Desarrollo de una aplicación móvil para búsquedas de menús alimenticios según intereses y basados en opiniones

---

Mobile application development for searching food menus according to interests and based on public opinions

Miriam Patricia Choy Castillo - Ingeniería del Software
Juan Pedro García Castaño - Ingeniería del Software
Pablo Sáenz Bullón - Ingeniería Informática

FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

Trabajo de fin de grado en Ingeniería del Software e Ingeniería Informática

Madrid, junio de 2022

Directores:

Bernabé García, Sergio
Muñoz Fernández, Emilio José

Agradecimientos

Primero de todo, gracias a nuestros familiares y seres queridos que siempre han estado presentes en todos los momentos durante estos años de carrera, apoyándonos psicológica y emocionalmente, sobre todo en los últimos empujones que es cuando uno necesita más apoyo.

Gracias a todas y cada una de las personas que nos hemos encontrado providencialmente a lo largo de la carrera. Por su inestimable ayuda y valioso compañerismo en muchas asignaturas, pero también gracias a su alegría y compañía hemos ido avanzando poco a poco cada día que ha conformado nuestra trayectoria universitaria.

También queremos agradecer a *Stack Overflow* que ha sido una fuente inagotable de resolución de dudas a lo largo de toda la carrera, nuestro pilar siempre disponible para apoyarnos.

Y, por último pero no por ello menos importante, queremos dar las gracias a nuestros tutores Sergio y Emilio por la comprensión, dedicación y disposición que siempre han mostrado con nosotros para poder llevar a cabo este Trabajo de Fin de Grado (TFG). Gracias por dedicarnos vuestro tiempo para las distintas reuniones que hemos tenido a lo largo de este proyecto y así obtener mejores resultados gracias al *feedback* continuo que nos proporcionabais en cada sesión.

Índice general

Índice de figuras

Índice de tablas

Resumen

Este Trabajo de Fin de Grado (TFG), tiene como objetivo el desarrollo de una aplicación móvil que realice búsquedas de restaurantes según los intereses del usuario, expresados a través de filtros. En caso de indecisión por los locales disponibles, los usuarios cuentan con reseñas de los mismos por parte de otros clientes, que pueden servir de apoyo a la hora de elegir.

La aplicación móvil se ha desarrollado para ser utilizada en el sistema operativo *Android* y estará en continua comunicación con una base de datos donde se almacenarán todos los datos del perfil, incluidas las reseñas mencionadas, que quedarán sincronizadas para todos los usuarios.

Además de esto, la *app* utiliza elementos *open source* como *OpenStreetMap*, para la obtención de datos en base a la localización del usuario y los filtros que haya aplicado, y cuenta con la implementación de *web scraping* para obtener más información acerca del local que se está consultando (platos de temporada, rangos de precios, etc).

Palabras clave

Filtro, reseña, restaurante, menú, carta, dieta, aplicación móvil, *Android*, *web scraping*, *OpenStreetMap*.

Lista de acrónimos

API - Application Programming Interface o Interfaz de Programación de Aplicaciones

CPU - Central Processing Unit o Unidad Central de Procesamiento

CSS - Cascading Style Sheets o Hojas de Estilo en Cascada

DOM - Document Object Model o Modelo de Objetos del Documento

GPS - Global Positioning System o Sistema de Posicionamiento Global

HTML - HyperText Markup Language o Lenguaje de Marcas de Hipertexto

JSON - JavaScript Object Notation o Notación de Objetos de JavaScript

NFC - Near Field Communication o Comunicación de Campo Cercano

NOSQL - Base de Datos no SQL o Base de datos no Relacional

OSM - OpenStreetMap

SDK - Software Development Kit o Kit de Desarrollo de Software

SQL - Structured Query Language o Lenguaje de Consulta Estructurada

UID - Unique User Identifier o Identificador Único de Usuario

UML - Unified Modeling Language o Lenguaje Unificado de Modelado

XML - Extensible Markup Language o Lenguaje de Marcado Extensible

Abstract

This Final Degree Project aims to develop a mobile application for searching restaurants according to the user's interests, expressed through filters. In case of indecision for available restaurants, users have reviews of them by other clients, which can serve as support when choosing.

The mobile application has been developed to be used in the Android operating system and will be in continuous communication with a database where all the profile data will be stored, including the mentioned reviews, which will be synchronized for all users.

In addition to this, the app uses open source elements such as OpenStreetMap, to get data based on the user's location and the filters applied, and also it has the implementation of web scraping to get more information about the place being consulted (seasonal dishes, price ranges, etc).

Keywords

Filter, review, restaurant, menú, food menu, diet, mobile application, Android, web scraping, OpenStreetMap.

Capítulo 1

Introducción

Debido a la globalización, el catálogo de restaurantes, comidas, hábitos alimenticios y regímenes es más amplio que nunca: desde la comida de distintos países hasta la gran variedad de bebidas, pasando por menús específicos según las condiciones del cliente (con o sin gluten), y llegando a tendencias alimentarias cada vez más presentes en el panorama gastronómico (veganismo, vegetarianismo, etc).

La diversidad es tan apabullante que pueden surgir bastantes dificultades a la hora de encontrar la comida que nos apetece, convirtiendo esto en una tarea incómoda y complicada. Por ello, es necesaria una solución que haga más fácil este proceso, sin olvidar la gran variedad de tipos de comida y gustos existentes mencionados anteriormente.

Cuando salimos a comer, queremos ir al sitio que mejor se ajuste tanto a nuestro presupuesto como a nuestro gusto, así que solemos ir a sitios de confianza o que nos han recomendado amigos o familiares. Sin embargo, cuando ya no queremos acudir a los establecimientos de siempre o nos encontramos haciendo turismo en otras localidades, tenemos que averiguar cuáles son los mejores restaurantes del lugar, momento en que el dispositivo móvil se vuelve nuestro mayor aliado.

1.1. Motivación

La tendencia que tiene el ser humano de buscar en *Internet* cada problema con el que se encuentra ha aumentado exponencialmente desde el surgimiento de la *web* y el proceso de

encontrar un sitio para comer no iba a ser diferente. Los diversos establecimientos se han dado cuenta de esto y, durante los últimos años, han promovido el desarrollo de páginas *web* no solo más atractivas para el cliente, sino que también aportasen toda la información necesaria a las aplicaciones de *delivery* para incluirlas en sus servicios.

Con esta idea en mente, se han aprovechado todas las ventajas que nos ofrece la tecnología hoy en día y se han tenido en cuenta las necesidades que demanda el mercado para crear una herramienta que resulte útil a la sociedad. De esta forma, ofrecemos *Food Feeltr*, una aplicación móvil que muestra los restaurantes en base a las preferencias de los distintos tipos de usuarios, así como parte de los contenidos disponibles en las *webs* de los establecimientos (imágenes de los platos disponibles, precios de los platos ofrecidos, etc) para hacer más sencilla la toma de decisión.

1.2. Objetivos

El objetivo principal del Trabajo de Fin de Grado (TFG) es la implementación de una aplicación móvil que recoja los datos sobre las preferencias de los distintos usuarios para, posteriormente, ofrecerles la mayor cantidad de información disponible sobre los restaurantes más afines a sus gustos y necesidades. Para conseguirlo, a lo largo de todo el proceso se perseguirán las siguientes metas establecidas:

- **Obtener la información de los distintos filtros elegidos por el usuario:** con la finalidad de poder ofrecer los mejores resultados.
- **Registrar reseñas:** las cuales proporcionan al usuario una idea general sobre el restaurante que le interese en base a las valoraciones y experiencias de otros comensales.
- **Registrar restaurantes favoritos:** con el propósito de que el consumidor consulte los restaurantes que más le interesa de una forma más rápida.
- **Modificar la reseña:** dándole al usuario la posibilidad de cambiar su opinión hacia un restaurante.

- **Utilización sin registro:** para que los usuarios reacios a proporcionar información personal, como puede ser un correo electrónico, puedan hacer uso de la *app*.

- **Open source:** la aplicación utilizará, siempre que sea posible, herramientas *open source* para que cualquiera que lo desee pueda comprender de una manera más sencilla el funcionamiento de la misma. Esto se puede implementar utilizando herramientas de obtención de datos como *OpenStreetMap (OSM)*, los cuales son generados por los usuarios de su comunidad y no se ven afectados por agentes externos como podría ser el uso de publicidad para un mejor posicionamiento de un local determinado, siendo así un método fiable de comparación entre distintos restaurantes.

- **Web scraping:** se utilizará este sistema para una mayor y mejor obtención de datos no disponibles o desactualizados por parte de la herramienta de obtención de datos principal. Gracias a ello, se ofrecen modificaciones generadas en tiempo real en las cartas de los restaurantes, buscando en su *web* directamente ciertas palabras clave.

Cabe destacar que todo esto irá acompañado de una interfaz elegante y sencilla que resulte atractiva para aquellas personas que hagan uso de la *app*, sin abandonar su fácil manejo para hacerla accesible a todo tipo de consumidores.

1.3. Plan de trabajo

Todo trabajo serio debe de incluir una buena planificación de la labor a realizar según el tiempo disponible. Se ha planificado el tiempo en tres fases principales: investigación y estrategia, desarrollo y, por último, despliegue. Dentro de cada una de estas fases hay distintas tareas que acometer. Por ejemplo antes de elegir los objetivos y construir los casos de uso hay que analizar la competencia para ver qué podemos ofrecer nuevo y, posteriormente, realizar el diseño.

De igual manera en la fase de desarrollo hay que distinguir la construcción de la parte *front* con la parte *back*, además de la base de datos. Y, por otro lado, hay que incluir una

parte muy importante como es el *testing*, para ir comprobando a lo largo de todo el proceso que las distintas utilidades de la aplicación están funcionando correctamente.

La fase final se compone de tres tareas principales: el despliegue real de la aplicación, el análisis de resultados y la realización de esta memoria, que si bien se ha ido pensando durante todo el trabajo, en este periodo es el momento de completarla. El código de este TFG se encuentra disponible en el siguiente repositorio de *GitHub*[1].

Para mostrarlo de una forma más gráfica se ha diseñado un diagrama de *Gantt*, visible en la Figura 1.1, donde se ha definido de forma visual la duración y las tareas a cumplir para el correcto cumplimiento de los plazos.

El diagrama de *Gantt* es una herramienta muy útil para planificar cualquier tipo de proyecto. Se muestra en una vista general todas la tareas programadas de manera ordenada junto con su seguimiento a lo largo del tiempo. Este diagrama se ha realizado en una página *web* especializada en estos diagramas: *Gantt.io*[2].

1.4. Organización de esta memoria

La memoria está organizada según los estándares propuestos para la realización de proyectos de este calibre. Esta sección describirá breve y ordenadamente el contenido de cada capítulo para poder aclarar la estructura de la memoria:

- **Capítulo 1 - Introducción:** se incluye un pequeño resumen de lo que es el proyecto en sí y los objetivos que se pretenden cumplir. Se explica de dónde y por qué sale la idea, así como lo que se pretende conseguir con el TFG. También se describen las tareas en un marco temporal, mediante un diagrama de *Gantt*.

- **Capítulo 2 - Estado del Arte:** se hace una comparativa de las tecnologías utilizadas con otras alternativas disponibles en el mercado. Además se muestran algunos ejemplos de uso reales.

[1] https://github.com/juanpegc/appMenus
[2] https://www.gantt.io/

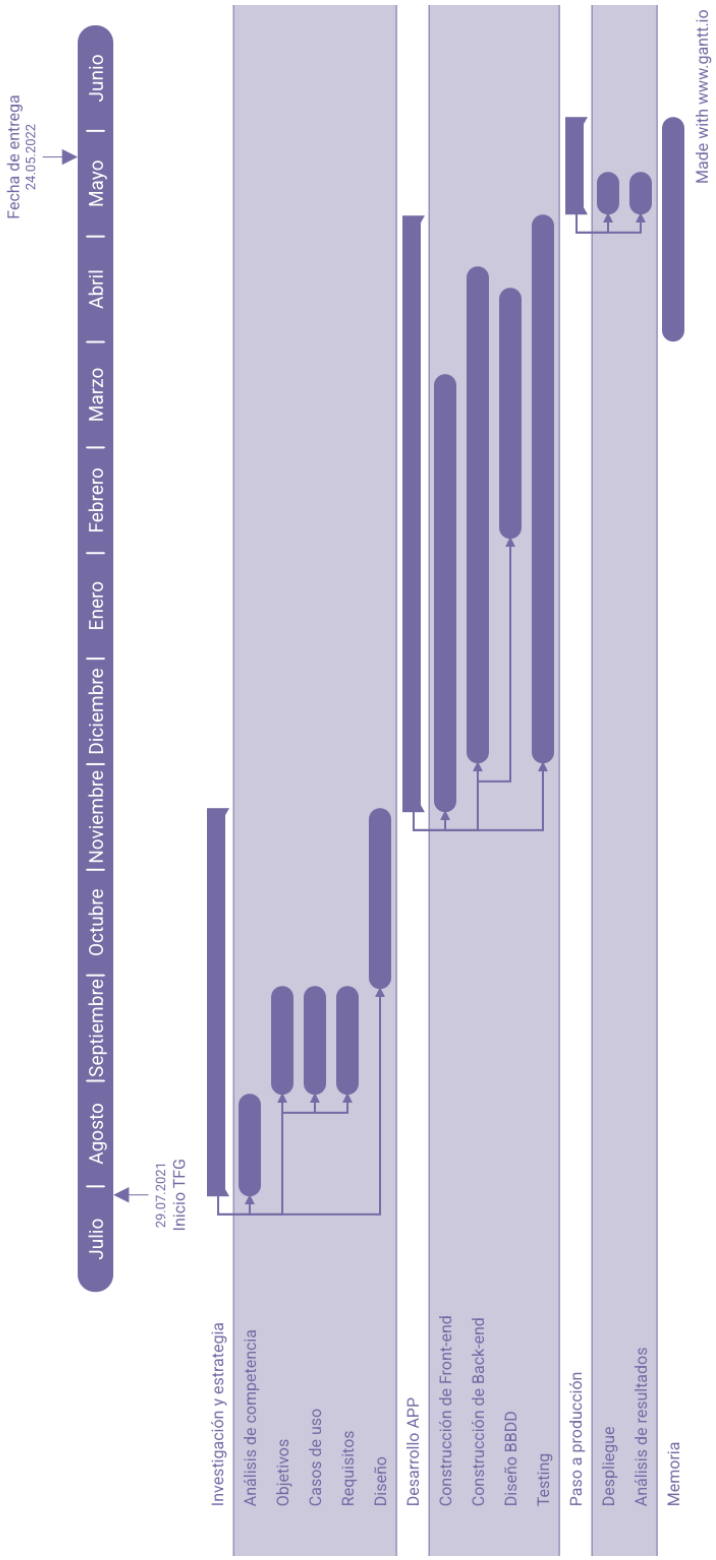

Figura 1.1: *Diagrama de Gantt.*

- **Capítulo 3 - Diseño de la aplicación:** se detalla de manera gráfica los actores involucrados junto con su diagrama de caso de uso. Además, se habla tanto del diseño de la interfaz gráfica como de la base de datos. También se mencionan las metodologías y herramientas usadas para la recolección de información, su transformación y la posterior introducción en la base de datos.

- **Capítulo 4 - Implementación de la aplicación:** se explica el diseño tanto *front-end* como en *back-end*. También se detalla en profundidad el *framework* utilizado y el modelo elegido, exponiéndose el resultado final a través de diferentes imágenes.

- **Capítulo 5 - Resultados obtenidos:** se analizan los resultados obtenidos de las pruebas técnicas realizadas del uso de la aplicación y las opiniones recibidas de los usuarios que la han podido probar.

- **Capítulo 6 - Conclusiones y Trabajo Futuro:** se comentan las conclusiones y consecución de objetivos del proyecto y algunas de las ideas que no han podido ser realizadas y podrían ser interesantes para un trabajo futuro.

Capítulo 2

Estado del arte

Una vez definidas la motivación y el objetivo de este proyecto, se van a analizar aquellas aplicaciones que cuenten con características similares a la que queremos crear y que sigan actualmente en el mercado.

Para mostrarlo de manera visual se hará una tabla comparativa: primero se estudiarán los diferentes sistemas operativos en los que se podría desarrollar, para desplegarla en el más adecuado; posteriormente, se prestará atención a los puntos fuertes y débiles de éstos y el porqué de *Android* como elección final; a su vez, se analizarán los dispositivos móviles compatibles con el sistema operativo citado que se encuentren disponibles actualmente en el mercado.

2.1. Sistemas operativos para dispositivos móviles

Actualmente, hay una variedad muy definida de sistemas operativos para móviles: lo que en su origen fue un mercado en expansión con gran variedad de los mismos, hoy en día se ha convertido en una simple bifurcación entre dos opciones. Todos conocen los más usados, véase *Android e iOS* [1]. Pero también hay otros como *Windows, KaiOS, Symbian OS* o *Ubuntu Touch*.

Sin embargo, como se puede observar en la Figura 2.1, los sistemas operativos más utilizados a día de hoy son *Android* (en primer lugar, con un 71 % de los dispositivos) e *iOS*, propiedad de *Apple* (con un 28 %).

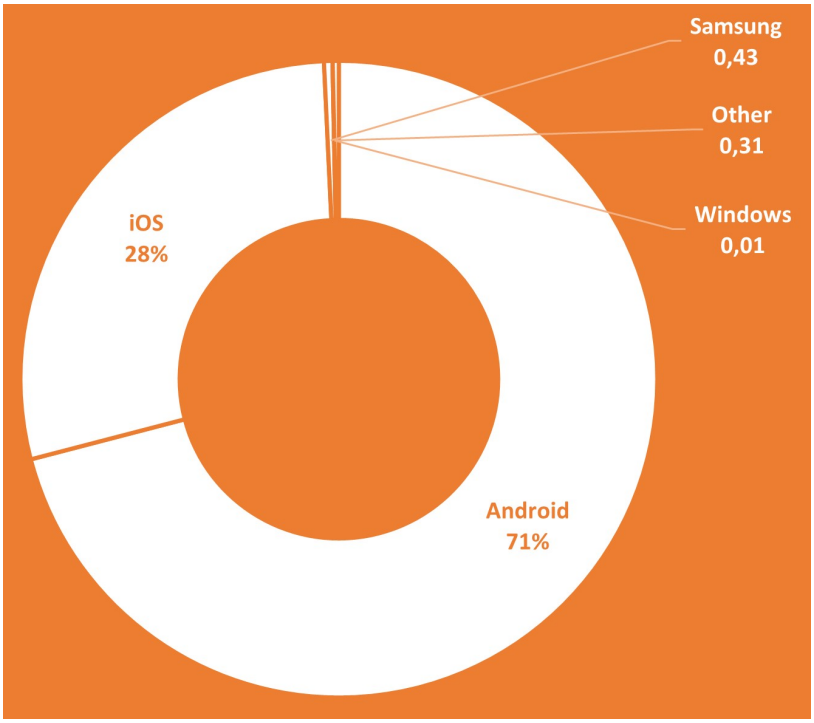

Figura 2.1: *Gráfico de abril de 2022 sobre el mercado de sistemas operativos para móvil. Fuente:* https://gs.statcounter.com

La opción de desarrollar la aplicación en otro sistema operativo que no fuera ninguno de estos dos se ha descartado por su reducido mercado actual. *Android e iOS* engloban prácticamente la totalidad de usuarios con *smartphone* y cuentan con una gran comunidad de desarrolladores para las dos plataformas.

2.1.1. Android

Es el sistema operativo con la cuota de mercado más alta a día de hoy (cuyo logotipo se muestra en la Figura 2.2), por lo que marcas de dispositivos muy conocidas, véase *Samsung, Xiaomi, Oppo* o *OnePlus*, hacen uso de él (coloquialmente se dice que todo dispositivo que no es de Apple tendrá instalado Android).

La última versión disponible es la 12 y, como se mencionó anteriormente, *Android* cuenta con una gran comunidad de desarrolladores que trabajan creando aplicaciones para aportar más funcionalidad a los dispositivos. Aunque a la hora de considerar este sistema operativo

Figura 2.2: *Logotipo de Android.*

como una de las opciones para implementar la aplicación, es más recomendable fijarse en aspectos como la cuota que tienen que pagar estos desarrolladores. El precio es muy reducido pues en este caso solamente hay que pagar 25$ en una cuota única para inscribirse como desarrollador de *Google Play*, la tienda oficial de *Android*.

Las aplicaciones en *Android* son programadas usando el *SDK*(*Software Development Kit*) y normalmente en *Java* [2]. En 2017 *Google* eligió *Kotlin* (desarrollado por *JetBrains*) como lenguaje de programación oficial de *Android*, dándole preferencia respecto a *Java*, aunque este se sigue soportando hoy en día [3].

Android Studio [4] (basado en *Intellij IDEA*) es el entorno de programación principal para desarrollo de *Android*, también lanzado por *Google*. Actualmente existe una gran cantidad de documentación en *Internet* por ser el entorno oficial de desarrollo para aplicaciones *Android*. Además de su editor de código y el sistema de compilación basado en *Gradle* cuenta con un emulador muy completo y cargado de funciones. Destaca además su modo de *debug*. Otra de las ventajas que tiene *Android Studio* es su conseguida integración con *Git* [5] donde se puede visualizar en una línea de tiempo todos los *commits*.

2.1.2. iOS

Es el sistema operativo de la multinacional *Apple*. La compañía que fundó Steve Jobs desarrolló este entorno con el que operan todos los dispositivos móviles desarrollados por *Apple*. Al principio solamente era para el *iPhone* pero después se extendió a otros dispositivos como *iPad* o *iPod*, aunque *Apple* no permite su instalación en *hardware* que no sea propio de la empresa. La última versión de este sistema operativo es *iOS* 15.

A la hora de desarrollar aplicaciones para *iOS*, cuyo logotipo es el mostrado en la Figura 2.3, el panorama cambia considerablemente con respecto a *Android*. Para desarrollar en

iOS es necesario el *iOS SDK*, el cual permite el desarrollo de aplicaciones para todos los dispositivos de *iOS* y dispone de un emulador de *iPhone*. El *SDK* sólo está disponible si eres usuario de *Macintosh*. Los lenguajes más utilizados son *Objetive-C o Swift* y el entorno de desarrollo es *XCode*. Al igual que el *SDK, XCode* se encuentra disponible gratuitamente para usuarios de *Macintosh* y no está disponible para los usuarios de *Windows*.

El coste de ser desarrollador de aplicaciones *iOS* es bastante más elevado que el de *Android*. El precio anual por utilizar el *Apple Developer Program* es de 99$ (esto es lo que se necesita si solamente quieres publicar una aplicación en la tienda oficial de Apple) y 299$ el *Apple Developer Enterprise Program*.

Figura 2.3: *Logotipo de iOS.*

2.1.3. Elección de sistema operativo

La decisión sobre para qué sistema operativo desarrollar la aplicación ha sido muy sencilla. Principalmente se han observado dos razones:

- **Lenguaje:** como se ha visto anteriormente el lenguaje oficial de desarrollo de *Android* es *Kotlin*. Este lenguaje no se ve en ninguna asignatura de la carrera, ni siquiera en la optativa *PAD* (Programación de Aplicaciones para Dispositivos móviles). La otra alternativa existente es *Java* [6]. Este es un lenguaje que, gracias a los conocimientos recibidos en distintas asignaturas como son Tecnología de la Programación o Ingeniería del *Software*, nos sentimos muy cómodos desarrollando en él. Las opciones de *iOS*

(Objetive-C y Swift) son opciones que al igual que *Kotlin* no podemos considerar al tener conocimiento nulo en estos lenguajes.

- **Incompatibilidades:** a parte de la razón monetaria hay que tener en cuenta que los tres miembros que formamos parte de este TFG somos usuarios de *Windows* y por lo tanto hacer uso del *SDK* de *iOS* y de su *IDE* nos hubiera sido imposible al ser alternativas diseñadas únicamente para usuarios de *Macintosh*.

Es por estas razones que la aplicación estará disponible solamente para usuarios de *Android*. Se reducirá así el número de usuarios que puedan usarla. Esta decisión ha permitido un mayor rendimiento y más rapidez. Un trabajo mucho más satisfactorio que si se si hubiera tenido que aprender otro lenguaje de programación o haber tenido que adquirir un *Macintosh*.

2.2. Dispositivos móviles

La variedad de dispositivos portátiles que se encuentran en el mercado actualmente es muy grande aun solamente deteniéndose en los dispositivos móviles, pues observando la aparición de los teléfonos recientes que se pueden doblar se da uno cuenta de que las posibilidades son infinitas. Ya si abrimos la mirada nos encontramos con dispositivos tan diferentes como *tablets*, *smartwatches* o *smartbands*. A continuación revisaremos los detalles de cada tipo de dispositivo:

- **Smartphones:** actualmente en España hay más teléfonos móviles que personas, es el dispositivo más extendido. Su tamaño es muy estándar y suele variar entre cuatro y siete pulgadas. Hoy en día el *smartphone* es una de las cosas que más llevamos con nosotros al salir de casa, con la evolución de la tecnología este dispositivo ha llegado a ser un objeto indispensable para la sociedad gracias a la gran cantidad de funciones que posee, lo que facilita y ayuda a muchos usuarios en todas sus tareas del día a día.

- **Tablets:** esencialmente las *tablets* son *smartphones* pero con una pantalla más grande. La principal diferencia que tenían con respecto a los teléfonos móviles era que no se podía llamar, pero hoy en día hay muchos de estos dispositivos que también cuentan con la facultad de poder hacer y recibir llamadas, mensajes e incluso tener acceso a *Internet*. El hecho de que tengan una pantalla más grande hace ciertas experiencias sean más disfrutables, como el consumo de vídeos, juegos o incluso para organizar las tareas de la universidad o del trabajo. Muchos de ellos cuentan con un lápiz táctil que hace que la experiencia sea muy parecida a estar con una hoja de papel en tus manos pero con muchas más capacidades.

- **Smartbands:** son pulseras inteligentes que están centradas en la monitorización de actividad. Se llevan en la muñeca y pueden hacer las veces de reloj pero añadiendo más funcionalidades, como por ejemplo medidor de pulsaciones, gestor de actividad deportiva, monitorización del sueño, notificación de eventos. Todo ello logrado generalmente mediante la conexión *Bluetooth* con el *smartphone*.

- **Smartwatches:** la evolución de los relojes ordinarios añadiendo prácticamente todas las funcionalidades de las *smartbands* e incluyendo otras nuevas. El tamaño no es ni demasiado grande ni demasiado pequeño y hace que la interacción táctil con ellos sea cómodo y agradable. Disponen de funciones tan interesantes como la recepción de las notificaciones del teléfono o el pago con la conectividad *NFC (Near Field Communication)*.

2.2.1. Elección del dispositivo

Una vez vistos todos los dispositivos que se encuentran hoy en en día en el mercado se ha estudiado la posibilidad de incluir la aplicación en alguno de ellos. Está claro que una aplicación de este estilo no sería posible en las *smartbands* ni tampoco en los *smartwatches* por la pantalla tan pequeña que disponen, incluso hay algunos de estos dispositivos que no disponen de pantalla como tal. De igual manera hacer esta *app* compatible con *smartwatches*

sería muy complicado y no valdría la pena ya que haría la experiencia de usuario muy complicada al tener que introducir todas las opciones disponibles y visualizar la información en una pantalla tan pequeña.

Por lo tanto la decisión final se situó entre las *tablets* y los *smartphones*. Sin duda la implementación en una *tablet* sería algo muy positivo porque la pantalla grande haría muy fácil la navegación por los componentes y las diferentes vistas. Por otro lado ya que este proyecto está orientado a una elección rápida e improvisada de un restaurante donde comer, no sería cómodo el tomar la decisión con una *tablet* por la calle, sino más bien en el domicilio. Como se puede entender esta no suele ser la situación en la que se encuentra una persona que quiere comer fuera de casa y está decidiendo el lugar.

Es por todos estos motivos anteriores que el dispositivo elegido es el *smartphone*, aun siendo posible su utilización en una *tablet Android*. El tamaño de los teléfonos móviles actuales y la posibilidad de llevarlos siempre en el bolsillo, hace que sea sencillo para el usuario disponer de la aplicación en cualquier momento, lugar y situación.

2.3. Aplicaciones para la búsqueda de menús alimenticios

En este apartado se van a analizar las aplicaciones disponibles para *Android* que tienen similitud a la aplicación que se va a desarrollar en este proyecto. A continuación se detallarán alguna de ellas como son *'Google Maps', 'El Tenedor', 'Trip Advisor' y 'Yelp'* y, como comparativa adicional, se puede consultar la Tabla 2.1.

- **Google Maps:** es una aplicación de navegación *GPS (Global Positioning System)* que viene instalada en la mayoría de los dispositivos móviles, esta *app*, cuyo logo se puede observar en la Figura 2.4, nos permite realizar consultas por distintos tipos de restaurantes (italiano, japonés, tapas, etc) y también es posible realizar filtros por valoración para visualizar los mejores locales del mapa. En la sección explorar se encuentran diferentes opciones para ir a comer, merendar, etc. En cada ficha del

Aplicaciones Similares					
	Food Feeltr	**Google Maps**	**Trip Advisor**	**Tenedor**	**Yelp**
Centrado en alimentación	Si	No	No	No	No
Múltiples filtros	Si	Si	Si	Si	No
Gratuito	Si	Si	Si	Si	Si
Búsqueda sin login	Si	Si	Si	Si	Si
Open source	Si	No	No	No	No

Tabla 2.1: *Tabla comparativa de aplicaciones con finalidades similares.*

restaurante se muestra su información, fotos, horas populares y reseñas.

Figura 2.4: *Logo de la aplicación Google Maps.*

- **Trip Advisor:** es la guía de viajes más popular del mundo, se puede ver su logo, mostrado en la Figura 2.5, en las puertas de muchos restaurantes y hoteles. Su aplicación permite seleccionar el tipo de establecimiento al que se quiere ir a comer y aplicar numerosos filtros para localizar el restaurante más cercano y que más se ajuste a las preferencias del usuario. Cada ficha de restaurante muestra su información, fotos y comentarios.

Figura 2.5: *Logo de la aplicación TripAdvisor.*

- **El tenedor:** es una aplicación que permite reservar en mas de 80.000 restaurantes de diversos tipos y distintos países, ofrece a los usuarios realizar reservas de una manera fácil sin necesidad de llamar y pudiendo beneficiarse de promociones y descuentos de entre el 20 % y el 50 %. A su vez, los restaurantes logran tener mayor visibilidad,

aumentar sus reservas, gestionar su sala fácilmente y fidelizar a los clientes. Su logo se muestra en la Figura 2.6.

Figura 2.6: *Logo de la aplicación el tenedor.*

- **Yelp:** Es una gran alternativa para encontrar cualquier tipo de negocio como salones de belleza, clínicas veterinarias, centros de *wellness* pero, sobre todo, para restaurantes y bares. Permite filtrar por restaurantes, ver si aceptan reservas, si hay alguna oferta vigente. Además te permite ordenar los resultados por valoración o distancia y te muestra los resultados en el mapa junto las valoraciones de su gran comunidad. Su logo es el mostrado en la Figura 2.7.

Figura 2.7: *Logo de la aplicación Yelp.*

La aplicación que vamos a desarrollar cuenta con muchas características compartidas con las ya comentadas. Sin embargo el valor que aporta nuestra aplicación reside principalmente en dos elementos: especialización y filtros.

Food Feeltr, a diferencia de las otras aplicaciones comentadas, es muy concreta, está únicamente centrada en alimentación. Una persona que quiera solamente buscar un lugar donde comer acudirá a la aplicación que más fácil y rápido pueda ofrecerle lo que busca. Por otro lado los filtros son el segundo gran elemento de *Food Feeltr*, pues se dispone de gran variedad de ellos, son muy fácilmente intercambiables y se pueden combinar para obtener unos resultados más específicos.

2.4. Herramientas para búsquedas de ubicación geográfica (Geolocalizadores)

Al inicio del desarrollo de este proyecto, la idea original fue utilizar *Google Maps* [7] para la búsqueda de los datos necesarios, así como la ubicación y que estos puedan ser mostrados debidamente a los usuarios de la aplicación. Debido a que *Google Maps* es una aplicación muy conocida por la inmensa mayoría de la población, muchos establecimientos de comida estarán registrados en este *software*, además de su fácil implementación y de que tiene un una gran compañía que es responsable de su mantenimiento, utilizarlo generaría un programa más resistente a errores.

Como se ha comentado anteriormente, esta era la idea original pero a la hora de poner en marcha la implementación de la aplicación nos encontramos con un problema: *Google* no permite su utilización de manera gratuita y somete su uso al pago de una cuota. *Google* proporciona 200$ mensuales para su uso, lo cual se traduce en aproximadamente 7000 búsquedas al mes. Aunque este número es más que suficiente para la realización de este proyecto, se hizo el estudio de otras herramientas posibles para que este *software* que se va a desarrollar, se pueda utilizar de manera gratuita pensando en el posible aumento futuro de usuarios.

Debido a esto se decidió utilizar *OpenStreetMap (OSM)* [8] ,que es un mapa de código abierto, editable por todo el mundo, compuesto por tres elementos básicos:

- ***Node:*** elemento del mapa definido por un único punto en el mismo, ha de tener una latitud y longitud así como un identificador único. Generalmente los nodos hacen referencia a lugares de interés, como monumentos o locales.

- ***Way:*** elemento del mapa que define una unión entre dos o más puntos (nodos), generalmente calles o carreteras.

- ***Relation:*** lista ordenada que contiene otros elementos (nodos, vías o relaciones) y la relación lógica o geográfica de los mismos.

OSM no cuenta con una gran compañía respaldando su funcionamiento en contraposición a *Google Maps*, pero ha sido utilizado en aplicaciones móviles muy exitosas como *Pokemon GO*, la cual implementa dicho mapa y, desde su lanzamiento, han animado a su comunidad de jugadores a mejorar los mapas, siempre de forma responsable. La implementación de *OSM* por parte de dicho juego tan popular ha implicado un aumento muy significativo del número de usos del mapa, así como de los elementos disponibles en el mismo, por lo cual se ha convertido en una herramienta con una gran cantidad de información disponible para su uso, de manera totalmente gratuita.

2.5. Búsqueda en páginas web: web scraping

Web scraping o raspado web en español, es un conjunto de técnicas desarrolladas que se utilizan para obtener información a partir de sitios *web* y realizar multitud de funcionalidades. Estas técnicas están basadas en la indexación, como las que realizan los motores de búsqueda de *Google* o *Bing*. Hay dos tipos de motores de búsqueda: los manuales y los automáticos. Por un lado, el *scraping* manual trata de copiar y pegar información concreta, por lo que es más laborioso. Por otro lado, el *scraping* automático se ayuda de *software* o algoritmos para el análisis y extracción de contenido *web*. Las utilidades del *web scraping* son casi ilimitadas, desde la caza de tendencias y optimización de precios hasta la monitorización de la competencia. En este proyecto se utilizan estas técnicas con el fin de extraer los datos de manera selectiva y presentarlos ordenadamente, ya que la mayoría de restaurantes muestran su información a través del formato *HTML* en sus páginas *web*. También remarcar que el *web scraping* es legal siempre y cuando los datos recabados estén disponibles libremente para terceros en la *web*.

Las diferentes opciones tenidas en cuenta para realizar el *web scraping* han sido las siguientes:

- ***JSoup*** [9]: biblioteca simple de código abierto que proporciona una funcionalidad muy útil para extraer y manipular datos usando el *DOM (Document Object Model)*, para

recorrer la jerarquía de nodos de *HTML* y también permite el uso de selectores *CSS* para encontrar datos basándose en elementos auxiliares, como etiquetas de clases. No acepta búsquedas mediante *XPath*.

- ***HTMLUnit***[1]***:*** biblioteca más potente que permite realizar interacciones como si se tratase de un navegador real, como *clicks* con el ratón o simular eventos en el navegador. Además tiene soporte para *JavaScript* y es compatible con *XPath*

- ***Jaunt***[2]***:*** biblioteca de *Java* para *web scraping*, automatización *web* y consultas *JSON*. El navegador proporciona funcionalidad de raspado *web*, acceso al *DOM* y control sobre cada solicitud/respuesta *HTTP*. Recientemente se ha añadido compatibilidad con *Javascript*.

Para este proyecto se optó por utilizar *JSoup* ya que para su uso no se requieren conocimientos muy avanzados de *web scraping*, ni disponer de un dispositivo potente (como puede ser un teléfono móvil). Tampoco se necesita el uso de elementos más avanzados como podría ser generar eventos de ratón o interpretar *Javascript*, dado que dichas técnicas más avanzadas solo se requieren si se desea desarrollar un *bot* o similar.

La finalidad que se le ha dado al uso de esta técnica en este proyecto ha sido obtener información puntual basada en determinados patrones fácilmente reconocibles. Por ejemplo, mediante *regex* para el reconocimiento de precios. Por todo ello se ha optado por utilizar esta herramienta. Así se consumirán un menor número de recursos *hardware* muy valiosos en los *smartphones*, como pueden ser la *CPU*, la memoria *RAM* y ocupará un menor espacio de almacenamiento.

[1] https://htmlunit.sourceforge.io/
[2] https://jaunt-api.com/

Capítulo 3

Diseño de la aplicación

En este capítulo se va a hablar de todo lo relacionado con el diseño de la aplicación. Se explicará tanto el componente funcional como la parte gráfica.

3.1. Arquitectura de la aplicación

El sistema que se ha utilizado en la aplicación ha sido, como se puede observar en la Figura 3.1, la arquitectura de cliente-servidor. El servidor almacena toda la información del cliente en la base de datos y el cliente accede a esos datos a través de *Internet*. El modelo se basa en que el cliente hace peticiones al servidor el cual le da las respuestas que necesita. En este caso no es solamente un dispositivo *Android* el que hace las peticiones si no que son todos los usuarios que estén usando la aplicación en ese momento. El servidor tiene que atender a todas las peticiones simultáneamente, es por eso que debe de tener capacidad para no verse saturado.

En la parte del dispositivo *Android* (o cliente), se necesita cumplir con unos requisitos. Primero de todo como es lógico hay que tener la aplicación instalada en el dispositivo. Además es necesario que el *GPS* del teléfono esté activado. Nada más abrir la aplicación se pedirá al usuario su permiso para activar la ubicación. Por último debe de haber acceso a *Internet*, independientemente de si es usando datos móviles o con *wifi*. En caso de no cumplirse alguna de estas condiciones no se podría garantizar el correcto funcionamiento de la aplicación.

Una vez se tienen las dos partes funcionando, el servidor se encarga de la autenticación de clientes, de la persistencia de datos y el almacenamiento de los mismos. Entre otras cosas, almacena información acerca de las reseñas, las valoraciones de los restaurantes y los filtros extras que han añadido los usuarios a los mismos. Esto es gracias a las funcionalidades nativas que tiene *Google Firebase* como se comentará en la Sección 4.2.

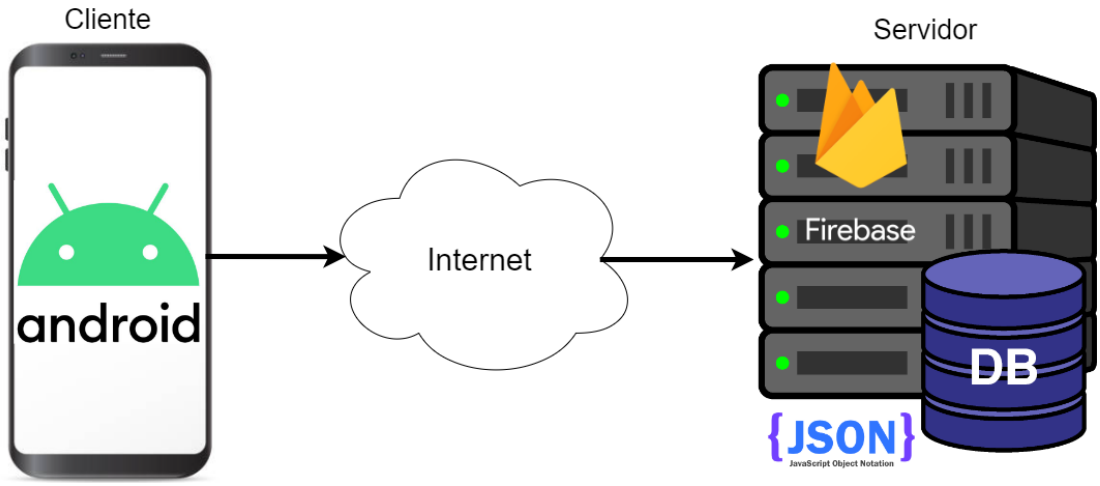

Figura 3.1: *Arquitectura del sistema.*

3.2. Fuente de información: OpenStreetMap

Para la utilización de *OSM* como fuente básica de información, se ha optado por utilizar la página *web Overpass Turbo* [10]. *OSM* no dispone de un método oficial de hacer búsquedas en base a ciertos criterios. En su lugar permite consultar su mapa haciendo *click* sobre el mismo. Debido a esto se ha recurrido a dicha *web* mencionada ya que permite de una forma relativamente sencilla, eficiente y gratuita realizar búsquedas específicas [11] en *OSM*.

Gracias a ella se puede seleccionar un gran número de filtros disponibles en *OSM* [12] y recibir los resultados en formato *XML* o *JSON* [13], habiendo optado por este último pues permite un manejo más sencillo de los datos recibidos y es compatible con nuestra base de datos en caso de necesitarse en algún momento del desarrollo.

Para realizar estas búsquedas se han realizado en nuevos *'threads'* del procesador, en parte por una mayor eficiencia al buscar varios datos de forma paralela a la ejecución del resto de la *app* y en parte por obligatoriedad de *Android*, ya que el sistema operativo exige que los procesos de gran coste de tiempo y *CPU* (como las búsquedas en *Internet*) se realicen en hilos diferentes del principal, para evitar que la aplicación se congele o dé esa sensación. Por ello cada vez que se realiza una nueva búsqueda, se procede a crear un nuevo hilo y a asignarle al mismo la tarea de construir la *query*, realizar la búsqueda e interpretar los datos recibidos.

3.3. Recolección de información extra

Como se ha comentado en la Sección 2.5, en este trabajo se ha utilizado el *web scraping* para obtener una mayor información acerca de los locales recibidos desde *OSM*. Por ello el funcionamiento del *web scraping* de esta aplicación va ligado al uso de *OSM*. Así cuando se recibe el resultado con todos los restaurantes que cumplen los criterios de búsqueda seleccionados por el usuario, se comienza a comprobar uno por uno si el local recibido contiene o no una página *web* y en caso de tenerla se inicia el proceso de buscar información extra en la misma.

Debido a que cada página *web* tiene un formato y organización diferente, no se puede garantizar que mediante esta técnica se puedan obtener más datos de interés. Por ejemplo hay *webs* cuyo único contenido es una imagen con la carta, de la cual no se puede buscar texto y hay otras donde la oferta alimenticia está tras varias pestañas de búsqueda.

La información general obtenida mediante esta técnica se divide en 3 categorías:

- **Imágenes:** se utilizan para que el usuario tenga un apoyo visual de la oferta alimenticia del local así como de otros elementos (en caso de encontrarse en la *web*) como pueden ser imágenes del restaurante o de la carta en caso de estar en este formato.

- **Filtros:** se utilizan para mejorar la información recibida de *OSM*, complementando así esos datos obtenidos con información en tiempo real de la oferta gastronómica del

local, ya que los resultados de *OSM* pueden no estar actualizados, mientras que las *webs* de los restaurantes suelen tener las últimas novedades culinarias del lugar.

- **Precios:** debido a que esta información no se encuentra disponible en *OSM*, se decidió buscar una manera de obtenerla y el resultado fue aplicar *web scraping*. Gracias a ello se puede ofrecer a los usuarios de la aplicación una manera de conocer el rango de precios del local, indicando el precio más bajo, el más alto y la mediana de los mismos, para que cualquier usuario, aun sin tener unos grandes conocimientos de matemáticas, pueda saber si puede comer en dicho local.

3.4. Casos de uso

La aplicación cuenta con un único actor que la utiliza, siendo este el usuario. Este actor podrá hacer uso de los siguientes casos de uso:

- **Buscar restaurantes,** reflejado en la Tabla 3.1. Muestra los pasos a realizar para buscar restaurantes aplicando filtros.
- **Aplicar filtros,** reflejado en la Tabla 3.2. Muestra los pasos a realizar para aplicar filtros.
- **Aplicar rango de búsqueda,** reflejado en la Tabla 3.3. Muestra los pasos a realizar para aplicar el rango de búsqueda.
- **Añadir a favoritos,** reflejado en la Tabla 3.4. Muestra los pasos a realizar para añadir un restaurante a favoritos.
- **Eliminar de favoritos,** reflejado en la Tabla 3.5. Muestra los pasos a realizar para eliminar un restaurante de favoritos.
- **Listar favoritos,** reflejado en la Tabla 3.6. Muestra los pasos a realizar para listar los restaurantes favoritos.

- **Añadir reseña,** reflejado en la Tabla 3.7. Muestra los pasos a realizar para añadir una reseña a un restaurante.
- **Modificar reseña,** reflejado en la Tabla 3.8. Muestra los pasos a realizar para modificar una reseña de un restaurante.
- **Mostrar reseña,** reflejado en la Tabla 3.9. Muestra los pasos a realizar para mostrar una reseña.
- **Listar reseñas,** reflejado en la Tabla 3.10. Muestra los pasos a realizar para listar las reseñas de un restaurante o de un usuario.
- **Mostrar restaurante,** reflejado en la Tabla 3.11. Muestra los pasos a realizar para mostrar los detalles de un restaurante.
- **Listar restaurantes,** reflejado en la Tabla 3.12. Muestra los pasos a realizar para listar los restaurantes de la búsqueda o marcados como favoritos.
- **Alta de usuario,** reflejado en la Tabla 3.13. Muestra los pasos a realizar para registrar a un nuevo usuario en la aplicación.
- **Baja de usuario,** reflejado en la Tabla 3.14. Muestra los pasos a realizar para eliminar a un usuario de la aplicación.
- **Edición de usuario,** reflejado en la Tabla 3.15. Muestra los pasos a realizar para modificar los datos de perfil del usuario.
- **Mostrar usuario,** reflejado en la Tabla 3.16. Muestra los pasos a realizar para mostrar los datos de un usuario público o del propio perfil.

Además de las tablas se incluye un diagrama *UML (Unified Modeling Language)* en la Figura 3.2. Se puede ver en ella todos los casos de uso divididos por entidades.

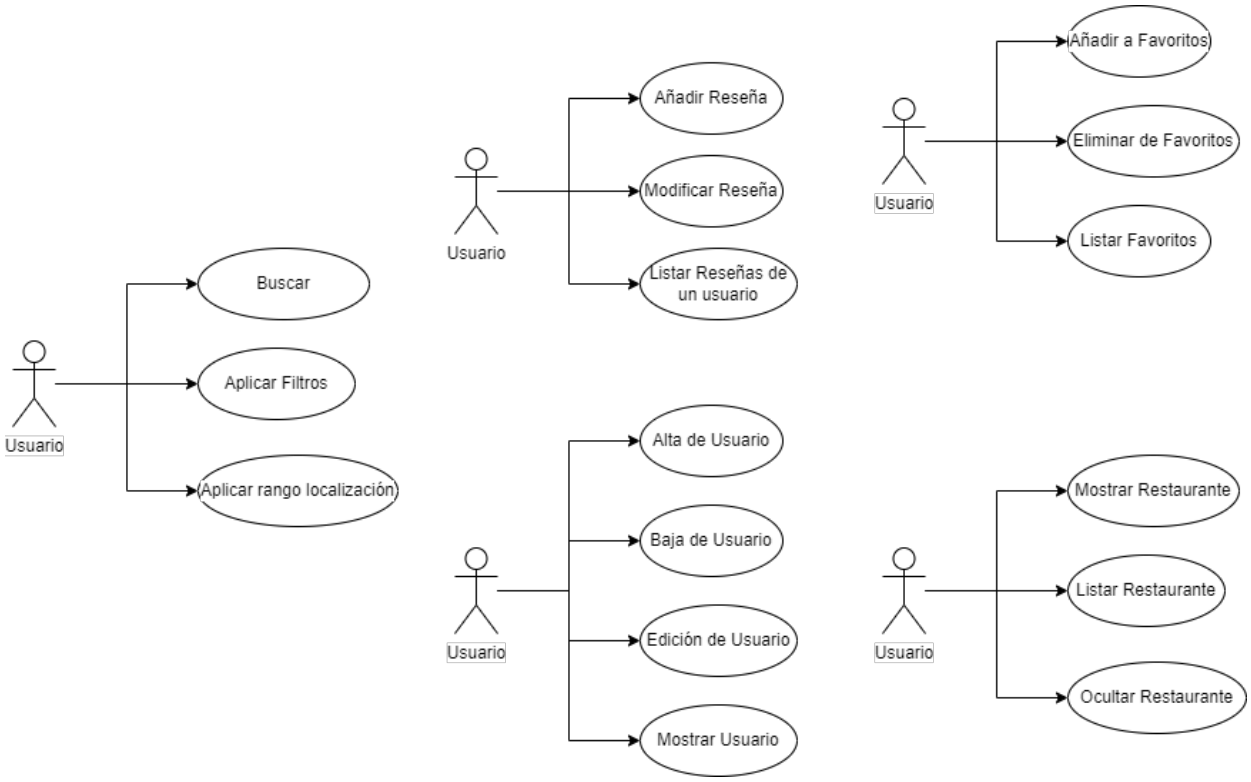

Figura 3.2: *Diagrama de casos de uso UML de Food Feeltr.*

CU-01	Buscar restaurantes
Autor	Usuario
Descripción	Buscar restaurantes con los filtros aplicados y la localización fijada
Entradas	Búsqueda Filtros seleccionados Localización
Salidas	Lista de restaurantes
Precondición	Tener activada la ubicación y haber aplicado al menos 4 filtros
Secuencia	Guardar los filtros aplicados Realizar la búsqueda en la base de datos y *API* según los filtros En caso de error con la *API* mostrar error En caso de no tener resultados mostrar error Mostrar lista de restaurantes
Postcondición	Se muestra una lista de restaurantes que cumplen con los filtros aplicados

Tabla 3.1: *Caso de uso 01: Buscar restaurantes.*

CU-02	Aplicar filtros
Autor	Usuario
Descripción	Aplicar los filtros sobre los que se va a realizar la búsqueda
Entradas	Filtros seleccionados
Salidas	Indicativo de filtro seleccionado
Precondición	Haber aplicado al menos 4 filtros
Secuencia	Mostrar los filtros seleccionables en el desplegable Marcar los filtros deseados
Postcondición	Se muestran los filtros aplicados y el número total

Tabla 3.2: *Caso de uso 02: Aplicar filtros.*

CU-03	Aplicar rango de búsqueda
Autor	Usuario
Descripción	Aplicar el rango de búsqueda que quiera determinar el usuario
Entradas	Rango a aplicar
Salidas	Indicativo del rango aplicado
Precondición	Tener activada la ubicación
Secuencia	Mostrar la ubicación del usuario Alterar el rango de búsqueda en kilómetros. Aplicar el rango seleccionado
Postcondición	Se aplica el rango seleccionado a las búsquedas

Tabla 3.3: *Caso de uso 03: Aplicar rango de búsqueda.*

CU-04	Añadir a favoritos
Autor	Usuario
Descripción	Añadir un restaurante a favoritos para tenerlo más accesible siempre
Entradas	Restaurante
Salidas	-
Precondición	Restaurante no añadido a favoritos
Secuencia	Mostrar información del restaurante Añadir a favoritos Mostrar mensaje de éxito
Postcondición	El restaurantes está marcado como favoritos y se encuentra en una lista

Tabla 3.4: *Caso de uso 04: Añadir a favoritos.*

CU-05	Eliminar de favoritos
Autor	Usuario
Descripción	Eliminar un restaurante que estaba añadido a favoritos
Entradas	Restaurante marcado como favorito
Salidas	-
Precondición	Restaurante añadido a favoritos
Secuencia	Mostrar información del restaurante Eliminar de favoritos Mostrar mensaje de éxito
Postcondición	El restaurantes ya no está marcado como favorito

Tabla 3.5: *Caso de uso 05: Eliminar de favoritos.*

CU-06	Listar favoritos
Autor	Usuario
Descripción	Listar en una misma pantalla todos los restaurantes que el usuario ha marcado como favoritos
Entradas	Restaurantes marcados como favoritos
Salidas	Lista de restaurantes
Precondición	Tener restaurantes marcados como favoritos
Secuencia	Marcar opción de ver restaurantes favoritos Ver lista de restaurantes favoritos
Postcondición	Los restaurantes se muestran en una lista

Tabla 3.6: *Caso de uso 06: Listar favoritos.*

CU-07	Añadir reseña
Autor	Usuario
Descripción	El usuario añade una reseña a un restaurante seleccionado
Entradas	Restaurante
Salidas	Reseña
Precondición	-
Secuencia	Ver restaurante Añadir reseña Rellenados todos los campos obligatorios de la reseña
Postcondición	Reseña añadida

Tabla 3.7: *Caso de uso 07: Añadir reseña.*

CU-08	Modificar reseña
Autor	Usuario
Descripción	El usuario modifica una reseña que había añadido a un restaurante alterando cualquiera de los campos
Entradas	Reseña
Salidas	Restaurante
Precondición	Que la reseña esté añadida
Secuencia	Ver reseña que se quiere modificar Modificar reseña
Postcondición	Reseña modificada

Tabla 3.8: *Caso de uso 08: Modificar reseña.*

CU-09	Mostrar reseña
Autor	Usuario
Descripción	Se muestra la descripción detallada de una reseña
Entradas	Lista de reseñas
Salidas	Reseña detallada
Precondición	Que la reseña exista
Secuencia	Ver lista de reseñas Mostrar detalles de reseña
Postcondición	Se muestra la reseña

Tabla 3.9: *Caso de uso 09: Mostrar reseña.*

CU-10	Listar reseñas
Autor	Usuario
Descripción	Hay dos listados, listado de todas las reseñas publicadas por el usuario y el listado de reseñas de un restaurante
Entradas	Reseñas de un usuario o de un restaurante
Salidas	Lista de reseñas
Precondición	Que las reseñas existan
Secuencia	Mostrar restaurante y ver sus reseñas O mostrar usuario y ver sus reseñas
Postcondición	Mostrar lista de reseñas

Tabla 3.10: *Caso de uso 10: Listar reseñas.*

CU-11	Mostrar restaurante
Autor	Usuario
Descripción	Se muestran todos los detalles del restaurante seleccionado
Entradas	Lista de restaurantes
Salidas	Restaurante
Precondición	Restaurante existente
Secuencia	Ver restaurante a seleccionar Mostrar detalles del restaurante
Postcondición	Se muestran los detalles del restaurante

Tabla 3.11: *Caso de uso 11: Mostrar restaurante.*

CU-12	Listar Restaurantes
Autor	Usuario
Descripción	Hay dos listados, listado de todos los restaurantes de la búsqueda y el listado de restaurantes favoritos
Entradas	Filtros de búsqueda
Salidas	Lista de restaurantes
Precondición	Restaurantes existentes o marcados como favoritos
Secuencia	Aplicar filtros de búsqueda O ver restaurantes favoritos Mostrar lista de restaurantes
Postcondición	Se muestra una lista de restaurantes

Tabla 3.12: *Caso de uso 12: Listar restaurantes.*

CU-13	Alta de usuario
Autor	Usuario
Descripción	Se da de alta a un nuevo usuario registrado
Entradas	Datos del nuevo usuario
Salidas	Usuario creado
Precondición	Usuario no existente
Secuencia	Usuario introduce sus datos Se comprueba que no exista El usuario selecciona sus primeros filtros
Postcondición	Se muestra una lista de restaurantes según filtros seleccionados

Tabla 3.13: *Caso de uso 13: Alta de usuario.*

CU-14	Baja de usuario
Autor	Usuario
Descripción	Se da de baja a un usuario registrado
Entradas	Datos del usuario
Salidas	Usuario eliminado
Precondición	Usuario existente
Secuencia	Usuario va a su perfil Selecciona darse de baja Introduce contraseña para confirmar Se da de baja al usuario
Postcondición	Se muestra la pantalla de inicio de sesión

Tabla 3.14: *Caso de uso 14: Baja de usuario.*

CU-15	Edición de usuario
Autor	Usuario
Descripción	Se modifican los datos del usuario
Entradas	Perfil del usuario
Salidas	Usuario modificado
Precondición	Usuario existente
Secuencia	Usuario va a su perfil Selecciona modificar sus datos Se modifican sus datos
Postcondición	Se modifican sus datos

Tabla 3.15: *Caso de uso 15: Edición de usuario.*

CU-16	Mostrar usuario
Autor	Usuario
Descripción	Se muestran los datos de un usuario
Entradas	Perfil de usuario
Salidas	Datos del usuario
Precondición	Usuario existente y perfil público (idea)
Secuencia	Se selecciona al autor de una reseña O se selecciona perfil de usuario Se comprueba que el perfil sea público Se muestran sus datos
Postcondición	Se muestran los datos

Tabla 3.16: *Caso de uso 16: Mostrar usuario.*

3.5. Diseño gráfico de la aplicación

En este apartado se explicará todo lo relacionado con diseño gráfico de la aplicación. Para ello se contará el origen y significado del logo junto con el diseño de la interfaz gráfica de la aplicación. Esto último de forma general, ya se explicará más adelante en profundidad.

3.5.1. Logo

Para decidir qué logo se iba a utilizar para la aplicación los miembros del equipo realizaron una serie de prototipos diferentes. Estos diseños fueron sometidos a votación junto con los tutores del trabajo. Todos ellos tenían en común dos elementos: los cubiertos y una selección de colores que recordasen a *Google Maps*. Todos los prototipos se hicieron usando *Adobe Photoshop*.

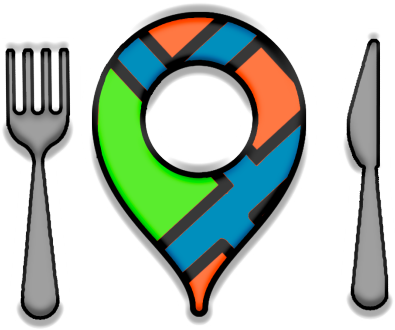

Figura 3.3: *Logo de Food Feeltr.*

Finalmente, la elección por unanimidad fue este logo, mostrado en la Figura 3.3. En él se muestran los dos elementos de los que se ha hablado anteriormente que simbolizan que la aplicación se centra en alimentación (los cubiertos) y una localización (el marcador típico que hace las veces de plato). Al fin y al cabo expresa el funcionamiento básico de la aplicación que es encontrar un sitio donde poder comer.

3.5.2. Aplicación

Al igual que el diseño del logo para diseñar la aplicación también se hizo una lluvia de ideas entre todos los miembros del equipo. Para que cada uno hiciese su propio diseño con

las mismas opciones disponibles se hizo uso de una página *web* especializada en hacer estos *mockups*[1]. En la Figura 3.4 se pueden ver tres pantallas que se diseñaron como prototipo.

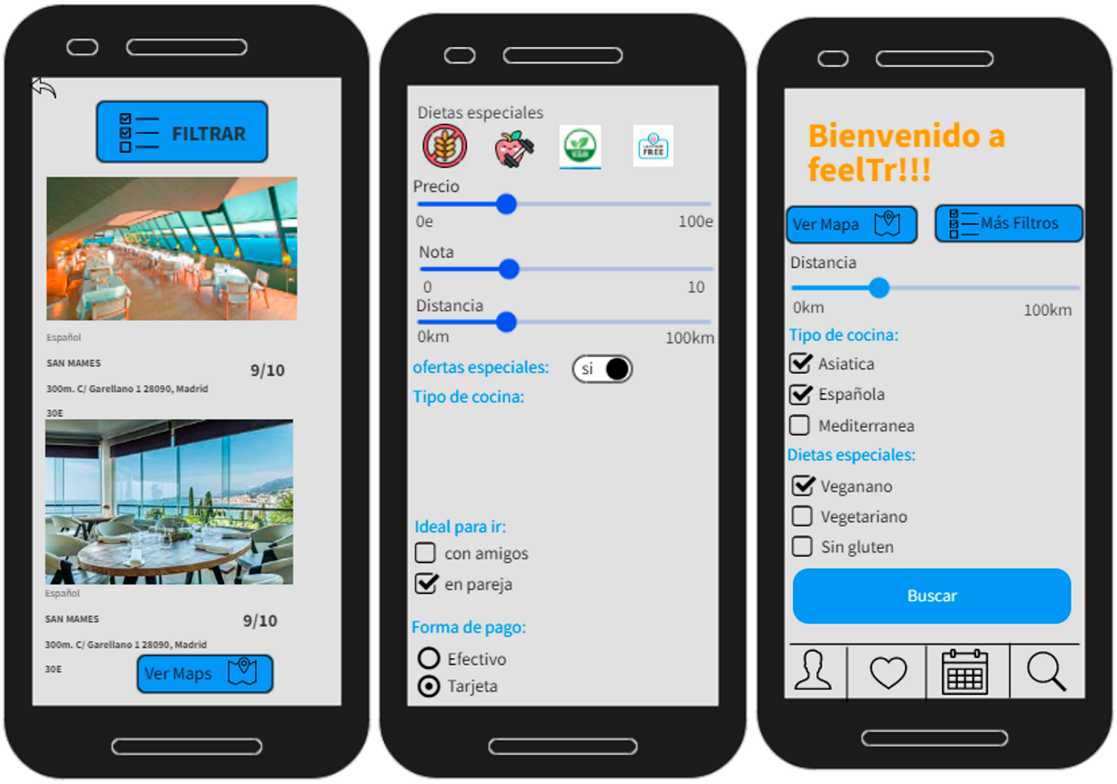

Figura 3.4: *Prototipo de la interfaz gráfica hecho en mockflow.com.*

El diseño de estos prototipos no se enfocó tanto para crear ya una interfaz bien definida con todos sus componentes bien seleccionados. Más bien fue de mucha utilidad para saber cuántas pantallas diferentes se querían, qué iban a contener esas pantallas y cómo iban a estar organizadas. Del diseño de los prototipos sacamos en claro que la aplicación tenía que tener una pantalla de inicio de sesión y otra de registrar que fueran muy sencillas para que hiciera fácil acceder a la aplicación.

Una vez que se está dentro de la aplicación se pueden ver tres grandes módulos. El primero y principal es donde se van a mostrar todos los restaurantes que se encuentran cerca del usuario. Después hay un segundo módulo donde se especificarán los filtros que

[1]Página web: https://mockflow.com

el usuario quiere poner para la búsqueda. Por último está el módulo de perfil donde el usuario puede ver sus reseñas y sus restaurantes favoritos. Más adelante se explicará más en profundidad los componentes que se encuentran en cada una de estas vistas principales.

Además de estas tres grandes vistas principales hay otras vistas menores para realizar acciones concretas como ver un restaurante con todos sus detalles, añadir una reseña o cambiar ajustes del usuario. La navegación entre todas y cada una de las vistas, tanto menores como principales, es muy fácil e intuitiva. Cada componente está donde se espera que esté, lo que genera una muy buena experiencia de usuario.

3.6. Base de datos

Se ha utilizado *Google Firebase* [14] como base de datos principal. *Google Firebase* es una plataforma ubicada en la nube integrada por *Google Cloud Platform* que usa un conjunto de herramientas para la creación de aplicaciones (web/móviles) procurando que sea más rápido y sin renunciar a la calidad requerida.

Se ha elegido *Google Firebase* como primera opción después de haber realizado una comparativa acerca de los *web services* posibles para el proyecto, una de la razones principales de su elección es que es una plataforma gratuita y de uso fácil e intuitivo lo cual hace más fácil su implementación, esto lo hace ideal para proyectos que están iniciándose.

En el proyecto se ha utilizado tres aplicaciones de *Google Firebase* que se detalla a continuación:

1. ***Authentication:*** ofrece un sistema de gestión de usuarios propio que permite tanto el registro(mediante correo electrónico y contraseña) como el acceso utilizando perfiles de otras plataformas externas (por ejemplo, de *Facebook*, *Google* o *Twitter*), una alternativa muy cómoda para usuarios reacios a completar el proceso.

2. ***Realtime Database:*** es una base de datos *NoSQL* en formato *JSON* alojada en la nube, con el fin de almacenar toda la información utilizada en la aplicación. Los datos

se sincronizan con cada cliente conectado en tiempo real y se mantienen disponibles aún cuando la aplicación no tiene conexión.

3. ***Storage:*** almacena archivos como imágenes, vídeos y audio, así como otro tipo de contenido generado por el usuario. Inicialmente se planteó el uso de esta utilidad para establecer por ejemplo imágenes de usuario para las reseñas o para almacenar imágenes obtenidas de *web scraping* y no tener que buscarlas cada vez, pero finalmente se descartaron ya que por un lado no se vio utilidad en establecer imágenes para los usuarios al no ser esta una red social o una aplicación donde sea necesario identificar al usuario mediante una imagen. Por el lado de *web scraping* como se necesita de cualquier manera acceder a *OSM*, se puede realizar en el mismo paso la búsqueda de imágenes, por lo que tener que realizar un acceso a la base de datos en ocasiones para obtener imágenes desactualizadas que además consumen parte del limitado espacio de almacenamiento es contraproducente.

La elección de una base de datos de formato *NoSQL* [15] ha sido tomada en base a varios aspectos, entre ellos:

- **Escalabilidad:** la facilidad de modificación de la estructura de los datos (lo cual implica una mejor adaptación) a futuras funcionalidades que puedan implementarse en el proyecto.
- **Rendimiento:** al utilizar una base de datos *NoSQL*, permite que la aplicación *Android* se pueda ejecutar en distintos dispositivos con poca potencia a diferencia de una base de datos *SQL* lo cual se vería afectado con el rendimiento.
- **Facilidad de almacenamiento de datos:** debido a la naturaleza de los datos almacenados (por ejemplo al crear un *array* de favoritos, reseñas), una base de datos *NoSQL* permite un acceso más fácil a lectura y escritura debido a que los datos recibidos de *OSM* son obtenidos en formato *JSON*, por lo que su *parseo* y uso es más sencillo.

Para explicar de una forma gráfica el diseño de la base de datos se muestra en la Figura 3.5 un diagrama entidad-relación. La base de datos está formada por tres tablas principales, utilizadas para representar toda la información disponible de la aplicación:

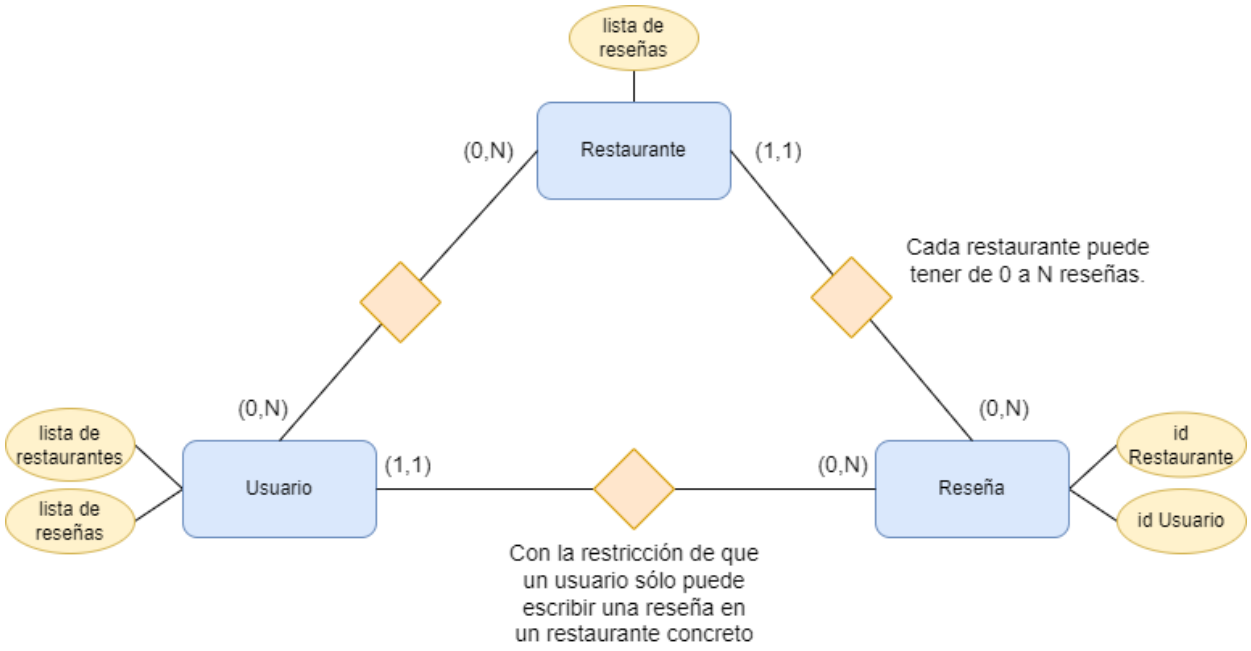

Figura 3.5: *Diagrama de entidad-relación.*

Por último, en las tablas siguientes se puede encontrar una especificación de la base de datos, donde se indica para cada una de las tablas, de manera esquematizada, qué campos contienen, el nombre del mismo, los datos que representan y las posibles restricciones que estos pudieran incluir.

- **Usuarios:** esta colección contiene toda la información relacionada con cada usuario en la cual se incluyen:
  - ***usuarioID:*** es la 'clave primaria' del usuario, creado automáticamente por *Google Firebase* a la hora de registrar al usuario mediante su correo electrónico.
  - ***usuarioEmail:*** es el correo electrónico con el cual se ha registrado el usuario.
  - ***usuarioNombre:*** es el nombre que se mostrará en los restaurantes cuando tengan reseñas de este usuario, para evitar mostrar el *email*, con un máximo de 20

caracteres.

- **Resenias:** lista de los *IDs* de las reseñas creadas por los usuarios.
- **Restaurantes:** lista de los *IDs* de los restaurantes favoritos del usuario.

- **Restaurantes:** esta colección contiene toda la información de los restaurantes que no se encuentra disponible en *OSM*. Su clave primaria es el *id* del nodo al cual corresponde en *OSM* para poder realizar una búsqueda de una forma más sencilla y coherente de la información relacionada a cada restaurante.

  - ***Valoración:*** valoración media del restaurante, creada a partir de la valoración de todas de reseñas de dicho restaurante. Este campo incluye el número de votos y el resultado de dichos votos, se utiliza para tener un acceso mucho más eficiente a datos y no tener que realizar la media de todas las reseñas cada vez que se cargue un restaurante.
  - ***Resenias:*** lista de los identificadores únicos de todas las reseñas puestas en este restaurante, para poder acceder a las mismas y cargarlas en caso de ser necesarias.
  - ***FiltrosNoAprobados:*** lista de filtros que han añadido los usuarios de la *app* y que, por lo tanto, no están disponibles en *OSM*. Cada filtro contiene el número de veces que se ha añadido para mostrarlo como un extra de información. A más veces sean añadidos en teoría más fiable es dicho filtro.

- ***Resenias:*** esta colección contiene toda la información de las reseñas escritas por los usuarios en los restaurantes:

  - ***idResenia:*** identificador único de la reseña, creado realizando un *hash* del identificador del usuario que la creó y el identificador del restaurante para el cual se creó.
  - ***idRestaurante:*** identificador único del restaurante al cual pertenece la reseña.
  - ***idUsuario:*** identificador único del usuario que creó la reseña.

- **Título:** es el título de la reseña que se mostrará de forma más destacada en la *app*, tiene un máximo de 50 caracteres.
- **Descripción:** es el texto detallado de la reseña, podrá ser de más caracteres que el título, en este caso un máximo de 350, y de acuerdo a ello se mostrarán con menor tamaño.
- **Valoración:** valoración numérica, en número entero, de las reseñas, indica como de 'bueno o malo' es el restaurante.

- **Tabla 1:** usuarios.
  - **Campo 1:** usuarioEmail.
    1. Almacena el correo electrónico del usuario.
  - **Campo 2:** usuarioID.
    1. Almacena el identificador único del usuario, el cual es creado al registrarse.
  - **Campo 3:** usuarioNombre.
    1. Almacena el nombre de pila del usuario.
    2. Tipo de datos: cadena de caracteres.
    3. Restricciones: máximo de 20 caracteres.
  - **Campo 4:** restaurantes.
    1. Almacena los restaurantes favoritos del usuario.
    2. Tipo de datos: lista que contiene los identificadores únicos de los locales.
    3. Restricciones: no puede aparecer el mismo identificador más de una vez.
  - **Campo 5:** reseñas.
    1. Almacena las reseñas que el usuario ha creado.

2. Tipo de datos: lista que contiene el identificador único de la reseña.
3. Restricciones: no puede aparecer el mismo identificador más de una vez.

- **Tabla 2:** restaurantes.
  - **Campo 1:** valoración.
    1. Almacena la valoración media del restaurante y el número de veces que se ha valorado.
  - **Campo 2:** reseñas.
    1. Almacena el identificador único de la reseña asociada a este restaurante.
    2. Restricciones: no puede aparecer el mismo identificador más de una vez.
  - **Campo 3:** filtros no aprobados.
    1. Almacena los filtros añadidos por los usuarios.
    2. Tipo de datos: mapa que contiene como clave el nombre del filtro y como valor el número de veces que se ha añadido.
    3. Restricciones: no se permiten filtros repetidos.
- **Tabla 3:** reseñas.
  - **Campo 1:** idReseña.
    1. Almacena el único de la reseña, creado a partir del identificador del restaurante y el del usuario que la escribió.
  - **Campo 2:** idRestaurante.
    1. Almacena el identificador único del restaurante donde está publicada.
  - **Campo 3:** idUsuario.
    1. Almacena el identificador único del usuario que la creó.

- **Campo 4:** título.
    1. Almacena el título de la reseña.
    2. Tipo de datos: cadena de caracteres.
    3. Restricciones: longitud máxima de 50 caracteres.

- **Campo 5:** descripción.
    1. Almacena el texto extenso escrito por el usuario.
    2. Tipo de datos: cadena de caracteres.
    3. Restricciones: longitud máxima de 350 caracteres.

- **Campo 6:** valoración.
    1. Almacena la valoración otorgada por el usuario al restaurante del cual se escribe la reseña.
    2. Tipo de datos: número entero positivo.

Capítulo 4

Implementación de la aplicación

En este capítulo, se explica en profundidad tanto el desarrollo de la aplicación del lado del cliente como el desarrollo del lado del servidor.

4.1. Back-End o lado del servidor

En esta sección se definirá de manera detallada la configuración de la base de datos de la aplicación, ya que generalmente esta es el pilar de todo sistema informático destinado a ofrecer o manejar información. Debido a que en su implementación hubo muchos problemas y dudas, se va a explicar de manera precisa el proceso que se siguió para que funcionase de manera correcta.

La finalidad de incluir esta guía de configuración de base de datos es para que los futuros estudiantes no les ocurra el mismo fallo que se tuvo que afrontar en este proyecto: 'La información no se registraba correctamente'. De esta manera podrán ahorrar tiempo a la hora de configurar la base de datos, invirtiéndolo en implementar otras funcionalidades importantes.

Como se comentó en la Sección 3.6, se ha utilizado *Google Firebase*, la cual dispone de un modelo de datos no relacional (*NoSQL*) donde la información se aloja en formato *JSON*.

A continuación se va a detallar la configuración de la base de datos paso a paso:

1. Acceder a la siguiente *URL*: **console.firebase.com**. Para acceder a esta *web* se debe

tener una cuenta de *Google* para poder autenticarse.

2. Acceder a la consola y pulsar en el botón **'Agregar Proyecto'**.

3. Introducir el nombre del proyecto, en este caso es el nombre de la aplicación, es decir **'Food Feeltr'**.

4. Una vez creado el proyecto en *Google Firebase* se tiene que vincular con el proyecto de la aplicación de *Android Studio*, esto se debe realizar para que exista comunicación entre ambas aplicaciones y así la información pueda almacenarse correctamente. Para ello se tiene que introducir el paquete de *Android*, en este caso **'ucm.appmenus'**, en *Google Firebase*:

```
<manifest xmlns:android='http://schemas.android.com/apk/res/android'
package='ucm.appmenus'>
```

5. Una vez registrado el proyecto, se debe descargar el fichero llamado **google.services.JSON**, mostrado en la Figura 4.1, y añadirlo en la *app* del proyecto.

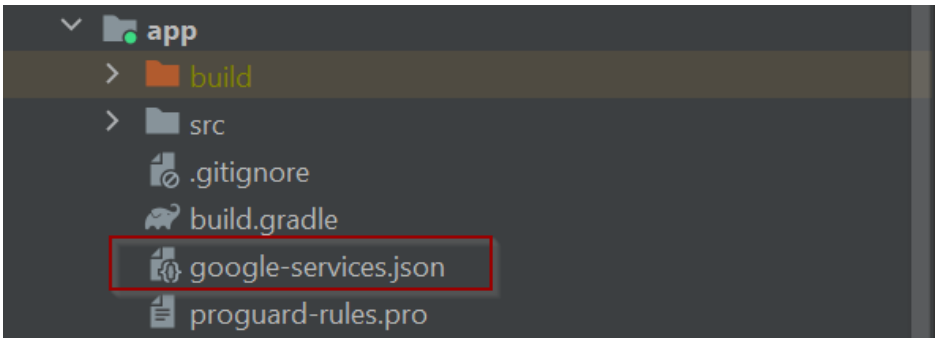

Figura 4.1: *Fichero de configuración 'Google Services JSON'.*

6. Agregar el *classpath* en el *build gradle* del proyecto, como se indica a continuación:

```
classpath 'com.google.gms:google-services:4.3.10'
```

7. Agregar el *plugin* de *Google Services* en el *build gradle* de la aplicación como se indica a continuación:

```
apply plugin:'com.google.gms.google-services'
```

8. Una vez sincronizados los cambios, la aplicación de *Android* ya estaría vinculada con *Google Firebase*.

9. Para que la configuración funcione correctamente y cumpla con el objetivo que tiene el proyecto, se deben añadir los siguientes módulos, mostrados en la Figura 4.2, utilizados en la *app*:

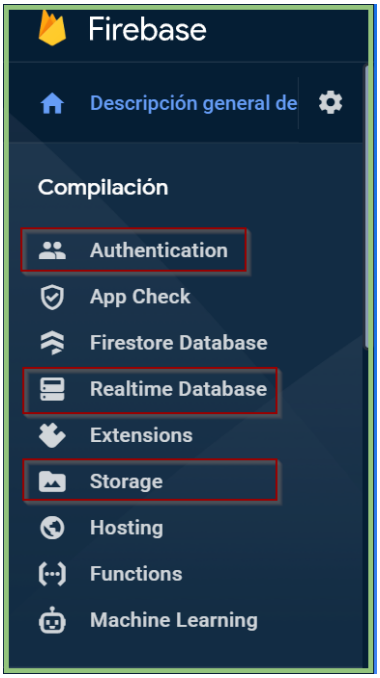

Figura 4.2: *Módulos utilizados en Google Firebase.*

- **Módulo Authentication**, representado en la Figura 4.3 contiene la información de todos los usuarios registrados, informa de la fecha y hora de registro y a su vez crea un *UID* (identificador único de usuario).

- **Módulo RealTime Database**, representado en la Figura 4.4 contiene la información de las reseñas, restaurantes y usuarios.

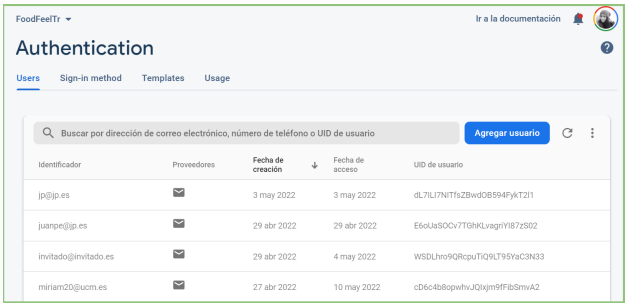

Figura 4.3: *Módulo Authentication.*

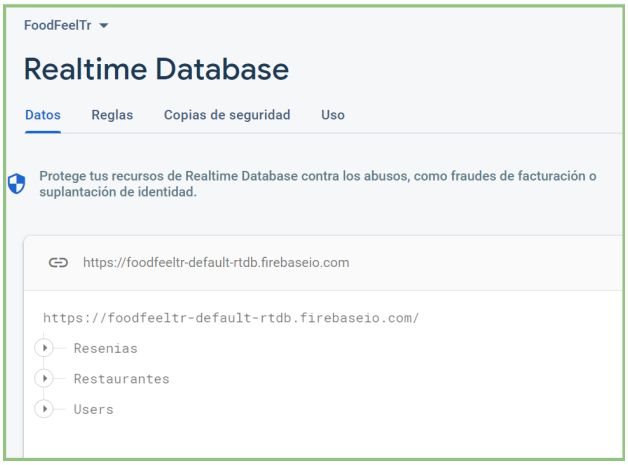

Figura 4.4: *Módulo Real Time Database.*

4.2. Front-end o lado del cliente

En el lado del cliente es donde se gestiona todo el uso de la aplicación [16, 17]: acceso a *OSM* para recibir la información necesaria de los lugares, *web scraping* y base de datos para complementar la información recibida de *OSM* e interpretar los datos obtenidos para posteriormente, mostrárselos al usuario de una forma que pueda comprender.

Para el diseño general de la aplicación, se han decidido usar como colores principales el verde, azul claro y naranja, los cuales son los usados en el logo seleccionado para la misma.

Además se ha decidido optar por colores grises para opciones no seleccionadas, como los filtros, y azules cuando sí lo están.

Todos los diseños se han comprobado que funcionan de manera correcta en los dispositivos *Pixel 4* y *Pixel 4XL*, de 4,7 y 6,3 pulgadas de pantalla respectivamente.

Para mantener una consistencia en todo el apartado visual de la aplicación [18], se han establecido como uso único los botones de *Material Design 1.4* [19], de esta manera todos los botones y filtros utilizan el color azul para facilitar su reconocimiento, se ha ampliado el tamaño de la mayoría de textos de la aplicación con el fin de facilitar su lectura en pantallas más pequeñas o para gente con ciertos problemas de visión y se ha mantenido un color de letra oscuro en fondos claros y un color blanco en fondos más oscuros.

4.2.1. Inicio de sesión y registro

Para que se puedan almacenar los usuarios que se den de alta en la aplicación es necesario tener el módulo *authentication* y habilitar el *sign-in method* habilitado en el *Google Firebase*, como se muestra en la Figura 4.5. *Sign-In method* ayuda a que los usuarios se registren o inicien sesión con un usuario y una contraseña.

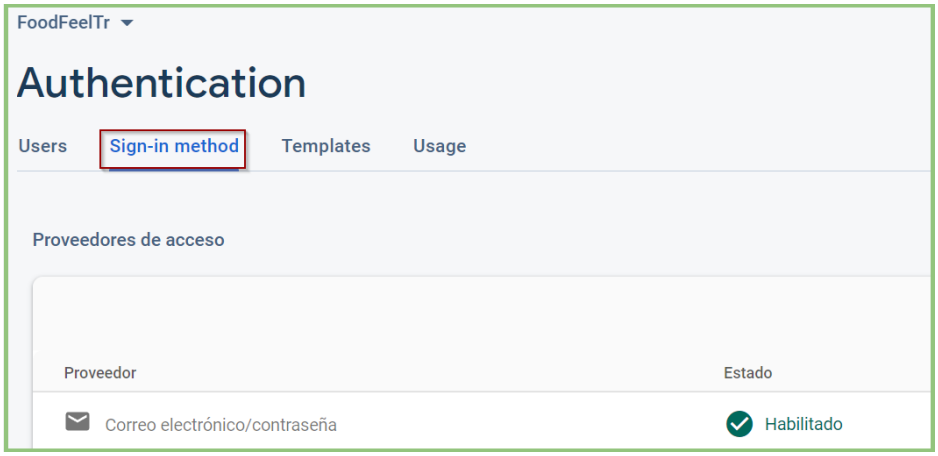

Figura 4.5: *SignInMethod habilitado.*

En la implementación del código se creó una carpeta específica llamada *login* donde se aloja las clases relacionadas con el registro y *login*.

- **Registro de usuario:** para registrar un usuario hace falta un nombre, un correo y una contraseña, como se muestra en la Figura 4.6. También existe la posibilidad de navegar por la aplicación sin un registro previo. Esa acción se realiza con el usuario invitado que tenemos implementado en la aplicación.

- **Login de usuario:** para hacer *login* es necesario acceder con las credenciales obtenidas a la hora de registrarse en la aplicación, como se muestra en la Figura 4.7. En el caso del usuario invitado te dejará entrar sin problema pulsando en el botón 'Acceder sin iniciar sesión'. Esta acción automáticamente permitirá navegar por la aplicación y dará la libertad de realizar búsquedas. Sin embargo no tendrás la posibilidad de añadir, guardar reseñas ni tampoco restaurantes favoritos, en ese caso sería necesario acceder con una cuenta existente de la aplicación.

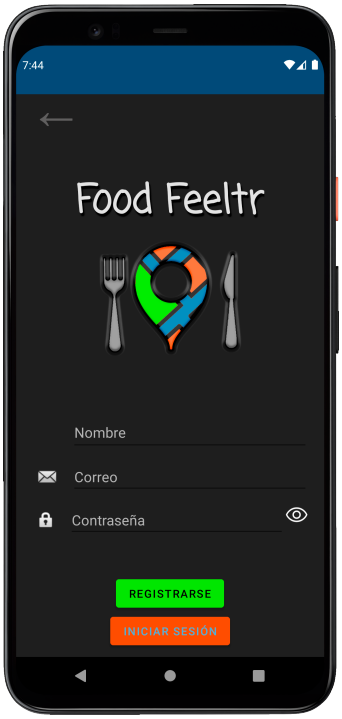

Figura 4.6: *Pantalla de registro.*

Figura 4.7: *Pantalla de login.*

Tanto en el registro como a la hora de hacer *login* se hacen las validaciones correspon-

dientes de todos los campos. Éstos tienen que tener un formato correcto a parte de que todos los campos son de cumplimentación obligatoria.

4.2.2. Página principal

Una vez el usuario abre la aplicación y avanza de la pantalla del *login* (ya sea habiéndose registrado o iniciado sesión o no), se le muestra inmediatamente una lista de todos los restaurantes en un radio de 1500 metros, para que así pueda tener una visión rápida y clara de las opciones que tiene disponibles cerca de su ubicación actual.

Esta página principal está dividida en 3 secciones:

Inicio

En esta vista, representada en la Figura 4.8 es donde se muestra el resultado de las búsquedas realizadas. En la zona superior se encuentra una pequeña sección de texto con los filtros que se han aplicado para la búsqueda, a continuación se muestra una lista con información básica de los restaurantes, entre la que se puede destacar:

- **Nombre del restaurante.**
- **Imagen principal del restaurante.**
- **Página *web*.**
- **Si está actualmente abierto o no:** también puede ser estado desconocido en caso de no tener los horarios del restaurante.
- **Dirección.**
- **Filtros que contiene.**
- **Rango de precios.**

La imagen principal y el rango de precios se obtienen mediante *web scraping* en caso de que en el nodo del restaurante recibido por *OSM* se encuentre la página *web* del mismo. Estos

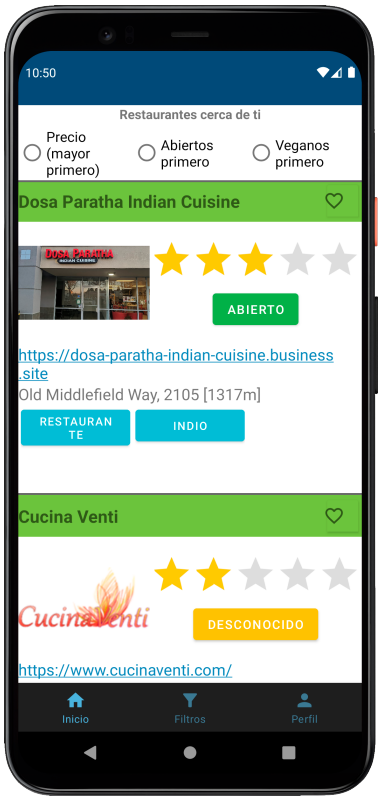

Figura 4.8: *Vista de la página principal con todos los resultados de las búsquedas de los filtros.*

elementos se cargan en un 'thread' secundario para que la aplicación no se quede congelada y permita una interacción con el usuario pese a que no esté aún toda la información disponible.

Además en ocasiones *OSM* no contiene la dirección del restaurante sino solo sus coordenadas, por lo que mediante búsqueda inversa utilizando la localización del mismo y del usuario, aplicando la fórmula de *Haversine*[1].

Como complemento a toda esta información, también se recibe de la base de datos la información relativa a la valoración del restaurante por parte de otros usuarios para una mejor elección del local.

[1] Distancia$=2*(radiodelplaneta)*\arcsin\left(\sqrt{\sin^2(\frac{\theta_1-\theta_2}{2})+\cos(\theta_1)*\cos(\theta_2)*\sin^2(\frac{\lambda_1-\lambda_2}{2})}\right)$

Filtros

En esta vista, cuya representación en la aplicación se puede apreciar en la Figura 4.9, se encuentran todos los parámetros necesarios para la búsqueda, se dispone de un deslizable para elegir el radio de búsqueda, seguido de los filtros, que son una selección que se consideraron importantes de entre todos los disponibles en *OSM*.

Una vez el usuario tiene seleccionados los filtros por los cuales desea buscar, como se puede observar en la Figura 4.10, pulsa el botón de filtrar y entonces la aplicación le lleva a la pestaña de inicio y busca los restaurantes que cumplan alguno de los criterios seleccionados. Debido a que *OSM* tiene sus filtros en inglés y la aplicación está en español, se introdujo un sistema de traducción mediante un *hashmap* donde la clave es el nombre en inglés y el valor es la traducción al español, y cada vez que hay que mostrar un filtro se traduce mediante dicho método.

Figura 4.9: *Vista de selección de filtros sin selección.*

Figura 4.10: *Vista con filtros seleccionados.*

Perfil

En esta vista se muestran los detalles relativos al perfil y la configuración, en la parte superior se muestra el nombre del usuario y su correo, y en la inferior se muestran o las reseñas que ha publicado, Figura 4.11 o los restaurantes que ha añadido a favoritos, Figura 4.12. Ambos datos se cargan de la base de datos al iniciar la aplicación en caso de que el usuario haya iniciado sesión y se mantienen durante todo el ciclo de ejecución de la aplicación para ahorrar tiempo y recursos del procesador.

En caso de que el usuario no haya iniciado sesión se muestra una pantalla con un botón dándole la opción de hacerlo, como se puede apreciar en la Figura 4.13.

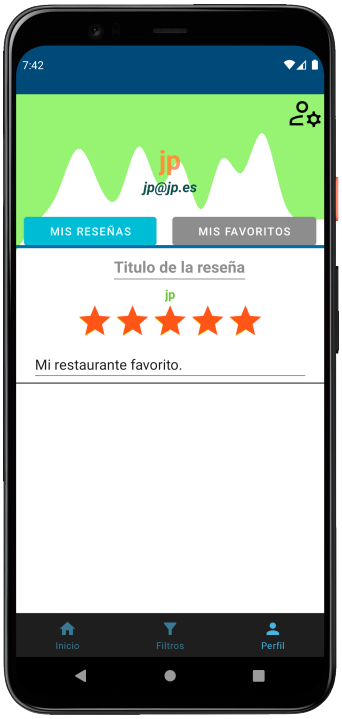

Figura 4.11: *Vista de las reseñas del usuario.*

Figura 4.12: *Vista de los favoritos del usuario.*

4.2.3. Detalles del restaurante

Esta vista, representadas en las figuras Figura 4.15 y 4.16, contiene información más específica del restaurante, entre ellas:

Figura 4.13: *Vista del perfil si no se inicia sesión.*

Figura 4.14: *Vista de añadir o editar una reseña.*

- **Imágenes:** se muestran todas las imágenes disponibles en la *web* del local, para que el usuario pueda comprobar si la oferta alimenticia se adapta a sus expectativas.
- **Valoración media del restaurante:** basada en las reseñas que se han publicado del mismo.
- **Filtros:** aquí se muestran tanto los filtros recibidos mediante *OSM* como los que han establecido otros usuarios y están cargados en la base de datos, lo cual se notifica con un texto en color naranja para que se sepa que pueden no ser correctos.
- **Información adicional:** entre la que se encuentra la *web*, el número de teléfono o el horario de apertura del local.
- **Reseñas:** una sección deslizable con todas las reseñas publicadas por otros usuarios acerca del local, cada una contiene un título, una valoración y una descripción con

más detalles acerca del lugar.

- **Botones:** utilizados para crear una reseña nueva, como se muestra en la Figura 4.14, y para añadir filtros de comida, para ampliar la información nutricional del local.

Para la obtención de la valoración, las reseñas y los filtros se necesita acceso a la base de datos [20], lo cual se realiza en un nuevo hilo del procesador en segundo plano así como la carga de imágenes de la *web*. Sin embargo no se vuelven a buscar los filtros ni los precios ya que se realizó durante la búsqueda general de restaurantes, así que se reutilizan dichos datos para mejorar ligeramente el uso de recursos del dispositivo y el tiempo.

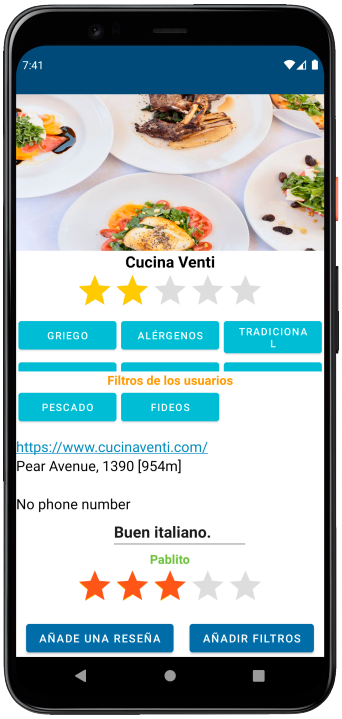

Figura 4.15: *Vista de información detallada de un restaurante.*

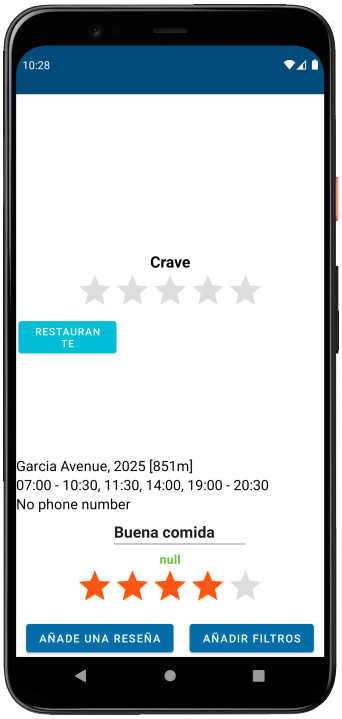

Figura 4.16: *Vista de información detallada de un restaurante sin web.*

4.2.4. Búsqueda de información en webs de los locales

Como se ha comentado en la Sección 3.3, la aplicación implementa *web scraping* debido a que los datos recibidos de *OSM* en ocasiones pueden no estar actualizados. Por ello se

decidió utilizar esta técnica para proporcionar una información extra al usuario acerca de los locales que se le han mostrado como resultado de su búsqueda.

Esta aplicación crea un nuevo *thread* para cada restaurante del cual se va a realizar *web scraping* (solo en caso de que el restaurante cuente con una página *web* registrada en *OSM*) para obtener la información a mayor velocidad como se ha explicado en la Sección 3.2. Gracias a esta técnica podemos recibir información de distintos tipos como:

- **Imágenes:** se utilizan para brindar al usuario con un apoyo visual del local seleccionado o de su oferta alimenticia. Al ser las imágenes elementos que en ocasiones pueden llevar una gran cantidad de información, sobre todo en contraposición a otros elementos como texto, se ha decidido realizar su búsqueda en dos partes para una mayor eficiencia de la aplicación:

  - **Pantalla de inicio:** en esta parte solamente se decodifica una imagen, generalmente el logo del local que suele ser la imagen situada en el *header* de la *web* o, en caso de no haberlo, en la parte superior de la misma.
  - **Pantalla de restaurante detallado:** en esta pantalla se buscan todas las imágenes disponibles en la *web* del local y se decodifican mostrándolas en la parte superior de la pantalla a modo de galería deslizable de derecha a izquierda.

  La elección de esta distribución ha sido esta ya que en la pantalla de inicio se presentan todos los resultados de las búsquedas, proceso que requiere una cantidad considerable de tiempo para lo esperable de un dispositivo móvil, además de que sobrecargar esta sección de información general con muchas imágenes de cada restaurante puede saturar al usuario, por todo ello se decidió mostrar solo una imagen representativa del local.

  Para realizar dicha búsqueda se han buscado los *tags HTML* de *'img'*. En caso de no recibir imágenes mediante *web scraping* se establece una imagen por defecto en la aplicación, como se puede observar en la Figura 4.17.

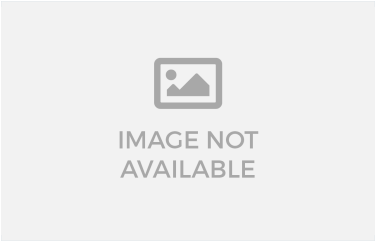

Figura 4.17: *Imagen mostrada cuando no se encuentran imágenes mediante web scraping.*

- **Filtros extras:** se utilizan para ofrecer filtros extras al local en base a la carta o al menú del mismo. Para ello se buscan palabras clave en la *web* principal como 'menú' o 'carta' pues generalmente estas palabras están incluidas en un botón que contienen la *URL* de la sección alimenticia (en cuyo caso se carga) o informan de que en la sección siguiente de la *web* se indican los alimentos del local.

  Una vez disponemos de la *URL* donde se encuentra toda la información de los alimentos, se utiliza *web scraping* para buscar en el texto de la página *web* palabras clave relacionadas con posibles alimentos como podrían ser carne, pescado, arroz, ensalada o elementos dietéticos como menú vegano o sin gluten.

- **Precios:** puesto que el coste económico del local es un elemento que una gran parte de la sociedad tiene en cuenta a la hora de elegir un local para comer y ya que *OSM* no proporciona dicho filtro, se ha utilizado *web scraping* para buscar sobre la carta los precios de los distintos platos. Para ello se utiliza una función *regex* [21] que busca los elementos numéricos, separados por comas o puntos para los decimales, que incluyen un símbolo monetario a continuación, para de esta forma obtener solamente elementos que indiquen precios y no obtener por error por ejemplo las horas de apertura.

  Una vez se han obtenido todos los precios disponibles del local, se ordenan de menor a mayor precio y se muestran al usuario el menor, el mediano y el mayor. La elección de estos tres elementos se ha tomado en cuenta pensando en no proporcionar una cantidad excesiva de información al usuario, se proporcionan el menor y mayor precio para que sepa en qué rango de precios se sitúa el local y se proporciona la mediana para

evitar caer en problemas que puede causar la media (dejarse influenciar por elementos extremos como por ejemplo un vino muy caro subiría el precio medio) y puesto que el elemento de la mediana generalmente (sabiendo el máximo y mínimo) es útil para analizar si los precios están desviados hacia la zona cara o hacia la barata.

Capítulo 5

Resultados obtenidos

En esta sección se analizan algunos resultados técnicos obtenidos de la aplicación desarrollada. Para ello se han medido los siguientes parámetros:

- **Tiempo de ejecución de la tarea:** el tiempo que tarda el sistema desde el inicio de la medición que se quiere realizar hasta su finalización.
- **Porcentaje máximo de utilización del procesador:** indicado por los instantes donde más carga de trabajo se genera en el mismo a raíz del uso de la aplicación.
- **Consumo de memoria *RAM*:** este dato incluye aspectos básicos de la ejecución de una aplicación de *Android*, como por ejemplo la interfaz de usuario. También indica la memoria consumida por el propio código en ejecución y el espacio que ocupan los datos obtenidos.
- **Uso de la red:** tanto en subida como en descarga de información, ya sea a la base de datos o al realizar tareas como *web scraping* o búsquedas en *OSM*.

Los datos mostrados son una media aritmética de 5 mediciones para cada sección que se desea probar. Se ha realizado la medición de este modo debido a que los resultados obtenidos pueden estar influidos por factores externos al *software* desarrollado. Por ejemplo, los resultados pueden verse alterados por factores como el uso de la *CPU* por otros procesos

o aplicaciones del sistema, la intensidad de conexión de la *red* en el momento de realización de las pruebas o incluso el nivel de carga de trabajo de los servidores de *OSM*.

Las mediciones se han realizado todas en el emulador de *Android Studio*, con un dispositivo *Google Pixel 4XL* que posee una memoria *RAM* de 1500MB y el sistema operativo *Android* 9.0.

5.1. Resultados técnicos

Se ha decidido, basándose en los casos de uso de la Sección 3.4, medir las categorías consideradas como más relevantes, mostrando en la Tabla 5.1 las medias de los 5 tests ejecutados para cada sección.

	Tiempo de carga (ms)	Uso máximo de CPU (%)	Uso máximo de RAM (MB)	Descarga de red (KB/s)	Subida de red (KB/s)
Inicio de sesión	784,90	60,40	73,18	0,18	0,22
Inicio *app*	722,42	78,00	116,06	237,12	3,86
Búsqueda 1000m	770,28	52,40	119,42	187,46	6,82
Búsqueda 1750m	769,66	67,20	149,60	426,40	21,54
Búsqueda 3000m	5884,06	84,40	130,82	922,98	68,54
Local con web	4180,54	55,20	115,94	376,70	6,12
Local sin web	513,02	29,20	103,54	376,70	0,98
Añadir reseña	656,67	26,40	114,64	2,32	1,86

Tabla 5.1: *Media de los resultados técnicos obtenidos.*

Además de los resultados anteriores, en las categorías de 'inicio aplicación' y 'local con web' se obtuvieron mediciones extra:

- **Inicio aplicación:** en esta categoría se midieron los tiempos que se tardaba en buscar los filtros adicionales mediante *web scraping* y la carga, ordenación y selección de los precios, mostrados en la Tabla 5.2.

	Filtros (ms)	Precios (ms)
Tiempo	4707,60	907,69

Tabla 5.2: *Media del tiempo de carga de filtros y precios.*

- **Local con web:** en esta categoría se midió el tiempo que se tardaba en buscar y cargar todas las imágenes de la web del local. También está calculado el tiempo de carga de las reseñas del mismo, todo ello mostrado en la Tabla 5.3.

	Imágenes (ms)	Reseñas (ms)
Tiempo	3220,67	599,86

Tabla 5.3: *Media del tiempo de carga de imágenes y reseñas*

5.2. Feedback de los usuarios

El objetivo principal de esta aplicación es ofrecer una forma más fácil de seleccionar el lugar donde se desea comer. Por esta razón se ha realizado una encuesta a una serie de usuarios que han podido probar el *software*. Así se han podido descubrir los puntos fuertes que han encontrado en la *app*.

Además se pretenden encontrar los posibles fallos o partes donde, bien por la interfaz (tamaño reducido, manejo poco intuitivo, etc) o bien por el funcionamiento de la *app* (tiempo de respuesta, errores que inutilizan la aplicación, etc), el uso se ve penalizado.

Para ello los usuarios que han probado la aplicación han respondido un pequeño formulario con preguntas genéricas acerca del funcionamiento de la aplicación. Existen respuestas numéricas con una escala del 1 al 10, siendo 1 la peor puntuación posible y 10 la mejor. De igual forma se les ha permitido dar sugerencias sobre qué apartados no les han convencido, que mejorarían o cuales son los puntos mas positivos según su opinión.

Formulario con preguntas de valoración

1. ¿Con qué frecuencia usarías la *app*?
2. ¿Qué tan fácil fue para ti usar la *app*?
3. ¿Qué tan atractiva te parece la *app* en cuanto a diseño?
4. ¿Con qué frecuencia recomendarías la *app*?

5. ¿Qué tan rápido responde la *app* a tus peticiones?

6. ¿Qué tan difícil fue para ti usar la *app*?

7. ¿Qué tan amigable te parece la *app*?

8. ¿Qué tan satisfecho estas con los filtros ofrecidos?

9. ¿Cuánto crees que tardaría otra persona en aprender a usar la *app*?

10. ¿Qué tantos conocimientos técnicos debes tener para interactuar con la *app*?

Preguntas con respuesta mas detallada

1. ¿Echas en falta alguna funcionalidad?

2. ¿Te sientes cómodo con el modo de registrarse e iniciar sesión?

3. ¿Qué es lo que menos te gusta la aplicación?

Análisis y conclusiones acerca del *feedback* de los usuarios

Tal y como muestran los resultados obtenidos, todos los usuarios, han valorado muy positivamente la aplicación. También todos coinciden en que no hace falta tener conocimientos técnicos para saber usar la aplicación. Otro aspecto que ha gustado mucho a los usuarios es la facilidad para iniciar sesión y registrarse. Por otro lado hemos recibido también valoraciones negativas, por ejemplo a un usuario la paleta de colores seleccionada no le gusta o le complica la utilización de la aplicación. Otro comentario que se ha repetido es que echan en falta un mapa donde poder ver los resultados o centrar la búsqueda. Este es un punto que ya se ha comentado a lo largo de la memoria y también nosotros lo echamos en falta, pero como se ha explicado en secciones anteriores finalmente no ha sido posible.

Usuario 1: Andrea
Diseñadora gráfica, 26 años

Formulario:
En la tabla 5.4 se pueden observar las valoraciones obtenidas de Andrea a las primeras 10 preguntas del formulario, y a continuación de la tabla se muestran las respuestas de las últimas 3 preguntas.

Nº pregunta	**Valoración**
1	9
2	7
3	5
4	10
5	8
6	7
7	7
8	10
9	5
10	3

Tabla 5.4: *Valoración de Andrea.*

1. Un pequeño tutorial de qué o cuántos filtros aplicar para obtener mejores resultados.
2. Sí
3. El texto a veces se desborda en los filtros/botones (por ejemplo restaurant-e)

Usuario 2: Rosse Mary
Ingeniera alimentaria, 35 años

Formulario:
En la tabla 5.5 se puede observar las valoraciones obtenidas de Rosse Mary a las primeras 10 preguntas del formulario, y a continuación de la tabla se muestran las respuestas de las últimas 3 preguntas.

Nº pregunta	**Valoración**
1	6
2	8
3	7
4	7
5	8
6	3
7	9
8	7
9	6
10	2

Tabla 5.5: *Valoración de Rosse Mary.*

1. El mapa, para que muestre los restaurantes cercanos a mi.
2. Sí, porque no piden tantos datos para registrarme en la aplicación, este le hace eficaz.
3. Que no tenga localización en tiempo real.

Usuario 3: Mónica
Estudiante, 16 años

Formulario:
En la tabla 5.6 se puede observar las valoraciones obtenidas de Mónica a las primeras 10 preguntas del formulario, y a continuación de la tabla se muestran las respuestas de las últimas 3 preguntas.

Nº pregunta	**Valoración**
1	8
2	10
3	7
4	9
5	8
6	1
7	10
8	10
9	1
10	1

Tabla 5.6: *Valoración de Mónica.*

1. Me gustaría ver un mapa los resultados para saber dónde está exactamente.
2. Es muy fácil, ojalá todas las aplicaciones permitiesen registrarse así de fácil.
3. Los colores no terminan de convencerme.

Usuario 4: Iñaki
Médico, 36 años

Formulario:
En la tabla 5.7 se puede observar las valoraciones obtenidas de Iñaki a las primeras 10 preguntas del formulario, y a continuación de la tabla se muestran las respuestas de las últimas 3 preguntas.

Nº pregunta	**Valoración**
1	8
2	7
3	8
4	6
5	7
6	6
7	7
8	9
9	7
10	3

Tabla 5.7: *Valoración de Iñaki.*

1. A veces la búsqueda ofrece pocos resultados.
2. Sí, cómodo e intuitivo y se puede acceder con una cuenta de Google.
3. Hay secciones del diseño que quedan raras (algunos filtros muy largos).

Usuario 5: Mayte
Consultora de marketing, 43 años

Formulario:
En la tabla 5.8 se puede observar las valoraciones obtenidas de Mayte a las primeras 10 preguntas del formulario, y a continuación de la tabla se muestran las respuestas de las últimas 3 preguntas.

Nº pregunta	**Valoración**
1	8
2	9
3	8
4	7
5	7
6	6
7	9
8	9
9	6
10	3

Tabla 5.8: *Valoración de Mayte.*

1. La opción de buscar en el mapa y cambiar la ubicación.

2. Sí.

3. No disponer de un mapa para ver los locales y su ubicación, o de un enlace a una aplicación como Google Maps que te permita buscar la ruta.

Capítulo 6

Conclusiones y trabajo futuro

6.1. Conclusiones

Una vez finalizado el desarrollo del proyecto es el momento de realizar una valoración sobre el trabajo desarrollado y lo aprendido hasta la fecha.

El objetivo de este proyecto era realizar el diseño, desarrollo e implementación de una aplicación nativa para dispositivos *Android*, el fin de ésta es realizar búsquedas de restaurantes según los intereses del usuario, expresados mediante filtros y teniendo en cuenta la opinión de aquellos que hayan compartido alguna reseña en los restaurantes. Este objetivo puede ser marcado como cumplido. Acompañado de una interfaz sencilla y atractiva para consultar restaurantes. La aplicación también es capaz de sincronizar con un servidor donde se almacenan ciertos datos para después mostrarlos a otros usuarios.

Existe una funcionalidad que siempre se quiso implementar pero que no ha sido posible: la visualización del mapa. Surgieron algunos problemas en el proceso de configuración de las herramientas previamente elegidas y problemas de rendimiento. Al ser *OSM* una herramienta cuyo uso es principalmente mediante páginas *web*, no hay implementaciones de la misma para *Android*. Por ello, una de las opciones era implementar en la aplicación una vista directa de su *web*, pero el uso del mapa mediante este método era muy complejo. La otra opción fue utilizar una *API* de terceros, en este caso *MapsForge* [22], pero después de implementarla el rendimiento era muy bajo, con constantes fallos de *renderización* del mapa

y complejidad en la selección de puntos en el mismo.

A lo largo de la carrera hemos cursado muchas asignaturas. Algunas de ellas han sido imprescindibles para el desarrollo de este proyecto.

- **Tecnología de la Programación:** en esta asignatura aprendimos los conceptos básicos de la programación orientada a objetos. Más en concreto vimos por primera vez *Java* que ha sido el lenguaje que hemos usado en la aplicación.
- **Bases de Datos:** tanto esta asignatura como Ampliación de Bases de Datos nos han servido para diseñar el modelo y gestionar toda la información.
- **Ingeniería de Software:** gracias a los conocimientos recibidos sobre patrones de diseño se ha podido realizar todo siguiendo un buen patrón de código.
- **Programación de Aplicaciones para Dispositivos Móviles:** al ser una optativa no todos la hemos podido cursar. Los que lo hemos hecho lo agradecemos porque hemos recibido en ella muchos conocimientos que nos han sido de gran utilidad.
- **Estructura de datos:** la utilidad de esta asignatura ha sido crear una mejor organización de los datos utilizados, utilizando por ejemplo *sets y maps* para una mayor eficiencia en la búsqueda y organización de locales y reseñas.

6.2. Trabajo futuro

En esta sección se describirán ideas que añaden más valor al proyecto, pero que por complejidad y por tiempo disponible no ha sido posible desarrollarlas. A continuación, se van a detallar las posibles mejoras o funcionalidades nuevas que se podrían añadir en versiones futuras.

- **Mostrar el mapa gráficamente:** para una mayor visibilidad se podrían mostrar en el mapa los restaurantes disponibles más cercanos. También se podría ofrecer la posibilidad de acotar una zona de búsqueda.

- **Acotar una zona de búsqueda:** ahora mismo se puede aplicar un rango de búsqueda. Con esta característica futura se podría dibujar en el mapa la zona en la que el usuario quiere buscar.

- **Selección de la localización en el mapa:** actualmente solo se puede cambiar la localización de búsqueda introduciendo manualmente las coordenadas. Esto no es cómodo ni accesible para una gran mayoría de usuarios. La posibilidad de señalarla con un toque en el mapa haría que la *app* fuese mucho más fácil de utilizar y sumar valor.

- **Múltiples idiomas:** poder tener la aplicación disponible para otros países y tener una base de datos de usuarios extranjeros. Esto extendería todavía más el número de usuarios que podrían usar la aplicación.

- **Modo nocturno intercambiable:** sería muy agradable que la aplicación soportase el modo nocturno en la totalidad de la aplicación con los colores deseados y de una forma correcta y uniforme. Esta opción podría ser seleccionada por el usuario al abrir la aplicación y en la zona de ajustes.

- **Soporte en otros dispositivos**: hacer que la aplicación funcione con vistas adaptadas a otros dispositivos como *tablets*, en las cuales actualmente funciona pero debido a que, como generalmente tienen un tamaño de pantalla bastante mayor al de un *smarphone*, hay espacio que se podría aprovechar de una mejor manera. También convendría estudiar la posible compatibilidad de ciertas funcionalidades en algunos *smartwatches*.

Bibliografía

[1] Leyva Adriana. ¿Android o iOS? *Mural*, Nov 2015. Accedido por última vez: 15-10-2021.

[2] Cadenhead Rogers and Garín Fominaya(Trad.) Lola. *Java 8*. Anaya Multimedia D.L., 2014. Accedido por última vez: 19-04-2022.

[3] Google. Android developer guide main page. https://developer.android.com/guide/. Accedido por última vez: 27-05-2022.

[4] Google. Android Studio main page. https://developer.android.com/studio. Accedido por última vez: 10-11-2021.

[5] Santacroce Ferdinando. *Git essentials : create, merge, and distribute code with Git, the most powerful and flexible versioning system available*. Birmingham, UK : Packt Publishing, 2015. Accedido por última vez: 13-02-2022.

[6] Oracle. The Java Tutorials. https://docs.oracle.com/javase/tutorial/. Accedido por última vez: 30-03-2022.

[7] Google Maps. Google Maps Places SDK for Android. https://developers.google.com/maps/documentation/places/android-sdk/. Accedido por última vez: 10-12-2021.

[8] OpenStreetMap. OpenStreetMap Wiki. https://wiki.openstreetmap.org/wiki. Accedido por última vez: 30-05-2022.

[9] Jsoup org. Jsoup: Java HTML Parser. https://jsoup.org/. Accedido por última vez: 02-03-2022.

[10] Overpass Turbo. Overpass Turbo query. https://overpass-turbo.eu/. Accedido por última vez: 08-05-2022.

[11] Overpass Turbo. Overpass API/Overpass QL. https://wiki.openstreetmap.org/wiki/Overpass_API/Overpass_QL. Accedido por última vez: 29-03-2022.

[12] OpenStreetMap. OpenStreetMap Category: Food and beverages. https://wiki.openstreetmap.org/wiki/Category:Food_and_beverages. Accedido por última vez: 23-02-2022.

[13] Smith Ben. *Beginning JSON*. Berkeley, CA, 2015. Accedido por última vez: 25-04-2022.

[14] Google. Google Firebase documentation. https://firebase.google.com/docs. Accedido por última vez: 30-05-2022.

[15] Meier Andreas and Kaufmann Michael. *SQL NoSQL Databases : Models, Languages, Consistency Options and Architectures for Big Data Management*. Wiesbaden : Springer Fachmedien Wiesbaden, 2019. Accedido por última vez: 15-02-2022.

[16] Ableson Frank, Collins Charlie, and Sen Robi. *Android: Guia para desarrolladores*. Anaya Multimedia, 2011. Accedido por última vez: 12-12-2021.

[17] Burnette Ed. *Android*. Anaya Multimedia, 2010. Accedido por última vez: 18-10-2021.

[18] Google. Android user interface and navigation. https://developer.android.com/guide/topics/ui/. Accedido por última vez: 27-04-2022.

[19] Google. Material design Android components. https://material.io/components?platform=android. Accedido por última vez: 28-04-2022.

[20] Google. Google Firebase read and write. https://firebase.google.com/docs/database/android/read-and-write. Accedido por última vez: 30-04-2022.

[21] RegExr. RegExr: learn, build, test Regular Expressions. https://regexr.com/. Accedido por última vez: 20-04-2022.

[22] Mapsforge org. OpenStreetMap MapsForge wiki. https://wiki.openstreetmap.org/wiki/Mapsforge. Accedido por última vez: 10-02-2022.

Apéndice A

Introduction

A.1. Motivation

Due to globalization, the catalog of restaurants, meals, eating habits and regimens is broader than ever: from food from different countries to the big variety of drinks, through specific menus according to the conditions of the client (with or without gluten) , and becoming food trends that are increasingly present in the gastronomic scene (veganism, vegetarianism, etc.).

The diversity is so overwhelming that a lot of difficulties can arise at the time to find the food that we want, making this an uncomfortable and complicated task. For this reason, a solution is needed that makes this process easier, without forgetting the big variety of types of food and existing tastes mentioned above.

When we go out to eat, we want to go to the place that best suits both our budget and our taste, so we tend to go to places that we trust or that have been recommended to us by friends or family. However, when we do not want to go to the usual establishments or we find ourselves doing tourism in other locations, we have to find out which are the best restaurants in the place, in that moment the mobile device becomes our greatest ally.

A.2. Objectives

The main objective of the Final Degree Project is the implementation of a mobile application that collects data on the preferences of the different users in order to subsequently offer them the greatest amount of information available about restaurants according to their tastes and needs. To achieve this, the following established goals will be pursued throughout the entire process:

- **Obtain the information of the different filters chosen by the user:** in order to offer the best results.
- **Register reviews:** which provide the user with a general idea about the restaurant that interests him based on the evaluations and experiences of other diners.
- **Register favorite restaurants:** in order for the consumer to consult the restaurants that most interest him more quickly.
- **Modify review:** giving the user the ability to change their opinion of a restaurant.
- **Use without registration:** so that users reluctant to provide personal information, such as an email, can use the app.
- **Open source:** the application will use open source tools whenever possible so that anyone who wants to can understand its operation more easily. This can be implemented using data collection tools such as OpenStreetMap, which are generated by users in their community and they are not affected by external agents such as the use of advertising for a better positioning of a certain location, being that way a reliable method. comparison between different restaurants.
- **Web scraping:** this system will be used for a greater and better collection of data not available or outdated by the main data collection tool. Thanks to this, modifications

generated in real time are offered in the menus of the restaurants, searching directly on their website for certain keywords.

Keeping in mind that all this will be accompanied by an elegant and simple interface that will be attractive to users who uses the app, without abandoning its easy handling to make it accessible to all types of consumers.

A.3. Workplan

All serious work must include a good planning of the work to be done according to the available time. The time has been planned in three main phases: research and strategy, development and, finally, deployment. Within each of these phases there are different tasks to do. For example, before choosing the objectives and building the use cases, we must analyze the competition to see what we can offer new and, later, carry out the design.

In the same way, in the development phase, it is necessary to distinguish the construction of the front part with the back part, in addition to the database. On the other hand, a very important part must be included, such as testing, to check throughout the entire process that the different application utilities are working correctly.

The final phase iscomposed of three main tasks: the actual deployment of the application, the analysis of results and the preparation of this report, which although it has been thought about throughout the work, in this period is the time to complete it. The code of this Final Degree Poject is available in the following repository of GitHub[1].

To show it in a more graphic way, a Gantt chart has been designed, visible in Figure A.1, where it has been defined visually the duration and the tasks to be fulfilled for the correct fulfillment of the objectives.

The Gantt chart is a very useful tool for planning any type of project. All scheduled tasks are displayed in an overview in an orderly manner along with their monitoring over

[1] https://github.com/juanpegc/appMenus

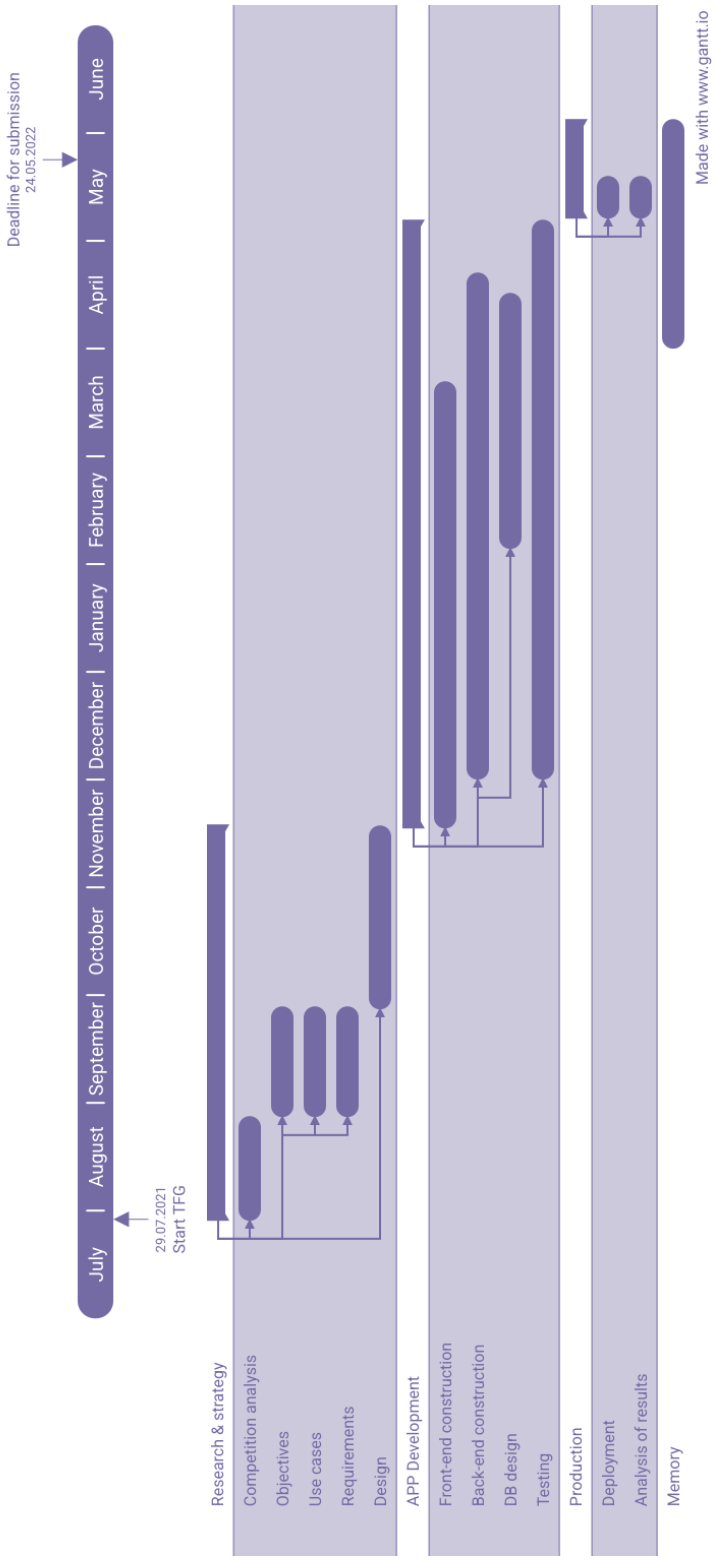

Figura A.1: *Gantt chart.*

time. This diagram has been made on a web page specialized in these diagrams: Gantt.io[2].

A.4. Memory organization

The memory is organized according to the standards proposed for carrying out projects of this caliber. This section will briefly and orderly describe the content of each chapter in order to clarify the structure of the report:

- **Chapter 1. Introduction** in this chapter, there is a brief summary of what the project itself is. It explains where and why the idea comes from, as well as it is intended to be achieved with the Final Degree Project. The tasks will also be described in a temporary frame, using a Gantt chart.

- **Chapter 2. State of the Art** a comparison is made of the technologies used with other alternatives available on the market. In addition, some real usage examples are shown.

- **Chapter 3. design of the application** the actors involved are graphically detailed along with their use case diagram. In addition, both the design of the graphical interface and the database are discussed. On the other hand, the methodologies and tools used for the collection of information, its transformation and the subsequent introduction into the database are also mentioned.

- **Chapter 4. Implementation of the application** both front-end and back-end design is explained. The framework used and the chosen model are also detailed in depth, exposing the final result through different images.

- **Chapter 5. Results obtained** the results obtained from the technical tests carried out on the use of the application and the opinions received from the users who have been able to test it are analyzed.

[2] https://www.gantt.io/

- **Chapter 6. Conclusions and Future Work** it is mentioned the conclusions and achievement of the project's objectives and some of the ideas that could not be carried out and could be interesting for future work.

Apéndice B

Conclusions and future work

B.1. Conclusions

Once the development of the project is finished, it is time to make an assessment of the work done and what has been learned to date.

The objective of this project was to carry out the design, development and implementation of a native application for Android devices, the purpose of which is to search for restaurants according to the user's interests, expressed through filters and taking into account the opinion of users who have shared a review in the restaurants.This goal can be marked as met. Accompanied by a simple and attractive interface to consult restaurants. The application is also capable of synchronizing with a server where certain data is stored in order to later show it to other users.

There is a functionality that has always been wanted to be implemented but has not been possible: the display of the map. Some problems arose in the process of setting up the previously chosen tools and performance issues. As OSM is a tool whose use is mainly through web pages, there are no implementations of it for Android. Therefore, one of the options was to implement a direct view of your web in the application, but the use of the map through this method was very complex. The other option was to use a third-party API, but after implementing it the performance was very low, with constant map rendering failures and complexity in selecting points on it. The Android application has an architecture based

on three elements: database, back-end and front-end.

Throughout the career we have studied many subjects. Some of them have been essential for the development of this project.

- **Programming Technology.** in this course we learned the basics of object-oriented programming. More specifically, we saw Java for the first time, which has been the language we have used in the application.
- **Databases.** both this course and Database Expansion have helped us to design the model and manage all the information.
- **Software Engineering.** thanks to the knowledge received about design patterns, everything has been done following a good code pattern.
- **Application Programming for Mobile Devices.** because of is an optative subject, not all of us have been able to take it. Who have done it appreciate it because we have received in it a lot of knowledge that has been very useful.
- **Data structure.** the usefulness of this course has been to create a better organization of the data used, using for example sets and maps for greater efficiency in the search and organization of restaurants and reviews.

B.2. Future Work

This section will describe ideas that add more value to the project, but due to complexity and time available it has not been possible to develop them. Next, the possible improvements or new functionalities that could be added in future versions will be detailed.

- **Show the map graphically:** for greater visibility, the closest available restaurants could be shown on the map. The possibility of limiting a search area could also be offered.

- **Narrow a search area:** a search range can now be applied. With this future feature, the area in which the user wants to search could be drawn on the map.
- **Selecting the location on the map:** currently you can only change the search location by manually entering the coordinates. This is not comfortable or accessible for a large majority of users. The possibility of pointing it with a touch on the map would make the app much easier to use and add value.
- **Multiple languages:** to have the application available for other countries and to have a database of foreign users. This would further extend the number of users who could use the app.
- **Swappable night mode:** it would be very nice if the application supported night mode throughout the application with the desired colors and in a correct and uniform way. This option could be selected by the user when opening the application and in the settings area.
- **Support on other devices**: make the app work for other devices like tablets with better UI and study the possible compatibility of certain features on some smartwatches.

Apéndice C

Flujo de uso y demo de la aplicación

En este anexo se va a enseñar el funcionamiento de la aplicación completa a través de un flujo de uso y una demostración en vídeo de como funciona cada caso de uso. Se dispone de una explicación en vídeo[1] accesible mediante la *URL* del fichero *README* en el repositorio de *Github* mencionado en la Sección 1.2.

A continuación se explicará paso a paso el recorrido por *Food Feeltr*.

C.1. Registro de un usuario

El registro de un usuario en *Food Feeltr* es muy sencillo. Nada más abrir la aplicación mostrará la pantalla de inicio de sesión. En esta pantalla se debe de pulsar en el botón [Registrarse]. Una vez en la pantalla de registro, representada en la Figura C.1, hay que introducir un nombre, un correo y una contraseña. Al pulsar en el botón [Registrarse], iniciará sesión automáticamente y mostrará la pagina principal.

A la hora de abrir la *app* si el usuario inició sesión anteriormente se mostrará directamente la página principal.

[1] Carpeta contenedora del vídeo: https://drive.google.com/drive/folders/1XaE0IhyrUBVN7mg7jIDMfS1UMSLSUF5_?usp=sharing

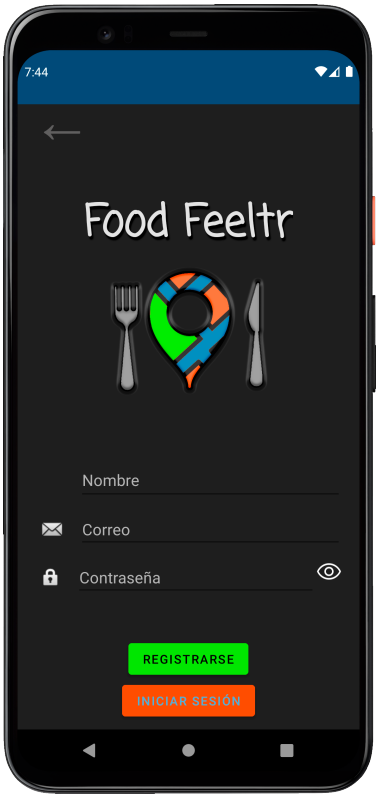

Figura C.1: *Vista de registro de usuario.*

Figura C.2: *Vista de inicio de sesión.*

C.2. Inicio de sesión de un usuario

La aplicación consta de 2 formas de iniciar sesión: con un usuario propio o con un usuario invitado. Ambas opciones se pueden realizar desde la pantalla de inicio de sesión, mostrada en la Figura C.2.

Para hacer *login* con un usuario propio se debe introducir un usuario y una contraseña. Las credenciales son las mismas con las que se registró anteriormente. Al pulsar en el botón [iniciar sesión] se mostrará automáticamente la página principal.

Para hacer *login* con un usuario invitado se debe pulsar en el botón [acceder sin iniciar sesión]. Esta opción permite navegar por la aplicación con algunas restricciones tales como: 'No podrá añadir una reseña', 'No podrá añadir un restaurante a favoritos', 'No podrá acceder a los ajustes del usuario', para realizar estas acciones es necesario iniciar sesión con

una cuenta propia.

C.3. Navegación por la aplicación

Una vez que se haya iniciado sesión, la aplicación mostrará la página principal y en la parte inferior se mostrará una barra de navegación compuesta por 3 botones: [Inicio], [Filtros] y [Perfil].

Figura C.3: *Vista al pulsar el botón de inicio.*

Figura C.4: *Vista al pulsar el botón de filtros.*

Figura C.5: *Vista al pulsar el botón de perfil.*

Cada uno de estos botones tiene una funcionalidad específica que detallaremos a continuación:

- Al pulsar sobre el botón [Inicio] mostrará el listado de restaurantes, como se muestra en la Figura C.3.

- Al pulsar en el botón [Filtros] mostrará una pantalla con todos los filtros disponibles en la *app*, como se muestra en la Figura C.4.

- Al pulsar en el botón [Perfil], los resultados que se muestren dependerá de con qué usuario se haya accedido a la *app*. Si es un usuario propio mostrará la pantalla de su perfil junto con 2 botones: [Mis reseñas] y [Mis favoritos], además de contar con un botón de [ajustes], en el caso de ser un usuario invitado te mostrará la pantalla con un botón [Iniciar sesión], como se muestra en la Figura C.5.

Disponer de una barra de navegación con diversas opciones hace la aplicación muy fácil de interpretar mostrando las características básicas de la aplicación en una sola vista.

C.4. Búsqueda de restaurantes mediante filtros

Para encontrar restaurantes acordes a las preferencias del usuario se debe pulsar en la opción [Filtros] que se encuentra en la barra de navegación, como se muestra en la Figura C.4. Al pulsar sobre la opción [Filtros] se mostrará en la parte superior un *slide* llamado 'radio de búsqueda' donde el usuario podrá especificar la distancia en la que se quiere aplicar la búsqueda, el máximo será de 3000 metros, además de eso la aplicación muestra una gran variedad de filtros ordenados por categorías.

Para elegir los filtros se debe pulsar en aquel que se adecue a las preferencias del usuario, una vez realizada esta acción, este cambiará de un color gris a azul claro para diferenciar aquellos filtros aplicados, como se muestra en la Figura C.6.

Una vez seleccionados los filtros deseados se debe pulsar en el botón [Filtrar], posteriormente nos mostrará una lista de restaurantes que cumplan con el filtrado de búsqueda.

Nota: No olvidar activar el *GPS* del teléfono para que nos muestren los resultados más cercanos al radio de búsqueda.

Figura C.6: *Vista de filtros al ser seleccionados.*

C.5. Detalle de los restaurantes

Para ver un restaurante en concreto y no sólo los datos básicos ofrecidos en el listado de inicio hay que pulsar sobre la imagen del restaurante. Una vez hecho esto se mostrará una nueva pantalla, reflejada en la Figura C.7. En esta pantalla se muestra toda la información del restaurante que se ha podido conseguir, datos como por ejemplo: valoración media, filtros, dirección, imágenes, reseñas, distancia (a la ubicación actual), número de teléfono y una *URL* para dirigirse a la página *web* del restaurante.

Dentro del detalle del restaurante muestra dos botones: [Añadir una reseña], [Añadir filtros], cada botón tiene una funcionalidad concreta, al pulsar en el primer botón se abrirá una pantalla donde el usuario podrá añadir una reseña al restaurante en cuestión, y al pulsar en el botón añadir filtros, te mostrará una nueva pantalla con todas las categorías de filtros y se permitirá añadir los filtros seleccionados al restaurante, para mejorar la calidad de la

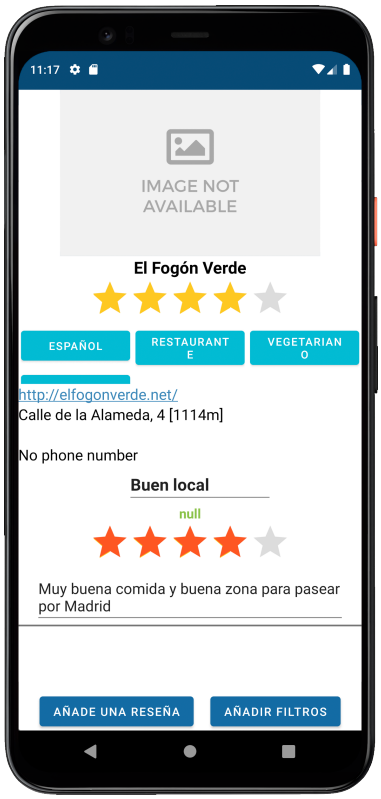

Figura C.7: *Vista de restaurante detallado.*

información aplicación.

C.6. Añadir reseña a un restaurante

Para añadir una reseña a un restaurante hay que pulsar al botón [Añade una reseña]. Este botón sólo se encuentra en la vista detallada de un restaurante, como se muestra en la Figura C.7. Cabe destacar que si el usuario ha accedido a la aplicación como invitado, al pulsar en el botón [Añadir una reseña] le mostrará la siguiente alerta: 'Login necesario'

En el caso de que se inicie sesión con un usuario propio, al pulsar en el botón [Añadir una reseña] mostrará una nueva pantalla donde se podrá introducir la valoración y la reseña. Esta nueva pantalla cuenta con dos campos a rellenar: el primero es el título de la reseña y el segundo es una descripción más extensa de la reseña. Además está la valoración que se le otorga al restaurante pulsando en las estrellas, como se muestra en la Figura C.8. Para que

la reseña se guarde hay que pulsar en el botón [Publicar]. Todas las reseñas que el usuario haya publicado se podrán ver en la sección de perfil, como se muestra en la Figura C.9 y a su vez se podrá visualizar dentro del restaurante donde se publicaron, debajo de la información general.

Figura C.8: *Vista de añadir una nueva reseña.*

Figura C.9: *Vista de las reseñas del usuario.*

C.7. Añadir un restaurante a favoritos

Esta opción se encuentra en el listado de restaurantes que se muestra en la página principal. En la esquina superior derecha de cada restaurante hay un icono con forma de corazón. Este es el botón para añadir a favoritos. Esto lo vemos representado en la Figura C.3. Por supuesto si el usuario ha accedido a la aplicación como invitado al pulsar el botón mostrará la siguiente alerta: '*login* necesario'

Al pulsar sobre el icono del corazón de un restaurante, se mostrará una alerta: 'Restaurante 'x' añadido a favoritos' y automáticamente este restaurante se añadirá a los favoritos del usuario que haya iniciado sesión. Todos los restaurantes que el usuario haya marcado como favoritos se pueden encontrar en la sección del perfil del usuario, como se ve en la Figura C.5. Para eliminar el restaurante de la lista de favoritos se debe pulsar nuevamente sobre el icono del corazón.

C.8. Perfil del usuario

Figura C.10: *Vista de perfil válido.*

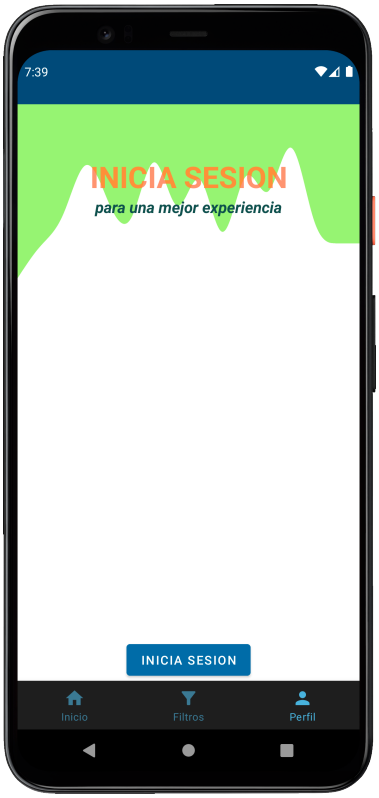

Figura C.11: *Vista de perfil si no se inicia sesión.*

Para entrar en esta sección hay que pulsar en la última opción de la barra de navegación. La vista de este módulo es muy diferente dependiendo de si el usuario ha iniciado sesión o ha entrado como invitado. Se pueden ver las diferencias en la Figura C.10 y la Figura C.11.

Como se puede ver si el usuario ha accedido como usuario invitado se mostrará el botón [Inicia sesión].Al pulsar en este botón te llevará a la página de *login*. Así el usuario se podrá identificar y tendrá acceso a todas las funcionalidades.

Sin embargo si el usuario ha iniciado sesión con un usuario propio, en esta pantalla podrá ver las reseñas que haya publicado junto con sus restaurantes favoritos. Para ver las reseñas hay que pulsar en el botón [Mis reseñas] y para ver los restaurantes en el botón [Mis favoritos]. Además de estos dos botones hay un tercero en la esquina superior derecha. Este botón llevará a la sección de opciones del usuario.

C.9. Opciones del usuario

Como se puede ver en la Figura C.12 el menú de opciones es muy sencillo y concreto. Se muestran seis botones:

Figura C.12: *Vista de ajustes de usuario.*

- **Cerrar sesión:** para que el usuario pueda volver a la pantalla de inicio de sesión.
- **Cambiar contraseña:** si el usuario desea cambiar la contraseña de su cuenta. Al pulsarlo se abrirá una ventana, como se ve en la Figura C.13, con unos campos que se deben rellenar correctamente.
- **Cambiar nombre:** este botón permite al usuario cambiar el nombre que se usa para las reseñas publicadas. Al pulsarlo aparecerá un *pop up*, reflejado en la Figura C.14 donde se pedirá al usuario rellenar dos campos.
- **Cambiar localización:** para que el usuario pueda cambiar la localización manualmente. Al presionar en este botón aparecerá una ventana pequeña, como se ve en la Figura C.15.
- **Borrar datos:** al pulsarlo aparecerá el *pop up* de la Figura C.16 para confirmarlo.
- **Borrar cuenta:** para que el usuario pueda borrar la cuenta de nuestra base de datos. Al pulsar el botón aparecerá el *pop up* de la Figura C.17 para confirmarlo.

Figura C.13: *Pop-up de cambiar contraseña.*

Figura C.14: *Pop-up de cambiar nombre.*

Figura C.15: *Pop-up de cambiar localización.*

Figura C.16: *Pop-up de borrar datos.*

Figura C.17: *Pop-up de borrar la cuenta.*

Apéndice D

Reparto de trabajo

En este apéndice se va a explicar detalladamente la aportación de cada uno de los miembros al Trabajo de Fin de Grado que se ha desarrollado.

D.1. Sáenz Bullón, Pablo

Este miembro del equipo inicialmente se dedicó a aprender y configurar los elementos básicos de una aplicación *Android*. Posteriormente, realizó unos bocetos de la misma (junto a los otros miembros del equipo) y en una de las reuniones se decidió elegir uno como el prototipo de la *app*, tras lo cual se dispuso a realizar una implementación básica del modelo visual elegido.

En la siguiente fase del proyecto, se dedicó a crear las clases básicas de las que estaría compuesta la aplicación, como 'restaurante o usuario' y, una vez creadas, se dispuso a crear un prototipo visual de la aplicación, que no utilizaba datos reales sino preestablecidos.

Posteriormente, se dedicó a explorar diferentes posibilidades para la obtención de la información de los locales, barajando como primera opción *Google Maps*, la cual se descartó en una de las reuniones, junto a los demás miembros del equipo, debido a su alto precio y a no ser *open source*. En esa misma reunión se acordó utilizar *OpenStreetMap*.

Durante la siguiente fase investigó la utilización de *OSM* y como incluirla en la aplicación, utilizando finalmente la herramienta de *Overpass Turbo*, para lo que tuvo que comprender su utilización y sintaxis de las *querys*. Todo ello mientras desarrollaba una forma muy básica

de representar los datos obtenidos mediante esta herramienta y actualizar las partes del proyecto requeridas para dicha implementación.

Una vez la búsqueda de datos funcionaba de forma óptima, su trabajo fue obtener los datos mediante *web scraping* de las páginas *web* de los locales, para ello valoró junto a los demás integrantes las distintas herramientas disponibles para su búsqueda, así como los datos relevantes a obtener. Una vez dichos requisitos estaban claros, se dispuso a realizar los cambios necesarios en la *app* para su implementación y probar su correcto funcionamiento, así como refinar la búsqueda para maximizar los resultados obtenidos de cada local.

Tras la implementación de *web scraping* y de *OSM* y una vez había sido configurada la base de datos, se dedicó a realizar la funcionalidad de lectura y modificación de los datos de la misma, así como una implementación visual básica del perfil del usuario.

Como último aporte a la aplicación, buscó una forma de implementar un mapa en la misma para así poder brindar una mejor experiencia a los usuarios, contemplando distintas opciones para realizar dicha tarea y llevando a cabo la implementación básica de una de ellas, aunque como se ha explicado en capítulos anteriores, surgieron muchos problemas durante su implementación y se descartó del resultado final.

Como aportación a la memoria colaboró en las secciones relacionadas a *web scraping, OSM* y estructura de la base de datos, así como la realización del *testing* de la aplicación desarrollada, realizando las mediciones oportunas. Además colaboró en la supervisión de los fallos que pudiese haber en la misma y añadiendo elementos o detalles que no se habían mencionado.

D.2. García Castaño, Juan Pedro

Este miembro del equipo primeramente estuvo informándose sobre la tecnología *Android*. Realizó un curso de *Google* Actívate para aprender los conceptos básicos sobre el desarrollo de aplicaciones móviles. A la vez llevó a cabo una investigación sobre aplicaciones en la *Play Store* similares a la que se iba a desarrollar, sobre todo *Yelp*.

Después, al igual que el resto de miembros del equipo, estuvo diseñando prototipos de una interfaz de usuario para la aplicación. También realizó con *Adobe Photoshop* las propuestas de logo que se sometieron a votación con todos los miembros del TFG. Además pensó en nombres para la aplicación, de los cuales salió el elegido: Food Feeltr.

Simultáneamente creó el repositorio de *GitHub* en el que se iba a alojar el código. Después de haber llegado a un acuerdo en la interfaz de la aplicación se dedicó a ir haciéndolas en *Android Studio*. Estuvo encargado de realizar estas interfaces y de encontrar un diseño que fuera atractivo. En concreto el menú de opciones fue labor suya, con todos los *pop up*.

Por otro lado, hizo el diagrama de casos de uso y lo propuso a los tutores. Después de hacer el diagrama hizo todas las tablas para cada uno de los casos de uso. Estas tablas fueron puestas en la memoria junto con el diagrama.

Además del diagrama de casos de uso también estuvo encargado de hacer el diagrama de Gantt (en español y en inglés) para expresar de forma gráfica las tareas y su duración. Por otra parte también realizó el diagrama de arquitectura del sistema y el diagrama de entidad-relación.

Con respecto a la memoria recibió la carga de trabajo equitativa al igual que el resto de miembros del grupo. Participó en la redacción de muchos puntos de la memoria y en la revisión de otros tantos. Junto a las tareas de redacción también llevó a cabo un trabajo de lectura global de la memoria para mejorar la consistencia de la misma. Particularmente se encargó de conseguir las capturas de pantalla de la *app* que a lo largo en la memoria.

D.3. Choy Castillo, Miriam Patricia

Este miembro del equipo al principio invirtió tiempo en buscar información por *Internet* acerca de *Android* por ser un lenguaje nuevo que debía aprender para ponerlo en práctica en este TFG.

Al inicio de la planificación se dedicó a hacer unos prototipos de diseño para la aplicación al igual que el resto de miembros del equipo. Después de realizar las comparaciones de los

bocetos se decidió de manera conjunta la que actualmente tiene la aplicación.

Posteriormente se encargó del diseño de la aplicación, empezó a crear algunas vistas principales de la *app* en *Android Studio*, al mismo tiempo que buscaba posibles servicios *web* para alojar la aplicación.

Por otro lado se encargó de montar la base de datos seleccionada: (*Google Firebase*). Para ello tuvo que buscar mucha información y ver vídeos en *YouTube* para poder realizarlo de manera satisfactoria. Al principio se enfrentó con ciertos problemas porque la información no se registraba bien. Después de volver a buscar información para solventar el problema que le ocurría llegó a la conclusión que era tema de configuración: faltaba configurar un módulo en *Google Firebase*. Por ese motivo en esta memoria se incluye una guía de configuración de la base de datos con la finalidad de ayudar a otros estudiantes.

Una vez configurada la base de datos, también implementó la parte del registro y *login* del usuario en la *app* ya que tenía que ver con la base de datos.

Finalmente se encargó de realizar la memoria junto con los otros miembros del equipo. También realizó pruebas funcionales de la aplicación y solucionó alguna cosa de diseño que faltaba por implementar.

Este TFG ha sido repartido entre todos los miembros del equipo, cada uno de ellos ha realizado cosas de distintas fases a pesar de que desde un principio habían elegido enfocarse en algo en concreto. Durante el proceso de este trabajo se ha tenido buena comunicación y comprensión entre todos, se ha notado el compromiso y las ganas de querer sacar adelante este proyecto.