GestioBar: Tu camarero Android

Nombre Estudiante: Jesús Ródenas Jaque
Máster Universitario en Desarrollo de Aplicaciones para Dispositivos Móviles

Nombre Consultor/a: Eduard Martín Lineros
Profesor/a responsable de la asignatura: Carles Garrigues Olivella

Fecha de Entrega: 03/01/2018

A Marta, mi motor y motivación para avanzar.
Gracias por apoyarme siempre, sin excepción.

FICHA DEL TRABAJO FINAL

Título del trabajo:	*GestioBar: Tu camarero Android*
Nombre del autor:	*Jesús Ródenas Jaque*
Nombre del consultor/a:	*Eduard Martín Lineros*
Nombre del PRA:	*Carles Garrigues Olivella*
Fecha de entrega (mm/aaaa):	01/2018
Titulación:	Máster Universitario en Desarrollo de Aplicaciones para Dispositivos Móviles
Idioma del trabajo:	*Castellano*
Palabras clave	*Bar, Camarero, Hostelería*

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

Como finalidad principal de este TFM se ha marcado la obtención de un producto principalmente transversal. De esta forma se quiere conseguir una herramienta que sirva para integrar la gestión común de los servicios de hostelería y colocarla al alcance de cualquier usuario con un smartphone o tablet. A día de hoy el sector de la hostelería es un sector cuya oferta de soluciones tecnológicas de gestión está fundamentada en herramientas realizadas a la medida del cliente en cuanto a software y hardware. Este aspecto es un factor que incrementa de manera notable el presupuesto destinado a a adquisición de estas herramientas. Es por tanto que, buscando la adaptabilidad de la aplicación, se pretende obtener una solución genérica que cubra las necesidades comunes de estos establecimientos, dejando las herramientas a medidas a realidades más complejas. De esta forma se reducirían gastos para aquellos hosteleros cuya gestión de negocio no implica una especialización en su herramienta de gestión: Las PDA's específicas se podrían sustituir por smartphones de los empleados, o por una batería de dispositivos de gama media/baja (es una aplicación con no muchos requisitos) conectados a una red wifi que además centralicen su información bajo la supervisión del gestor del establecimiento.

La verdadera potencia de esta aplicación radicará en la definición de un conjunto de funcionalidades para la gestión del establecimiento: La aplicación deberá proveer de las herramientas necesarias para permitir estructurar la información que dará soporte a las sucesivas descargas de la app por parte de los empleados y que estos compartan el mismo contexto de trabajo:

Estas sucesivas descargas posteriores deberán incorporar a sus sistemas la información definida por el usuario encargado de la gestión

Abstract (in English, 250 words or less):

The main aim of this TFM is to get a product mainly transversal. In this way, it is wanted to get a software tool which can integrate the comun gestion of hostelry services and place it at the service of any user with a smartphone or tablet.

Nowadays, hostelry sector has a offer of technological management solutions based on tools according to customer needs regarding software and hardware. For this reason, the Budget of these software tools is increased. So, looking for the adaptability of this application, it is intended to provide a general solution in order to satisfy the comun needs of these establishments, without considering particular situations which will need another kind of application more specific.

This application would reduce costs in those hoteliers whose business management don't imply specialization in its management tool: Specific PDAs would be able to be replaced by smartphones that belongs to employees or connected devices to the same WiFi.Also these devices could be in the low-mid range because this application has few requirements. Application is able to centralize the infomation and to make available to the manager's supervision.

The real power of this application will be a group of functionalities to the gestion of establishment: the application will be able to provide the necessary software tools to allow structuring the information that will give support to the downloads that employees will make. With this technique, the employees will share the same work that the manager has previously defined.

Índice

1. Introducción

1.1 Contexto y justificación del Trabajo

El proyecto surge desde un caso real: La existencia de hosteleros que, con la finalidad de mejorar sus procesos productivos, han optado por implantar un sistema de gestión para agilizar el despacho de sus comandas.

Sin embargo, y pese a que la gestión de los servicios de hostelería no ofrece grandes particularidades entre diferentes establecimientos, la solución más extendida es la de desarrollar software a medida con el fin de automatizar la gestión que realizan sus trabajadores.

Este es un aspecto que, obviamente, supone un incremento en los costes de desarrollo.

En la actualidad existen soluciones específicas tanto para la plataforma iOS como para Android. Partiendo de que el alcance de la aplicación está pensado para abarcar a la plataforma Android (aspecto que se explica más en detalle en capítulos posteriores), el estudio de mercado debería ceñirse a la plataforma de Google. Lo cual no impide que un análisis global de las soluciones existentes en el mercado pueda desembocar en la captación de buenas ideas, en depurar algunas que no teníamos claras y, por supuesto, en intentar predecir cuál es el calado y nicho de mercado que puede tener una aplicación de las características de la que se va a desarrollar.

En el mercado existen diversas aplicaciones, algunas android como:

- Restaurant Waiter (1) : Esta aplicación pretende simular la gestión de un camarero en un bar. Al descargarla se observa una navegación poco intuitiva, una organización de los datos poco usable y la imposibilidad de compartir datos de soporte como pretende realizar la aplicación GestioBar. Sin embargo la calidad de la interfaz puede dar una idea de cómo afrontar la de la aplicación de la que es objeto este TFM. Quizá refinar el estilo pero puede servir para captar ideas.

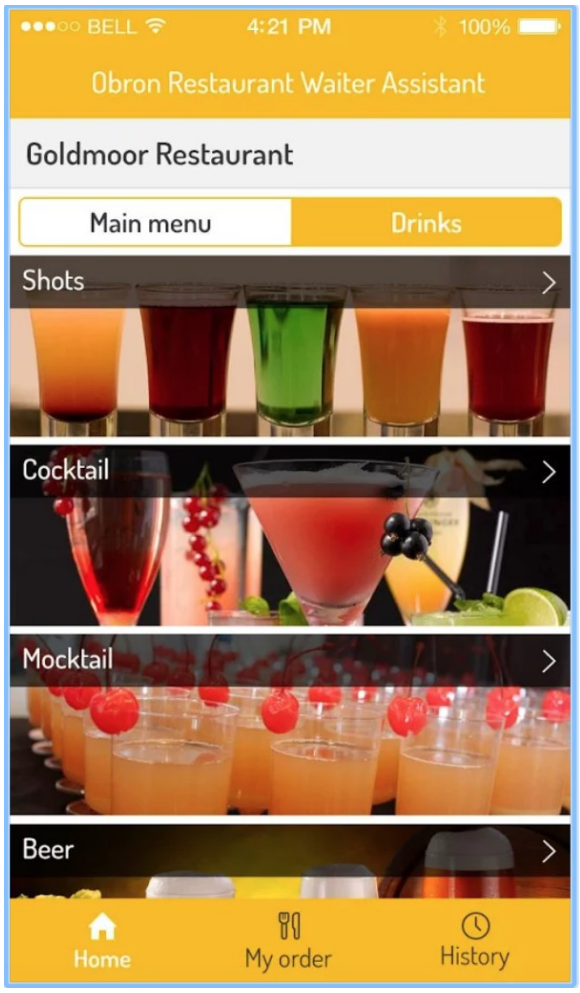


Figura 1. Tipos de bebidas de la app Resturant Waiter

También existen soluciones multiplataforma: adaptadas a dispositivos Apple:

- Cuiner (2): Salta a la vista la profesionalidad de este sistema. La experiencia no sólo les avala, sino que ofrecen un producto de calidad con un aspecto difícilmente mejorable. Gran ejemplo a seguir en cuanto a procedimientos de ingeniería del software para conseguir una solución usable y transversal. No existe para plataformas móviles con sistemas Android: Podría ser una opción de futuro como proyecto de integración.

Figura 2. La web de cuiner.com

Todos son buenos ejemplos para analizar el nivel de profundidad del producto en el mercado o las estrategias que plantean para alcanzar el producto final.

1.2 Objetivos del Trabajo

El objetivo prioritario de este proyecto es la consecución de una aplicación eficaz e intuitiva que facilite la gestión de un negocio de hotelería. Para ello en primer lugar deberá definirse un perfil de usuario encargado de la gestión del establecimiento y especificación de la información sobre la que se va a trabajar. De aquí en adelante se hablará de perfil de "Gestor" al usuario que adopte este rol dentro del establecimiento. Existirá, en consecuencia, un perfil para empleados cuya primera acción con la aplicación será la de incorporar la información correcta. Esta información habrá sido definida previamente por el "Gestor" y será incorporada a sus dispositivos. La aplicación perseguirá aportar las herramientas necesarias para hacer posible la propagación e intercambio de la información preparada desde el perfil "Gestor" del sistema hasta las aplicaciones relacionadas (los empleados del establecimiento).

Tras descargar la aplicación podrá definirse la información necesaria para la gestión que harán los empleados del establecimiento. La aplicación dará el soporte para generar la estructura de la información desde el perfil "Gestor" del establecimiento, así como para hacer posible su incorporación a un dispositivo relacionado: A través del perfil "Gestor" se definirá, sobre el modelo de datos genérico y transversal a todas las aplicaciones, el contenido específico consistente en todos los distintos elementos que formarán parte de gestión propia de la que será objeto el establecimiento en cuestión. A partir de este punto, las aplicaciones descargadas en los dispositivos de los empleados podrán incorporar esa información a sus herramientas de gestión.

Estableciendo un paralelismo entre hitos y objetivos, se podrían enumerar como objetivos funcionales:

– **Definición de un modelo de datos flexible y lo más genérico posible** de manera que se aleje de la realidad de un único servicio de hostelería y se acerque a cubrir el mayor abanico de casos. Se perseguirá la definición de lo que en ingeniería del software se conoce como "Metamodelo": Se generará un modelo de datos que permitirá la generación de distintos modelos para cada uno de los establecimientos que descargue la aplicación.

– **Establecimiento de la gestión y permisos de los distintos perfiles de acceso a la aplicación**. En este caso dos: perfil de normalización de los datos (perfil "Gestor") para dar soporte a los elementos de la estructura de la información, y un perfil "Empleado" cuya

gestión sea la de registrar pedidos sobre la estructura de datos definida anteriormente.

– **Conexión con back-end firebase**: La aplicación se integrará con la solución transversal “firebase” para almacenar los datos del modelo de la aplicación.

– **La aplicación tendrá que ser capaz de almacenar en firebase las entidades que se registrarán en la gestión del establecimiento**: En la primera descarga por parte del establecimiento y, a través del perfil de “Gestor” del servicio, se registrarán los datos en cuanto a las entidades cuya gestión será el objeto de la aplicación.

– **La plataforma aportará las herramientas necesarias** para permitir al “Gestor” de la aplicación **la definición** de manera intuitiva de los distintos elementos que darán el **soporte de datos** para la gestión de los empleados.

– En consecuencia con el objetivo anterior, la aplicación ofrecerá un **front-end usable para la definición de los datos que se van a gestionar por parte de los empleados del establecimiento.**

– **Definir el protocolo de autenticación** de los usuarios empleados del establecimiento. Será necesario discriminar la información que le corresponde a un establecimiento entre toda la almacenada en firebase. El diseño de la aplicación y su implementación debe servir de herramienta para incorporar la información correspondiente al usuario que accede a la aplicación. Como propuesta inicial se usará un token identificador, se valorará la idoneidad de una lectura de código QR.

– De la misma forma, la aplicación ofrecerá un **front-end usable para la gestión propia del establecimiento**, permitiendo el **registro de distintos pedidos y los importes de los servicios**.

Desde este planteamiento, se podrían concluir una serie de requisitos no funcionales tales como:

– Se tratará de obtener una aplicación usable, ya que desde el perfil “Gestor” se producirá una interacción durante períodos extensos de tiempo. En caso del usuario empleado también será necesario adecuar la usabilidad al acceso que efectuará el usuario empleado a la aplicación: éste será en numerosas ocasiones y durante menos tiempo continuado. Será crucial definir interfaces amigables para cada una de las situaciones.

– Se tratará de obtener un producto **transversal**, de manera que una solución genérica sea capaz de solucionar distintas realidades de establecimientos**.**

– **Como consecuencia de la integración con firebase** obtendremos una aplicación fácilmente migrable, ya que el soporte de la información de todos los establecimientos estará en un punto común externo, lo que nos permitiría cambiar el front-end de forma independiente a la implementación del back-end, ya que éste lo delegaremos a firebase.

1.3 Enfoque y método seguido

Entre las distintas posibilidades existentes a la hora de decidir qué tecnología a adoptar de cara a emprender cualquier desarrollo para aplicaciones móviles, me he decantado por la plataforma Android. Principalmente se ha optado por la plataforma de Google por un motivo: La difícil compatibilización de la vida laboral y el estudio en estos meses que vienen de aquí a final de año. Se me exigirá una implicación personal en la implantación de un nuevo proyecto, por lo que se hará ciertamente complicado poder afrontar un desarrollo basado en una tecnología de desarrollo cuya curva de aprendizaje suponga un obstáculo temporal para la consecución de los objetivos marcados en la planificación del proyecto.

Es innegable la potencia del desarrollo híbrido, máxime para software enfocado a la gestión. Ionic2 surge como principal ejemplo: Una tecnología que permite reutilizar desarrollos para distintas plataformas, ofreciendo así un gran atractivo a la hora de establecer un perfil de gestión que sea accesible desde la web, que para nuestra aplicación sería el usuario con perfil “Gestor”, cuya función sería la de normalizar la información del establecimiento de cara a incorporar eso datos para el trabajo de los usuarios con perfil de empleado. Como hándicap, Ionic2 presenta inestabilidad en la interfaz de alguno de sus componentes, y ofrece cierta dependencia del navegador web para su visualización. Dependencia que al tratarse de un cliente final de escaso nivel en informática, puede llevar a provocar problemas de mantenibilidad. Además, para emprender un desarrollo en Ionic2 tendría que mejorar mis capacidades sobre esta tecnología en primer lugar.

También Swift para dispositivos iOS ofrece una calidad en el producto final difícilmente superable por otras plataformas, no en vano muchos desarrolladores profesionales consideran la plataforma de Apple como único destino de sus desarrollos. Una interfaz atractiva avalaría sin duda optar por esta decisión. Como factor en contra se detecta rápidamente que el uso de estas aplicaciones implica un sobre coste en el dispositivo. En este aspecto Apple y Android no pueden competir: La marca de Google abarca un espectro de dispositivos (y de bolsillos) muchos más amplio. También he tenido en cuenta que la realidad dice que voy a compatibilizar el desarrollo de este TFM con el de la asignatura de especialización en iOS, por lo que se puede deducir que tras cursar una asignatura de iniciación, no alcanzo los conocimientos mínimos para el desarrollo de un producto de calidad. Lo dejaré en el debe de este TFM con vistas a una versión mejorada a futuro de la aplicación.

Es por tanto que, basándome en una curva de aprendizaje más liviana, en una calidad del producto final de un nivel notable y una accesibilidad al mercado en porcentajes superiores a los de cualquier otra plataforma, por lo que la decisión tomada en cuanto a la tecnología de desarrollo para este TFM sea la tecnología Android.

Como idea de punto de partida del producto optamos por iniciar un producto nuevo debido a la flexibilidad de la que se quiere dotar al sistema. Desde la normalización de la información de un sólo establecimiento, hasta la definición de una estructura de información que de soporte a todo el espectro de establecimientos que pueden llegar descargarla, muchos aspectos implican realizar un desarrollo desde cero de manera que se consiga autosuficiente. Mediante una adaptación, sería necesario conocer con mucha profundidad y nivel de detalle la base de otra aplicación de terceros además de, por supuesto, disponer de la posibilidad de ampliarla. En cualquier caso, y tratándose de una aplicación ideada desde cero, creo que iniciando todo el proceso del ciclo del software desde cero me veo capaz de alcanzar a cubrir todos los objetivos funcionales y no funcionales definidos anteriormente.

La metodología de desarrollo a seguir vendrá marcada por los hitos propios que marcarán las PEC's del TFM y se verá más en detalle en la planificación temporal en el epígrafe a continuación.

De la manera que está planteada la asignatura del TFM, se podría desprender que el uso de una de las metodologías ágiles más populares, como *Scrum*, se adapta de forma ideal.

Scrum (3) basa su desarrollo en una generación incremental del producto consistente en una serie entregables al cliente con cada uno de ellos. Cierto es que su máxima expresión surge en proyectos colaborativos con posibilidades de kanbanizar (4) el proyecto. De todas formas, en este caso se podría considerar óptimo, ya que los entregables serían las propias PEC's de la asignatura, y el *cliente* encajaría con el rol del consultor del TMF.

1.4 Planificación del Trabajo

Para la el desarrollo del TFM contaremos con los siguientes recursos materiales y humanos:

- Como equipo humano se contará con el autor del proyecto para realizar las labores de análisis y desarrollo. También se contará con el mismo recurso para las pruebas y se añadirá al consultor del TFM Eduard Marín Lineros como testeador final del producto y evaluador de la documentación.

- Como recursos técnicos se contará con:
  - Portátil MacBook Pro con macOS Sierra 10.12.6

- Entorno de desarrollo Android Studio actualizado.
- Smartphone Xiaomi Redmi pro S3.
- Base de datos documental Firebase.
- Cuenta de Google como desarrollador de Android.
- Repositorio de versiones GitLab y cuenta de usuario.
- OpenOffice para la elaboración de documentación.
- https://app.smartsheet.com/ para la gestión de los hitos del proyecto y la planificación temporal.
- Gimp para la edición de imágenes.

- Se ha establecido una lista de tareas estableciendo un cruce entre los hitos propios en el desarrollo de la asignatura y los objetivos perseguidos para la aplicación.

  La estimación horaria de nuevo se ha basado en la alta implicación personal que tengo actualmente en mi proyecto laboral, por lo que intentando ser realista he tomado como medida válida la siguiente: Se contabilizará una hora diaria de TFM en días laborables y 3 en días festivos. La razón de esta estimación a la baja de los días festivos viene motivada a que: quizá un festivo finalmente puedan ser 7 horas de trabajo, pero habrá días laborables en lo que no se alcanzará el cumplimiento de la hora de trabajo. Dicho esto, los hitos y planificación de los mismos están plasmados en la siguiente cronología y diagrama de Gantt:

N horas	Nombre de la tarea	Fecha de Inicio	Fecha final
36	**PEC1: Plan de trabajo**	**18/09/17**	**10/10/17**
10	Contexto y justificación del trabajo	18/09/17	24/09/17
10	Objetivos del trabajo	24/09/17	01/10/17
8	Enfoque y método elegido	01/10/17	06/10/17
7	Planificación del trabajo	07/10/17	09/10/17
1	Sumario productos obtenidos	10/10/17	10/10/17
34	**PEC2: Diseño**	**11/10/17**	**30/10/17**
7	Usuarios	11/10/17	14/10/17
4	Análisis, recogida de requisitos	11/10/17	12/10/17
3	Contextualización y casos de uso	13/10/17	14/10/17
27	Diseño	15/10/17	30/10/17
7	Flujos de interacción del sistema	15/10/17	18/10/17
6	Modelo de datos conceptual	19/10/17	22/10/17
6	Diseño de la arquitectura	23/10/17	26/10/17
8	Prototipo de alto nivel	27/10/17	30/10/17
62	**PEC 3: Implementación**	**02/11/17**	**10/12/17**
8	Integración con Firebase	02/11/17	05/11/17
18	Perfil "Gestor"	06/11/17	18/11/17
16	Perfil "Empleado"	18/11/17	27/11/17
20	Plan de pruebas	28/11/17	10/12/17
24	**PEC 4: Entrega**	**14/12/17**	**02/01/18**
16	Finalización y corrección de la documentación	14/12/17	23/12/17
8	Elaboración de la presentación	26/12/17	02/01/18

Figura 3. Diagrama de Gantt del TFM

Se han eliminado del cronograma los días 24, 25 y 30 de Diciembre, también el 1 de Enero al considerarlos días no válidos para la realización del TFM.

En cuanto al desglose de la fase de implementación (PEC 3) quizá se ha pecado de ser poco profuso, pero se da por supuesto que implica desarrollo de parte de comunicación con Back-End y desarrollo del Front-End, entre otros aspectos.

Cabe apuntar que en la iteración correspondiente a la PEC 4, la elaboración de la presentación debe servir de base para la generación del manual de usuario de la aplicación. De esta forma, las ilustraciones e

ideas de la misma, deben desprender la idea para la definición del manejo de la aplicación y sirva, además, para generar este manual que ha de servir al usuario final como guía de uso para sacar el máximo partido a la aplicación final.

Se anexa a esta documentación el diagrama de Gantt correspondiente a dicha planificación y temporalización de hitos.

1.5 Breve sumario de productos obtenidos

Aplicación: Será el producto final a entregar. El instalable en un dispositivo con las funcionalidades definidas en este documento.

Código fuente de la aplicación: Junto con la aplicación, también irá el código fuente debidamente comentado de manera que sea fácilmente legible para un evaluador, y mantenible para otro desarrollador.

Presentación del TFM: Se realizará de cara a exponer en qué ha consistido el desarrollo del TFM, las fases y vicisitudes por las que se ha pasado y las previsiones que se han cumplido y las que no

Vídeo de presentación: Se elaborará un vídeo que de relieve a la presentación comentada anteriormente, en la que se exponga de manera dinámica y audiovisual la presentación del TFM.

Manual de usuario: A partir del material generado para la presentación del TFM, paralelamente se confeccionará un manual de uso para usuarios con escaso nivel en el manejo de aplicaciones móviles: Atractivo, sencillo y enfocado a objetivos de uso.

Memoria del TFM: Contendrá el contenido de las distintas PEC's que se vayan realizando en cuanto al TFM durante el semestre y concluirá con los hitos definidos en la planificación.

1.6 Breve descripción de los otros capítulos de la memoria

Dentro del análisis, estudio y desarrollo que supondrá este TFM, se destacan otros capítulos a lo largo de la siguiente memoria:

- Usuarios y contexto de uso: Al tratarse de una app destinada al sector servicios y a ser usada por usuarios de distintos tipos de preparación, será fundamental un análisis pormenorizado de los destinatarios potenciales de la misma. De esta forma se conseguirá una mayor capacidad de adaptación.

- Diseño conceptual: A partir del análisis realizado para los usuarios y los contextos de uso, se definirán una serie de situaciones sobre las que vertebrar una línea de trabajo y el planteamiento a llevar a cabo para el desarrollo de la aplicación en sí.

- Prototipado: Imprescindible para la vertebración antes comentada la realización del árbol de navegación de la aplicación y los prototipos de pantallas de la aplicación. Mediante esta técnica se consiguen depurar muchos problemas de usabilidad y amigabilidad de la aplicación, de forma que se consigue un producto mucho más profesional en cuanto al acercamiento al usuario final. Es en este punto donde cobrará mucho más peso semántico la separación que hace GestioBar de los perfiles que intervienen en su flujo de interacción, de manera que se harán mucho más visibles los motivos por los que se definen estos dos actores.

- Diseño de la arquitectura: Mostrará desde el plano técnico cuáles son las soluciones por las que se ha optado en cuanto al desarrollo de la aplicación. Para esta en concreto, basaremos nuestro diseño de la arquitectura en los siguientes aspectos básicos:
  - Diagrama de base de datos: Se mostrará un diseño de base de datos que servirá de justificación conceptual para la estructura de la información que se almacenará en nuestra base de datos NoSQL. En otras palabras, pese a no tratarse de tablas de bases de datos propiamente dichas, sí que nos basaremos en las relaciones entre entidades para estructurar la información que llegará a Firebase.
  - Arquitectura del sistema: Se presentará y argumentará la decisión técnica por la que se ha optado para la realización de la aplicación. En el caso de GestioBar se ha optado por la robustez y mantenibilidad que ofrece el patrón Modelo-Vista-Controlador. Además, se establecerá un paralelismo en cuanto a la estructura de clases Java generadas y la elección del paradigma de desarrollo a llevar a cabo.
  - Arquitectura de clases del sistema: Se presentarán las distintas clases que compondrán las capas que formarán parte de nuestro patrón de desarrollo: Desde las clases destinadas a la renderización de las vistas (Activities) hasta las clases transversales que mantendrán la información del usuario en vuelo en la aplicación y las distintas interacciones que vaya realizando con GestioBar, así como los elementos destinados a la comunicación con el backend.

- Plan de pruebas: Se establecerán las pruebas que servirán para testear la amigabilidad de la navegación. Para ello se ha optado por una serie de funciones de callback que servirán para el ajuste de las vistas tras la información obtenida (setUpView). Otro aspecto fundamental para la evaluación de la viabilidad de la aplicación serán las distintas interacciones que se producen contra firebase, tanto con wifi como con datos 3G/4G.

- Conclusiones y líneas de acción para la mejora en futuro.

2. Usuarios y contexto de uso

En este apartado se analizarán los usuarios potenciales de la plataforma en base a la gestión actual que se hace del sector destino del aplicativo. Se establecerán como consecuencia de la evolución histórica de las herramientas de gestión para los servicios de hostelerías y se plantearán distintos escenarios de uso para este tipo de sistemas de gestión. Se justificará cómo han desembocado los métodos más rudimentarios en la generación de sistemas de información que faciliten el trabajo de las personas. Además, se intentarán identificar los usuarios en base a unos paradigmas que se plantearán en los siguientes epígrafes. A partir de éstos, y como consecuencia de los mismos, se establecerán como consecuencia los distintos perfiles que interactuarán con la aplicación final que se persigue obtener.

2.1 Evolución histórica

Inicialmente, como en cualquier establecimiento del sector servicios, la contabilidad y movimientos económicos derivados de su actividad se efectuaba y consolidaba mediante los medios coetáneos vigentes.

Se contabilizaba de manera disgregada la actividad de cada camarero, llegando a un punto común en formato papel en el que se efectuaba una contabilización general del establecimiento. Posteriormente, se pasaba a registrar esta información en un formato físico. De esta forma se podían explotar estos datos en forma de resultados globales. Esta explotación en ocasiones derivaba en conclusiones de negocio para evaluar la viabilidad del mismo, o bien se trataba simplemente de una herramienta para su contabilidad.

Una vez la era digital alcanzó la gestión de este tipo de negocios, se mantenía la gestión disgregada de cada uno de sus empleados, cambiando en este momento el punto de información común del soporte papel al soporte digital. Es decir, cada uno de los camareros gestionaba de manera independiente su comanda mediante libreta o similar, pasando posteriormente a forma parte esta información del punto común en el medio establecido, bien fuera un pc, bien fuera un panel táctil.

Y a pesar de todas estas herramientas, incluso la llegada de formatos digitales a la gestión, era un sector que no eludía la pérdida de datos propia del trasvase de la información del individuo al colectivo, del humano a la máquina y de un formato a otro.

En pleno apogeo de la era de los dispositivos móviles, no podía escaparse a su alcance una gestión tan móvil como la de un camarero en un bar, por lo que pensar en el smartphone o la tablet como herramientas de este trabajo, es una conclusión ineludible y un destino que ha de mejorar la gestión propia del negocio.

2.2 Objetivos que se pretenden conseguir para los usuarios

Los objetivos que se definieron a nivel sistema de información y aplicativo, se añaden a los objetivos que perseguirá cubrir la plataforma de cara al usuario. Principalmente serán objetivos enfocados a la estructuración de la información, así como la gestión de la misma.

La aplicación deberá cubrir las necesidades que surgen de la gestión del establecimiento tanto en cuanto quiera ofrecer una funcionalidad a sus trabajadores. Esto quiere decir que existirá un tipo de usuario potencial de la aplicación cuyo uso derivará en conformar y organizar información que usará otro tipo de usuario, éste encarnado en el empleado del establecimiento.

Se desprende rápidamente del diseño centrado en el usuario que existirán dos tipos de usuarios de la aplicación, el encargado o regente del establecimiento, cuya gestión en cuanto a la aplicación consistirá en la gestión de la información. Registrar categorías de productos, registrar los propios productos, gestionar la información de los empleados, la cantidad de los mismos, etc.

Por otro lado existirá otro tipo de usuario de la aplicación, que será el empleado del establecimiento, cuya gestión en la aplicación consistirá en registrar comandas en la aplicación, estado de las mismas (abierta, cerrada) y toda esta gestión en función de la información estructurada anteriormente.

3. Diseño conceptual

3.1 Escenarios de uso

Me basaré en lo expuesto anteriormente para hacer desprender el diseño conceptual a partir de lo analizado y basar este en una serie de “Fichas de personas” y “Escenarios de uso” de la aplicación. Las fichas han sido creadas en base a la experiencia de conocidos, necesidades observadas en el día a día y en experiencias reales:

<u>Ficha de persona 1:</u>
Nombre: Santiago
Edad: 34
Profesión: Regente de una abacería
Descripción: Santiago regenta desde hace un año y medio un joven negocio en su ciudad natal. Los gastos que conlleva empezar en cualquier negocio han hecho que tras un año ya sepa dónde debe guardar y dónde debe gastar para intentar mantener la línea de ingresos ascendente que afortunadamente está llevando. Decide incorporar un sistema de información digital a su gestión debido a que la oscuridad en el despacho de comandas ha sido detectada como factor de mejora en su gestión.

Escenario de uso 1:
Necesita una herramienta útil para la toma de las comandas y la recepción de éstas en la barra del bar. Decide incorporar una herramienta a través de la cual sus empleados marcan las comandas a través de su smartphone, en presencia de sus clientes, y van entrando en el panel central para su despacho en barra o cocina. Todo es visible para sus clientes y genera un clima de confianza entre establecimiento y clientela.

Ficha de persona 2:
Nombre: María
Edad: 26
Profesión: Camarera
Descripción: Ana acaba de empezar a trabajar en un bar por primera vez, y a la falta de hábito en la profesión se suma su mala memoria. Debe ir siempre con la carta en la mano mientras redacta los productos a los clientes. Sus constantes olvidos e inseguridad están suponiendo un factor diferencial negativo del establecimiento a la hora de elegir un lugar para tomar algo en reunión de amigos.

Escenario de uso 2:
Tras la instalación de *GestioBar* en el bar que trabaja, no tiene que sacar más que su smartphone para recordar rápidamente a sus clientes cuál es la carta. De esta forma puede hacer sus propias recomendaciones o las del establecimiento y, en definitiva, consigue que la información fluya de manera más armónica y eficiente desde el cliente hasta la barra.

Ficha de persona 3:
Nombre: Pablo
Edad: 45
Profesión: Regente de Bar de tapas
Descripción: Pablo posee un bar cuyos veladores se encuentran al otro lado de la calle. Sus empleados necesitan cruzar el paso de peatones para hacer llegar las comandas a la barra, por lo que suelen tomar varias comandas y llevarlas de una vez. Estos tiempos, unidos a los de cruzar la calle hacen que determinados clientes más exigentes hayan hecho expresar en más de una ocasión por la lentitud de la cocina. Lentitud que no es tal, sino que se debe al cúmulo de tiempos de retraso en cada uno de los puntos en los que se toma la comanda en dicho este establecimiento.

Escenario de uso 3:
Los camareros que trabajan en el establecimiento de Pablo, ya tiene *GestioBar* instalado en sus smartphones, por lo que primero toman varias comandas haciéndolas llegar a barra, de forma que es barra la

que solicita a sus empleados vayan a recoger las comandas a servir una vez estén preparadas.

Ficha de persona 4:
Nombre: Marta
Edad: 34
Profesión: Cocinera de alta cocina
Descripción: Marta no entiende la cocina si no es de la mano de la innovación. En constante crecimiento profesional, hace que su vocación se refleje en su establecimiento ofreciendo constantes novedades en la carta. Esto hace que el trabajo de sus empleados gane en complejidad, así como continuos constantes en cartas y herramientas para sus trabajadores. Necesita una herramienta flexible, que le permita dar rienda suelta a su creatividad en la cocina, y despreocuparse en cómo hacer llegar esta información a sus empleados de cara a atender a los clientes en su establecimiento.

Escenario de uso 4:
Ha decidido que *GestioBar* es su herramienta. Únicamente tiene que preocuparse de registrar los nuevos productos que ofrece a su clientela en la herramienta y sus empleados automáticamente tendrán el soporte de la información necesaria para poder trabajar con las novedades de la cocina registradas al momento. Es más, una vez habituada a la herramienta, ha efectuado una transmisión de conocimientos y ha delegado esta gestión en otro empleado, ciñéndose ella únicamente a la creatividad entre fogones.

De este breve estudio de usuarios potenciales, se desprenden una conclusión principal en cuanto a la tipología de usuarios que tendrá la aplicación: Existirá un tipo de usuario que se encargará de la estructuración de la información, y otro que se encargará de registrar la información sobre el modelo preparado por el anterior.

A partir de ahora estableceremos este consenso:

Se hablará de **perfil de usuario** para definir a la propia tipología del usuario, identificados por el conjunto de funcionalidades que identifican su gestión dentro de la aplicación.

Se hablará de **perfil Gestor** al encargado de estructurar la información del establecimiento. Registrará la información propia de su establecimiento sobre el metamodelo definido por la plataforma. Esta información dará el soporte para el trabajo de sus empleados.

Se hablará de **perfil Empleado** al tipo de usuario cuya interacción con la aplicación se basará en el registro de comandas sobre la información definida por el perfil gestor.

Es conveniente aclarar en este punto dos aspectos importantes de aquí en adelante en cuanto al diseño de la aplicación:

Por un lado es conveniente anotar que es mejor asumir un bajo nivel técnico en los usuarios de cara a efectuar un diseño eficiente de la aplicación, ya que es un fallo que solemos tener los desarrolladores el de asumir que los usuarios potenciales de la plataforma poseen un nivel alto en este aspecto. Fallo generado o bien por la falta de perspectiva debida a un entorno técnico o bien por falta de experiencia.

También es importante apuntar que un usuario en nuestro aplicativo podrá poseer ambos perfiles, tanto en cuanto la tablet principal del establecimiento puede servir tanto para normalizar la información propia de la gestión del establecimiento como para anotar comandas en base a la información registrada.

3.2 Estructura de la aplicación y flujos de interacción

El flujo de interacciones que se producirá en la aplicación será lo más simple posible y responde al siguiente diagrama en forma gráfica:

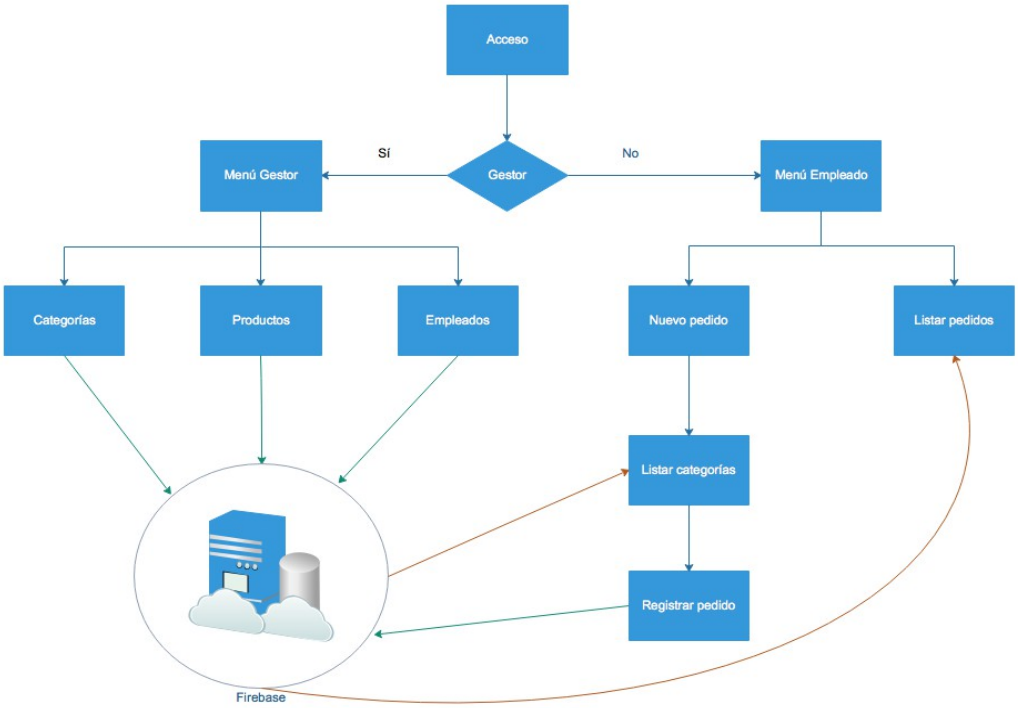


Figura 3. Diagrama de Flujo de la aplicación

La simplicidad es uno de los objetivos que se busca de cara a optimizar la usabilidad de la aplicación, y de ahí se desprende que la funcionalidad debe ser fácilmente accesible sin navegaciones interminables.

En el primer acceso, ofrecerán dos posibilidades: *Registrar un nuevo dispositivo*, operación a través de la cual se otorgará al usuario los

perfiles de Gestor y Empleado, e *Identificar un nuevo dispositivo*, operación que asigna al usuario de acceso el perfil de empleado.

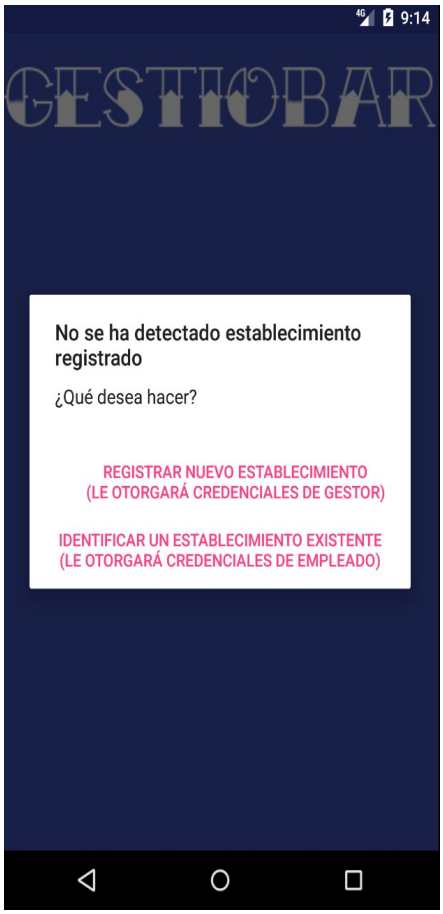


Figura 3. Primer acceso a la aplicación

Una vez registrado o identificado el establecimiento en el dispositivo, el usuario podrá acceder en modo empleado y/o en modo gestor, en función de los perfiles que tenga asignados. Al acceder en modo gestor lo hará a la sección destinada a la normalización de la información. Tendrá la posibilidad de visualizar las categorías, los productos de cada una de las categorías y los pedidos registrados en el establecimiento, así como la edición de esta información.

Si el acceso se produce en modo empleado podrá acceder al pedido abierto que está realizando, o bien acceder a un nuevo pedido, para lo que listará los distintos productos de cada una de las categorías. De esta forma se podrán tomar por separado los productos de las distintas categorías, de manera análoga a como se hace de la manera tradicional.

Se observa en el diagrama de flujo de la información, que las entradas de información en el back-end de almacenamiento de la información, en este caso firebase, se representan con flechas de color verde y flechas de color rojo para el flujo de la información cuando se produce desde firebase hacia la aplicación.

3.3 Diagramas de casos de uso

En base a lo que se ha ido concluyendo a lo largo del documento y los objetivos que se persiguen en cuanto a funcionalidad, los diagramas de casos de uso para cada uno de los perfiles, serían los siguientes:

3.3.1 Perfil Gestor

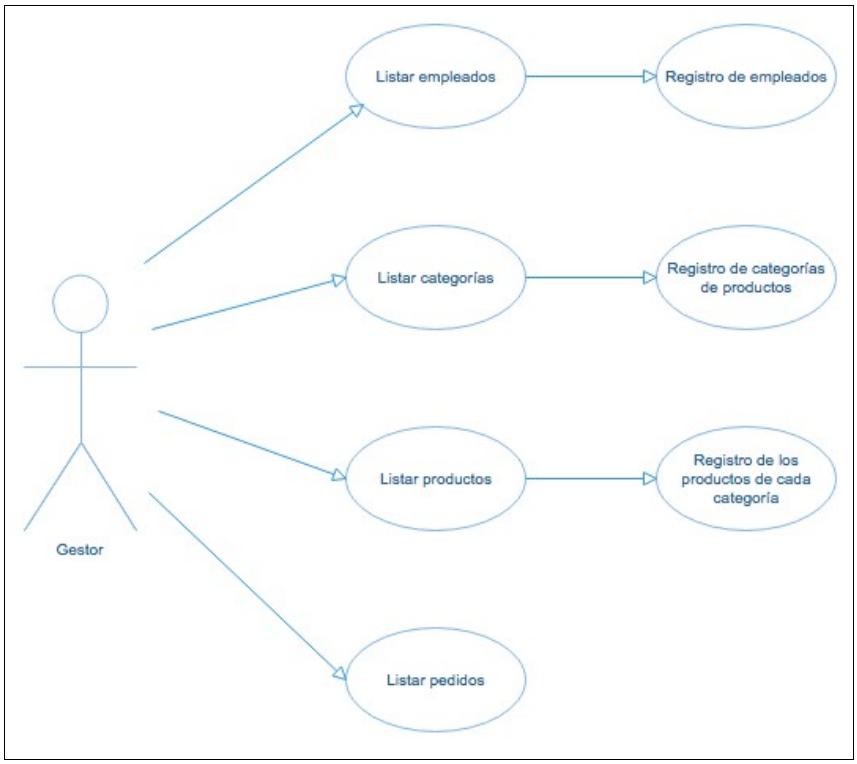


Figura 3. Diagrama de casos de uso para el perfil Gestor

El perfil Gestor poseerá en la aplicación las atribuciones que se muestran en la ilustración y que se procede a enumerar e informar a continuación:

Caso de Uso UC-01	Listar empleados
Descripción	Acción que consulta en firebase los empleados registrados para el establecimiento de consulta.
Precondiciones	Para listar empleados, previamente tendrán que haber sido insertados. En caso de no haber registrado aún empleados se mostrará un mensaje informativo.
Flujo	• El gestor selecciona la opción de empleados. • La aplicación consulta en firebase y obtiene los datos. • Los datos son presentados por pantalla para su visualización.

Postcondiciones	La lista de usuarios está visible para el gestor.

Caso de Uso UC-02	Registro de empleados
Descripción	Acción por la que el gestor registra un nuevo empleado del establecimiento a través de la aplicación.
Precondiciones	Listar empleados, se accederá a esta opción a través de ella.
Flujo	• El gestor selecciona la opción de empleados. • La aplicación consulta en firebase y obtiene los datos. • Los datos son presentados por pantalla para su visualización. • El gestor selecciona la opción para registrar un nuevo empleado. • El gestor inserta la información relevante del empleado. • La aplicación registra en firebase el nuevo empleado.
Postcondiciones	Tras el registro se generará el token para la identificación en el dispostivo del empleado.

Caso de Uso UC-03	Listar categorías
Descripción	Acción que consulta en firebase las categorías de productos registrados para el establecimiento de consulta.
Precondiciones	Para listar categorías, previamente tendrán que haber sido insertadas. En caso de no haber registrado aún categorías se mostrará un mensaje informativo.
Flujo	• El gestor selecciona la opción de categorías. • La aplicación consulta en firebase y obtiene los datos. • Los datos son presentados por pantalla para su visualización.
Postcondiciones	La lista de categorías está visible para el gestor.

Caso de Uso UC-04	Registro de categorías
Descripción	Acción por la que el gestor registra una nueva

	categoría de producto a través de la aplicación.
Precondiciones	Listar categorías, se accederá a esta opción a través de ella.
Flujo	• El gestor selecciona la opción de categorías. • La aplicación consulta en firebase y obtiene los datos. • Los datos son presentados por pantalla para su visualización. • El gestor selecciona la opción para registrar una nueva categoría. • El gestor inserta la información relevante a la categoría. • La aplicación registra en firebase la nueva categoría. nuevo empleado.
Postcondiciones	La categoría pasa a formar parte del listado de categorías.

Caso de Uso UC-05	Listar productos
Descripción	Acción que consulta en firebase los productos de una determinada categoría.
Precondiciones	Para listar los productos habrá que seleccionar previamente la categoría.
Flujo	• El gestor selecciona la opción de categorías. • El gestor selecciona del listado de categorías la opción de listar productos. • La aplicación consulta en firebase y obtiene los datos. • Los datos son presentados por pantalla para su visualización.
Postcondiciones	La lista de categorías está visible para el gestor.

Caso de Uso UC-06	Registro de productos
Descripción	Acción por la que el gestor registra un nuevo producto de una determinada categoría.
Precondiciones	Listar categorías, y al pulsar sobre alguna de ellas se accederá a esta pantalla.
Flujo	• El gestor selecciona la opción de categorías. • La aplicación consulta en firebase y obtiene los datos. • Los datos son presentados por pantalla para

	su visualización. • El gestor selecciona la categoría acerca de la cual quiere ver el listado de productos. • El gestor inserta la información relevante a la categoría. • La aplicación consulta en firebase y obtiene los datos. • Los datos son presentados por pantalla para su visualización.
Postcondiciones	El listado de productos es visible para el gestor.

Caso de Uso UC-07	Listar pedidos
Descripción	Acción que consulta en firebase los productos de una determinada categoría.
Precondiciones	Los pedidos han de haber sido registrados en firebase por parte del perfil empleado.
Flujo	• El gestor selecciona la opción de pedidos. • La aplicación consulta en firebase y obtiene los datos. • Los datos son presentados por pantalla para su visualización.
Postcondiciones	La lista de pedidos estará visible para el gestor posibilitando diferentes opciones de filtrado y ordenación.

3.3.2 Perfil Empleado

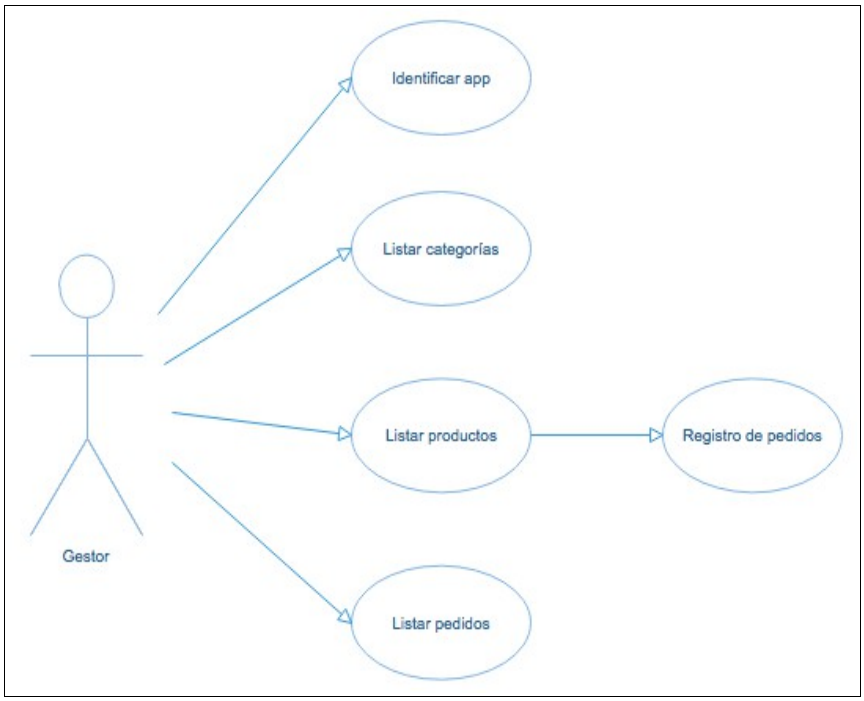


Figura 4. Diagrama de casos de uso para el perfil Empleado

El perfil Empleado poseerá en la aplicación las atribuciones que se muestran en la ilustración y que se procede a enumerar e informar a continuación:

Caso de Uso UC-08	Identificar app
Descripción	Acción que consulta identifica al empleado y lo relaciona con su establecimiento correspondiente en el backend de firebase.
Precondiciones	La aplicación ha de haber sido instalada en el dispositivo del empoleado, el empleado debe haber sido registrado por parte del perfil Gestor y generado el token de autenticación..
Flujo	• El empleado selecciona la opción de identificar app. • La aplicación envía el token de autenticación para identificar en firebase la relación empleado-establecimiento. • Firebase devuelve un ok y el listado de productos del empleado, en este caso vacío. • La información es mostrada en la pantalla.
Postcondiciones	La identificación empleado-establecimiento ha sido consolidada.

Caso de Uso UC-09	Listar categorías
Descripción	Acción que consulta en firebase las categorías de productos registrados para el establecimiento de consulta.
Precondiciones	Para listar categorías, previamente tendrán que haber sido insertadas. En caso de no haber registrado aún categorías se mostrará un mensaje informativo.
Flujo	• El empleado selecciona la opción de nuevo pedido. • La aplicación consulta en firebase y obtiene los datos acerca de las categorías. • Los datos son presentados por pantalla para su visualización.
Postcondiciones	La lista de categorías está visible para el empleado y seleccionable para listar los productos de las mismas.

Caso de Uso UC-10	Listar productos
Descripción	Acción que consulta en firebase los productos registrados para la categoría de consulta.
Precondiciones	Para listar productos, previamente tendrán que haber sido insertados.
Flujo	• El empleado selecciona la opción de nuevo pedido. • La aplicación consulta en firebase y obtiene los datos acerca de las categorías. • El empleado selecciona la categoría determinada. • La aplicación consulta en firebase y obtiene los productos de la categoría seleccionada. • Los datos son presentados por pantalla para su visualización.
Postcondiciones	La lista de productos está visible para el empleado y seleccionable para añadir al pedido.

Caso de Uso UC-11	Registrar pedido
Descripción	Acción que registra los productos asociados a un pedido.
Precondiciones	El pedido aún no ha sido registrado
Flujo	• El gestor selecciona la opción de registrar

	pedido. • La aplicación registra en firebase la información.
Postcondiciones	El pedido es registrado.

Caso de Uso UC-12	Listar pedidos
Descripción	Acción que consulta en firebase los pedidos asociados al empleado de consulta
Precondiciones	Los pedidos han de haber sido registrados en firebase por parte del perfil empleado.
Flujo	• El empleado selecciona la opción de pedidos. • La aplicación consulta en firebase y obtiene los datos. • Los datos son presentados por pantalla para su visualización.
Postcondiciones	La lista de pedidos estará visible para el empleado posibilitando diferentes opciones de filtrado y ordenación.

4. Prototipado

4.1 Perfil Gestor

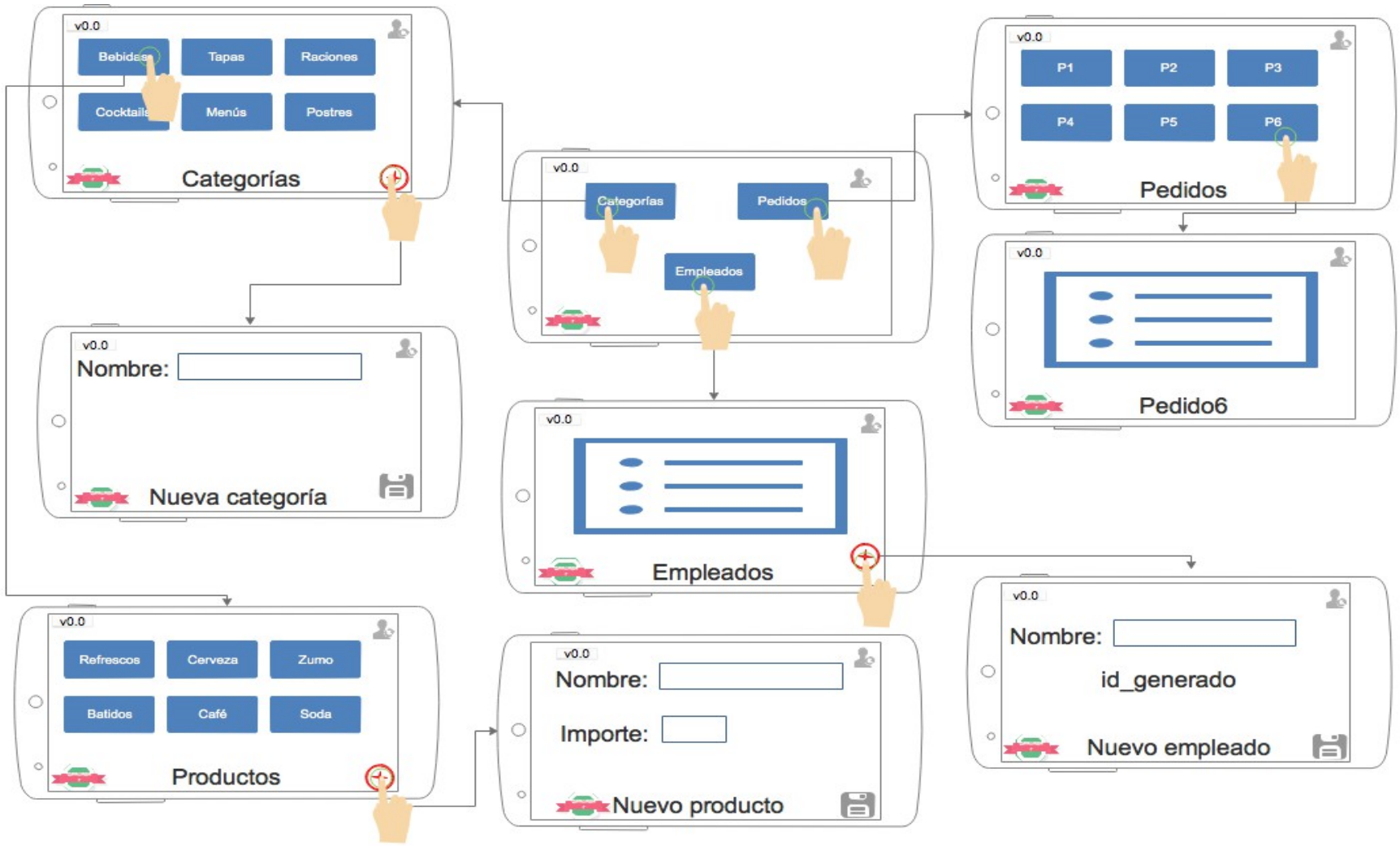


Figura 5. Prototipo para el perfil Gestor

4.1 Perfil Empleado

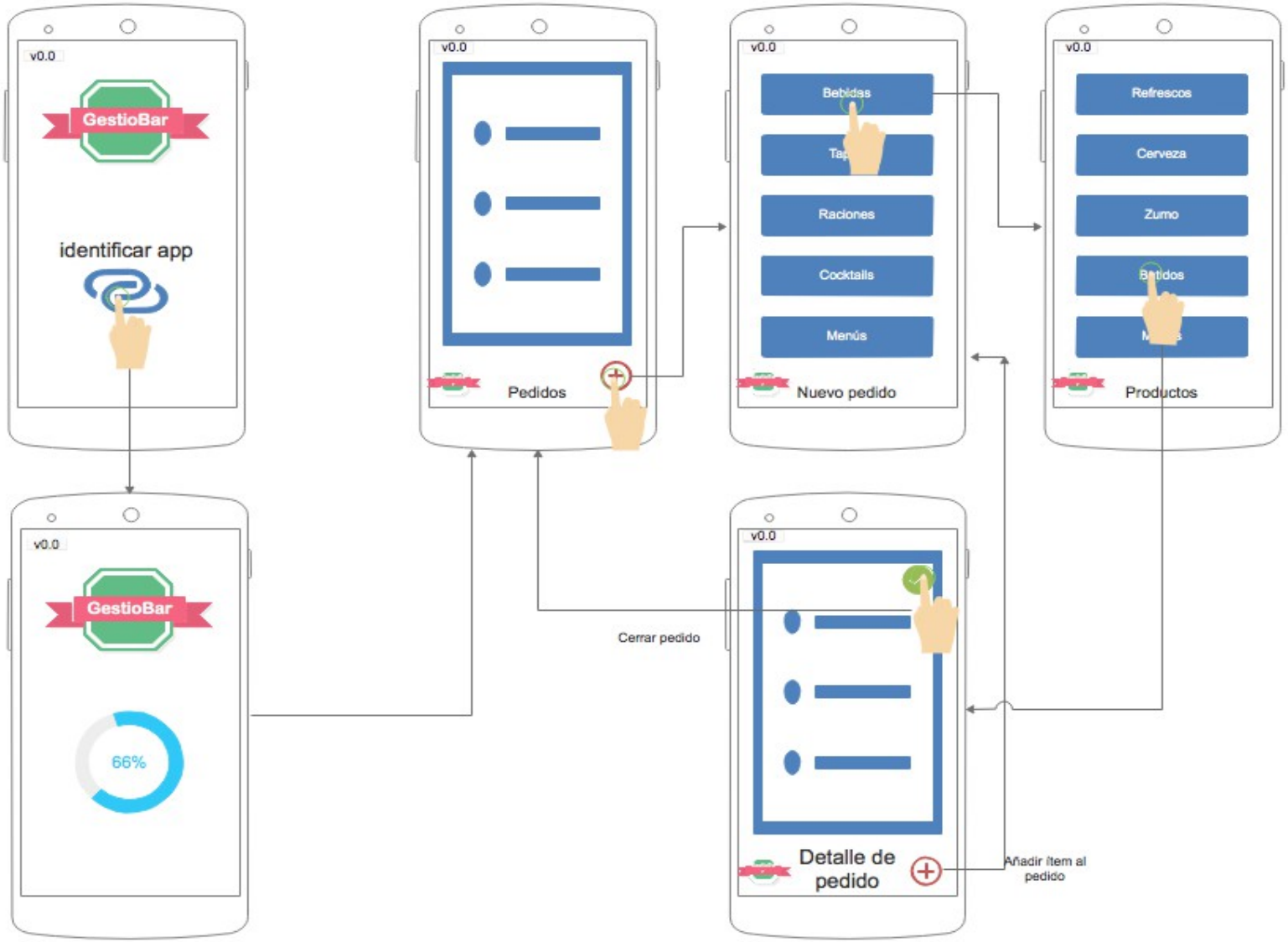


Figura 6. Prototipo para el perfil Empleado

En las dos ilustraciones anteriores se intenta plasmar un esbozo de aspecto y navegación de la aplicación, para ir introduciendo la filosofía que tendrá en su operativa el aplicativo.

5. Diseño de arquitectura de la aplicación

5.1 Diagrama de base de datos

Usaremos un soporte en la nube para el almacenamiento y consulta de datos en tiempo real. La opción elegida es la base de datos NoSQL **Firebase Realtime Database**. Los datos se almacenan en formato JSON y se sincronizan en tiempo real con cada cliente conectado. Cada cliente comparte una instancia de Realtime Database y reciben actualizaciones de forma automática con los datos actualizados.

GestioBar consultará e interactuará con esta estructura de datos a usar a modo de Back-End.

En el siguiente diagrama se muestra el modelado semántico que se ha hecho de las entidades que serán usadas en el sistema:

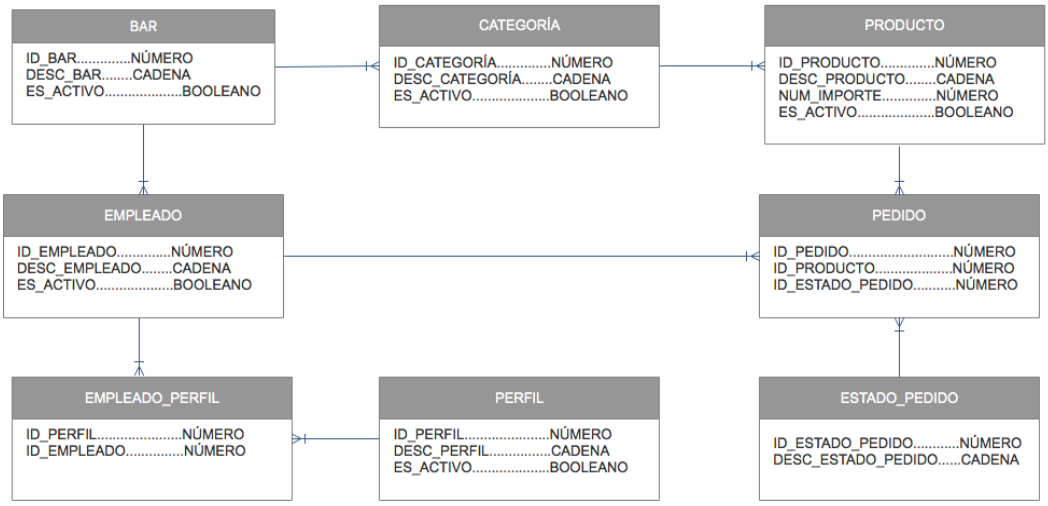


Figura 7.Modelo de datos de la aplicación.

En base a estas entidades será registrada y tratada la información por parte del backend firebase y las entidades que formarán parte de la aplicación.

El aspecto de la estructura de datos almacenada será el mostrado en el Anexo III: Estructura de datos almacenados en firebase.

La consola a través de la que se hará el control y gestión de la fuente del datos será la aportada por firebase, que presentará el siguiente aspecto en el caso particular de la aplicación que se está desrrollando.

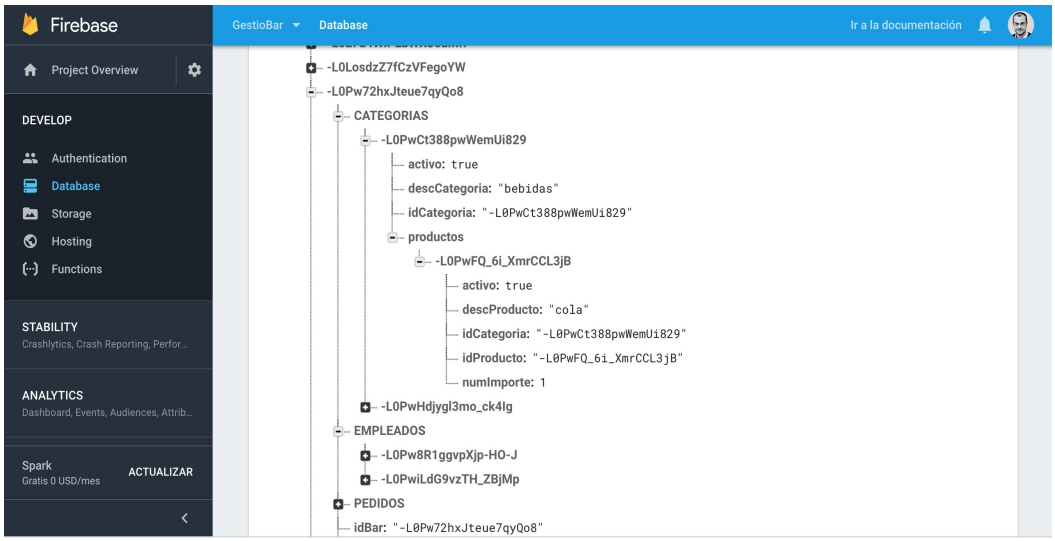


Figura 8. Consola de administración Firebase con la base de datos de GestioBar.

5.1 Arquitectura del sistema

En vistas a conseguir un software mantenible y robusto, de entre las opciones que se ofrecen dentro del catálogo de patrones de diseño existentes, se va a optar por emplear la arquitectura Modelo-Vista-Controlador (MVC) de forma que:

– Modelo: Representará la lógica de negocio, y tendrá una interpretación directa de la paquetería y clases del sistema:

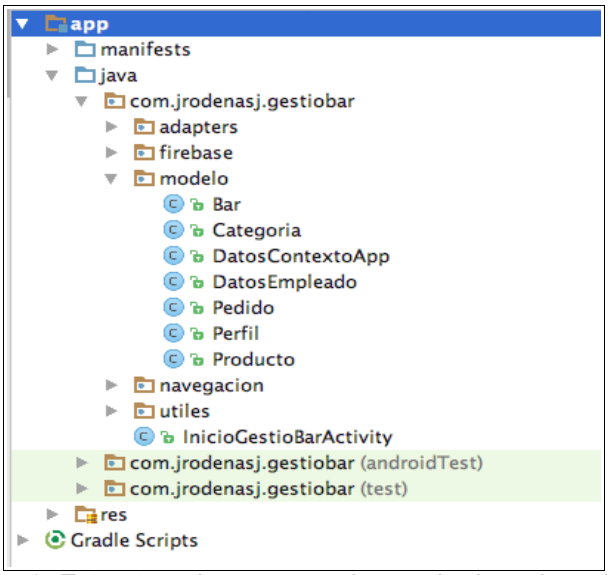


Figura 9. Estructura de paquetes de com.jrodenasj.gestiobar.modelo

Será el responsable de acceder a la capa de almacenamiento de la información. En esta capa, la responsabilidad de la interacción con la información registrada en firebase recaerá sobre una clase Java que se encargará de ello:

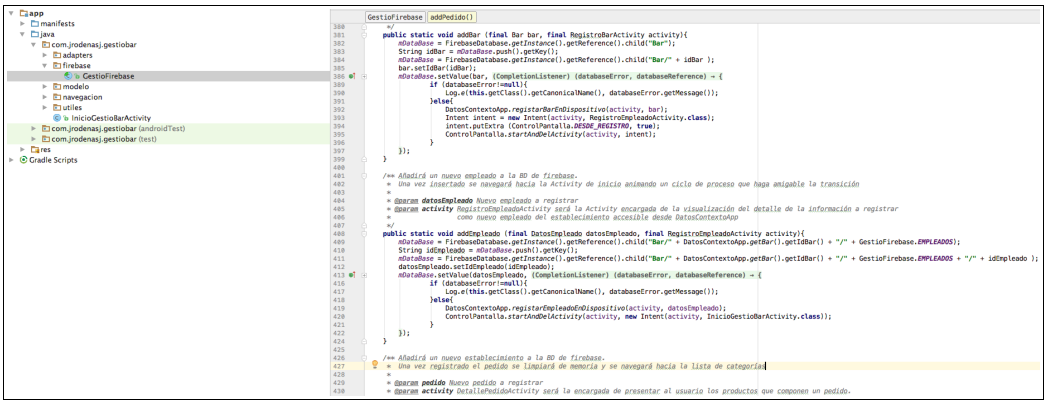


Figura 10. Clase de interacción con firebases: com.jrodenasj.gestiobar.firebase.GestioBar

Será necesaria la implementación de las clases que hagan de intérprete con el sistema de almacenamiento de manera que facilitemos su mantenimiento y mejoremos la robustez.

– Vista: Será la responsable de posibilitar la visualización a los usuarios. En el caso de una aplicación implementada en layout y diferentes elementos gráficos que va a visualizar el usuario. Como se ha visto en los distintos casos de usos, también estará conformada por las distintas clases que serán las responsables de la presentación de los datos obtenidos desde firebase por parte de los intérpretes. Cabe apuntar que esta aplicación se valdrá **adapters**, elementos recomendados desde el paradigma **material design** de Android para optimizar la visualización de colecciones de datos en las aplicaciones móviles:

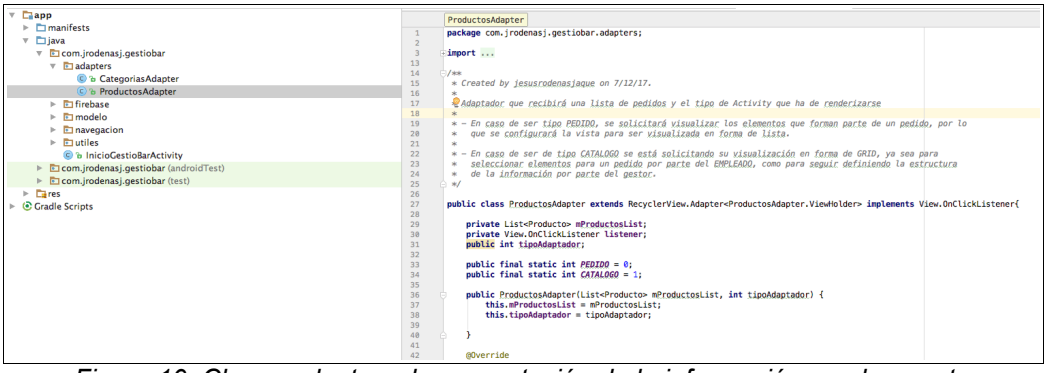


Figura 10. Clases adapters de presentación de la información en el paquete com.jrodenasj.gestiobar.adapters

– Controlador: Recibirá los eventos de entrada y gestionará el flujo de información entre la vista y el modelo. Será el nexo entre firebase y la capa sobre la que recaerá la responsabilidad de la vista.

5.1 Arquitectura de clases del sistema

El diseño de entidades de la aplicación favorece la generación de una clase por cada una de las entidades que intervendrán en el flujo de interacción de la información a lo largo del ciclo de vida de la aplicación,

así tendremos. A grandes rasgos, el diagrama de clases que se define será:

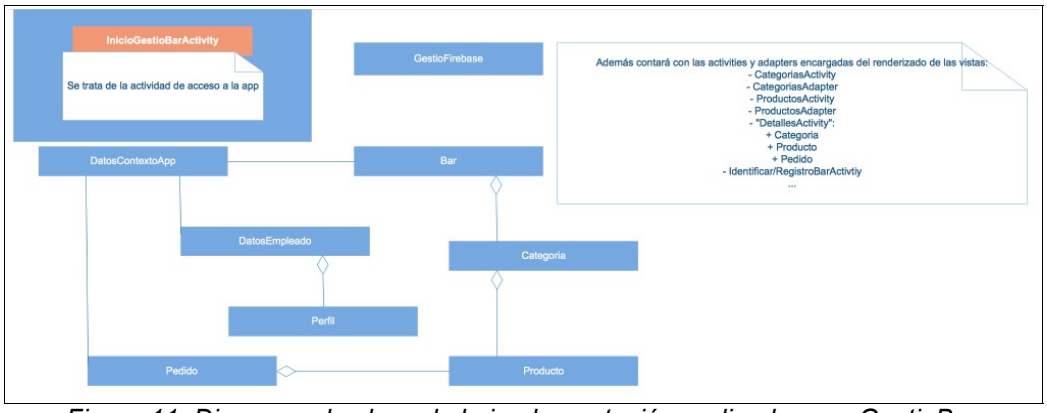


Figura 11. Diagrama de clase de la implementación realizada para GestioBar

En cuanto al inicio de la aplicación la parte de la arquitectura invitada:

– **InicioGestioBarActivity**: Será la clase de acceso a la aplicación. Establecerá la primera conexión con firebase, preparará los datos a albergar de forma local en caso de que no existan, o los consultará para realizar la primera carga de perifiles de usuario. Redigirá el flujo de la aplicación hacia donde corresponda en función del perfil del usuario.
– Archivo **gestiobarapp.properties**: Archivo de propiedades que se ubicará de forma local en el dispositivo. La finalidad del mismo es la de almacenar los datos que han sido registrados en firebase para ese usuario-dispositivo. Así se posibilitará la identificación en la primera carga de acceso a la aplicación. Será un fichero con la siguiente estructura:

ID_EMPLEADO=-L0PwCt388pwWemUi829
NOMBRE_EMPLEADO=Ejemplo_empleado
ID_BAR=-L0PwFQ_6i_XmrCCL3jB
NOMBRE_BAR=Ejemplo_Bar

En cuanto al modelo de datos, las clases de la arquitectura implicadas serán:

– Una clase **Bar** que encapsulará los datos registrados en firebase para un determinado establecimiento:
  - `String` **`idBar: Identificador firebase del establecimiento.`**
  - `String` **`nombreBar: Nombre del establecimiento.`**
  - `List<Categoria>` **`categorias: Listado de categorías registradas para el establecimiento.`**

– Una clase **DatosEmpleado** que poseerá los datos del empleado de acceso:
  - `String` **`idEmpleado: Identificador firebase del empleado.`**
  - `String` **`descEmpleado: Nombre.`**
  - **`boolean activo: Empleado activo.`**

- List<Perfil> **perfiles: Perfiles del usuario.**
- **int perfil: Perfil activo del usuario.**

– Una clase **Perfil** que encapsulará la información relativa a los perfiles de usuario. Será atributo en forma de lista de la clase DatosEmpleado:

- **int idPerfil: Identificador del perfil.**
- String **descPerfil: Nombre del perfil.**
- **boolean activo: Perfil activo.**

– Una clase **Categoria** destinada a representar la información de las categorías de productos que utilizan los establecimientos para la organización del establecimiento:

- String **idCategoria: Identificador firebase de la categoría.**
- String **descCategoria: Nombre de la categoría.**
- List<Producto> **productos: Productos que componen la categoría.**
- **boolean activo: Categoría activa.**

– Una clase **Pedido** para representar la informació que componen los pedidos. Será atributo de la clase DatosContextoApp:

- String **idPedido: Identificador firebase del pedido.**
- List<Producto> **productos: Lista de productos que conforman el pedido.**
- **int estadoPedido: Estado en el instante del pedido.**
- String **idEmpleado: Identificador firebase del usuario que registra el pedido.**

– Una clase **Producto** para estructurar la información relativo a los productos que componen cada una de las distintas categorías. Los productos serán atributo en form de listas en las clases de Categoria y Pedido:

- String **idProducto: Identificador firebase del producto.**
- String **descProducto: Nombre del producto.**
- Float **numImporte: Precio del producto.**
- **boolean activo: Pedido activo.**

– La clase DatoContextoApp que albergará el grueso de la información necesaria para la navegación por la plataforma y la gestión del archivo *gestiobarapp.properties*:

- **static** DatosEmpleado *datosEmpleado*: Datos del empleado que está en navegación.
- **static** Bar *bar*: Bar registrado en el dispositivo.
- **static** Pedido *pedidoEnCurso*: Pedido en curso en el dispositivo.

– La **interacción con firebase** será responsabilidad de la clase **GestioFirebase**. Será la encargada de mantener el sistema de peticiones hacia el backend consiguiendo así una mayor mantenibilidad de la aplicación al centralizar el acceso a datos en esta clase:

- **static void** setupFirebaseInfo(**final** InicioGestioBarActivity activity)

- `static void cargaPerfilesUsuario (final InicioGestioBarActivity activity)`
- `static void cargaCategoriaList(final CategoriasActivity activity)`
- `static void chequearIdBar (final String idBar, final IdentificarBarActivity activity)`
- `static List<Categoria> getCategoriaList()`
- `static void cargaProductoList(final ProductosActivity activity){`
- `static void checkCurrentUser()`
- `static List<Producto> getProductoList()`
- `static void addProducto (String id_categoria, Producto producto, final DetalleProductoActivity activity)`
- `static void addCategoria (Categoria categoria, final DetalleCategoriaActivity activity)`
- `static void addBar (final Bar bar, final RegistroBarActivity activity)`
- `static void addEmpleado (final DatosEmpleado datosEmpleado, final RegistroEmpleadoActivity activity)`
- `static void addPedido (Pedido pedido, final DetallePedidoActivity activity)`

También exisitrá una capa de Activity's que permitirán el renderizado de la información que se vaya obteniendo del backend y se encargará de gesitonar los eventos que vayan disparándose por parte de los usuarios en la interacción con la aplicación. Para ellos se apoyarán en una capa de adaptadores que el paradigma de Android ofrece para la optimización de aplicaciones y que Material Design (5) recomienda para la mejora de su usabilidad de las aplicaciones. Son las que en el diagrama anterior se presentan en forma de nota.

6. Plan de pruebas

Para la comprobación del correcto funcionamiento de GestioBar, se han utilizado principalmente dos herramientas:

- Emulador Android Studio
- Dispositivo Xiaomi s3 pro

En primer lugar y para ambas plataformas se ha relizado la correcta instalación. A partir de ahí se realizan las siguientes pruebas:

6.1 Registro de un nuevo establecimiento

Al acceder a la aplicación por primera vez se ofrece la posibilidad de registrar un nuevo establecimiento y con ello asignar al usuario de instalación los perfiles de Gestor y Empleado:

Figura 12. Primera pantalla tras la instalación

En primera instancia registraremos un nuevo establecimiento y en consecuencia un nuevo Empleado y Gestor:

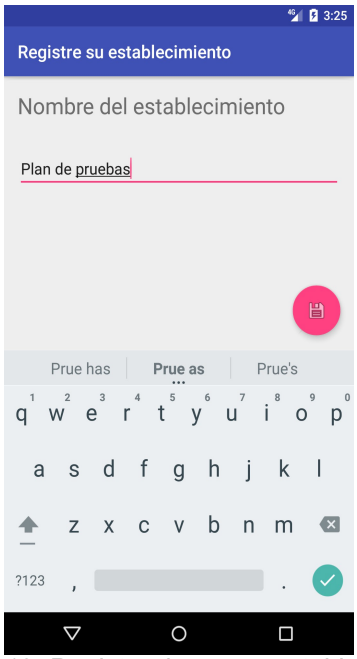


Figura 13. Registro de nuevo establecimiento

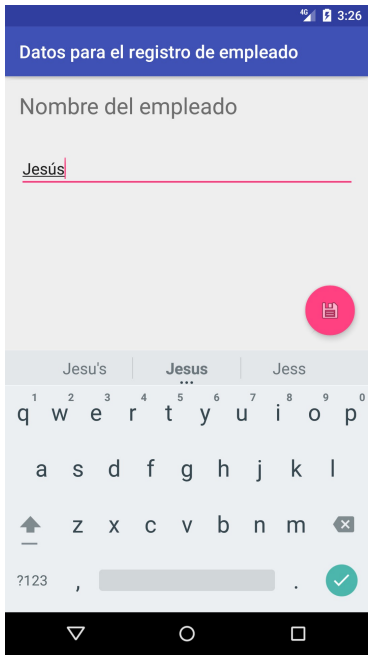


Figura 14. Registro de nuevo empleado

Tras estas operaciones efectivamente Firebase registra el nuevo establecimiento y los nuevos datos:

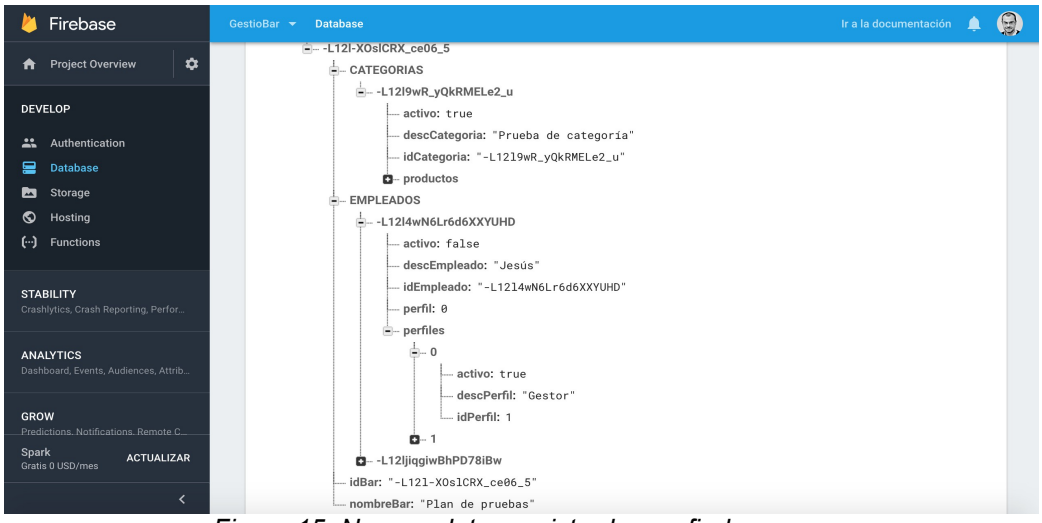


Figura 15. Nuevos datos registrados en firebase

6.2 Registro de nueva categoría para un establecimiento

Una vez se ha registrado el establecimiento en GestioBar, se cominenzan a definir las categorías correspondientes para el establecimiento credo. Para ello seleccionaremos la opción “+” en la pantalla de categorías, a la que la aplicación nos lleva tras finalizar el registro del establecimiento:

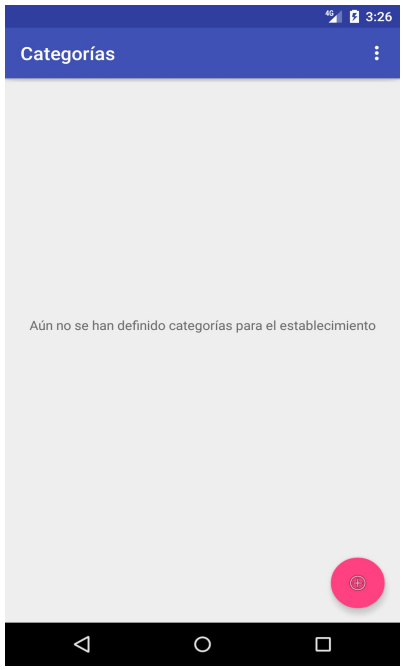


Figura 16. Pantalla inicial de categorías sin elementos

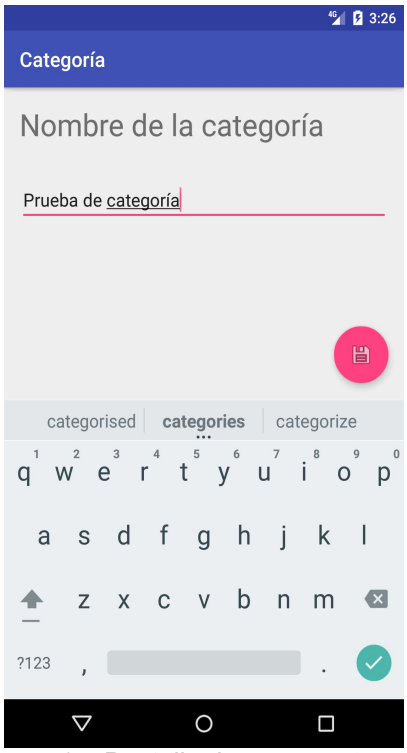


Figura 17. Pantalla de nueva categoría

Se comprueba que efectivamente se ha registrado la nueva categoría para el establecimiento del plan de pruebas:

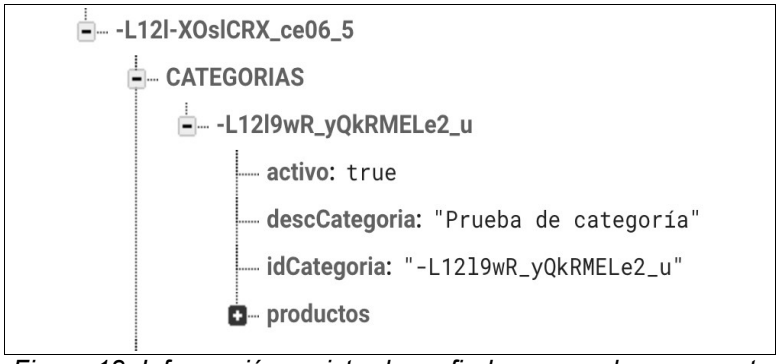


Figura 18. Información registrada en firebase para la nueva categoría

Además, en GestioBar se listará la nueva categoría registrada para el establecimiento de prueba:

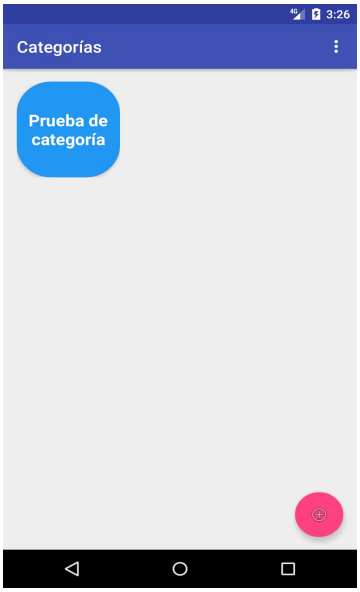


Figura 19. Listado de categorías para el bar de prueba en GestioBar

6.3 Registro de productos relacionados con una categoría creada

En función a las categorías han de definirse los productos relacionados con la misma. De esta forma al registrar los distintos productos de las distintas categorías se habrá definido la información necesaria para el uso de los Empleados. A continuación se definirá un producto de la categoría de prueba y se testeará su registro en firebase: Para ello se selecciona por parte del gestor la categoría para la cual quieren definirse los productos relacionados, mostrando el panel vacío para esta prueba:

Figura 20. Listado vacío de productos para la categoría de prueba en GestioBar

Al pulsar sobre el botón “+” nos llevará a un formulario en el que ingresar nombre y precio del producto:

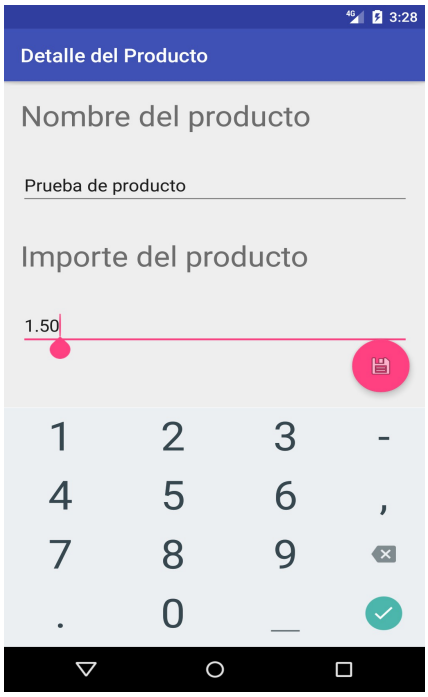


Figura 21. Formulario de alta para nuevo producto de GestioBar

Al registrar el nuevo producto se podár visualizar relacionado con la categoría de prueba el nuevo producto de prueba tanto en la base de datos firebase como en el frontend que supone la aplicaicó de gestiobar:

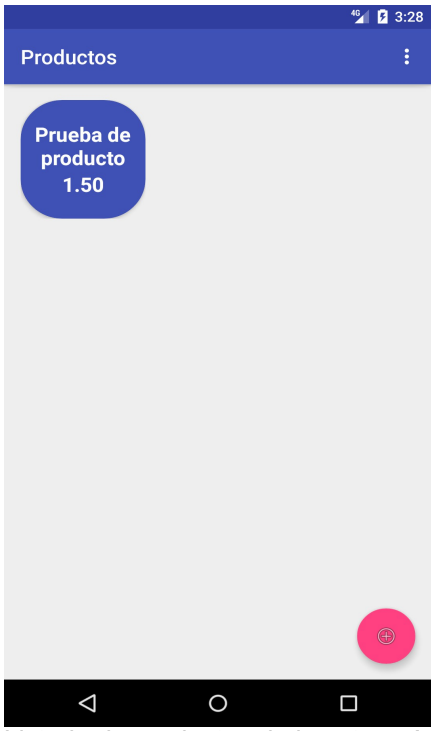


Figura 22. Listado de productos de la categoría de prueba

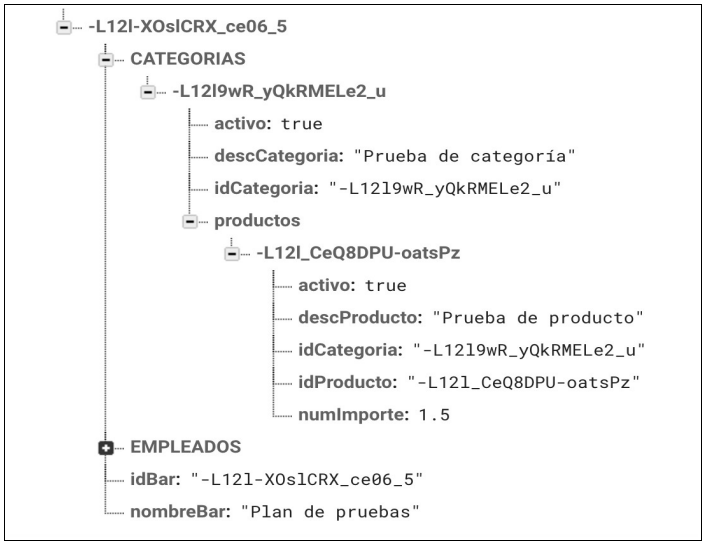


Figura 23. Información plasmada en firebase

6.4 Menú de Gestor

Los usuarios que poseen dos perfiles en GestioBar deben tener disponible el cambio de perfil en la plataforma, ya que la aplicación en modo Gestor está pensada para la normalización de la información y en modo Empleado para el registro de pedidos. Por tanto para el empleado “Jesús”, que posee ambos perfiles, debe aparecerle entre las opciones de menú la posibilidad de cambiar de perfil.

Además del cambio de perfil, al Gestor le debe ser posible copiar el ID del establecimiento para enviárselo por cualquier sistema de mensajería móvil a sus empleados para que puedan proceder a la identificación de usuarios que veremos probar más adelante.

El menú del gestor sería el siguinte:

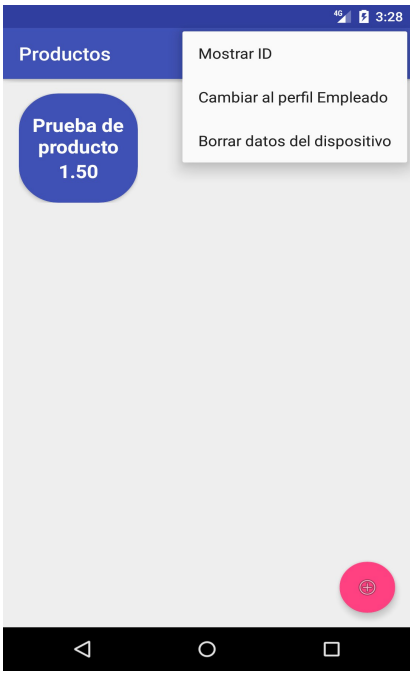


Figura 24. Menú perfil Gestor de GestioBar

A través de la opción de mostar ID podrán copiar el identificador del establecimiento para compartirlo:

Figura 25. Opción de copiar ID de establecimiento

En la sección de identificar establecimiento, se usará este ID para el perfil empleado.

Otra prueba a realizar en este menú es la de cambio de perfil, y se ve claramente al efectuarle que el menú del Gestor varía, ya que se encuentra con perfil Empleado dentro de la plataforma GestioBar:

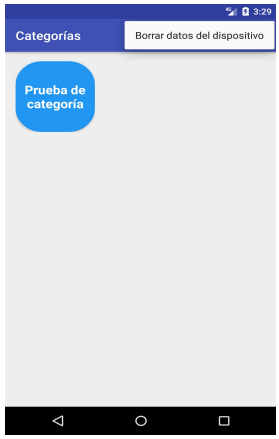


Figura 26. Menú empleado de GestioBar

6.5 Identificar establecimiento existente

Los empleados recibirán el identificador del establecimiento mediante cualquier sistema de mensajería móvil para poder pegarlo mediante la identificación del establecimiento. Como se citó anteriormente, las dos primeras operaciones que pueden realizarse en un primer acceso a la aplicación son las de registrar nuevo establecimiento o la de identificar un establecimiento existente. Un empleado en GestioBar la opción que seleccionaría sería la de identificar un establcimiento existente mediante la funcionalidad ofrecida para ello. De esta forma se identificará el bar y se registará el nuevo usuario únicamente con el perfil de Empleado.

Tras acceder mediante la identificación, al usuario Empleado se le mostrarán las categorías registradas en firebase para el establecimiento en cuestión por parte del Gestor del mismo.

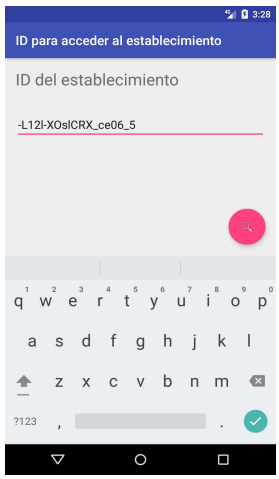


Figura 27. Identificar establecimiento existente

Una vez identificado el establecimiento y que GestioBar haya comprobado efectivamente que el establecimiento existe, se mostrará el

listado de las categorías y productos registrados para el establecimiento. Se comprueba que la información ha sido registrada correctamente en firebase:

Figura 28. Empleado registrado

6.6 Registro de pedidos

Una función reservada exclusivamente para los empleados será la del registro de pedidos. La aplicación para esto funciona de la siguiente forma: El Empleado accederá al listado de categorías a través de la pantalla de registro de pedido:

Figura 29. Activity de inicio del perfil empleado

A través del listado de categoría accederá al listado de productos relacionados mediante un tap. Al pulsar sobre el producto lo registará en el pedido en curso mostrando la siguiente información por pantalla:

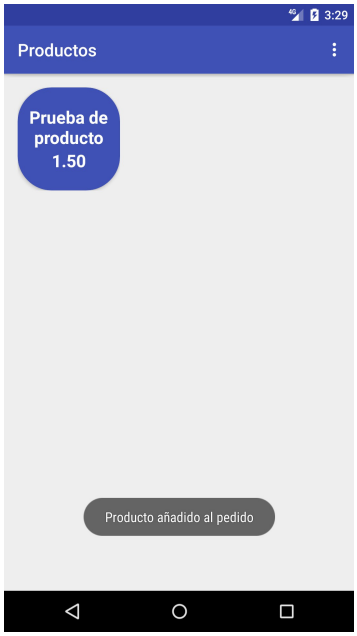


Figura 30. Productos añadido al pedido

El empleado podrá visualizar el pedido en curso mediante su menú de empleado para el cual, además de eliminar los datos de identificación del dispositivo, podrá seleccionar la opción comentada:

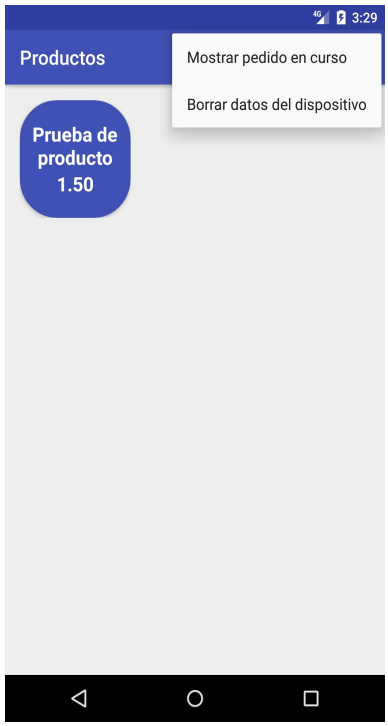


Figura 31. Menú de listar pedido en curso

La información a la que accederá a través de esta opción será la lista de productos que componen un pedido:

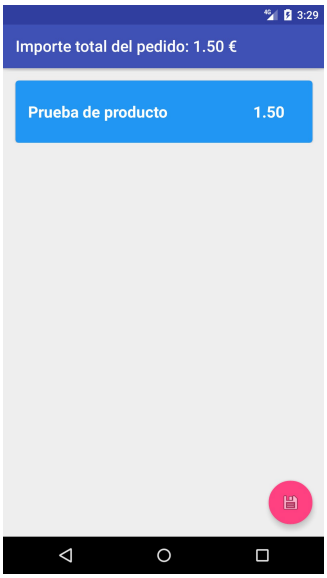


Figura 32. Listado de productos de pedido en curso

Mediante el botón de guardar que presenta la siguiente pantalla se enviará la información a firebase y se puede comprobar que la información ha sido correctamente registrada:

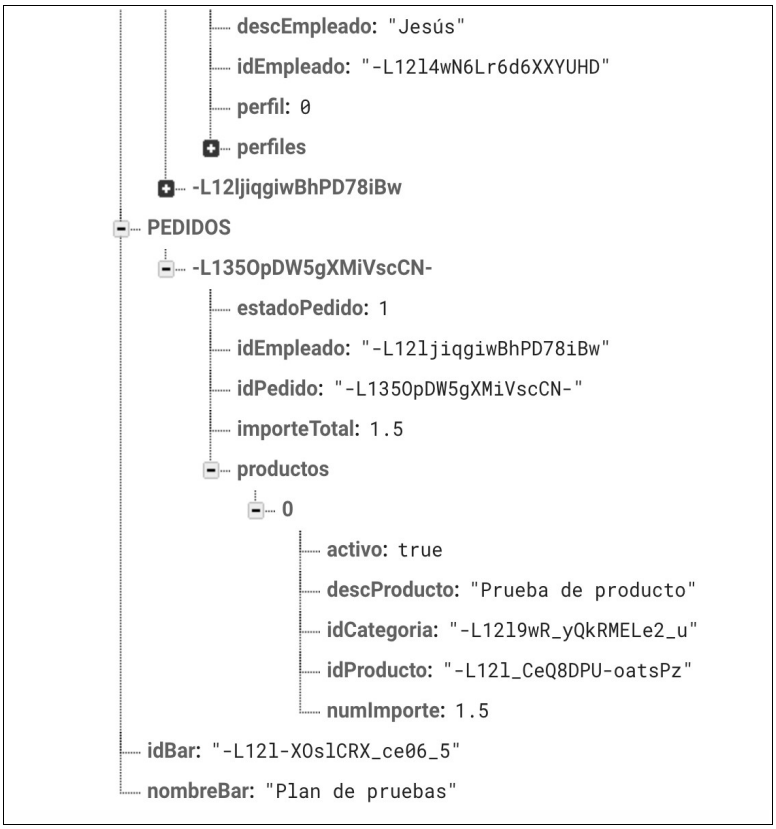


Figura 33. Pedido registrado correctamente en firebase

7. Conclusiones

El principal objetivo con el que nació la idea de GestioBar como aplicación objeto del TFM fue la de ofrecer una solución a bolsillos más modestos de cara a informatizar la gestión de sus negocios de hostelería. Además, se perseguía ofrecer una solución centralizada, en la que fuera posible residir cualquier establecimiento cuya lógica de negocio pudiera ser enmarcada en la estructura de datos que permite crear el soporte implementado para la interacción con el backend. Estos objetivos han sido cubiertos a lo largo del desarrollo de este TFM.

Se estableció como objetivo técnico la definición de un metamodelo que diera soporte al mayor espectro de establecimientos, y siempre que un establecimiento pueda categorizar sus productos y acogerse a la política de datos establecida en GestioBar, puede usar como herramienta la aplicación desarrollada en este TFM. Tras un análisis se desprende que tendría que ser muy particular la gestión de este servicio para no acogerse a las funcionalidades que ofrece GestioBar.

Otra de las prioridades de la idea por la que se optó para la implementación de la app fue del uso y comunicación con el backend de Firebase. Una base de datos NoSql que ofrece una solución transversal, fácilmente migrable y exportable. La primera versión de GestioBar está implementada en Android, pero una app desarrollada para iOS o con tecnología de desarrollo híbrido (Ionic2 por ejemplo) podrían comunicarse con el mismo backend, teniendo que programar únicamente la vista y acceso a datos en las correspondientes soluciones elegidas. La fuente del dato sería la misma y cubriría las mismas interacciones y funcionalidades.

Se perseguía ofrecer las herramientas necesarias para que el Gestor pudiera normalizar la información con la que establecimiento iba a trabajar, así como la integración de la misma en las aplicaciones descargadas e instaladas en los dispositivos de los empleados, asignando los perfiles, y por tanto los permisos necesarios, a cada uno de los actores implicados en la gestión del establecimiento.

Tras el análisis de los objetivos funcionales y no funcionales, en líneas generales existe un grado satisfactorio de consecución de los mismos.

Dentro de los requisitos no funcionales, se perseguía un frontend atractivo y usable. Si bien la usabilidad de la aplicación se puede considerar apta, no es menos cierto que el atractivo de la misma podría ofrecer mejoras notables. La mayoría de las aplicaciones que existen a nivel profesional presentan una interfaz mucho más atractiva que la que ofrece GestioBar: Se podrá apuntar también como acción de mejora de cara a futuro.

Otro aspecto a mejorar es la gestión de versiones de la app. Cualquier aplicación en un entorno productivo debe tener un estricto control de

cada una de las versiones que ha liberado, principalmente para la corrección de errores y para mejorar las mismas con el fin de obtener un producto cada vez mejor. Como acción de mejora se plantearía una integración con distintas DevOps que ofrecen integración y construcción contínua, aspectos hoy de los que un ingeniero de software no puede desligarse.

Desde el punto de vista de desarrollador, si bien poseía sólidos conocimientos en Java y avanzados en Android, he podido experimentar cómo se han mejorado mis tiempos en resolución de bugs, entendiendo mucho mejor la política en la gestión de errores que nos ofrece Android. Entender el funcionamiento de layouts y adapters, además del uso de funciones callback ha propiciado una navegación con una amigabilidad más que óptima, y que no presenta inconsistencias a pesar de la red a la que esté conectado. Otra cuestión que se desprendería de una hipotética puesta en producción, sería la de la implementación de un buen subsistema de gestión de errores para poder registrar aquellos que sucediesen con el fin de depurar la versión de la app en producción.

A nivel de gestión de proyectos he comprobado de primera mano las dificultades que se desprenden del cumplimiento de una planificación: A pesar de ofrecer una planificación exhaustiva y coherente con mi realidad laboral, las dificultades se han multiplicado llegando a ser imposible su cumplimiento. Solo gracias a las múltiples horas de sueño privadas, la flexibilidad encontrada a la hora de entregar documentación, la paralelización de tareas gracias a la metodología usada para el desarrollo de TFM y las tazas de café consumidas en proporción, se ha conseguido obtener un producto final que puede servir de base a una versión profesional futura.

En resumen, los principales cambios que se han tenido que ir introduciendo sobre la previsión y análisis inicial han sido consensuados y considerados óptimos con el fin de obtener un producto ideal para ser entregado como TFM. Todo puede desprender aprendizaje y de todo hay que intentar captarlo e incorporarlo.

Es por tanto que se podría desprender de las conclusiones obtenidas, que para mejorar GestioBar habría que:

- Implementar las versiones de la app para otras plataformas, principalmente iOS o bien elegir una tecnología de desarrollo híbrido de apps.
- Mejoras funcionales de la app que se desprenderían de una hipotética liberación en un entorno productivo.
- Mejoras a nivel estético de la app: Apoyarse en un diseñador.
- Subsistema de control de versiones: Imprescindible de cara a depurar mejoras en cada una de las versiones liberadas, seria automatizar la gestión de versiones usando herramientas de integración/construcción continua.
- Subsistema de errores: De cara a la depuración citada anteriormente y derivado del uso masivo de la app por parte de

usuarios, sería ideal la implementación de un subsistema de errores y poder depurar posteriormente la implementación de la aplicación. Podríamos valernos del backend seleccionado para almacenar estos errores.

8. Glosario

Perfil: Conjunto de funcionalidades agrupadas en forma de rol de permisos y posibilidades que se asigna a un usuario de la aplicación.

Comanda: Pedido o encargo que se le hace a un camarero.

Scrum: Scrum es una metodología ágil y flexible para gestionar el desarrollo de software, cuyo principal objetivo es maximizar el retorno de la inversión para su empresa (ROI). Se basa en construir primero la funcionalidad de mayor valor para el cliente y en los principios de inspección continua, adaptación, auto-gestión e innovación.

Material Design: Es una normativa de diseño enfocado en la visualización del sistema operativo Android , además en la web y en cualquier plataforma. Fue desarrollado por Google y anunciado en la conferencia Google I/O celebrada el 25 de junio de 2014

9. Bibliografía

- Documentación oficial de Android
  https://developer.android.com/index.html

- Documentación oficial de firebase
  https://firebase.google.com/docs/

- Estructuras de datos en firebase
  https://firebase.google.com/docs/database/web/structure-data

- Guía de desarrollo android
  https://desarrollador-android.com/

- Material design para android
  https://developer.android.com/design/material/index.html?hl=es-419

- Paradigma de material design
  https://material.io/guidelines/

- Blog oficial de firebase
  https://firebase.googleblog.com/

- Tutoriales Gimp
  http://todogimp.com/

- Tutoriales OBS
  https://obsproject.com/forum/

10. Anexos

Anexo I: Enlaces

(1) Restaurant Waiter
https://play.google.com/store/apps/details?id=com.restaurantwaiter&hl=es

(2) Cuiner
http://cuiner.com/

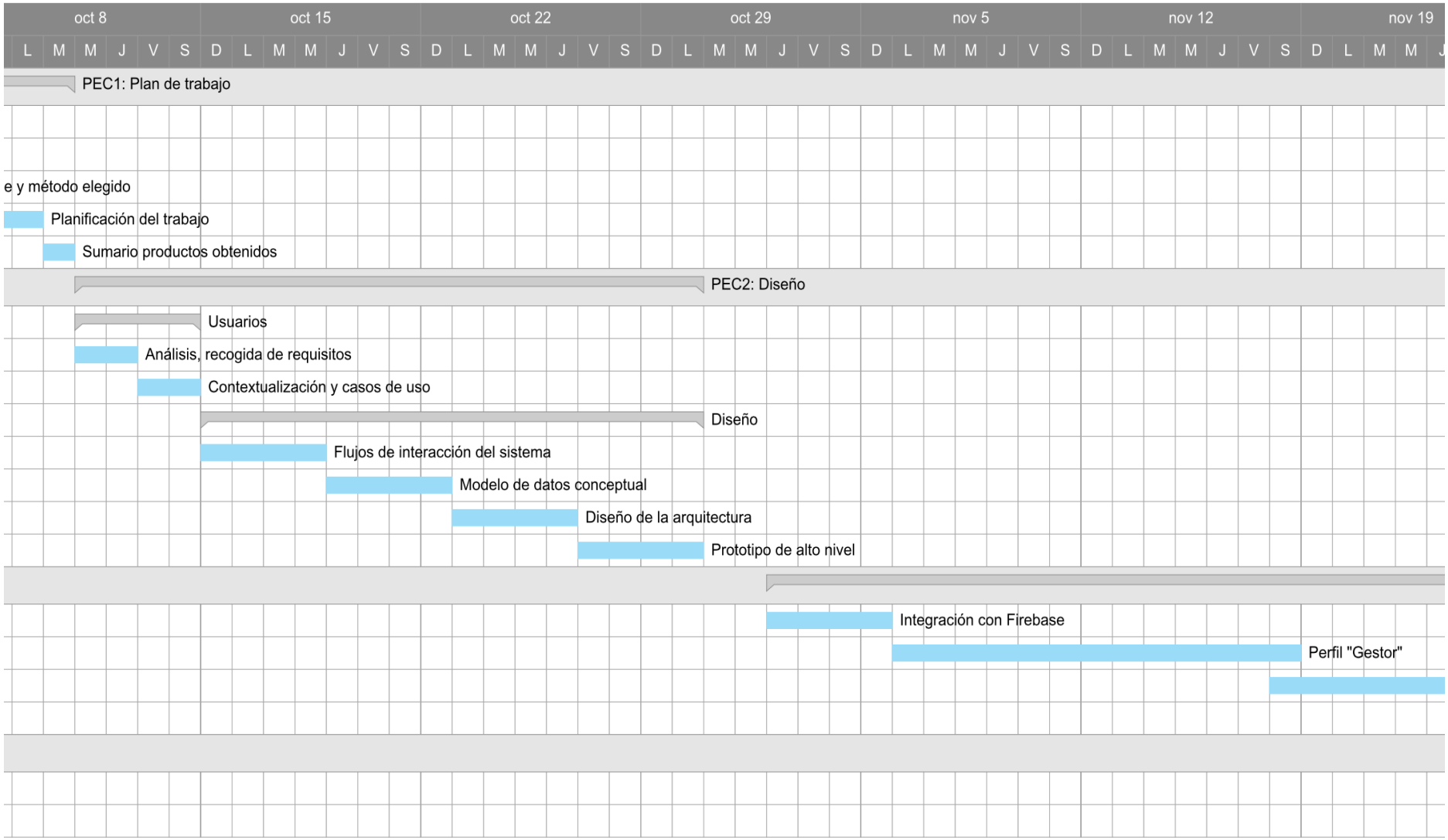
(3) Scrum
https://www.softeng.es/es-es/empresa/metodologias-de-trabajo/metodologia-scrum.html

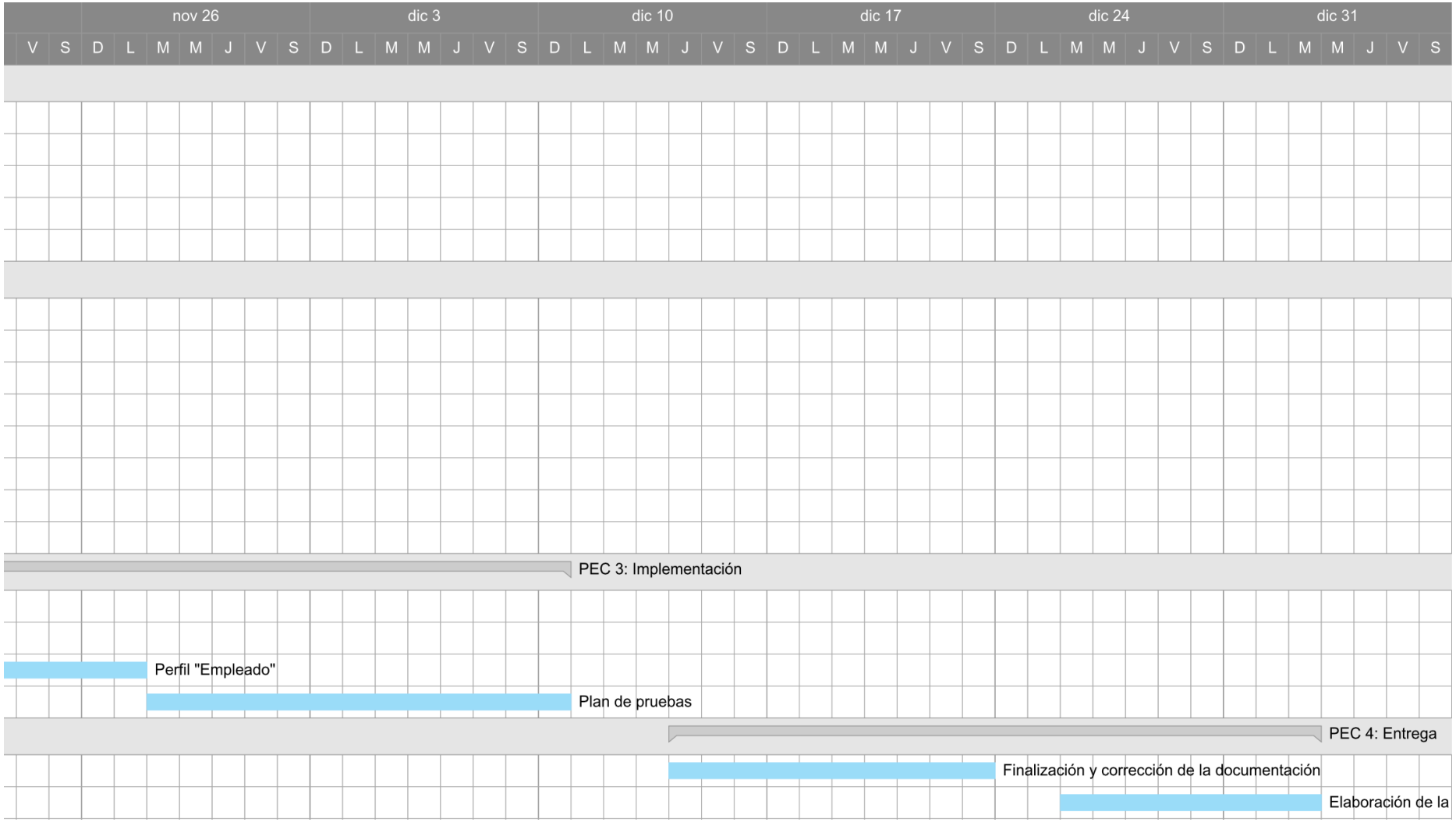
(4) Kanban
http://www.javiergarzas.com/2011/11/kanban.html

(5) Material Design
https://developer.android.com/design/material/index.html

Anexo II Diagrama de Gantt

N horas	Nombre de la tarea	Fecha de Inicio	Fecha final
36	PEC1: Plan de trabajo	18/09/17	10/10/17
10	Contexto y justificación del trabajo	18/09/17	24/09/17
10	Objetivos del trabajo	24/09/17	01/10/17
8	Enfoque y método elegido	01/10/17	06/10/17
7	Planificación del trabajo	07/10/17	09/10/17
1	Sumario productos obtenidos	10/10/17	10/10/17
34	PEC2: Diseño	11/10/17	30/10/17
7	Usuarios	11/10/17	14/10/17
4	Análisis, recogida de requisitos	11/10/17	12/10/17
3	Contextualización y casos de uso	13/10/17	14/10/17
27	Diseño	15/10/17	30/10/17
7	Flujos de interacción del sistema	15/10/17	18/10/17
6	Modelo de datos conceptual	19/10/17	22/10/17
6	Diseño de la arquitectura	23/10/17	26/10/17
8	Prototipo de alto nivel	27/10/17	30/10/17
62	PEC 3: Implementación	02/11/17	10/12/17
8	Integración con Firebase	02/11/17	05/11/17
18	Perfil "Gestor"	06/11/17	18/11/17
16	Perfil "Empleado"	18/11/17	27/11/17
20	Plan de pruebas	28/11/17	10/12/17
24	PEC 4: Entrega	14/12/17	02/01/18
16	Finalización y corrección de la documentación	14/12/17	23/12/17
8	Elaboración de la presentación	26/12/17	02/01/18


oct 8
oct 15
oct 22
oct 29
nov 5
nov 12
nov 19
L M M J V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S D L M M J
PEC1: Plan de trabajo
e y método elegido
Planificación del trabajo
Sumario productos obtenidos
PEC2: Diseño
Usuarios
Análisis, recogida de requisitos
Contextualización y casos de uso
Diseño
Flujos de interacción del sistema
Modelo de datos conceptual
Diseño de la arquitectura
Prototipo de alto nivel
Integración con Firebase
Perfil "Gestor"


nov 26
dic 3
dic 10
dic 17
dic 24
dic 31
V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S
PEC 3: Implementación
Perfil "Empleado"
Plan de pruebas
PEC 4: Entrega
Finalización y corrección de la documentación
Elaboración de la

Anexo III Estructura de datos almacenados en firebase:


```
{
  "Bar" : {
    "-L0UfQ6CWKG5m-tesl8Q" : {
      "CATEGORIAS" : {
        "-L0UfjuzSGwHD7GxnQdJ" : {
          "activo" : true,
          "descCategoria" : "hamburguesas",
          "idCategoria" : "-L0UfjuzSGwHD7GxnQdJ",
          "productos" : {
            "-L0UfyR7XeB7R5ACwr2d" : {
              "activo" : true,
              "descProducto" : "perritos",
              "idCategoria" : "-L0UfjuzSGwHD7GxnQdJ",
              "idProducto" : "-L0UfyR7XeB7R5ACwr2d",
              "numImporte" : 2.5
            },
            "-L0Ug-JTLWg3vzf7MCHs" : {
              "activo" : true,
              "descProducto" : "simple",
              "idCategoria" : "-L0UfjuzSGwHD7GxnQdJ",
              "idProducto" : "-L0Ug-JTLWg3vzf7MCHs",
              "numImporte" : 3
            }
          }
        },
        "-L0UfutlmHirzzXRUwgH" : {
          "activo" : true,
          "descCategoria" : "bebidas",
          "idCategoria" : "-L0UfutlmHirzzXRUwgH",
          "productos" : {
            "-L0Ug1rNQ8lsAanMCDDD" : {
              "activo" : true,
              "descProducto" : "cocacola",
              "idCategoria" : "-L0UfutlmHirzzXRUwgH",
              "idProducto" : "-L0Ug1rNQ8lsAanMCDDD",
              "numImporte" : 1.5
            }
          }
        }
      },
      "EMPLEADOS" : {
        "-L0UfS08aZHtoZGc73Et" : {
          "activo" : false,
          "descEmpleado" : “Manuel”,
          "idEmpleado" : "-L0UfS08aZHtoZGc73Et",
          "perfil" : 0,
          "perfiles" : [ {
            "activo" : true,
```

"descPerfil" : "Gestor",
"idPerfil" : 1
}, {
"activo" : true,
"descPerfil" : "Empleado",
"idPerfil" : 0
}]
},
"-L0UgFAX7CzTKANk8m2U" : {
"activo" : false,
"descEmpleado" : “Jesús”,
"idEmpleado" : "-L0UgFAX7CzTKANk8m2U",
"perfil" : 0,
"perfiles" : [{
"activo" : true,
"descPerfil" : "Empleado",
"idPerfil" : 0
}]
}
},
"PEDIDOS" : {
"-L0UgL5LvhPCgvYenWka" : {
"estadoPedido" : 1,
"idEmpleado" : "-L0UgFAX7CzTKANk8m2U",
"idPedido" : "-L0UgL5LvhPCgvYenWka",
"importeTotal" : 8.5,
"productos" : [{
"activo" : true,
"descProducto" : "cocacola",
"idCategoria" : "-L0UfutlmHirzzXRUwgH",
"idProducto" : "-L0Ug1rNQ8lsAanMCDDD",
"numImporte" : 1.5
}, {
"activo" : true,
"descProducto" : "cocacola",
"idCategoria" : "-L0UfutlmHirzzXRUwgH",
"idProducto" : "-L0Ug1rNQ8lsAanMCDDD",
"numImporte" : 1.5
}, {
"activo" : true,
"descProducto" : "simple",
"idCategoria" : "-L0UfjuzSGwHD7GxnQdJ",
"idProducto" : "-L0Ug-JTLWg3vzf7MCHs",
"numImporte" : 3
}, {
"activo" : true,
"descProducto" : "perritos",
"idCategoria" : "-L0UfjuzSGwHD7GxnQdJ",
"idProducto" : "-L0UfyR7XeB7R5ACwr2d",
"numImporte" : 2.5

```
        } ]
      }
    },
    "idBar" : "-L0UfQ6CWKG5m-tesl8Q",
    "nombreBar" : “Gastro Burguer"
  }
 }
}
```