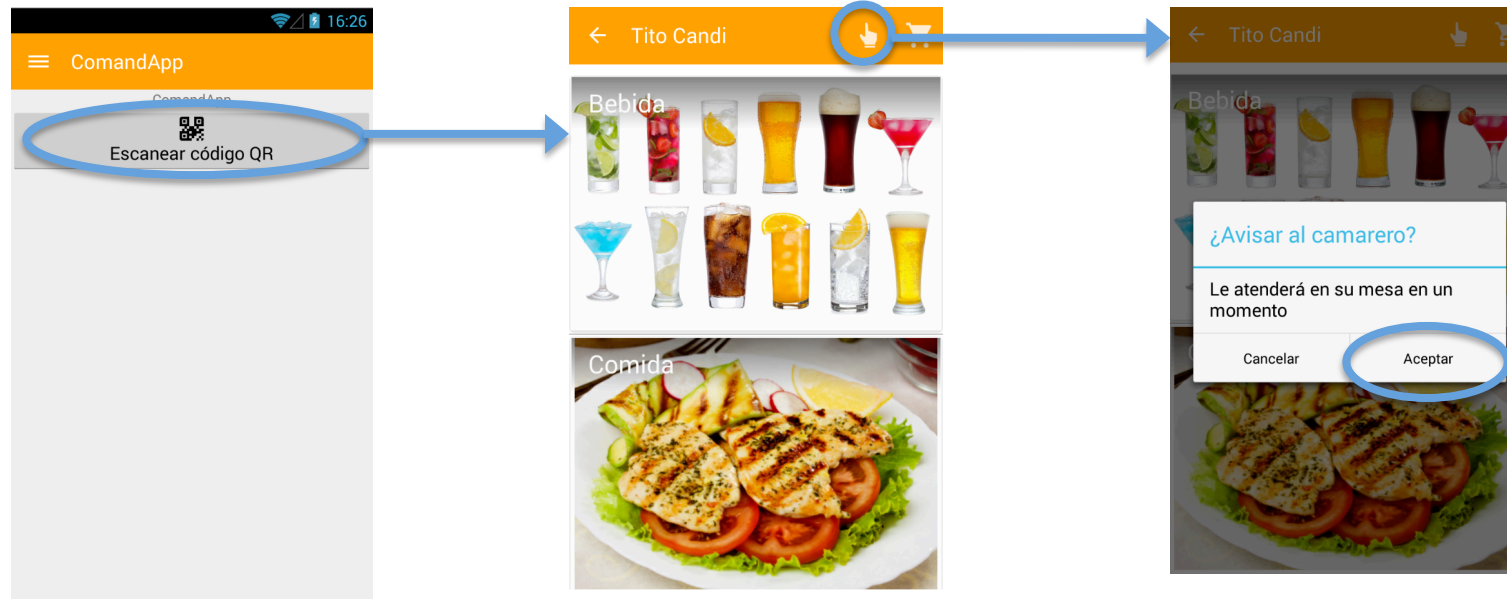
1. Registro de usuario



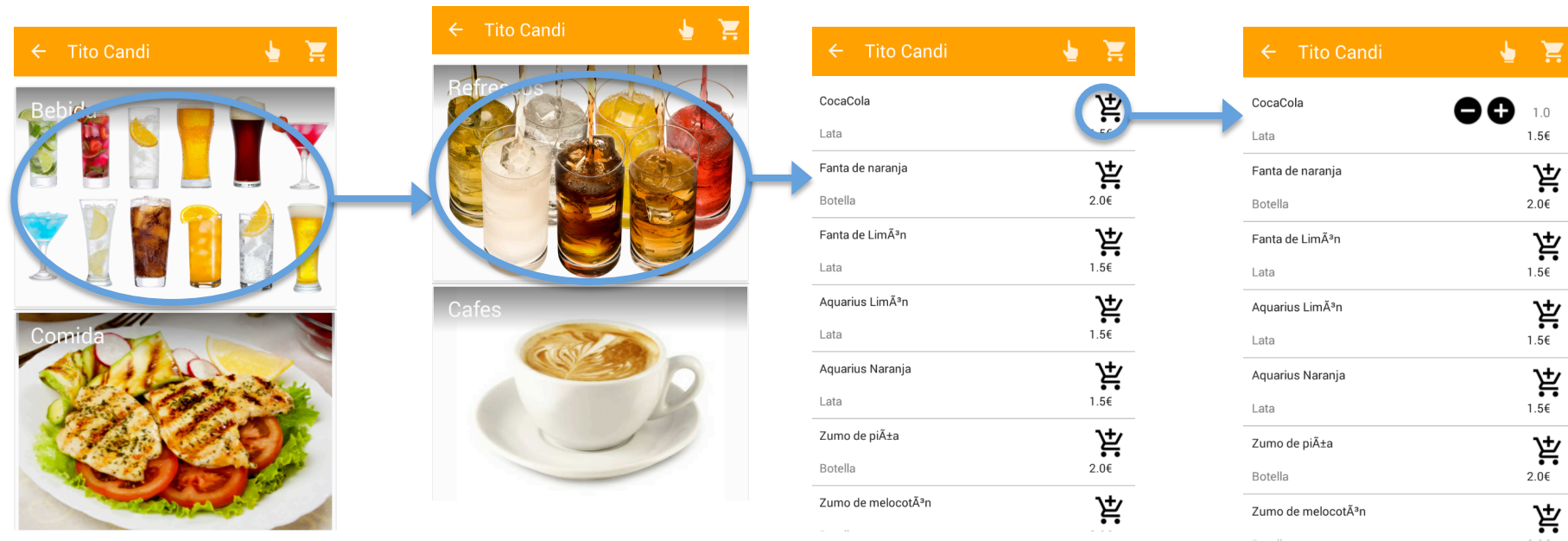
2. Inicio de sesión



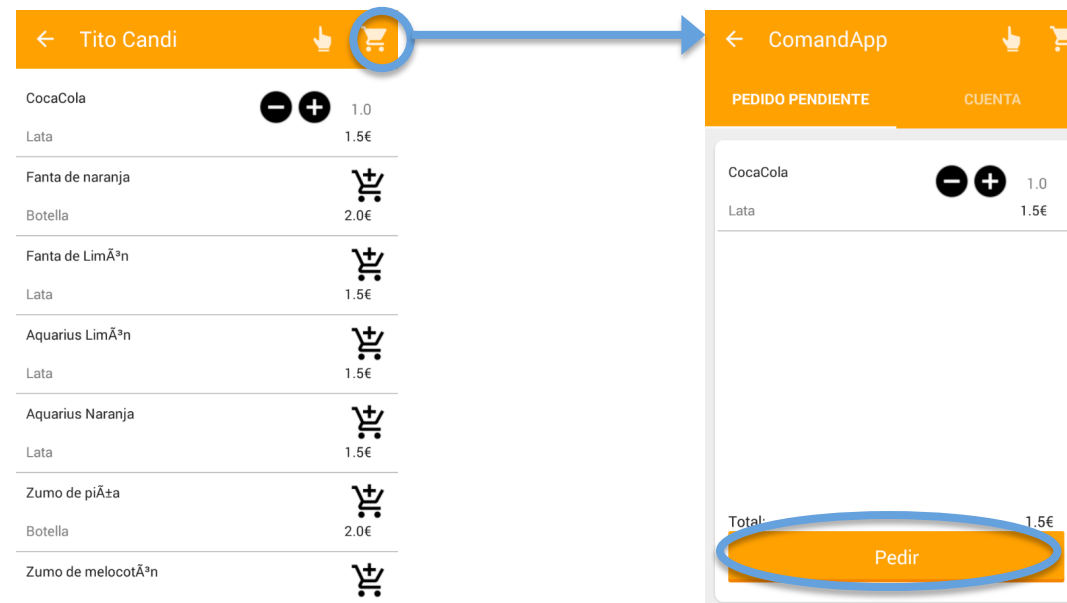
3. Llamar al camarero



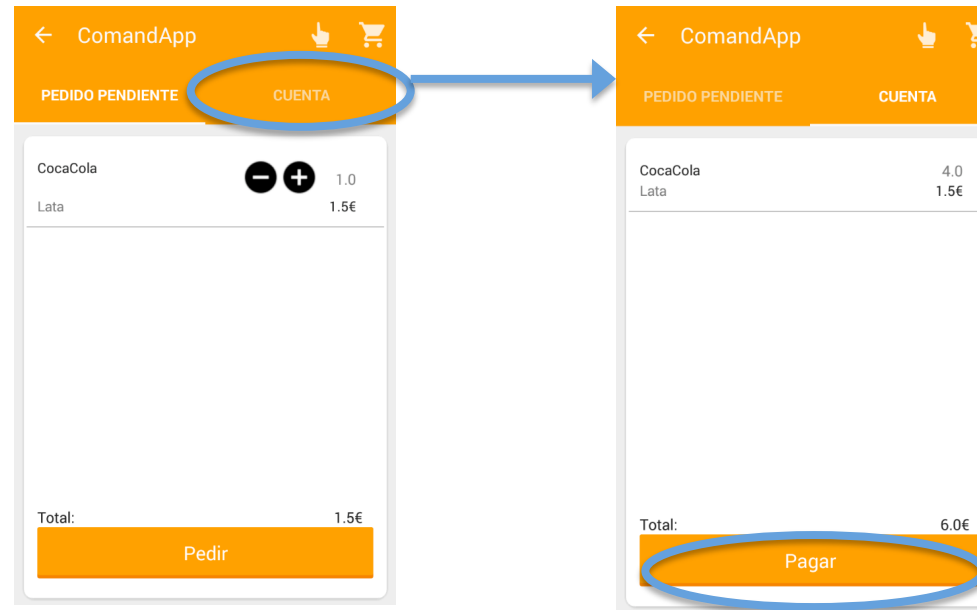
4. Añadir productos al pedido



5. Realizar pedido



6. Pedir la cuenta



5. IMPLEMENTACIÓN

5.1. Descripción

En este punto de la documentación se explicará la implementación realizada en el proyecto.

5.2. Arquitectura del sistema

El sistema estará montado sobre una arquitectura cliente-servidor, donde el cliente será el dispositivo móvil o el navegador web el cual solicitará o enviará datos. El servidor proporcionará una API donde los clientes podrán realizar o enviar datos y el servidor lo procesará y le dará respuesta a los diferentes clientes.

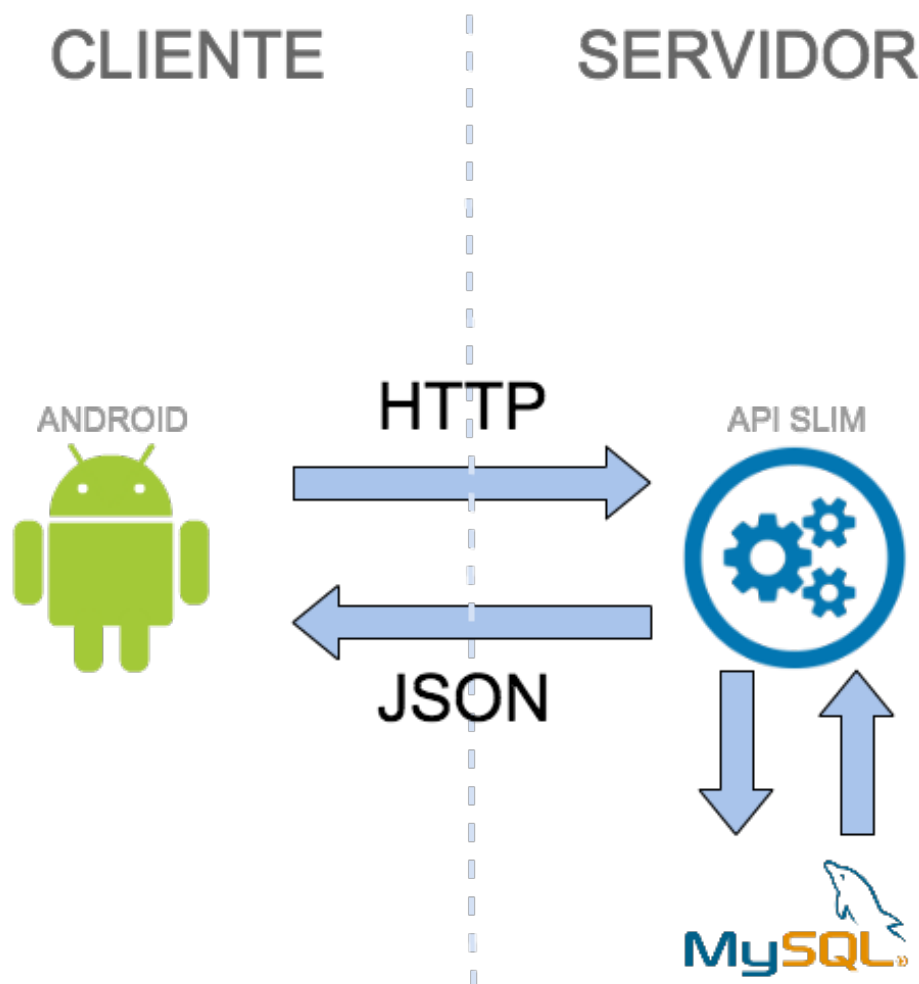


Ilustración 5.1: Esquema de la arquitectura general del sistema

5.2.1. *Arquitectura del cliente móvil*

El cliente móvil tendrá una arquitectura conocida como “Clean Architecture” lo que nos permite que la interfaz de usuario [35], la base de datos o cualquier otro agente externo sea independiente a la lógica de negocio, pudiendo ser reemplazado cualquiera de estos elementos sin gran esfuerzo.

A continuación se muestra un esquema de la arquitectura adaptada a este proyecto

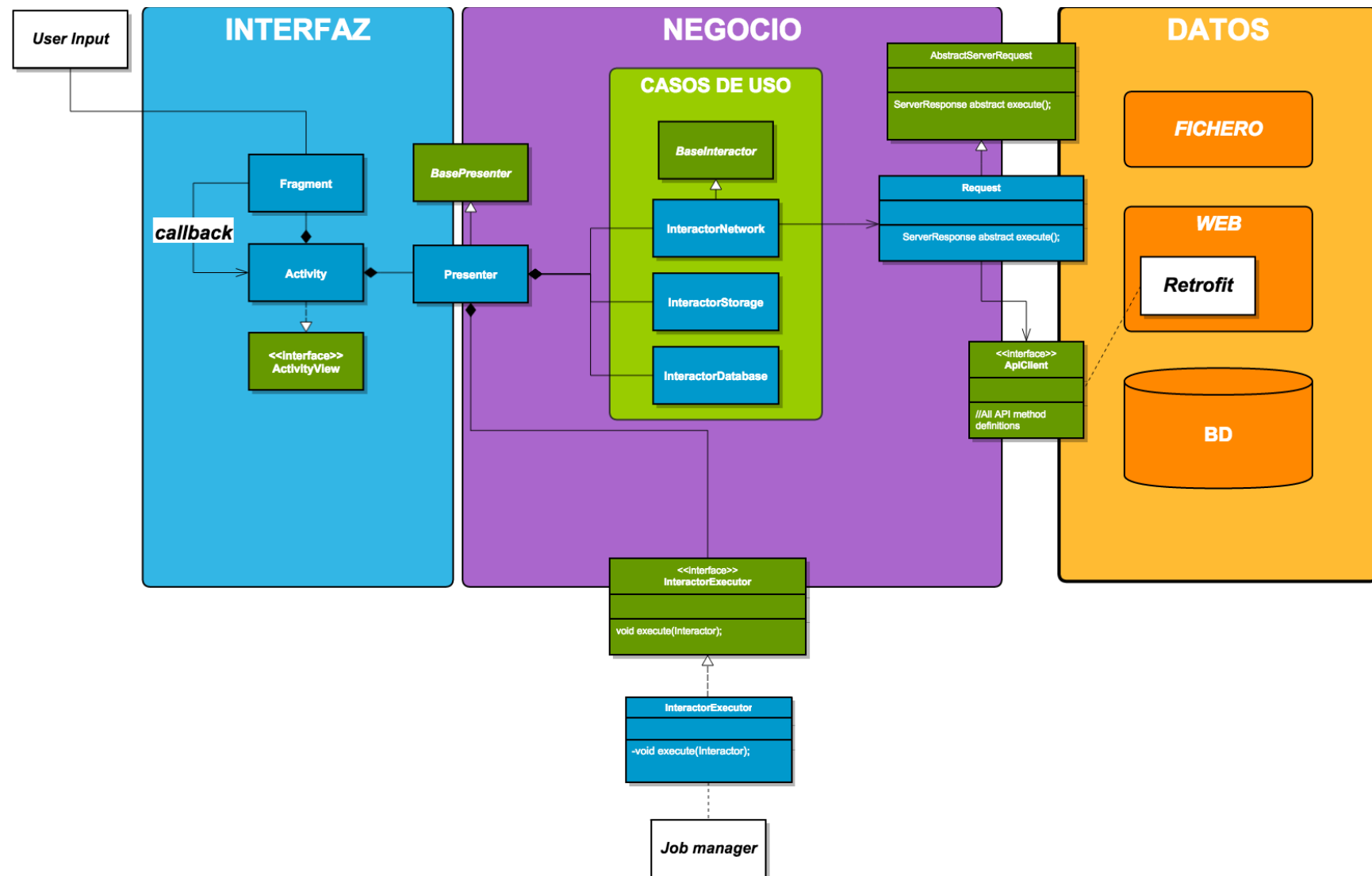


Ilustración 5.2: Esquema de la arquitectura Android

5.3. Ficheros

A continuación se describen los ficheros y directorios más importantes del servidor y del cliente.

Servidor

El servidor web se divide en 4 partes:

1. Index.php

Existen dos páginas de acceso, una cuenta con interfaz gráfica para acceder desde el navegador, permite realizar las funciones de “Inicio de sesión” y “Consultar pedidos pendientes” y se encuentra en el directorio *raíz* y la otra pagina de acceso se encuentra en el directorio *api*, se encarga de recibir todas las peticiones de los móviles, esta no cuenta con interfaz gráfica y contiene todas las funcionalidades del sistema.

2. src

Toda la lógica del servidor desarrollada en este proyecto se encuentra en el directorio *src*, se divide en otros 3 directorios, conteniendo las entidades, recursos y servicios.

3. templates

La interfaz gráfica que se muestra desde la web se encuentra en el directorio *templates*, este directorio se ha dividido en otros dos directorios, *admin* contiene los ficheros html y el directorio *assets* contiene las imágenes, ficheros css y javascript que utilizan los ficheros html.

4. vendor

El directorio vendor contiene los diferentes recursos externos que utilizamos en el sistema como el framework y las librerías para su correcto funcionamiento.

Cliente

El cliente móvil se divide en dos grandes partes:

1. app

En este directorio encontraremos el código que se ejecuta sobre el SDK de Android, dentro del directorio *src* se encuentra dos directorios.

- debug

En este directorio se encuentran la lógica y los recursos necesarios para la aplicación cuando se esta probando facilitando opciones de red y de interfaz de usuario. Este directorio solo esta presente durante el desarrollo del proyecto.

- main

Dentro de este directorio se encuentra la lógica y los recursos de la aplicación propiamente dicha.

2. core

En este directorio se encuentra código java, este directorio tiene la finalidad de extraer el código que no depende del SDK de Android para que pueda ser reutilizado en otros proyectos.

6. CONCLUSIONES

Este proyecto nació de una necesidad, en ese momento no existía ninguna solución parecida a la que propongo en este proyecto en España, creándome la sensación de que no fuera una necesidad real o que fuera inviable un sistema de estas características.

Durante el desarrollo del proyecto he visto como han aflorado multitud de soluciones para esta necesidad. Algunas soluciones trabajan sobre servidores locales montados en los establecimientos, hay quienes ofertan aplicaciones únicas para cada establecimiento, otros integran tablets en las mesas, etc... cada uno da su solución con sus pros y contras, creo que la solución que propongo tiene dos valores diferenciadores con respecto a las otras propuestas.

El principal valor para el establecimiento es el coste de tener un servicio de este tipo, ya que el coste de poseer un servidor dedicado a un establecimiento incrementaría bastante el precio final, hay que tener en cuenta que no solo es tener un servidor, si no todo lo que ello conlleva, como la gestión y el mantenimiento del mismo.

Con los servidores centralizados, como se propone en esta solución, el coste del servidor, su mantenimiento y su gestión sería costado por todos los establecimiento que utilicen nuestros servidores, abaratando costes tanto de despliegue como de gestión y mantenimiento de los servidores. Esta solución también nos permite dar un servicio de actualizaciones a todos los establecimientos en tiempo real y con un bajo coste, ya que no tendríamos que actualizar cada establecimiento, simplemente actualizando los servidores centrales y la aplicación Android se benefician todos los establecimientos. Puede surgir la duda de que al final todos los establecimientos tendrían la misma interfaz y no se diferenciarían unos de otros, pero esto no es cierto ya que más adelante podría incluirse la opción de personalización de la aplicación por parte del establecimiento, teniendo un aspecto único cada aplicación una vez leído el código QR.

El otro valor diferencial recae en los datos, al estar toda la información centralizada se puede hacer minería de datos sobre todos los establecimientos y

ofrecer a cada establecimiento nuevas estrategias de ventas a cada establecimiento como pudiera ser conocer cuales son los días de menor ventas en el barrio que se encuentra situado el establecimiento y proponer ofertas de los productos que tienen más salida o que generan más beneficio al establecimiento para atraer a la clientela.

Otro beneficio para los establecimientos es la obtención de actualizaciones en tiempo real tanto para el aplicativo web como para la aplicación Android.

También considero que es una ventaja para los usuarios de la aplicación Android ya que no necesitará descargarse una aplicación para cada establecimiento, con una única aplicación podrá realizar pedidos en los diferentes establecimientos adheridos al servicio y tener toda la información de sus pedidos en los diferentes establecimientos visitados.

Tras el desarrollo del proyecto puedo confirmar que se han realizado las funcionalidades descritas al comienzo del proyecto y que existe un amplio abanico de funcionalidades que se pueden ampliar como pudiera ser integrar el pago con tarjeta desde la aplicación, fidelización de clientes, almacenamiento de ingredientes de cada plato, tener en cuenta el rol “Gerente del establecimiento” para la consulta de estadísticas. También se puede mejorar en la seguridad de la aplicación con utilización de conexiones seguras por SSH.

Para terminar, comentar que a raíz de este proyecto he aprendido mucho sobre diferentes tecnologías, patrones de diseño y filosofías de desarrollo como puede ser SOLID, la cual supone un esfuerzo al principio pero que los resultados obtenidos son tremendamente satisfactorios.

BIBLIOGRAFÍA

- [1] Interbrand. (2015, Abril) Interbrand. [Online]. <http://bestglobalbrands.com/2014/ranking/>
- [2] Fundación Telefónica, "Sociedad de la Información en España," Barcelona, 2014.
- [3] Ministerio de Trabajo e Inmigración. (2009, Abril) Boletín oficial del estado. [Online]. <http://www.boe.es/boe/dias/2009/04/04/pdfs/BOE-A-2009-5688.pdf>
- [4] Wikipedia. (2014, Septiembre) Servidor HTTP Apache - Wikipedia, la enciclopedia libre. [Online]. http://es.wikipedia.org/wiki/Servidor_HTTP_Apache#cite_note-3
- [5] Wikipedia. (2014, Diciembre) MySQL - Wikipedia, la enciclopedia libre. [Online]. <http://es.wikipedia.org/wiki/MySQL>
- [6] Wikipedia. (2014, Diciembre) Android - Wikipedia, la enciclopedia libre. [Online]. <http://es.wikipedia.org/wiki/Android>
- [8] Wikipedia. (2014, Diciembre) HTML - Wikipedia, la enciclopedia libre. [Online]. <http://es.wikipedia.org/wiki/HTML>
- [7] Wikipedia. (2014, Diciembre) PHP - Wikipedia, la enciclopedia libre. [Online]. <http://es.wikipedia.org/wiki/PHP>
- [9] Wikipedia. (2014, Diciembre) JavaScript - Wikipedia, la enciclopedia libre. [Online]. <http://es.wikipedia.org/wiki/JavaScript>
- [10] Wikipedia. (2014, Diciembre) Java (lenguaje de programación) - Wikipedia, la enciclopedia libre. [Online]. http://es.wikipedia.org/wiki/Java_%28lenguaje_de_programaci%C3%B3n%29
- [11] Josh Lockhart. (2015, Abril) Slim Framework. [Online]. <http://www.slimframework.com/>
- [12] Nacho Pacheco. (2015, Abril) Bases de datos y Doctrine — Manual de Symfony2 en Español. [Online]. <http://gitnacho.github.io/symfony-docs-es/book/doctrine.html>
- [13] tuupola. (2014, Diciembre) <http://www.appelsiini.net/>. [Online]. <http://www.appelsiini.net/projects/slim-basic-auth>
- [14] ircmaxell. (2014, Diciembre) [Online]. https://github.com/ircmaxell/password_compat
- [15] firebase. (2014, Diciembre) [Online]. <https://github.com/firebase/php-jwt>
- [16] Michael Pardo. (2015, Abril) Activeandroid by pardom. [Online]. <http://www.activeandroid.com/>
- [18] Square. (2015, Abril) Retrofit. [Online]. <http://square.github.io/retrofit/>

- [17] fricolaB. (2013, Jan.) ZXing - Wiki Formación - fricolaB - herramientas mayúsculas para seres pequeños. [Online]. <http://fricolab.com/wiki/index.php?title=ZXing>
- [19] Square. (2015, Abril) OkHttp. [Online]. <http://square.github.io/okhttp/>
- [20] Square. (2015, Abril) Dagger. [Online]. <http://square.github.io/dagger/>
- [21] Square. (2015, Abril) Otto. [Online]. <http://square.github.io/otto/>
- [22] Jake Wharton. (2015, Abril) Butter Knife. [Online]. <https://github.com/JakeWharton/butterknife>
- [23] Android. (2015, Abril) <https://developer.android.com/>. [Online]. <https://developer.android.com/tools/support-library/features.html>
- [24] Wharton Jake. (2015, Abril) <https://github.com/JakeWharton/timber>. [Online]. <https://github.com/JakeWharton/timber>
- [25] Android. (2015, Abril) <https://developer.android.com/google/play-services>. [Online]. <https://developer.android.com/google/play-services>
- [26] Square. (2015, Abril) Picasso. [Online]. <https://github.com/square/picasso>
- [28] Jeff Gilfelt. (2015, Abril) Android System Bar Tint. [Online]. <https://github.com/jgilfelt/SystemBarTint>
- [27] Benjamin Weiss. (2015, Abril) Crouton. [Online]. <https://github.com/keyboardsurfer/Crouton>
- [29] Jake Wharton. (2015, Abril) Madge. [Online]. <https://github.com/JakeWharton/madge>
- [30] Jake Wharton. (2015, Abril) Scalpel. [Online]. <https://github.com/JakeWharton/scalpel>
- [31] Wikipedia. (2014, Junio) Especificación de requisitos de software - Wikipedia, la enciclopedia libre. [Online]. http://es.wikipedia.org/wiki/Especificaci%C3%B3n_de_requisitos_de_software
- [32] BENET CAMPDERRICH FALGUERAS, *INGENIERIA DEL SOFTWARE*, 1st ed. España: UOC (UNIVERSITAT OBERTA DE CATALUNYA), 2003.
- [33] Carlos Azaustre. (2015, Febrero) <https://carlosazaustre.es/>. [Online]. <https://carlosazaustre.es/blog/que-es-la-autenticacion-con-token/>
- [34] Manoj Jaggavarapu. (2015, Abril) Presentation Patterns : MVC, MVP, PM, MVVM | Manoj Jaggavarapu. [Online]. <http://manojjaggavarapu.wordpress.com/2012/05/02/presentation-patterns-mvc-mvp-pm-mvvm/>
- [35] The clean Architecture. (2015, Abril) <http://www.8thlight.com/>. [Online].

<http://blog.8thlight.com/uncle-bob/2012/08/13/the-clean-architecture.html>

[36] mvcsharp. (2014, Aug.) MVC#. [Online]. <http://mvcsharp.org/Overview/Default.aspx>

[38] Tiphaine MARCADÉ. (2015, Enero) E-Sports: Red social para deportistas. [Online]. http://laurel.datsi.fi.upm.es/~ssoo/DAW/Trabajos/2008-2009/001/func_es

[37] IAB. (2015, Abril) IAB Spain. [Online]. http://www.iabspain.net/wp-content/uploads/downloads/2013/06/I_estudio_ecommerce.pdf

[39] Path. (2015, Abril) android-priority-jobqueue. [Online]. <https://github.com/path/android-priority-jobqueue>

ANEXO 1

MANUAL DE INSTALACIÓN DEL SERVICIO WEB

En este anexo hablaremos de cómo configurar los componentes necesarios para poder ejecutar el servicio Web en el hosting contratado con todas sus funcionalidades, pero antes de esto existe unos requisitos mínimo que debe de cumplir el hosting.

Requisitos mínimos:

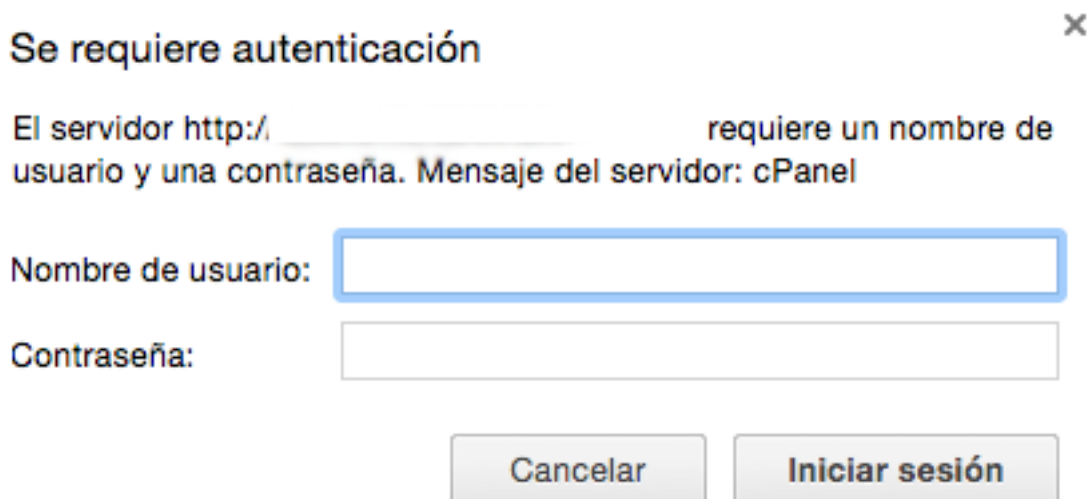
- Soporte para PHP 5.5
- Motor de Base de datos MySQL 5

Una vez encontrado un hosting o configurando un servidor propio con estos requisitos mínimos, podemos proceder a la instalación del servicio web.

En nuestro caso utilizaremos un cliente FTP y phpMyAdmin (Herramienta disponible desde el panel de control del hosting).

Lo primero que vamos a hacer es crear la base de datos y crear un usuario para la base de datos desde el panel de control del hosting.

Para acceder al panel de control, la empresa donde hemos contratado el hosting nos facilitara los datos de acceso (url, usuario y contraseña). Introduciremos la url en el navegador, a continuación nos solicitará el usuario y contraseña, a partir de este punto habrá datos sensibles ocultos en las imágenes por seguridad.



Se requiere autenticación X

El servidor http://[redacted] requiere un nombre de usuario y una contraseña. Mensaje del servidor: cPanel

Nombre de usuario:

Contraseña:

Ilustración I

Una vez introducidos los datos accederemos al panel de control de todo lo relativo al alojamiento web.

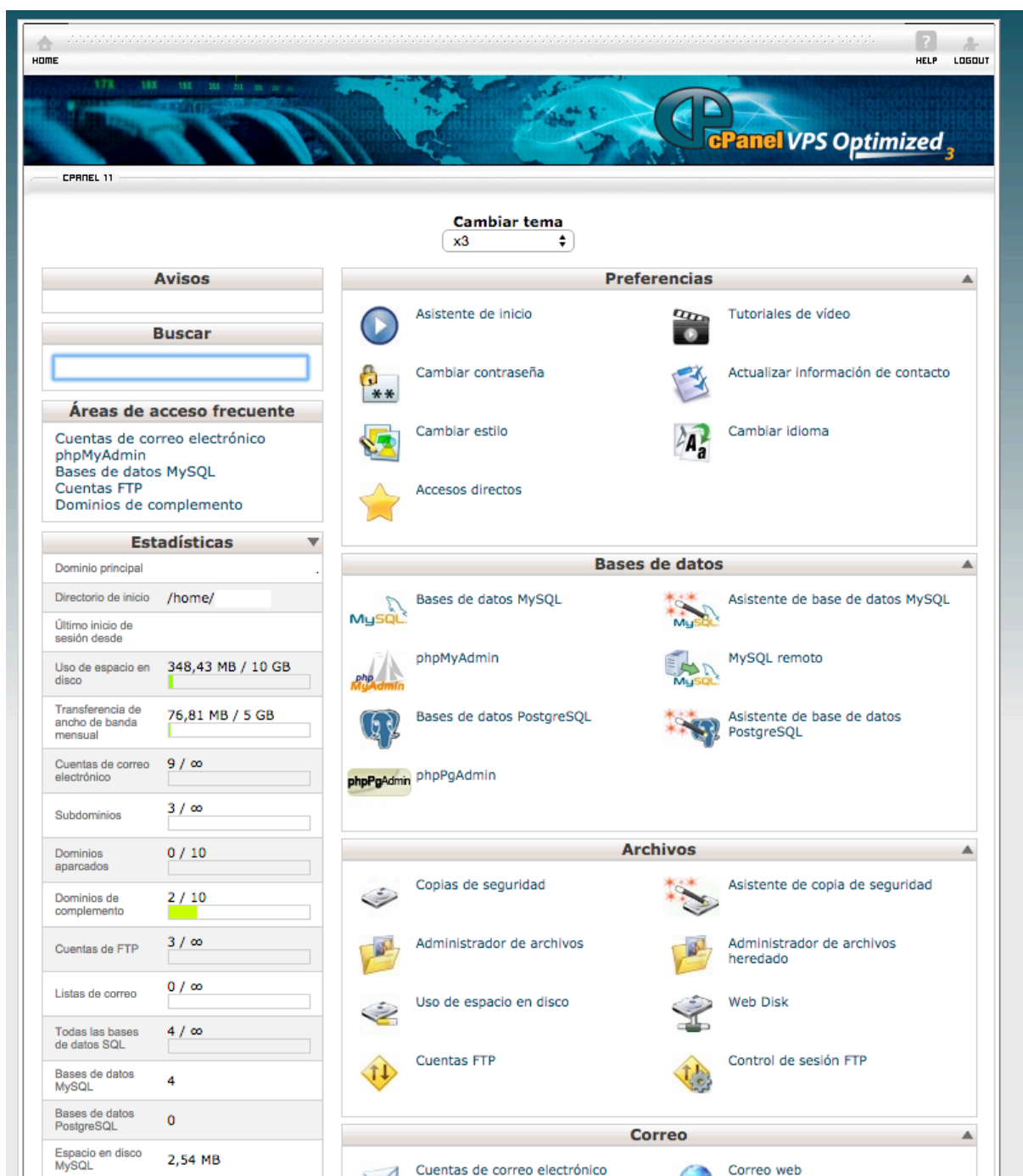


Ilustración II

Para crear la base de datos y el usuario de la base de datos accederemos a la opción “*Bases de datos MySQL*” dentro de la categoría “*Bases de datos*”.

Desde esta pantalla podremos crear la base de datos simplemente poniéndole el nombre que deseemos de la base de datos acompañado del prefijo “idHosting_”, en este caso le pondremos “idHosting_tfgUjaen” y pulsaremos “*Crear base de datos*”

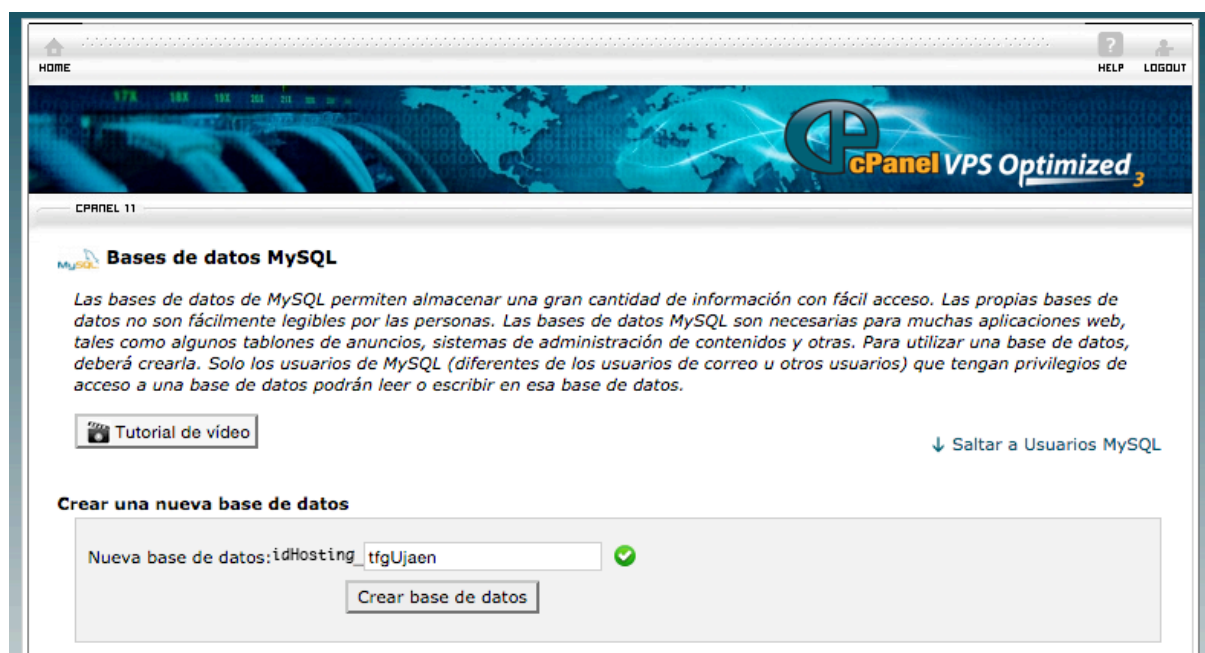


Ilustración III

Una vez creada la base de datos tenemos que crear el usuario, en la misma pantalla que hemos creado la base de datos tenemos el formulario para crear usuarios.

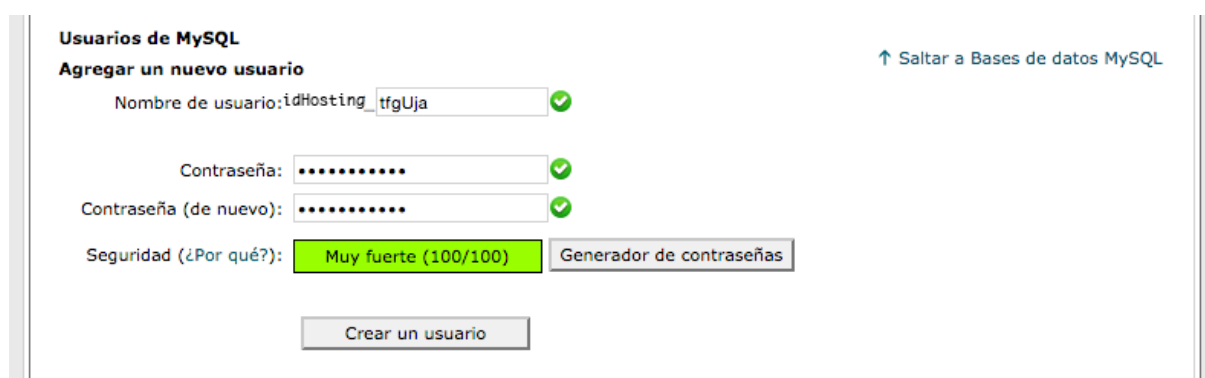
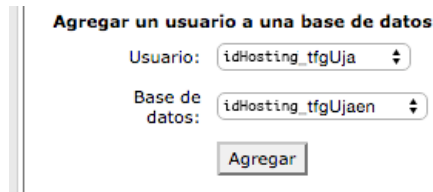


Ilustración IV

Teniendo creada la base de datos y el usuario el ultimo paso que nos queda en esta pantalla es añadir el usuario a la base de datos.



Agregar un usuario a una base de datos

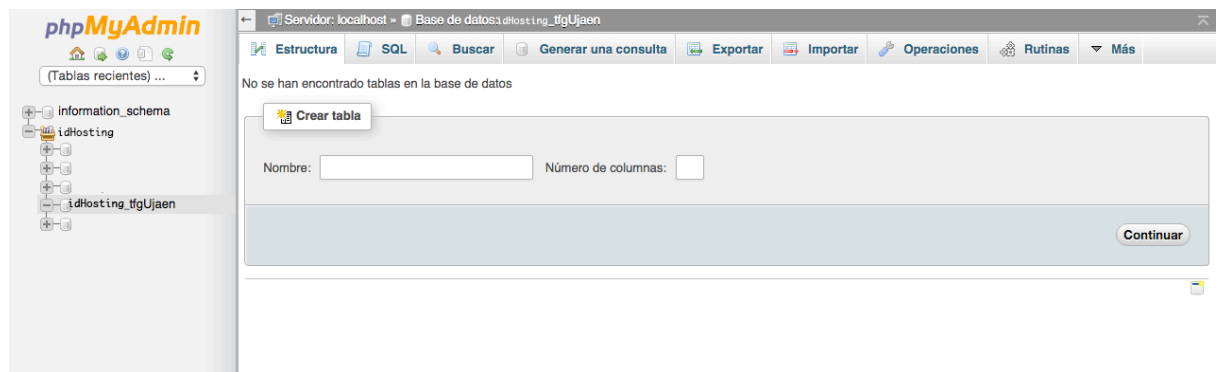
Usuario:

Base de datos:

Ilustración V

Y con este ultimo paso ya tenemos configurada la base de datos, ahora tenemos que importarle las tablas con sus tuplas y relaciones.

Desde el panel de control accederemos a la opción “phpMyAdmin”.



phpMyAdmin

Servidor: localhost - Base de datos: idHosting_tfgUjaen

Estructura SQL Buscar Generar una consulta Exportar Importar Operaciones Rutinas Más

No se han encontrado tablas en la base de datos

Crear tabla

Nombre: Número de columnas:

Continuar

Ilustración VI

Desde esta pantalla seleccionamos la base de datos desde el menú izquierdo y una vez dentro de la base de datos pulsamos “Importar” en el menú superior.

The screenshot shows a web interface for importing data into a database. The title is 'Importando en la base de datos "idHosting_tfgUjaen"'. The interface is divided into several sections:

- Archivo a importar:** A section for selecting the file to import. It includes instructions: 'El archivo puede ser comprimido (gzip, bzip2, zip) o descomprimido. Un archivo comprimido tiene que terminar en .[formato].[compresión]. Por ejemplo: .sql.zip'. There is a button 'Seleccionar archivo' and a status 'Ningún archivo seleccionado (Máximo: 50MB)'. Below this is a dropdown for 'Conjunto de caracteres del archivo:' set to 'utf-8'.
- Importación parcial:** A section for partial import. It has a checkbox 'Permitir la interrupción de una importación en caso que el script detecte que se ha acercado al límite de tiempo PHP. (Esto podría ser un buen método para importar archivos grandes; sin embargo, puede dañar las transacciones.)' which is checked. Below it is a text input for 'Número de filas a omitir, iniciando de la primer fila:' with the value '0'.
- Formato:** A dropdown menu set to 'SQL'.
- Opciones específicas al formato:** A section for format-specific options. It includes a dropdown for 'Modalidad SQL compatible:' set to 'NONE' and a checked checkbox 'No utilizar AUTO_INCREMENT con el valor 0'.

At the bottom of the form is a 'Continuar' button.

Ilustración VII

En esta pantalla tenemos que pulsar en “*Seleccionar archivo...*” y seleccionaremos el archivo “*instalacionBaseDeDatos.sql*” disponible en el CD adjunto, dentro del directorio “*/software/servidor*”.

Una vez seleccionado el archivo pulsaremos “*Continuar*” y tendremos lista nuestra base de datos.

Ahora configuraremos el fichero de configuración de la api para que detecte y pueda trabajar con la base de datos y lo subiremos todos los ficheros al servidor mediante un cliente FTP.

Dentro del CD accedemos al directorio “*/software/servidor/ficheros/*” y copiamos el contenido al ordenador. Abrimos el archivo “*local.ini*” que se encuentra dentro del directorio “*config*”, este archivo es un texto plano por lo que podremos abrirlo con cualquier editor de textos.

Modificamos el fichero poniendo los datos de la base de datos configurada previamente en el servidor, quedando el fichero de la siguiente manera.

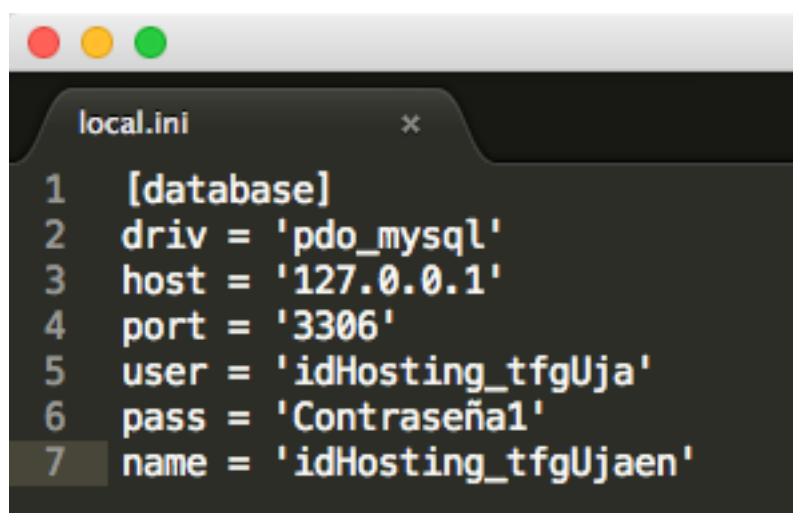


Ilustración VIII

Ya tenemos configurada la api con la base de datos, ahora nos queda subir los ficheros al servidor, para ello voy a utilizar el cliente de FTP “CyberDuck” aunque se puede utilizar cualquier otro cliente.

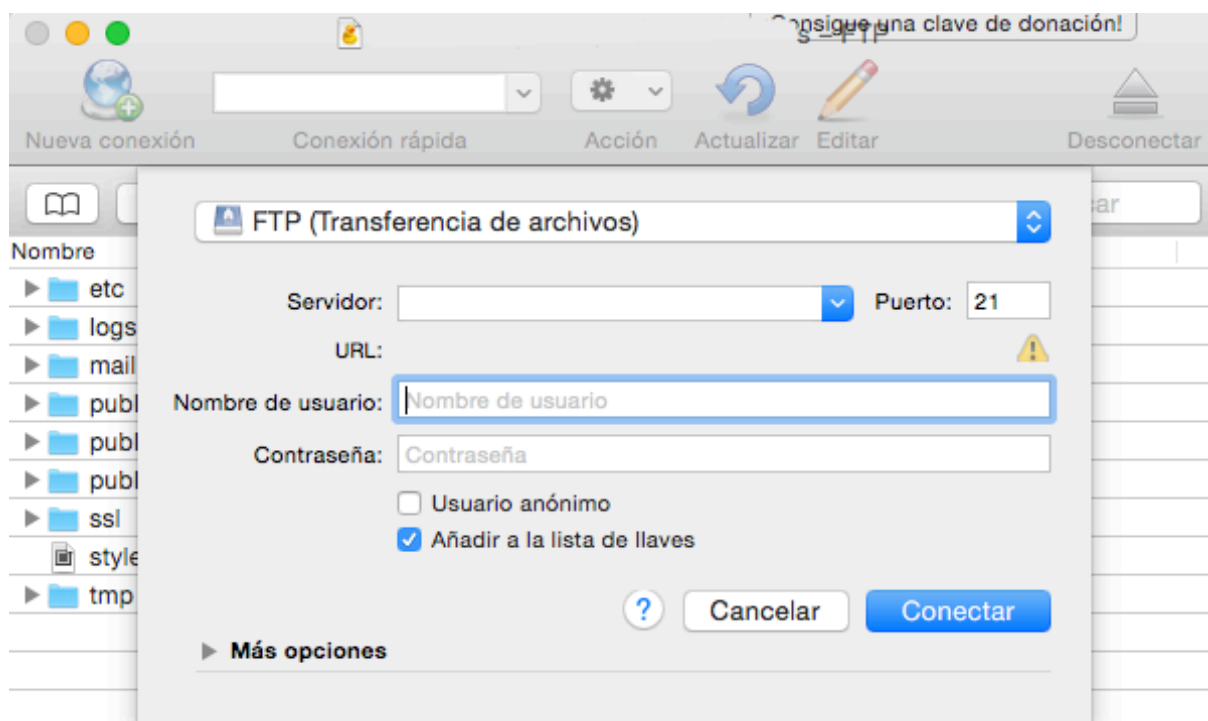


Ilustración IX

Al abrir el cliente configuramos una nueva conexión, poniendo los datos que nos haya facilitado la empresa servidora.

Una vez se haya realizado la conexión correctamente, se mostrará la estructura de directorios por la que podremos navegar. En nuestro caso subiremos los ficheros a “/public_html/nombreDominio/tfgUjaen”.

Para subir los ficheros hay que tener abierto en el cliente FTP la carpeta y arrastrar los ficheros desde la carpeta del ordenador al cliente ftp. Este proceso tardará unos minutos.

Una vez finalizada la subida de ficheros tendremos nuestro servidor totalmente configurado y listo para usarse.

ANEXO 2

MANUAL DE INSTALACIÓN DEL CLIENTE ANDROID

En este anexo explicaremos cómo Instalar la aplicación en cualquier dispositivo Android con una versión superior a 2.2.

Para enviar el fichero “proyecto.apk” desde el CD al móvil existen infinidad de métodos. Podemos enviarlo por el puerto usb, bluetooth, por correo electrónico o subirlo a un servicio de alojamiento en la nube desde el ordenador y descargarlo en el dispositivo, en mi caso he decidido enviarlo por correo electrónico por su sencillez y rapidez.

Antes de abrir el fichero “proyecto.apk” en el dispositivo, tenemos que activar el permiso para instalar aplicaciones que no sean descargadas de “Play Store”. Para activar esta opción abrimos Ajustes, pulsamos en el botón “Seguridad” y activamos la opción “Orígenes Desconocidos” como en la ilustración I y II

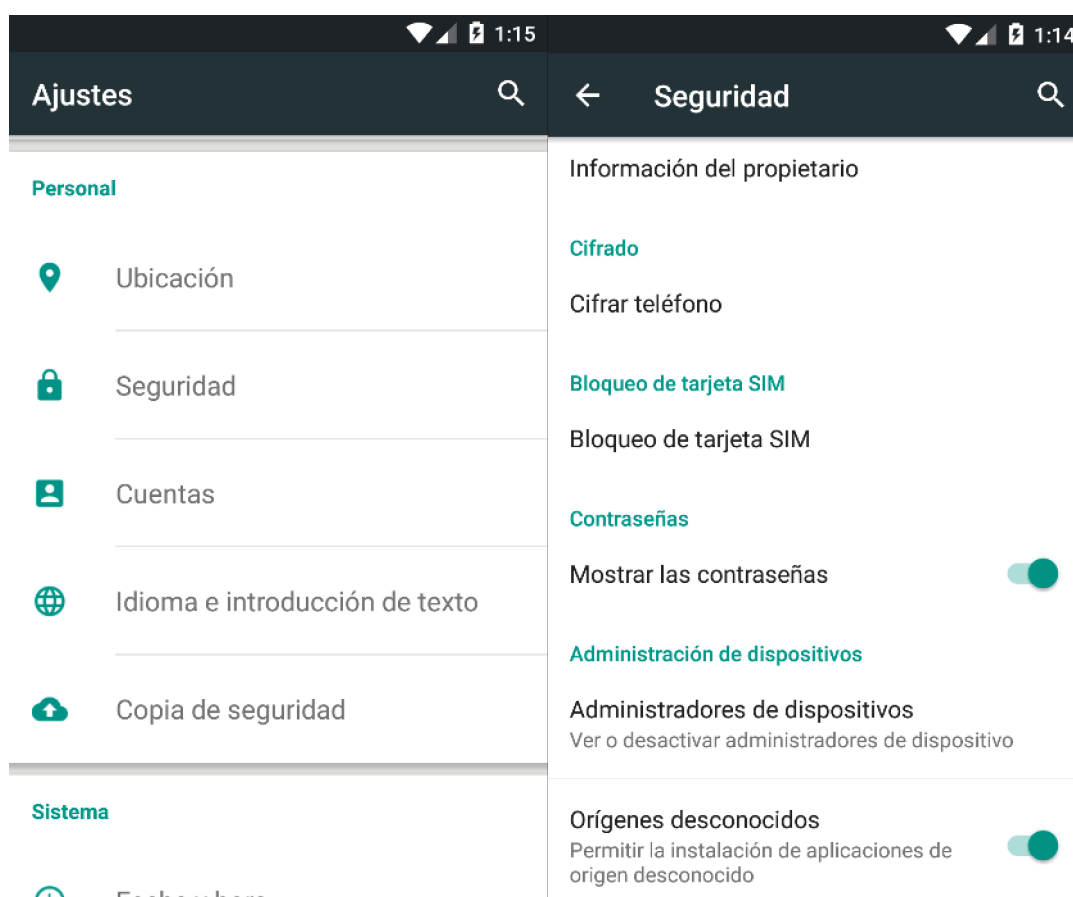


Ilustración I y II: Pantalla de Ajustes y Pantalla de Seguridad

Una vez activada podemos abrir el fichero y comenzar con la instalación.

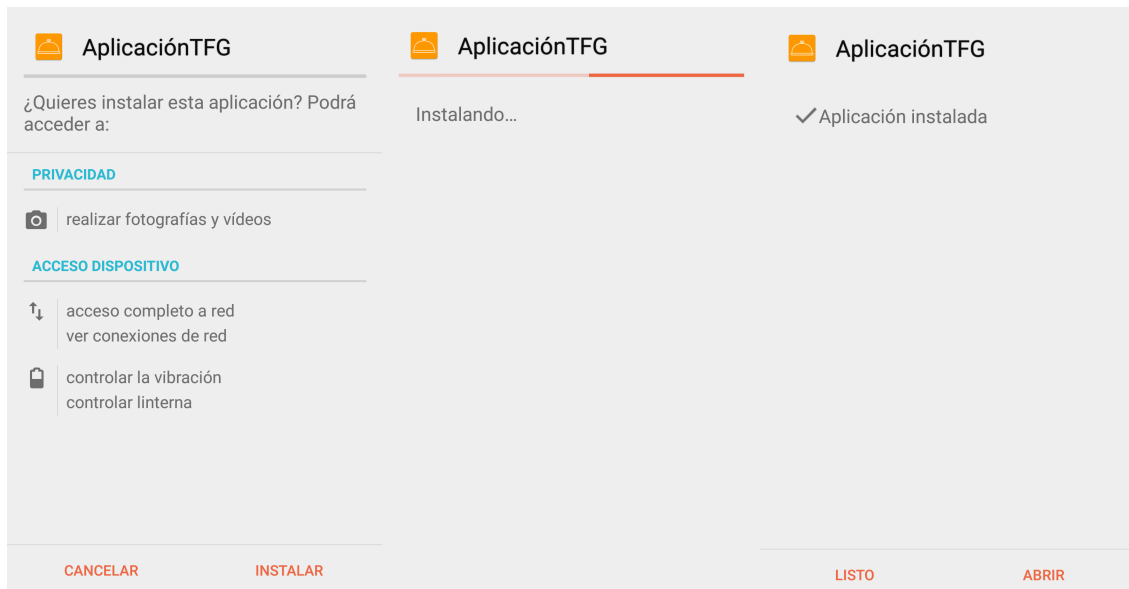


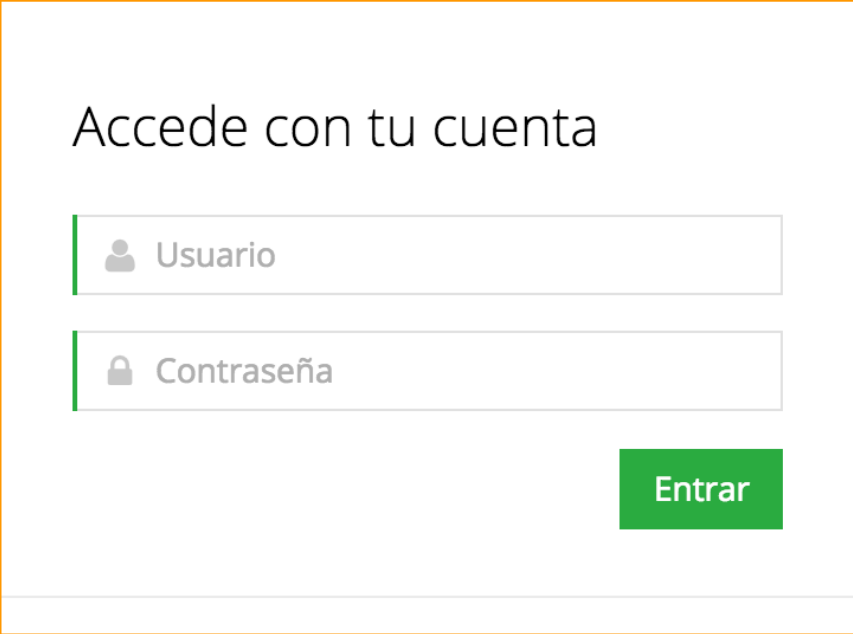
Ilustración III, IV y V: Proceso de instalación de la aplicación

Como podemos ver en las ilustraciones solo tenemos que darle a instalar para aceptar los permisos de la aplicación y tenerla disponible en el dispositivo. Una vez instalada podemos cerrar esta pantalla pulsando sobre “Listo” o podemos abrir la aplicación pulsando “Abrir”.

ANEXO 3

GUÍA DE USUARIO DEL CLIENTE WEB

En este anexo hablaremos de cómo se utiliza el cliente web, esta parte esta orientada para el establecimiento hostelero, los clientes del establecimiento no tendrán acceso a esta parte del proyecto.



Accede con tu cuenta

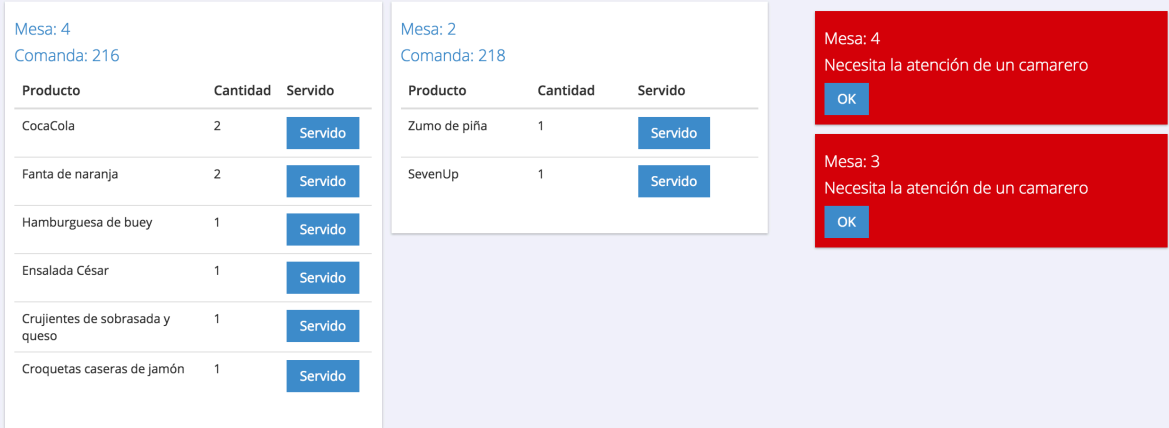
Usuario

Contraseña

Entrar

Ilustración I: Panel de acceso a la web

Lo primero que tenemos que hacer es acceder a la URL del servidor e identificarnos con el usuario y contraseña.



Mesa: 4
Comanda: 216

Producto	Cantidad	Servido
CocaCola	2	<button>Servido</button>
Fanta de naranja	2	<button>Servido</button>
Hamburguesa de buey	1	<button>Servido</button>
Ensalada César	1	<button>Servido</button>
Crujientes de sobrasada y queso	1	<button>Servido</button>
Croquetas caseras de jamón	1	<button>Servido</button>

Mesa: 2
Comanda: 218

Producto	Cantidad	Servido
Zumo de piña	1	<button>Servido</button>
SevenUp	1	<button>Servido</button>

Mesa: 4
Necesita la atención de un camarero
OK

Mesa: 3
Necesita la atención de un camarero
OK

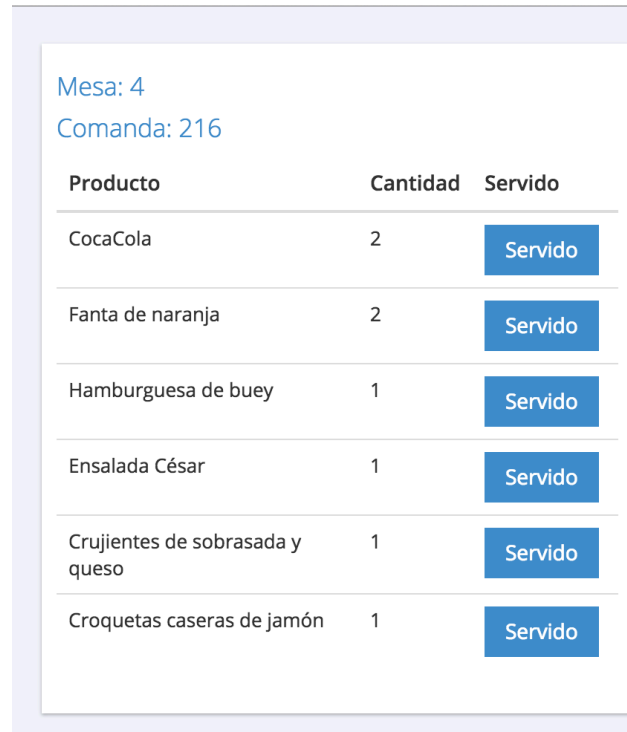
2015 © Trabajo de Fin de Grado hecho por Sergio Muriana.

Ilustración II: Panel de pedidos y notificaciones

Una vez puesto el usuario y contraseña accederemos al panel de pedidos y notificaciones.

El panel esta organizado en dos partes: Pedidos y Notificaciones.

- Pedidos



Mesa: 4 Comanda: 216		
Producto	Cantidad	Servido
CocaCola	2	<button>Servido</button>
Fanta de naranja	2	<button>Servido</button>
Hamburguesa de buey	1	<button>Servido</button>
Ensalada César	1	<button>Servido</button>
Crujientes de sobrasada y queso	1	<button>Servido</button>
Croquetas caseras de jamón	1	<button>Servido</button>

Ilustración III: Detalle de un pedido

Los pedidos se encuentran en la izquierda de la pantalla y se organizan en dos columnas de izquierda a derecha y de arriba abajo por orden de llegada.

Tienen fondo blanco y contiene la mesa que lo ha pedido, el número de comanda recibida en el establecimiento y una tabla con el nombre del producto, la cantidad y un botón para marcarlo como servido.

Cuando pulsamos el botón servido, se le notifica al servidor y se borra de la lista del pedido, una vez estén todos los productos servidos el pedido desaparecerá del panel.

- Notificaciones

Las notificaciones aparecen en el lado derecho del de la pantalla y se organizan en una columna de arriba abajo por orden de llegada.

Podemos diferenciar dos tipos de notificaciones, las notificaciones que solicitan la atención de un camarero en la mesa, donde nos mostrará la mesa que lo ha solicitado y un botón “OK” para dejar constancia en el servidor que se ha atendido a esa petición y para ocultar la notificación del panel.



Ilustración IV: Detalle de una notificación de llamada al camarero

El otro tipo de notificaciones son las solicitudes de pago, donde el camarero sabrá que mesa quiere pagar la cuenta y la cuantía de la cuenta desglosada. También incluye el botón “Pagado” para notificar al servidor que se ha pagado y para que se oculte la notificación del panel.



Mesa: 4		
Producto	Cantidad	Subtotal
CocaCola	3	4.5€
Fanta de naranja	2	4€
Hamburguesa de buey	1	9.95€
Ensalada César	1	4.5€
Crujientes de sobrasada y queso	1	3.95€
Croquetas caseras de jamón	1	4.5€
Rabo de Toro	1	9.95€
		45.35€
Pagado		

Ilustración IV: Detalle de una notificación de pago de cuenta

ANEXO 4

GUÍA DE USUARIO DEL CLIENTE ANDROID

En este anexo se explicará la utilización de la aplicación Android, esta aplicación esta orientada a los clientes del establecimiento.

Una vez instalada la aplicación (Manual en el Anexo II) vamos al menú de aplicaciones y abrimos “AplicaciónTFG”.

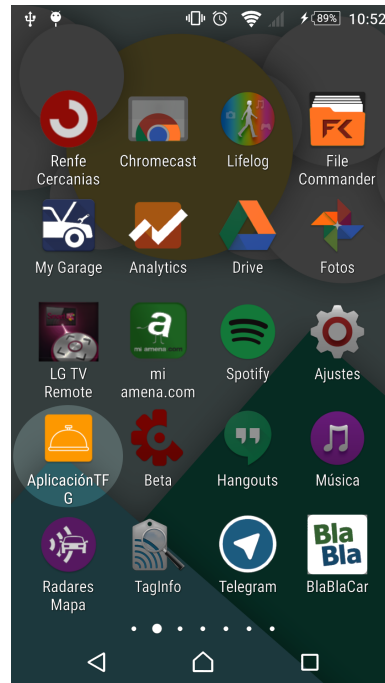


Ilustración I: Menú de aplicaciones

En la pantalla de inicio tendremos un botón para escanear el código QR en el centro de la pantalla y otro botón en la esquina superior izquierda para mostrar el menú de la aplicación.

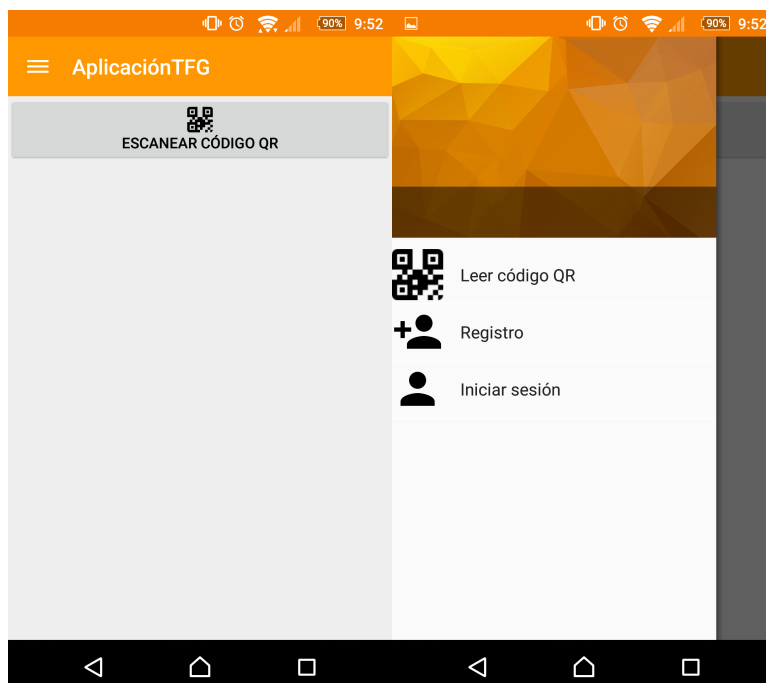


Ilustración II y III: Pantalla Inicial y menú de la aplicación

Si pulsamos sobre “Escanear Código QR” accederemos a la cámara, pero antes es necesario registrarnos en el caso de que no estemos registrados, para ello accedemos al menú y pulsamos sobre “Registro”.

Ilustración IV y V: Pantalla de Registro vacía y pantalla de registro rellena

En la pantalla de registro debemos de rellenar el nombre, apellidos, correo electrónico y contraseña. Una vez rellenados estos campos del formulario pulsamos sobre “Registro” y se enviará la información al servidor, en caso de que el correo

electrónico no este registrado por otro usuario nos llevará a la pantalla de inicio y en el menú aparecerá nuestro nombre y apellido.

En el caso de que ya estuviéramos registrado en el sistema omitiremos este paso e iremos directamente a “Iniciar Sesión” disponible en el menú de la ilustración III. Rellenamos los campos con nuestro correo electrónico y contraseña, a continuación pulsamos “Iniciar sesión” y en el caso de que estemos registrados y la contraseña sea correcta la aplicación nos llevará a la pantalla inicial.

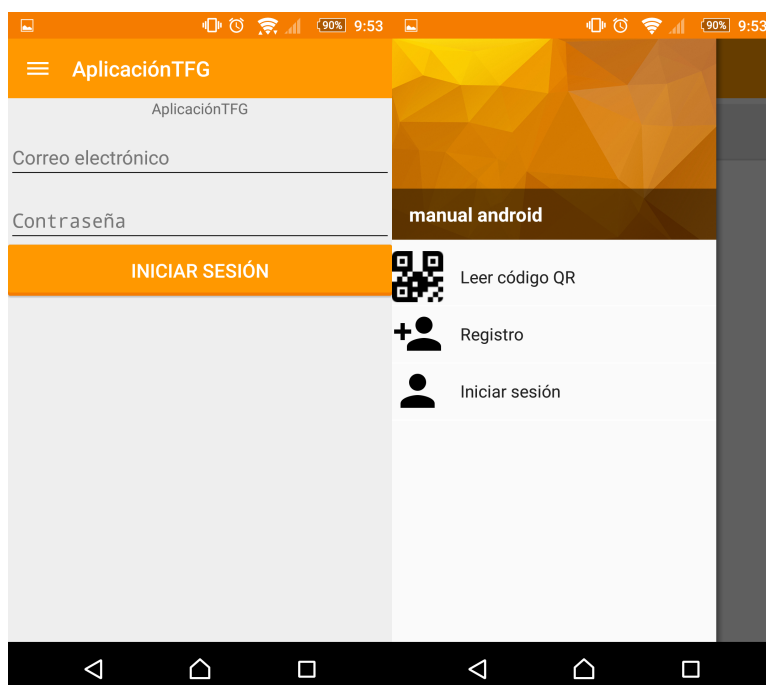


Ilustración VI y VII: Pantalla de inicio de sesión y menú con datos del usuario

Desde la pantalla inicial ya podremos pulsar sobre el botón “Escanear código QR” o visualizar el menú con nuestro nombre y apellidos.

Para escanear el código QR que se encuentre en la mesa pulsamos sobre el botón “Escanear código QR” y la aplicación nos mostrará la cámara frontal con el mensaje “Escanear código QR de la mesa” en la parte inferior, tendremos que apuntar con la cámara al código QR para que se quede en la parte central de la pantalla, una vez que la aplicación detecte el código QR automáticamente pasará a la pantalla de la carta.

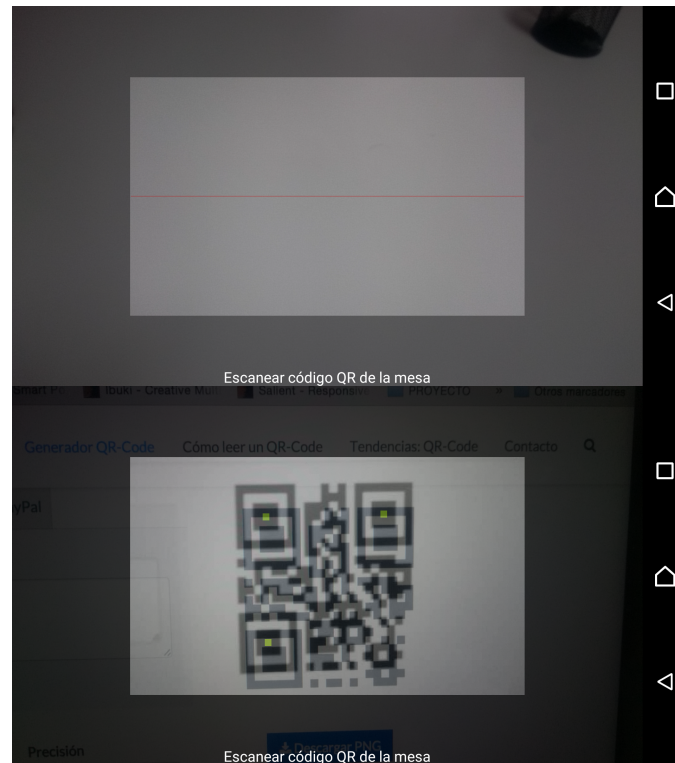


Ilustración VIII y IX: Pantalla con cámara para escanear QR y código QR centrado en la pantalla

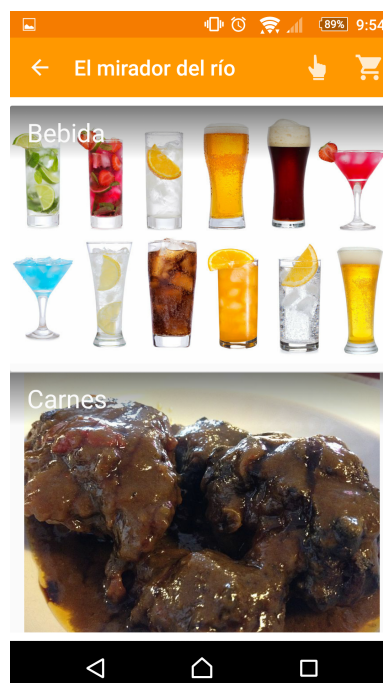


Ilustración X: Carta del establecimiento

Como podemos observar en la ilustración X en la pantalla de la carta del establecimiento tenemos un menú superior y una lista con las categorías en las que se organiza la carta. En el menú superior tenemos las opciones de volver a la

pantalla principal, llamar al camarero y el carrito de la compra, también nos mostrará el nombre del establecimiento.

Si accedemos a una categoría puedes ser que nos muestre subcategorías, donde la navegación es similar a la pantalla anterior como se puede observar en la ilustración XI o los productos de esa categoría como podemos ver en la ilustración XII, donde podremos ver el nombre del producto, el precio y un carrito al lado de cada producto para añadirlo a nuestro pedido. Si añadimos un producto al carrito podremos añadir o quitar unidades desde la misma pantalla como podemos ver en la ilustración XIII.

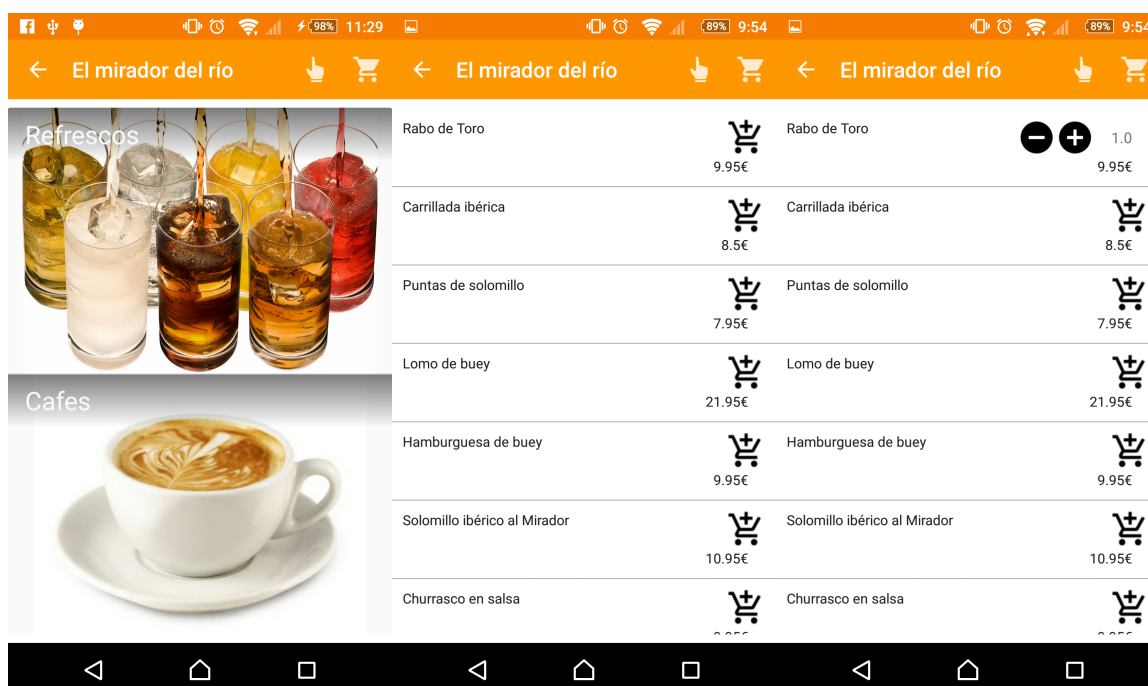


Ilustración XI, XII y XIII: Pantalla con subcategorías, pantalla con productos, pantalla con producto añadido

Una vez añadidos todos los productos que deseemos pedir pulsamos sobre el botón del carrito disponible en el menú superior.

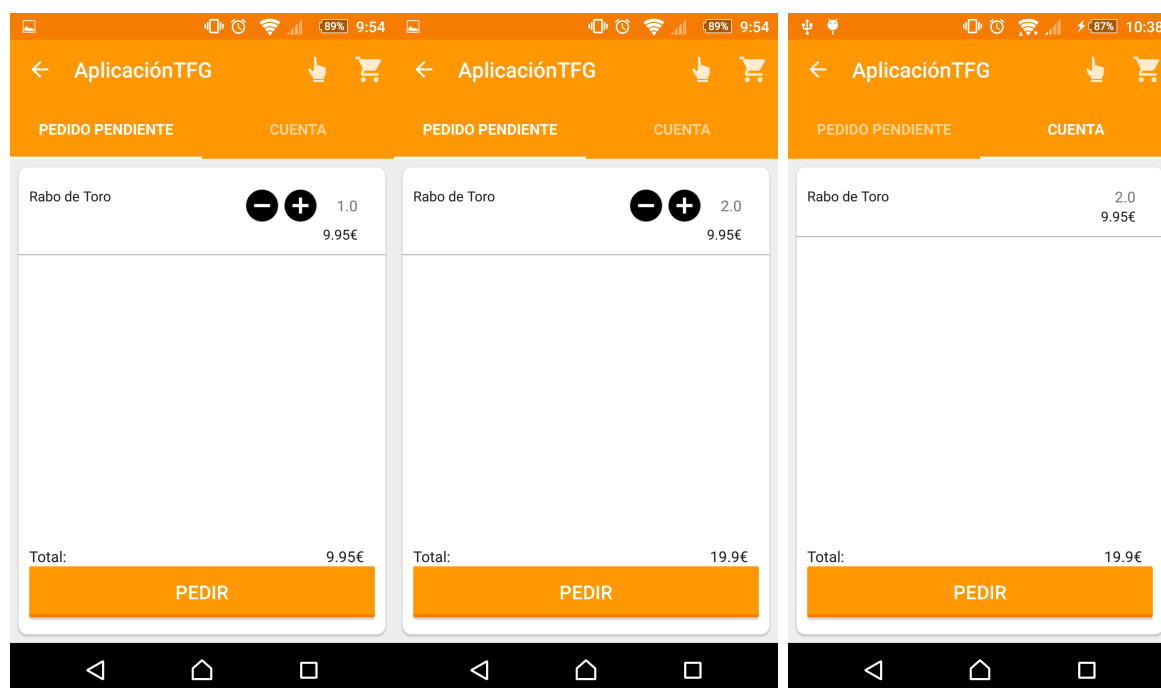


Ilustración XIV, XV y XVI: Pantalla de pedido pendiente, pantalla de pedido pendiente modificada y pantalla de cuenta

Al acceder al carrito nos mostrará dos pestañas “Pedido pendiente” y “Cuenta”, en el pedido pendiente tenemos los productos pendientes de pedir con la opción de añadir y quitar unidades, el coste unitario de cada producto, el coste total del pedido y el botón “Pedir”, si pulsamos el botón “Pedir” se notificará al establecimiento del pedido y procederán a prepararlo y servirlo en la mesa. En la pestaña “Cuenta” tenemos los productos que ya hemos pedido, las unidades y el coste unitario de cada producto, el coste total de la cuenta y el botón de “Pedir” para solicitar y cerrar la cuenta, si se pide la cuenta el establecimiento sabrá que desea pagar y le traerá la cuenta a la mesa.

La opción disponible de “Llamar al camarero” podremos utilizarla en cualquier parte de la aplicación una vez leído el código QR. Simplemente tenemos que pulsar sobre el icono de la mano disponible en el menú superior y confirmar que queremos la atención de un camarero en la mesa como podemos observar en la ilustración XVII.

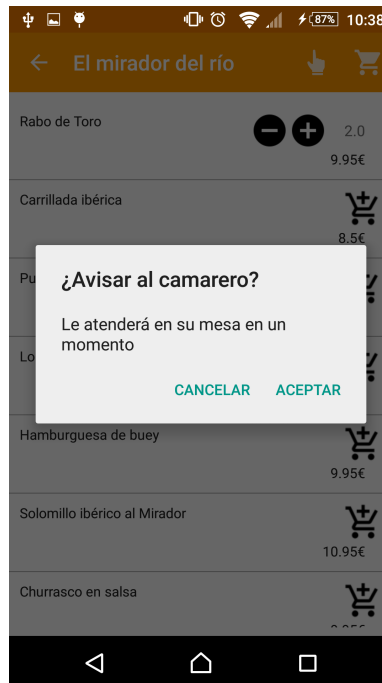


Ilustración XVII: Pantalla de confirmación para solicitar un camarero en la mesa.