TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

UNIVERSIDAD DE ALMERIA

ESCUELA SUPERIOR DE INGENIERÍA

"Desarrollo de una aplicación Android para la geolocalización de comercios almerienses"

Curso 2016/2017

Alumno/a:

Diego Mateo Moya Hurtado

Director/es:

Nicolás Padilla Soriano

Javier Criado Rodríguez

AGRADECIMIENTOS

Quiero agradecer a mis padres todo su apoyo económico, emocional y educativo prestado de forma incondicional durante toda la carrera y elaboración de este proyecto. Sin ellos, no hubiera sido posible llegar hasta aquí. También quiero agradecer al resto de mi familia y amigos por toda su ayuda y buenos momentos que he pasado junto a ellos, los cuales me han ayudado a reunir el ánimo y entusiasmo para enfrentar este trabajo.

También me gustaría agradecer a mis tutores Nicolás Padilla Soriano y Javier Criado Rodríguez. Su guía, apoyo y disponibilidad me han ayudado mucho a lo largo de todo el proyecto. Quiero agradecer también a todos los profesores, quienes me han ayudado y enseñado mucho durante la carrera de Informática.

También quiero agradecer a toda la comunidad de stackoverflow, de la que yo mismo me he hecho participante durante el desarrollo de este proyecto, por prestar su ayuda y resolver dudas a programadores menos experimentados.

Por último, quiero agradecer a mis compañeros de carrera con los que he compartido asignaturas difíciles y quiénes me han ayudado en muchas ocasiones a resolver dudas y seguir adelante.

Gracias a todos.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Capítulo 1:
Introducción

1. INTRODUCCIÓN

Los habitantes de un municipio necesitan disponer de un directorio donde poder encontrar información acerca de los comercios de los que dispone. Existen algunos directorios como las Páginas Amarillas o la información disponible en Google acerca de los comercios y servicios disponibles en una localidad. Sin embargo, dichos directorios no son herramientas dirigidas de forma específica a mostrar los distintos comercios presentes en la localidad, si no que éstos requieren de una búsqueda previa. Como consecuencia, desde nuestro punto de vista, creemos que la existencia de una aplicación más especializada en los comercios de una determinada ciudad puede aportar una serie de ventajas, como por ejemplo ofrecer al usuario una vista de todos los negocios, pudiendo ser filtrados por nombre o categoría.

Con la aplicación web de Comercio de Almería (disponible en www.comerciodealmeria.es), el Grupo de Informática Aplicada de la Universidad de Almería realizó un prototipo de aplicación web para ofrecer una primera versión de esta solución al Ayuntamiento de Almería. Sin embargo, en la aplicación web no se explotan una serie de funcionalidades que sí podrían ser mejor implementadas con una aplicación para dispositivos móviles. Además, dicha aplicación podría estar optimizada para el uso de características como la geolocalización, el envío de notificaciones o el cálculo de rutas. Además, se pretende que el usuario pueda buscar negocios por categoría o por nombre, buscar mercados y mercadillos de la zona, noticias y productos locales. Y desde el punto de vista del comerciante da la posibilidad de que los negocios se anuncien favoreciendo así el comercio de la región.

El hecho de haber elegido este proyecto como TFG se debe principalmente a mi entusiasmo por las aplicaciones móviles y mi creciente interés por la programación. Para mí no eran algo nuevo, pues a lo largo de toda la carrera he estudiado lenguajes de programación como Java y además en las prácticas curriculares trabajé con una aplicación Android de gestión destinada a técnicos agrónomos. Fue en ese momento cuando me percaté de que realmente me gustaba este campo de la informática, cosa que no me hubiera imaginado unos años antes.

Durante los primeros años de carrera, nunca tuve una gran destreza en las asignaturas de programación, además de que siempre me parecieron las más exigentes y difíciles de aprobar. De hecho, escogí la especialización de Tecnologías de la Información para evitar asignaturas de este tipo e ir en otra dirección. Sin embargo, con el tiempo y la práctica aprendí a desenvolverme con la programación y haber sido capaz de aplicarlo a un problema real en las prácticas de empresa me causó una gran satisfacción. Es por ello, por lo que me gustaría seguir aprendiendo y mejorando mis habilidades en este sector. Así que llegado el momento de elegir un TFG, me hice la pregunta ¿por qué no desarrollar una aplicación móvil desde cero? Y por suerte encontré una propuesta por parte de mi director de proyecto sobre geolocalización, algo que siempre me había llamado la atención.

Elegir Android como sistema operativo con el que trabajar no era una de mis prioridades, pero llegué a esa conclusión teniendo en cuenta que la aplicación debía ser nativa, por ofrecer una mejor experiencia de usuario y que Android es la plataforma con mayor cuota del mercado.

1.1. Objetivos

El objetivo principal del TFG es el de crear una aplicación para dispositivos móviles que permitan al usuario localizar negocios en la localidad de Almería.

Para cumplir este objetivo, la aplicación debe cumplir con una serie de sub-objetivos:

- Ofrecer al usuario una interfaz sencilla y eficiente que permita encontrar un negocio determinado en base a una serie de criterios o filtros, como la categoría del negocio, la zona, la proximidad, la valoración, etc.
- Ver los resultados de la búsqueda en un mapa o aplicar los filtros directamente sobre éste.
-Dar la posibilidad al usuario de ver anuncios de comerciantes relacionados con sus productos o buscar ofertas para un producto o categoría en concreto.
- Dar a conocer las noticias relacionadas con los comercios de la provincia.
- Permitir a los negocios darse de alta en la aplicación para anunciarse a los clientes.
- Recordar los negocios buscados por el usuario a través de un historial para así mostrar la información en base a sus gustos o intereses.
- Recordar al usuario mediante notificaciones o alertas que un negocio quería ser visitado al encontrarse en un radio de proximidad determinado.
- Dar a conocer los productos locales de la región y fomentar su consumo.
- Poder añadir comercios a una sección de Favoritos para facilitar la consulta en el futuro.
- Incluir una sección de Ajustes para que el funcionamiento de la aplicación sea configurable por el usuario.
- Ofrecer el mejor rendimiento y fluidez posible dentro de los límites permitidos por la tecnología escogida.
- Ofrecer un buen funcionamiento en los distintos dispositivos.

Capítulo 2:
Análisis DAFO

2. ANÁLISIS DAFO

2.1. Debilidades

-Poca experiencia en desarrollo de aplicaciones.

-Al tratarse de una aplicación específica para Android, no se cubre la totalidad de la cuota de mercado.

-La aplicación no está disponible para todas las versiones de Android, si no a partir de la versión 4.0 de Ice Cream Sandwich.

2.2. Amenazas

-Actualmente ya existen aplicaciones relacionadas con geolocalización de negocios. Algunos ejemplos de ellas son Foursquare, Tiendeo, Yelp o el propio Google maps.

-Constante innovación en el mundo de las apps.

2.3. Fortalezas

-La app impulsa el comercio local frente a grandes superficies.

-La app impulsa el producto local elaborado en la provincia.

-La app ofrece la posibilidad de publicitar cualquier negocio de forma gratuita.

-Ofrece información y orientación sobre cómo montar un negocio, favoreciendo el emprendimiento.

-Permite el envío de alertas de proximidad y trazado de rutas para una mejor localización de un negocio determinado.

2.4. Oportunidades

-La versatilidad de la app permite la introducción de nuevas funcionalidades.

-Puede interesar a cualquier persona localizada en la provincia.

-La app puede extenderse en el futuro a otras regiones.

-Puede distribuirse de forma gratuita para una mayor captación de usuarios.

Capítulo 3:
Fases de realización del trabajo

3. FASES DE REALIZACIÓN DEL TRABAJO

El trabajo a realizar se puede dividir en las siguientes fases:

1. Estudio de otras soluciones del mercado y tecnologías existentes. En esta fase, se pretende realizar un estudio de otras aplicaciones móviles existentes en el mercado en el ámbito del proyecto. Se determinará las funcionalidades que ofrecen así como los requisitos necesarios para su funcionamiento. También se incluye en esta fase el estudio de tecnologías que pueden ser utilizados para la construcción del producto a realizar.
2. Análisis de requisitos y diseño. En esta fase, a partir de la información extraída de la fase anterior y del portal de comercio de Almería existente, se determinará los requisitos que deberá cumplir el producto a realizar así como el diseño del mismo (funcionalidad, interacción y diseño de la interfaz gráfica). Aspecto fundamental en esta fase es seleccionar la tecnología necesaria para la implementación de la app.
3. Implementación. En esta fase se realizará la codificación de la app utilizando la tecnología seleccionada en la fase 2. También se incluye las tareas de depuración que se irán haciendo durante el desarrollo del producto.
4. Pruebas. Esta fase incluye las pruebas de la aplicación en la plataforma para la que se desarrolla para comprobar su correcto funcionamiento en diferentes dispositivos con especificaciones distintas, así como de todos los aspectos del programa y todos los casos de error.
5. Compilación. Generación de un ejecutable (.apk) a partir de la compilación del código fuente.
6. Evaluación. En esta última fase se procede a redactar la memoria y a exponer el trabajo realizado frente al tribunal de evaluación.

3.1. Cronograma

Estudio del mercado	
02/05/2016 - 15/05/2016	-Estudio de las distintas tecnologías presentes en el mercado -Estudio de viabilidad para el desarrollo en cada tecnología -Elección de la tecnología -Investigación sobre otras soluciones similares presentes en el mercado
Análisis de requisitos y diseño	
16/05/2016 - 01/05/2016	-Análisis de la página web de Comercio Almería y de sus funcionalidades -Elección de las funcionalidades -Diseño de la aplicación en base a la web y las funcionalidades elegidas -Estudio de requisitos necesarios para la implementación en base al diseño elegido -Estudio de servicios de hosting para la base de datos remota -Diseño de las bases de datos
Implementación	
02/06/2016 - 31/10/2016	-Instalación de todas las herramientas necesarias para la implementación de la aplicación (Eclipse, SDK, APIS, plugins…) -Documentación de Android y estudio de manuales y ejemplos acerca de las funciones a implementar.

	-Implementación de las bases de datos y sus ficheros de acceso -Codificación de las clases Java -Depuración en diferentes dispositivos -Optimización del código implementado
Pruebas	
01/11/2016 - 14/11/2016	-Prueba del código de la aplicación y corrección de errores -Análisis de la adaptabilidad en pantallas de otras dimensiones -Diseño de layouts para pantallas de Tablet. -Prueba de la aplicación en diferentes dispositivos
Compilación	
15/11/2016 - 18/11/2016	-Estudio de herramientas para reducir el tamaño de la aplicación -Generación del certificado y fichero .apk
Evaluación	
19/11/2016 - 20/12/2016	-Evaluación de todo el trabajo realizado

Tabla 1. Cronograma asociado al desarrollo de la aplicación

3.2. Diagrama de Gantt

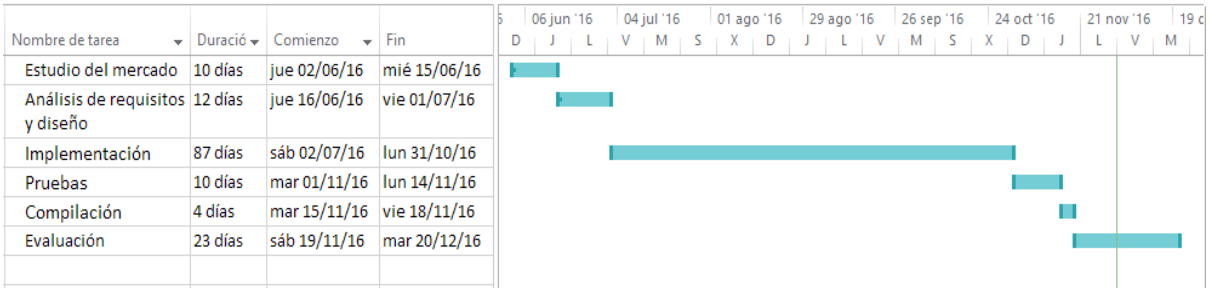

Figura 1. Diagrama de Gantt asociado al desarrollo de la aplicación

Capítulo 4:
Especificaciones generales

4. ESPECIFICACIONES GENERALES

4.1. Portal Comercio Almería

Como ya se ha mencionado el Grupo de Informática Aplicada de la Universidad de Almería realizó un prototipo de aplicación web para ofrecer una primera versión de esta solución al Ayuntamiento de Almería. Dicha aplicación web ofrece al usuario un mapa con los negocios de la localidad y la posibilidad de realizar búsquedas a partir de la categoría o nombre del negocio. Para cada negocio del mapa, se muestra información como el nombre del negocio, descripción, dirección, horario y teléfono. También cuenta con las secciones Noticias, Productos locales, Información y Contacto, donde se incluye por ejemplo, información relativa a los productos y empresas de la provincia.

La siguiente imagen muestra una captura de la pantalla principal de esta web:

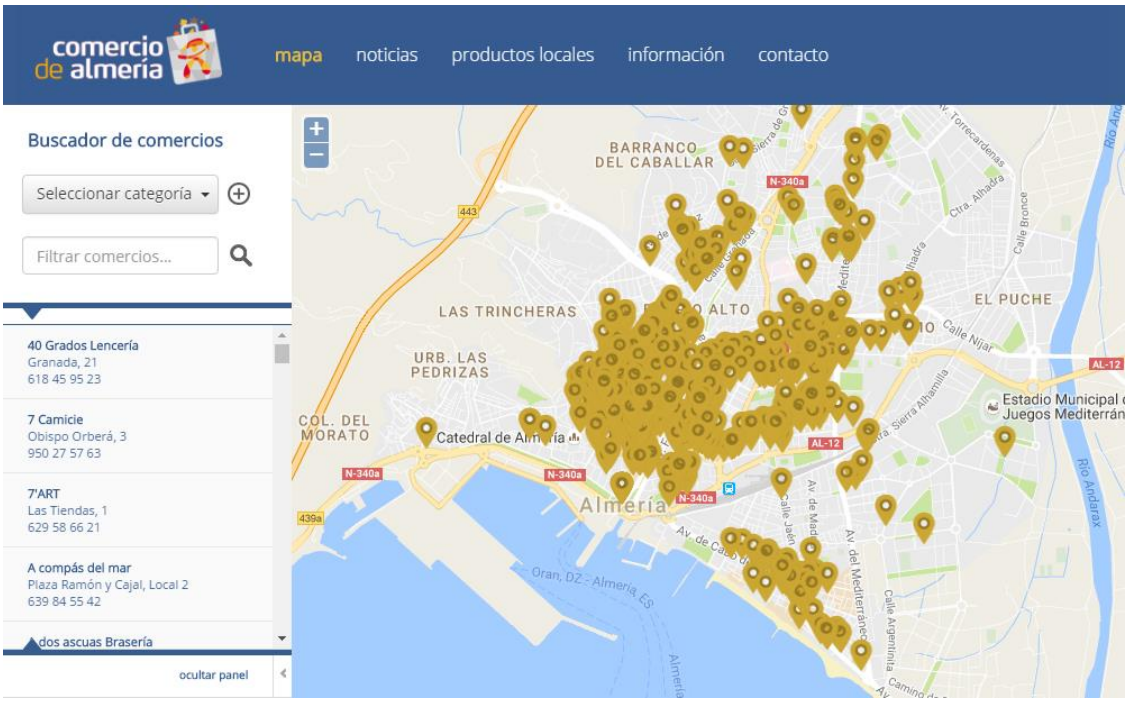

Figura 2. Captura de la página web de Comercio Almería

Por otro lado, la web también cuenta con limitaciones, como no estar optimizada para dispositivos móviles ni tampoco dar la posibilidad al usuario de hacer uso de características como el GPS, propia de estos dispositivos.

Tampoco se hace una distinción de los negocios mostrados en el mapa, pues como podemos observar, se le aplica el mismo color a todos los marcadores, haciendo difícil su identificación. Además, si observamos otras soluciones de mercado podemos apreciar que los servicios que ofrece esta web no hacen de ella un producto competitivo en comparación con otras aplicaciones de este tipo. Es por ello,

que se considera necesario la implementación de una aplicación móvil que ofrezca una serie de funcionalidades que la ayuden a ser más competitiva y captar mayor número de usuarios.

4.2. Propuesta de mejora

Se propone el desarrollo de una aplicación para dispositivos Android, para los cuales la página web no está optimizada. Dicha aplicación ofrecerá las funcionalidades propias de una solución de este tipo, es decir, explotará las características del GPS para permitir al usuario conocer su situación con respecto a un negocio determinado, trazar una ruta o incluso alertar de la proximidad a un comercio de interés. También se incorporarán funciones que saquen provecho de ciertas ventajas de estos dispositivos, como lanzar el servicio de llamada al hacer click sobre un número teléfono, compartir ubicación vía Whatsapp…

Además, en contraste con esta primera versión de la web, también se proporcionará una manera de diferenciar los negocios por categorías, asignando un color diferente a cada uno de los marcadores del mapa en base a su categoría. También incluiremos las secciones de Historial y Favoritos, para agilizar al usuario la búsqueda de negocios ya visitados o para etiquetarlos como favoritos si siente por ellos un interés especial.

Una aportación fundamental de la aplicación con respecto a la web, podría ser el de suministrar la mayor parte de las funcionalidades sin requerir de conexión a Internet. Para ello se puede hacer uso de las bases de datos locales y de ficheros .xml, mostrando al usuario información predeterminada a partir de ficheros de la misma aplicación o actualizándola en el momento en que se disponga de conexión.
Otro aspecto importante es el de proporcionar un entorno configurable en la medida de lo posible, a partir de una sección de ajustes, para hacer que la aplicación se adapte a las necesidades de cada usuario.

Todo ello hará de nuestra aplicación una herramienta competitiva en el mercado actual y aportará características nuevas no disponibles en la versión web.

Capítulo 5:
Especificaciones técnicas

5. ESPECIFICACIONES TÉCNICAS

A lo largo de todo el desarrollo del TFG se ha trabajado con diferentes dispositivos y herramientas para labores de diversa índole como la administración de bases de datos, programación, depuración…

Entre todas ellas destacamos las siguientes:

5.1. Dispositivos utilizados

-PC: El ordenador con el que se ha trabajado es un portátil Toshiba Satellite L500, con las siguientes características:

- Procesador Intel Core 2 Dúo P4750 a 2,13 GHz.
- Memoria RAM de 4 GB.
- Sistema operativo Windows 10 de 64 bits.
- Tarjeta gráfica ATI Mobility Radeon HD 4650.
- Disco duro TOSHIBA MK-5055GSX de 500 GB.

-Samsung Galaxy S4: dispositivo de telefonía móvil utilizado para la mayor parte de las tareas de depuración. Dispone de las siguientes características:

- Versión: 5.0.1
- Procesador Quad Core ARMv7 a 1.9 GHz.
- Memoria RAM de 2 GB.
- Memoria interna de 16 GB.
- Pantalla de 5”.
- Resolución 1920x1080 pixeles.

-Tablet Woxter Nimbus 101Q: dispositivo tablet utilizado también para las tareas de depuración. Tiene las siguientes características:

- Versión: 4.0.2
- Procesador Quad Core Cortex-A9 a 1.6 GHz.
- Pantalla de 10”.
- Resolución de 1280x800 pixeles.
- Memoria RAM de 1 GB.
- Memoria interna de 2 GB ampliada hasta 14 GB con tarjeta SD.

5.2. Herramientas

-Windows 10: sistema operativo instalado en el PC.

-Eclipse Mars 4.5: entorno de desarrollo integrado utilizado para las tareas de programación en Java y depuración de la aplicación Comercio Almería. En principio se pensó utilizar Android Studio dado

que es el IDE oficial de Android pero no ofrecía un buen rendimiento en cuanto a requisitos de memoria RAM, así que se determinó que era mejor usar Eclipse. Para el trabajo con proyectos Android ha sido necesario la instalación del Android SDK y el plugin ADT, así como las APIS correspondientes a la versión de Android para la que está destinada la aplicación.

-JDK 7 (Java Development Kit): es un superconjunto de un JRE (Java Runtime Environment) y contiene herramientas para programadores de Java, por ejemplo, un compilador de javac. El kit de desarrollo de Java se proporciona de forma gratuita, ya sea directamente por Oracle Corporation, o por el proyecto de código abierto OpenJDK, que se rige por Oracle. Para trabajar con Comercio Almería se instaló la versión 7 del JDK.

-Android SDK (Software Development Kit): plugin para Eclipse que incluye un conjunto de herramientas de desarrollo que comprenden un depurador de código, biblioteca, un simulador de teléfono basado en QEMU, documentación, ejemplos de código y tutoriales. Las plataformas de desarrollo soportadas incluyen GNU/Linux, Mac OS X 10.5.8 o posterior, y Windows XP o posterior.

-ADT (Android Development Tools): es un plugin para el IDE Eclipse que está diseñado para hacer de éste un entorno potente e integrado en el que construir aplicaciones de Android. Amplía las capacidades de Eclipse para que pueda configurar rápidamente nuevos proyectos Android, crear una interfaz de usuario de la aplicación, agregar paquetes basados en la API del marco de Android, depurar sus aplicaciones utilizando las herramientas del SDK de Android, e incluso exportar firmados (o sin firmar) los archivos .apk con el fin de distribuir una aplicación.

-ADB (Android Debug Bridge): es un conector o puente para depuración de Android, un juego de herramientas incluido en el paquete SDK de Android. Consiste en programas con función tanto de cliente, como de servidor, que se comunican entre ellos. El uso normal del ADB se realiza desde la línea de comandos , aunque existen numerosos interfaces gráficos para controlarlo.

-Sublime Text: Sublime Text es un editor de texto y editor de código fuente que está escrito en C++ y Python. Fue desarrollado originalmente como una extensión de Vim, pero con el tiempo fue creando una identidad propia. Se puede descargar y evaluar de forma gratuita, sin embargo no es software libre o de código abierto y se debe obtener una licencia para su uso continuado, aunque la versión de evaluación es plenamente funcional y no tiene fecha de caducidad.

-Adobe Photoshop: Adobe Photoshop es un editor de imágenes desarrollado por Adobe Systems Incorporated. Usado principalmente para el retoque de fotografías y gráficos, su nombre en español significa literalmente "taller de fotos". Es líder mundial del mercado de las aplicaciones de edición de imágenes y domina este sector de tal manera que su nombre es ampliamente empleado como sinónimo para la edición de imágenes en general. Fue utilizado para cambiar la resolución de imágenes, generación de transparencias en archivos .png, retoque de iconos…

-MySQLWorkbench 6.3 CE: MySQL Workbench es una herramienta visual de diseño de bases de datos que integra desarrollo de software, administración de bases de datos, diseño de bases de datos, creación y mantenimiento para el sistema de base de datos MySQL. Es el sucesor de DBDesigner 4 de fabFORCE.net, y reemplaza el anterior conjunto de software, MySQL GUI Tools Bundle.
Se utilizó para realizar el diseño de la base de datos que se encuentra en este documento.

-SQLite: gestor de bases de datos muy ligero y potente. Por sus características se utiliza en una gran variedad de aplicaciones, como Skype, Mozilla Firefox, Adobe Photoshop Elements, el navegador web Opera… Es el gestor de bases de datos que usa Android y con el cual se ha trabajado. Es utilizado en la aplicación mayoritariamente para realizar consultas sobre la base de datos local.

-SQLite Manager: plugin para el navegador Mozilla Firefox que trabaja con archivos SQLite y que nos permite la administración y consulta de nuestra base de datos local. Ha sido utilizado para la manipulación de las tablas y registros de los que se nutre la aplicación.

-Hostinger: servicio de hosting gratuito para subir nuestros archivos. En él disponemos de un administrador de archivos en el que se subieron vía FTP todas las imágenes que utiliza la aplicación además de contar con la herramienta PHPMyAdmin para albergar las tablas necesarias de la base de datos en el servidor. Las características principales del servicio gratis de Hostinger son las siguientes:

- 2000 MB de espacio en disco.
- 100 Gb de ancho de banda
- Almacenamiento ilimitado de dominios
- Panel de control basado en cPanel
- Sin publicidad o anuncios
- Auto-Instalador (Joomla, WordPress, etc.)
- Servicios de e-mail (IMAP/POP3/Webmail)
- Soporte de PHP y bases de datos MySQL

5.3. Lenguajes de programación

-Java: es un lenguaje de programación de propósito general, concurrente, orientado a objetos. Fue originalmente desarrollado por James Gosling de Sun Microsystems (la cual fue adquirida por la compañía Oracle). Su sintaxis deriva en gran medida de C y C++, pero tiene menos utilidades de bajo nivel que cualquiera de ellos. Las aplicaciones de Java son generalmente compiladas a bytecode (clase Java) que puede ejecutarse en cualquier máquina virtual Java (JVM) sin importar la arquitectura de la computadora subyacente. Es el lenguaje nativo de Android y el que se ha utilizado en la fase de implementación de la aplicación Comercio Almería.

-PHP: es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Es el lenguaje que se ha utilizado en los archivos que solicitan el acceso a la base de datos y realizan consultas sobre esta para pasarle

posteriormente los resultados a nuestra aplicación. Dichos archivos se encuentran subidos al servidor y se accede a ellos a través de una URL.

-MySQL: es un sistema de gestión de bases de datos relacional desarrollado bajo licencia dual GPL/Licencia comercial por Oracle Corporation y está considerada como la base datos open source más popular del mundo. Dado que hostinger trabaja con bases de datos MySQL, el código incluido en los archivos PHP mencionados anteriormente utiliza una sintaxis MySQL.

5.4. Librerías

-Android-support-v7-appcompat: Esta biblioteca añade soporte al diseño de la barra de acciones de la interfaz de usuario. Esta biblioteca depende de android-support-v4, por lo que se debe incluir ésta última como parte del classpath. Para añadir android-support-v7-appcompat al proyecto, fue necesario incluirla primero como un proyecto Android separado y después añadir la dependencia en nuestro proyecto de la aplicación.

-Android-support-v4: Esta biblioteca está diseñada para utilizarse con Android 1.6 (API nivel 4) y superior. Incluye el mayor conjunto de API en comparación con las otras bibliotecas, incluyendo soporte para componentes de aplicaciones, características de interfaz de usuario, accesibilidad, manejo de datos, conectividad de red y utilidades de programación.

-Google-play-services_lib: biblioteca en la que Google incorpora muchas de sus nuevas APIs para desarrolladores Android, por ejemplo la API de mapas, la de localización, o la de mensajería push. Ha sido utilizada principalmente para trabajar con el mapa de la aplicación.

-Apache-httpcomponents-httpcore: La librería HttpCore es un conjunto de componentes de transporte HTTP de bajo nivel que se pueden utilizar para crear servicios HTTP personalizados del cliente y del servidor. Se utiliza para las peticiones HTTP al servidor.

-Apache-httpcomponents-httpclient: Esta librería es una implementación del agente HTTP, compatible con HTTP/ 1.1 basada en HttpCore. También proporciona componentes reutilizables para la autenticación del cliente, la administración de estado HTTP y la administración de la conexión HTTP. Se utiliza para las peticiones HTTP al servidor.

-Picasso-2.5.2: Esta librería se utiliza para realizar la carga de imágenes que se encuentran alojadas en el servidor. Permite el cacheo de imágenes, redimensionamiento y mostrar imágenes de carga y error, entre otros, y además su utilización es muy sencilla.

-YouTubeAndroidPlayerApi: La API del reproductor de YouTube para Android permite incorporar la funcionalidad de reproducción de video en las aplicaciones de Android. Define métodos para cargar y reproducir videos de YouTube y para personalizar y controlar la experiencia de reproducción de video. Al usar la API, se pueden cargar videos en una vista de reproductor insertada en la interfaz de

usuario de la aplicación. Dispone de la función auxiliar de admitir los cambios de orientación, así como las transiciones y la reproducción en pantalla completa. Se utiliza en el proyecto para cargar los videos de que constan algunas empresas.

5.5. Permisos

La aplicación requiere acceso a diferentes características del dispositivo. Para conceder dicho acceso el usuario debe aceptar los permisos necesarios durante la instalación de la aplicación (en dispositivos con Android 5.1 o versiones anteriores) o durante su ejecución (en dispositivos con Android 6.0 o versiones posteriores).

En la captura siguiente se muestran los permisos declarados en el AndroidManifest.xml para la aplicación:

```
<!-- Permissions -->
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.CALL_PHONE" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.VIBRATE"/>
```

Figura 3. Captura de los permisos declarados en AndroidManifest.xml

-ACCESS_NETWORK_STATE: permite verificar el estado de la conexión para determinar si pueden descargarse datos.

-INTERNET: se usa para conceder acceso a Internet a la aplicación para descargar mosaicos de mapas de los servidores de Google Maps, acceder a la base de datos remota, imágenes y otros.

-WRITE_EXTERNAL_STORAGE: Permite que la aplicación escriba en un almacenamiento externo.

-CALL_PHONE: Permite que la aplicación inicie una llamada telefónica sin pasar por la interfaz de usuario Dialer.

-ACCESS_FINE_LOCATION: Permite determinar la ubicación de la manera más precisa posible a través de los proveedores de ubicaciones disponibles, entre los que se incluyen el sistema de posicionamiento global (GPS) y los datos de Wi-Fi y los datos móviles telefonía celular.

-VIBRATE: Concede acceso a la vibración del dispositivo.

En la siguiente imagen se muestra la pantalla de instalación de la aplicación con todos los permisos solicitados al usuario para instalarla.

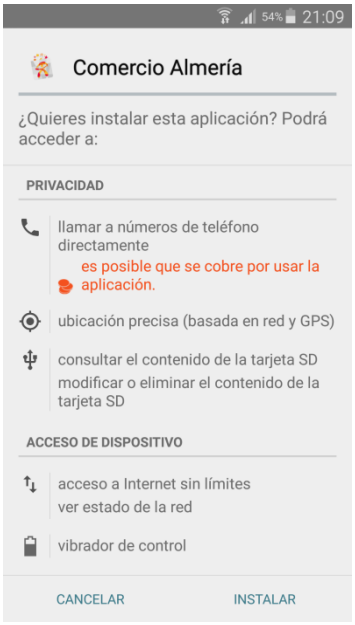

Figura 4. Captura de la pantalla de instalación de Comercio Almería

También es necesario especificar el requisito de OpenGL ES 2. La API de Google Maps usa OpenGL ES 2 para representar el mapa. Si no está instalado OpenGL ES 2, el mapa no aparecerá. Para especificar este requisito, hay que escribir lo siguiente en el manifest:

```
<uses-feature
    android:glEsVersion="0x00020000"
    android:required="true" />
```

Figura 5. Captura de especificación del requisito OpenGL ES 2

Esto notifica el requisito a los servicios externos. En particular, tiene el efecto de evitar que la aplicación se muestre a través de la Google Play Store en dispositivos que no sean compatibles con OpenGL ES 2.

5.6. Claves

Para el uso de las APIS de Google, es necesario disponer de una clave de API. Las claves de API identifican los proyectos para comprobar las cuotas y el acceso.

Figura 6. Esquema de uso de APIS por medio de claves

Las claves de la Android API están vinculadas a pares de paquetes y certificados específicos. Solo se necesita una clave para cada certificado, independientemente de la cantidad de usuarios que usen la aplicación. En nuestro caso, necesitamos dos claves: una para utilizar la API de mapas, y otra para la API de Youtube.

Para obtener las claves para la aplicación, se debieron completar varios pasos. Estos pasos se indican a continuación:

1. Obtener información sobre el certificado de la aplicación, conocido como huella digital SHA-1.
2. Registrar un proyecto en Google Developers Console y agregar la API correspondiente como servicio para el proyecto.
3. Crear la clave especificando el nombre del paquete de la aplicación y la clave SHA1.
4. Agregar la clave a la aplicación declarándola en el manifest.

A continuación, se muestra una imagen a modo de ejemplo sobre como declarar una API en el manifest:

```
<!-- Google MAP API key -->
<meta-data
    android:name="com.google.android.maps.v2.API_KEY"
    android:value="CLAVE_API" />
<meta-data
    android:name="com.google.android.gms.version"
    android:value="@integer/google_play_services_version" />
```

Figura 7. Captura de ejemplo de adición de la calve API a AndroidManifest.xml

5.7. Requerimientos técnicos

Para el correcto funcionamiento de la aplicación, el dispositivo en el que sea instalada debe disponer de las siguientes características técnicas:

-Sistema operativo Android.
-Versión 4.0 (Ice Cream Sandwich)
-Espacio de aproximadamente 22 MB de memoria interna.
-Disponer de tarjeta SD.
-Localizador GPS.

-Conexión a Internet para disfrutar de todas las funcionalidades.
-Pantalla de entre 2” y 10”.
-Tener instalado Google Play Services.
-Tener instalado OpenGL ES 2

5.8. Lista de requisitos funcionales

En la siguiente tabla se indican el conjunto de funciones que han sido implementadas en este proyecto. Para cada una de ellas se indica el nombre y la descripción. Para obtener más información de cada funcionalidad, en el Anexo A se encuentran descritas más detalladamente cada una de ellas.

Nombre	Descripción
Actualizar negocios	El sistema actualizará los negocios de la base de datos local con los negocios de la base de datos del servidor.
Comprobar tablas	El sistema comprobará las tablas de la base de datos local para eliminar algún registro correspondiente a negocios eliminados del servidor
Cargar mapa	El sistema cargará los negocios de la base de datos local y los situará en el mapa
Mostrar contador de negocios	El sistema mostrará un contador con los negocios visibles en el mapa
Actualizar contador de negocios	El sistema mostrará un contador con los negocios visibles en el mapa
Generar lista de negocios	El sistema mostrará los negocios del mapa a través de una lista
Cargar negocios por categoría	El sistema mostrará los negocios asociados a la categoría seleccionada
Buscar negocio por nombre	El sistema mostrará el negocio buscado en el campo de búsqueda
Mostrar negocio de la lista de sugerencias	El sistema mostrará el negocio pulsado de la lista de sugerencias
Mostrar información del negocio	El sistema mostrará una ventana emergente con el nombre y dirección del negocio clicado
Ver detalles del negocio del mapa	El sistema mostrará la información detallada del negocio
Ver detalles del negocio de la lista	El sistema mostrará la información detallada del negocio
Ver detalles del negocio del historial	El sistema mostrará la información detallada del negocio
Ver detalles del negocio de favoritos	El sistema mostrará la información detallada del negocio
Ver detalles del negocio del menú de favoritos	El sistema mostrará la información detallada del negocio

Añadir negocio a favoritos	El sistema añadirá a favoritos el negocio seleccionado
Borrar favorito	El sistema borrará de favoritos el negocio seleccionado
Borrar favorito del menú de favoritos	El sistema borrará de favoritos el negocio seleccionado
Ver favoritos	El sistema mostrará los negocios de la sección de favoritos
Ver historial	El sistema mostrará los negocios de la sección de historial
Añadir negocio al historial	El sistema guardará un historial de negocios visualizados
Trazar ruta hasta el negocio	El sistema mostrará una ruta en el mapa de detalle del negocio, para llegar desde la ubicación del usuario hasta el negocio
Compartir ubicación del negocio	El sistema enviará un mensaje con la ubicación del negocio
Notificar negocio	El sistema añadirá el negocio a notificaciones
Dejar de notificar negocio	El sistema eliminará el negocio de notificaciones
Ver galería del negocio	El sistema mostrará la galería de imágenes del negocio
Ver galería ampliada del negocio	El sistema mostrará la galería de imágenes ampliada del negocio
Mostrar versión web de la aplicación	El sistema mostrará la versión web de la aplicación
Mostrar noticias	El sistema cargará las noticias presentes en la versión web
Mostrar mercados/mercadillos	El sistema mostrará información e imágenes de los mercados y mercadillos
Ver galería ampliada de mercados y mercadillos	El sistema mostrará la galería de imágenes ampliada de los mercados/mercadillos
Mostrar productos locales	El sistema mostrará los productos locales y empresas que los trabajan
Ver mapa de productos	El sistema mostrará una imagen ampliada del mapa
Ver detalles de empresa	El sistema mostrará los detalles de la empresa seleccionada
Contactar con negocio	El sistema ejecutará el servicio de contacto indicado
Contactar con empresa	El sistema ejecutará el servicio de contacto indicado
Ver galería ampliada de empresa	El sistema mostrará la galería de imágenes ampliada de una empresa
Ver video de empresa	El sistema reproducirá un video de la empresa
Mostrar ayuda al comerciante	El sistema mostrará las secciones de Ayuda al comerciante disponibles en la web

Ver información sobre la aplicación	El sistema mostrará informativa relativa al uso de la aplicación
Cambiar tipo de mapa	El sistema mostrará el mapa del tipo seleccionado
Habilitar/Deshabilitar guardar historial	El sistema almacenará/no almacenará los negocios vistos en la pantalla de detalles del negocio en la tabla historial
Limpiar historial	El sistema eliminará los registros de la tabla historial
Elegir radio de alertas	El sistema cambiará el radio de proximidad de las alertas
Elegir frecuencia de alertas	El sistema cambiará la frecuencia con que se notifican las alertas
Habilitar alertas de favoritos	El sistema enviará/no enviará alertas de proximidad de los negocios favoritos
Seleccionar tono de alerta	El sistema asignará un tono a las alertas de proximidad
Habilitar vibración para alertas	El sistema habilitará / deshabilitará la vibración del dispositivo cuando se notifique la alerta
Habilitar LED para alertas	El sistema habilitará / deshabilitará el LED en las notificaciones
Recomendar aplicación	El sistema enviará un mensaje al contacto elegido publicitando la aplicación
Puntuar aplicación	El sistema mostrará el store de la aplicación para que pueda ser puntuada
Ver versión de la aplicación	El sistema mostrará la versión de la aplicación
Notificar alerta de proximidad	El sistema mostrará una notificación de proximidad a un negocio de interés

Tabla 2. Lista de requisitos funcionales

5.9. Lista de requisitos no funcionales

En la siguiente tabla se indican el conjunto de criterios a seguir para la implementación de este proyecto. Para cada uno de ellos se indica el nombre y la descripción. Para obtener más información de cada criterio, en el Anexo A se encuentran descritos más detalladamente cada uno de ellos.

Nombre	Descripción
Tecnología de la aplicación	La aplicación se desarrollará en Android
Funcionalidades de la aplicación	El sistema implementará las funcionalidades presentes en la versión y se le incluirán otras nuevas.

Estilo de la aplicación	El estilo de la aplicación irá en consonancia con el estilo de la versión web

Tabla 3. Lista de requisitos no funcionales

Capítulo 6:
Aplicación Comercio Almería

6. APLICACIÓN COMERCIO ALMERÍA

6.1. Arquitectura

Nuestra aplicación sigue una arquitectura cliente/servidor, es decir, el cliente realiza peticiones a otro programa, el servidor, quien le da respuesta. Observando el siguiente esquema podemos entender cómo se cargan los negocios que se muestran en el Mapa:

Figura 8. Esquema de la arquitectura de la aplicación

-Smartphone: realiza una petición al servidor para cargar el fichero *conexionbd.php* a través de una URL.
-Servidor: El fichero *conexionbd.php* ejecuta una consulta sobre la base de datos para devolver a la aplicación los registros de la tabla Negocios.
-Base de datos: Almacena la tabla Negocios que será consultada.

6.2. Estructura del proyecto

Nuestro proyecto tiene la estructura predeterminada de ficheros y directorios de un proyecto Android creado con Eclipse:

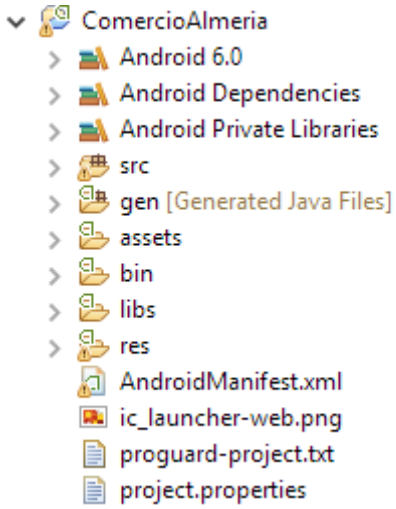

Figura 9. Estructura general del proyecto de Eclipse

-Carpeta /src: En esta carpeta se almacena todo el código fuente de las clases y los archivos java. La estructuración del código en esta carpeta se divide en paquetes que almacenan las clases .java según su contenido. Los paquetes que se incluyen en esta carpeta son los siguientes:

- com.example.comercioalmeria.activities: en este paquete se guardan las activities.
- com.example.comercioalmeria.adapterel paquete adapter almacena los adaptadores que se utilizan en componentes, como listas.
- com.example.comercioalmeria.bd: contiene el fichero de acceso a la base de datos.
- com.example.comercioalmeria.fragments: aquí se guardan las clases .java definidas para los fragments.
- com.example.comercioalmeria.model: almacena las clases que son modelo de objetos, como la clase Negocio.java, Empresa.java o ItemGaleria.java.
- com.example.comercioalmeria.service: en este paquete se guardan las clases relacionadas con nuestro servicio de alertas o detección de GPS activado.
- com.example.comercioalmeria.task: en task se encuentran las clases que envían solicitudes a través de Internet mediante la ejecución de una tarea asíncrona o AsyncTask.
- com.example.comercioalmeria.util: este paquete hace referencia a utilidades y las clases .java que contiene son accedidas a menudo en cualquiera de las otras clases. Algunos ejemplos de ello son las clases Utilidades.java, JSONParser.java o XMLParser.java.

En la siguiente imagen podemos ver la estructura descrita:

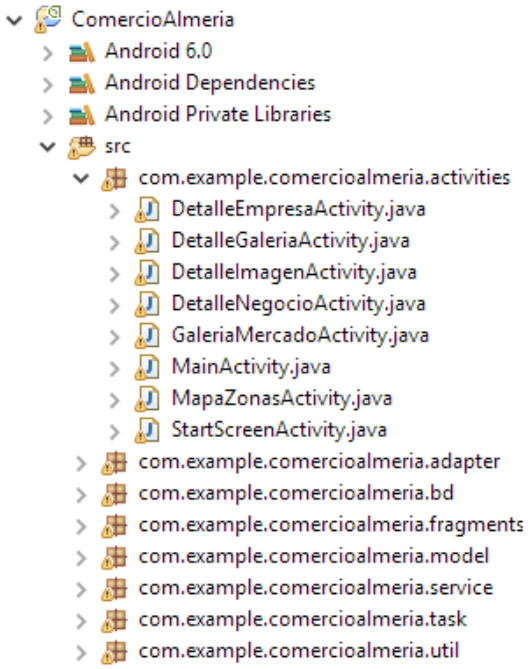

Figura 10. Estructura de la carpeta /src del proyecto de Eclipse

-Carpeta /gen: En esta carpeta Eclipse genera de forma automática recursos para el proyecto. Es recomendable no modificarla.

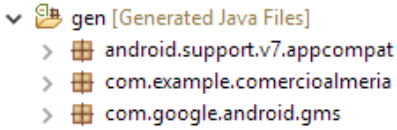

Figura 11. Contenido de la carpeta /gen del proyecto de Eclipse

-Carpeta /assets: Esta carpeta se usa para almacenar recursos utilizados en la aplicación.

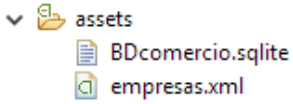

Figura 12. Contenido de la carpeta /assets del proyecto de Eclipse

-Carpeta /res: En esta carpeta se encuentra el directorio general de recursos utilizados en la aplicación, como imágenes, iconos, estilos y layouts. Los subdirectorios más destacados de esta carpeta son los siguientes:

-Carpeta /drawable-(hdpi/ldpi/mdpi/xhdpi/xxhdpi): Estas carpetas son los directorios de recursos gráficos que almacenan las imágenes organizadas por densidad.

-Carpeta /layout: La carpeta layout almacena ficheros xml de la aplicación, estos ficheros son los que definen cada sección de interfaz de usuario.

-Carpeta /values: Dentro de la carpeta values podemos definir entre otros elementos las cadenas de texto o los colores separándolos del resto del código de la aplicación mediante archivos xml (strings.xml, colors.xml …).

La imagen siguiente muestra la estructura mencionada:

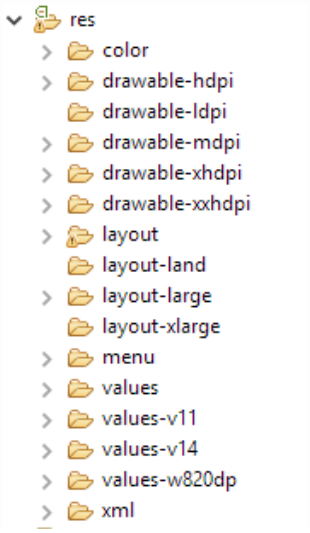

Figura 13. Estructura de la carpeta /res del proyecto de Eclipse

-/AndroidManifest.xml: Este fichero define los permisos de nuestra aplicación, el nombre y otros elementos como las actividades y los permisos especiales para acceder a los recursos del sistema.

6.3. Diseño.

El diseño de nuestra app se ha llevado a cabo teniendo en cuenta el que existe en la página web, ya que se trata de una herramienta complementaria. Se han incluido en las opciones del menú principal las opciones del menú de la cabecera que aparecen en la web, se han implementado funciones propias de la web como mostrar una lista de los negocios que son visibles actualmente en el mapa y se han incluido otros apartados novedosos.

La navegación a través de la app se basa en el uso de activities y fragments. Un activity (actividad) es el conjunto de acciones que forman parte de una interacción directa con el usuario y que afectan a una parte de la aplicación. Está conformada una parte lógica y una parte gráfica:

-La parte lógica es una archivo .java, que es la clase que se crea para poder manipular, interactuar y colocar el código de esa actividad.
-La parte gráfica es un XML que tiene todos los elementos que estamos viendo de una pantalla declarados con etiquetas.
Una actividad se caracteriza por tener un ciclo de vida, es decir consta de una serie estados y transiciones, que abarcan desde que es creada hasta que es destruida.

El siguiente diagrama muestra los diferentes estados de una Actividad.

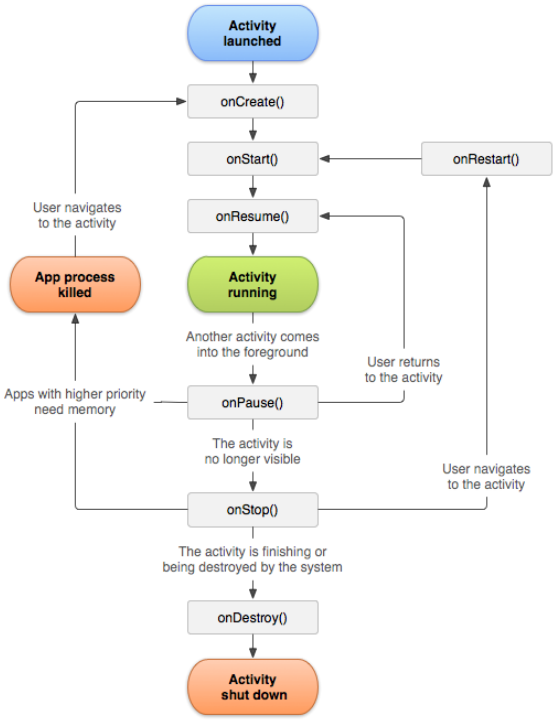

Figura 14. Ciclo de vida de un Activity

Por otro lado, los fragments representan un comportamiento o una parte de la interfaz de usuario en una activity. Un *fragment* podría definirse como una porción de la interfaz de usuario que puede añadirse o eliminarse de forma independiente al resto de elementos de la actividad, y también puede ser reutilizado en otras actividades.

Los fragments también poseen un ciclo de vida, que se ve directamente afectado por el ciclo de vida de la actividad anfitriona. La siguiente captura muestra el ciclo de vida de un fragmento mientras su actividad se encuentra en ejecución:

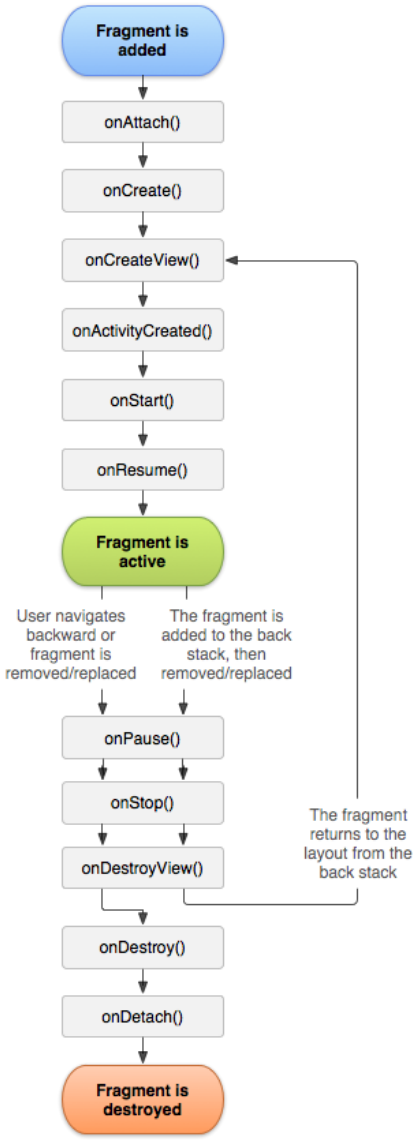

Figura 15. Ciclo de vida de un Fragment

El uso de fragments en nuestra aplicación nos proporciona una serie de ventajas:

-Posibilita la modificación de la apariencia de una Activity en tiempo de ejecución y preserva los cambios en la pila de procesos de la Activity.
-Proporciona diseños más dinámicos y flexibles, facilitando la tarea de adaptación a pantallas de distintas dimensiones.
-Cada fragment es independiente del resto, y por lo tanto reutilizable.

En nuestro proyecto, disponemos de un MainActivity.java con el menú lateral principal que será el esqueleto de la aplicación. Al seleccionar una opción de este menú, cargaremos el fragment correspondiente, siendo "Mapa" la opción por defecto y mostrada al inicio.

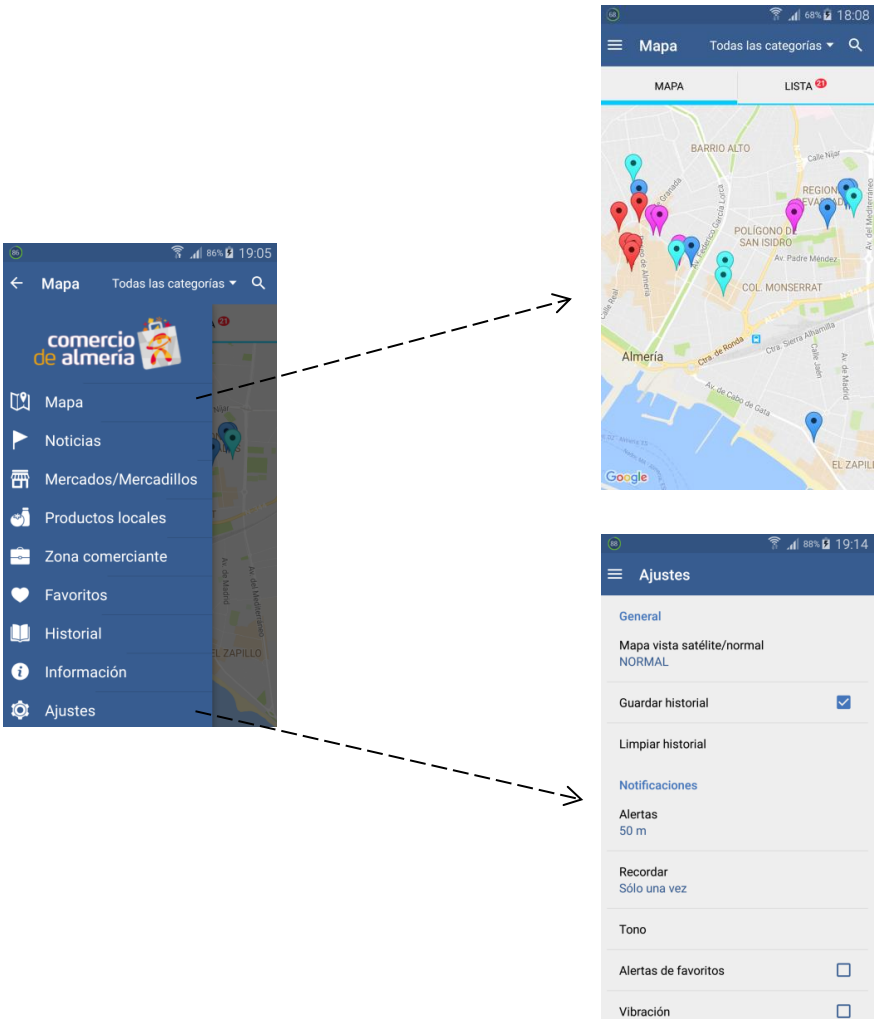

Figura 16. Funcionamiento del menú lateral para cargar los fragments de cada sección

Para ver en detalle los elementos incluidos en algunos de los fragments se volverá a lanzar otro Activity. Por ejemplo, si hacemos click sobre uno de los markers del mapa lanzaremos la Activity DetalleNegocioActivity.java con los detalles sobre el negocio seleccionado:

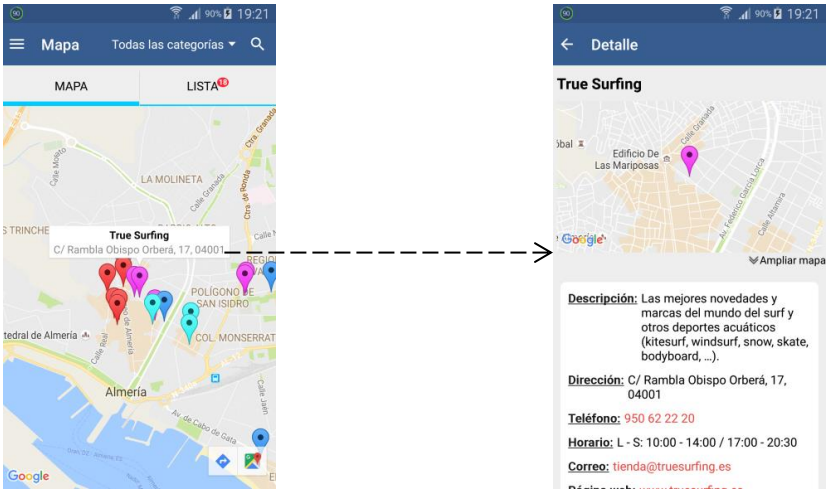

Figura 17. Visualización de detalles del fragment en un nuevo Activity

Es decir, la aplicación sigue la siguiente lógica:

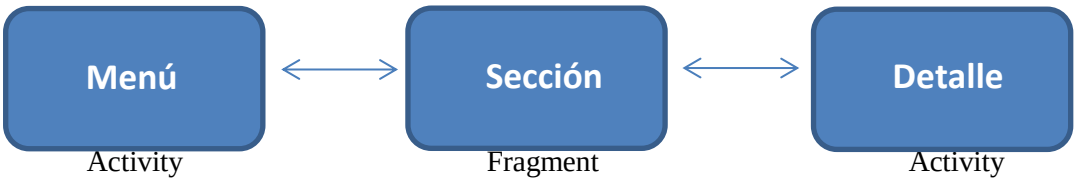

Figura 18. Lógica de navegación entre los fragments y activities de la aplicación

6.3.1. Componentes.

Nuestra aplicación dispone de multitud de componentes de distinto tipo, como layouts, widgets, vistas…

Por ejemplo, la interfaz principal incorpora lo siguiente:

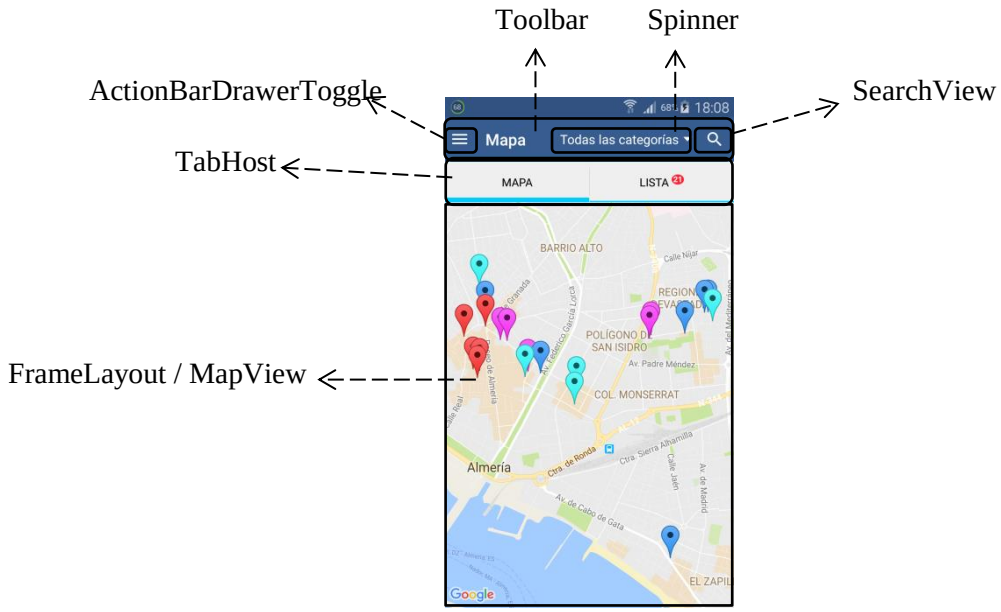

Figura 19. Muestra de algunos de los componentes de la aplicación

A continuación se detallan algunos de los más destacados en cuanto a la funcionalidad que ofrecen a la aplicación:

-Toolbar: widget que reemplaza al ActionBar. Utiliza la biblioteca de compatibilidad v7 appcompat.

Figura 20. Cabecera o Toolbar de la aplicación

-ActionBarDrawerToggle: Esta clase proporciona una manera práctica de unir la funcionalidad de DrawerLayout y el Toolbar para implementar el diseño recomendado para cajones de navegación.

Figura 21. Botón ActionBarDrawerToggle para mostrar/cerrar el menú principal

-DrawerLayout: layout sobre el que se implementa el menú desplegabe compuesto por un ListView.

Figura 22. Menú principal de la aplicación implementado en un DrawerLayout

-Spinner: widget que se compone de una lista desplegable hacia abajo.

Figura 23. Spinner para la búsqueda de negocios por categorías

-SearchView: widget sobre el que se realizan operaciones de búsqueda mediante la inserción de texto a través de teclado.

Figura 24. SearchView para la búsqueda de negocios por nombre

-TabHost: Contenedor para una vista de ventana con pestañas. Este objeto contiene dos hijos: un conjunto de etiquetas de pestañas que el usuario hace clic para seleccionar una pestaña específica y un objeto FrameLayout que muestra el contenido de esa página.

Figura 25. TabHost para intercambiar las vistas de "Mapa" y "Lista"

-FragmentTabHost: TabHost especial que permite el uso de objetos Fragment para el contenido de cada pestaña.

Figura 26. FragmentTabHost para almacenar contenido de tipo fragment

-FrameLayout: tipo de layout diseñado para bloquear un área en la pantalla para mostrar un solo elemento. Es el layout sobre el que se carga el contenido de los fragments.

-MapView: Una vista que extiende de FrameLayout y que muestra un mapa (con datos obtenidos del servicio de Google Maps). Cuando está enfocado, capturará pulsaciones de teclas y gestos táctiles para mover el mapa. Es el elemento idóneo y recomendado para la representación de mapas dentro de fragments.

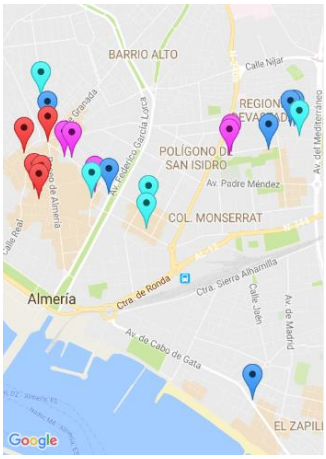

Figura 27. MapView para mostrar el mapa dentro de un fragment

-ListView: Una vista que muestra elementos en una lista de desplazamiento vertical. Los elementos provienen del ListAdapter asociado con esta vista.

Figura 28. ListView para mostrar listas de elementos

-TextView: Muestra texto al usuario y está configurada para no permitir la edición. Admite eventos como el onClick lo que le confiere una gran versatilidad pudiendo funcionar a modo de botón.

Figura 29. Menú de TextViews funcionando a modo de botón

-EditText: es una capa sobre TextView que se configura para ser editable.

Figura 30. EditText para la inserción de texto a través del teclado

-Button: Representa un widget pulsador. El usuario puede presionar o hacer clic en los botones pulsadores para realizar una acción.

Figura 31. Button para la ejecución de acciones

-CheckBox: Una casilla de verificación es un tipo específico de botón de dos estados que puede estar marcado o desmarcado. Un ejemplo de uso de una casilla de verificación sería el siguiente:

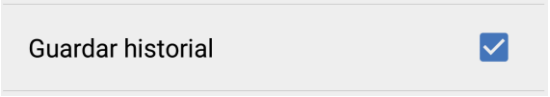

Figura 32. CheckBox para habilitar/deshabilitar funciones

-ImageView: La clase ImageView puede cargar imágenes de varias fuentes (como recursos o proveedores de contenido), se encarga de calcular su medición de la imagen para que pueda ser utilizada en cualquier gestor de diseño y ofrece varias opciones de visualización, como la ampliación y el tintado.

Figura 33. ImageView para mostrar imágenes

-WebView: Una vista que muestra páginas web. Esta clase es la base sobre la cual se puede mostrar algún contenido en línea dentro de tu Actividad. Utiliza el motor de renderizado WebKit para mostrar páginas web e incluye métodos para navegar hacia delante y hacia atrás a través de un historial, acercar y alejar, realizar búsquedas de texto y mucho más.

Figura 34. WebView para mostrar contenido web dentro de la aplicación

-Gallery: Una vista que muestra elementos en una lista de desplazamiento horizontal con bloqueo central.

Figura 35. Gallery para mostrar galerías de imágenes en miniatura

-ViewPager: Administrador de diseño que permite al usuario desplazarse a izquierda y derecha a través de páginas de datos. Proporciona una implementación de un PagerAdapter para generar las páginas que muestra la vista. ViewPager se utiliza con más frecuencia junto con Fragment, que es una forma conveniente de suministrar y administrar el ciclo de vida de cada página.

Figura 36. ViewPager para mostrar galerías de imágenes

-GridView: Una vista que muestra elementos en una cuadrícula de desplazamiento bidimensional. Los elementos de la cuadrícula provienen del ListAdapter asociado con esta vista.

Figura 37. GridView para la estructuración de elementos a modo de rejilla

-ScrollView: Contenedor de diseño para una jerarquía de vistas que el usuario puede desplazar, permitiendo que sea mayor que la presentación física. Un ScrollView es un FrameLayout, lo que significa que debe colocar un hijo en él que contiene todo el contenido para desplazarse.

-AlertDialog: Una subclase de Diálogo que puede mostrar botones

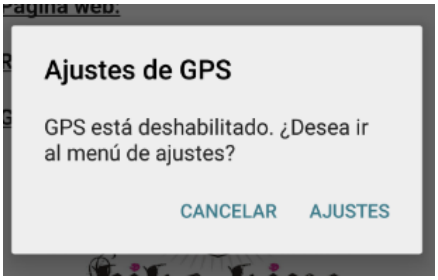

Figura 38. AlertDialog para mostrar cuadros dialógo

-ProgressDialog: Un cuadro de diálogo que muestra un indicador de progreso y un mensaje o una vista de texto opcional.

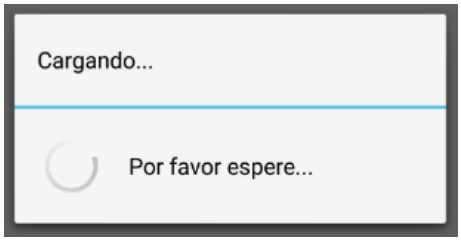

Figura 39. ProgressDialog para mostrar indicadores de progreso

-ProgressBar: una barra de progreso para indicar un proceso de carga.

Figura 40. ProgressBar para mostrar indicadores de progreso

-Toast: es una vista que contiene un pequeño mensaje rápido para el usuario. Cuando la vista se muestra al usuario, aparece como una vista flotante sobre la aplicación. Nunca recibirá el foco.

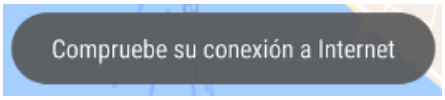

Figura 41. Toast para la notificación de mensajes al usuario

-CustomToast: Toast personalizado que agrega un recurso drawable (imagen .png) y un fondo rojo para representar algún error o advertencia.

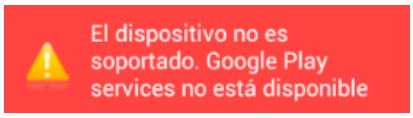

Figura 42. CustomToast para la notificación de mensajes al usuario

-Notification: Una notificación es un mensaje que puede mostrarse al usuario fuera de la interfaz de la aplicación.

Figura 43. Captura de una notificación

-PopupMenu: muestra un Menú en una ventana emergente anclada a una vista. La ventana emergente aparecerá debajo de la vista de anclaje si hay espacio, o sobre ella si no lo hay. Tocar fuera de la ventana emergente lo descartará.

Figura 44. PopupMenu para mostrar menús en ventanas emergentes

6.3.2. Estilo

El estilo o theme definido en *res/values/styles.xml* que utiliza nuestra app es el de *Theme.AppCompat.Light.NoActionBar*. Este estilo pertenece a la librería *android-support-v7-appcompat* y es el que le da el estilo al propio Toolbar.

```
<style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
    <item name="android:textColorPrimary">@color/white</item>
    <item name="android:textColorSecondary">@color/white</item>
    <item name="android:actionMenuTextColor">@color/white</item>
    <item name="android:textColor">@color/black</item>
    <item name="android:textColorHint">@color/color_hint</item>
    <item name="colorPrimary">@color/dark_blue</item>
    <item name="colorPrimaryDark">@color/status_bar</item>
    <item name="android:windowNoTitle">true</item>
    <item name="drawerArrowStyle">@style/DrawerArrowStyle</item>
</style>
```

Figura 45. Declaración del theme principal en el fichero styles.xml

Dentro de este theme se han declarado los colores de la barra, textos y menús del Toolbar, que se encuentran declarados en el fichero *res/values/color.xml*:

-Color del título (*android:textColorPrimary*): #FFFFFF (Blanco)
-Color de los menús de acción *(android:textColorSecondary*): #FFFFFF (Blanco)
-Color de los menús de acción por debajo de API 21 *(android:actionMenuTextColor*): #FFFFFF (Blanco)
-Color del texto *(android:textColor*): #000000 (Negro)
-Color del texto de indicación (*android:textColorHint*): #CCCCCC (Blanco grisáceo)
-Color de la barra (*colorPrimary*): #375C90 (Azul)
-Color de la barra de estado (*colorPrimaryDark*): #325381 (Azul oscuro)
-Color de la flecha del drawer (*drawerArrowStyle - color*): #FFFFFF (Blanco)

Algunos activities o fragments como por ejemplo AjustesFragment.java, tienen su propio estilo que hereda del theme principal y se ha definido aparte:

```
<style name="AjustesFragmentTheme" parent="AppTheme">
    <item name="colorAccent">@color/blue</item>
    <item name="android:textColorPrimary">@color/blue</item>
    <item name="android:textColorSecondary">@color/dark_blue</item>
    <item name="android:textColor">@color/black</item>
    <item name="android:scrollbarThumbVertical">@drawable/scrollbar_style</item>
    <item name="android:textSize">16sp</item>
    <item name="android:alertDialogTheme">@style/MyAlertDialogStyle</item>
</style>
```

Figura 46. Declaración de un theme particular en el fichero styles.xml

-Color de imagen, botón y widget (*colorAccent*) : #4475BA (Azul)
-Color de categoría de preferencia (*android:textColorPrimary*): #4475BA (Azul)
-Color de opción de preferencia seleccionada (*android:textColorSecondary*): #325381 (Azul oscuro)
-Color del texto (*android:textColor*): #000000 (Negro)
-Estilo del scrollbar vertical (*android:scrollbarThumbVertical*): #5F8EC0 (Azul claro)
-Tamaño del texto (*android:textSize*): 16 sp

Los diálogos también tienen su propio estilo que heredan de *Theme.AppCompat.Light.Dialog.Alert*, como podemos ver a continuación:

```
<style name="MyAlertDialogStyle" parent="Theme.AppCompat.Light.Dialog.Alert">
    <!-- Used for the buttons -->
    <item name="android:windowTitleStyle">@style/MyTitleTextStyle</item>
    <item name="android:textColorPrimary">@color/black</item>
</style>

<style name="MyTitleTextStyle">
    <item name="android:textColor">@color/black</item>
    <item name="android:textSize">20sp</item>
    <item name="android:textAppearance">@style/TextAppearance.AppCompat.Title</item>
</style>
```

Figura 47. Declaración de un estilo para cuadros de diálogo

Estos estilos son de carácter general pero también puede haber elementos que incorporen su propio estilo de forma individual.

-Icono launcher:

El icono elegido para la aplicación ha sido extraído de la página web y cómo podemos observar es bastante representativo para la temática que nos ocupa:

Figura 48. Icono launcher de la aplicación

6.3.3. Colores

Los colores utilizados en la aplicación son similares a los de la página web, siendo el color predominante el azul en distintas tonalidades. Aunque dependiendo del tipo de vista o elemento, se han podido aplicar otros colores diferentes, como por ejemplo:

-TextViews descriptivos: #000000 (Negro)
-TextViews con enlaces: #E74741 (Bermellón)
-TextViews de títulos: #375C90 (Azúl oscuro)
-Fondo: #EEEEEE (Blanco grisáceo)
-Layout contenedor: #FFFFFF (Blanco)
-Layout contenedor presionado: #41D8E9 (Cián)

En la siguiente tabla podemos ver todo el conjunto de colores utilizados en la aplicación:

Nombre	**Hexadecimal**	**Color**
azure	*#007FFF*	
blue	*#4475BA*	
dark_blue	*#375C90*	
status_bar	*#325381*	
scrollbar_color	*#5F8EC0*	
white	*#FFFFFF*	
color_hint	*#CCCCCC*	
black	*#000000*	
dark	*#1A1A1A*	

grey	*#8B8B8B*	
link	*#E74741*	
cyan	*#41D8E9*	
magenta	*#FF00FF*	
default_background	*#EEEEEE*	
yellow	*#E4AD42*	
orange	*#F06E1D*	
red	*#ED0800*	

Tabla 4. Paleta de colores utilizada en la aplicación

6.3.4. Fuentes

Todos los elementos textuales de la aplicación utilizan la fuente predeterminada de Android, es decir, "sans". Esta fuente se encuentra presente en nuestra aplicación en diferentes tamaños y variantes dependiendo del elemento del que se trate, como podemos ver en la siguiente tabla:

Elemento	Variante	Subrayado	Tamaño (en sp)
Título	Bold	No	20
Etiqueta	Bold	Sí	18
Descripción	Normal	No	18

Tabla 5. Tipos y tamaños de fuentes utilizadas en la aplicación

6.3.5. Imágenes

Las imágenes de Comercio Almería presentes en ImageViews se pueden encontrar en diferentes formatos y medidas. En la siguiente tabla podemos observar el elemento que representa el ImageView, el contenedor en el que se encuentra, el formato de la imagen, su tamaño y el tipo de escala determinado por el parámetro ("android:scaleType"):

Elementos	Contenedor	Formato	Tamaño	Tipo de Escala
Iconos del menú principal	ListView	.jpg	25 x 25 dp	Ninguno
Imagenes de negocios en la lista de Mapa	ListView	.jpg	36 x 36 dp	Ninguno
Items de Favoritos e Historial	GridView	.jpg	120 x 120 dp	centerCrop
Imagenes del fragment Mercados/Mercadillos	LinearLayout	.png	300 x 188 dp	Ninguno
Imagenes del fragment Productos Locales	LinearLayout	.png	300 x 188 dp	Ninguno
Imagenes del fragment Zona Comerciante	LinearLayout	.png	300 x 188 dp	Ninguno
Imagenes del fragment Información	LinearLayout	.png	300 x 100 dp	Ninguno
Items de empresas	GridView	.jpg	125 x 100 dp	fitCenter
Mapas de productos	LinearLayout	.png	350 x 230 dp	Ninguno
Mapas de productos en miniatura	LinearLayout	.png	50 x 50 dp	Ninguno
Galerias de negocios	LinearLayout	.jpg	275 x 125 dp	Ninguno
Galerias de empresas	GridView	.jpg	125 x 100 dp	centerCrop
Galerias de mercados/mercadillos	GridView	.jpg	100 x 100 dp	Ninguno
Imagenes ampliadas	FrameLayout	.jpg, .png	match_parent x match_parent	fitCenter
Galerias ampliadas	ViewPager	.jpg	match_parent x match_parent	Ninguno

Tabla 6. Propiedades de las imágenes utilizadas en la aplicación

6.3.6. Otros elementos de estilo

A lo largo de todo el proceso de creación del proyecto se ha trabajado con otros elementos que confieren un estilo determinado a los componentes de nuestra aplicación. Se han utilizado gradientes, selectores, shapes, animaciones y otro tipo de drawables. Las imágenes siguientes son una muestra representativa del uso de este tipo de elementos:

Figura 49. Ejemplo 1 de aplicación de drawables como recursos de estilo

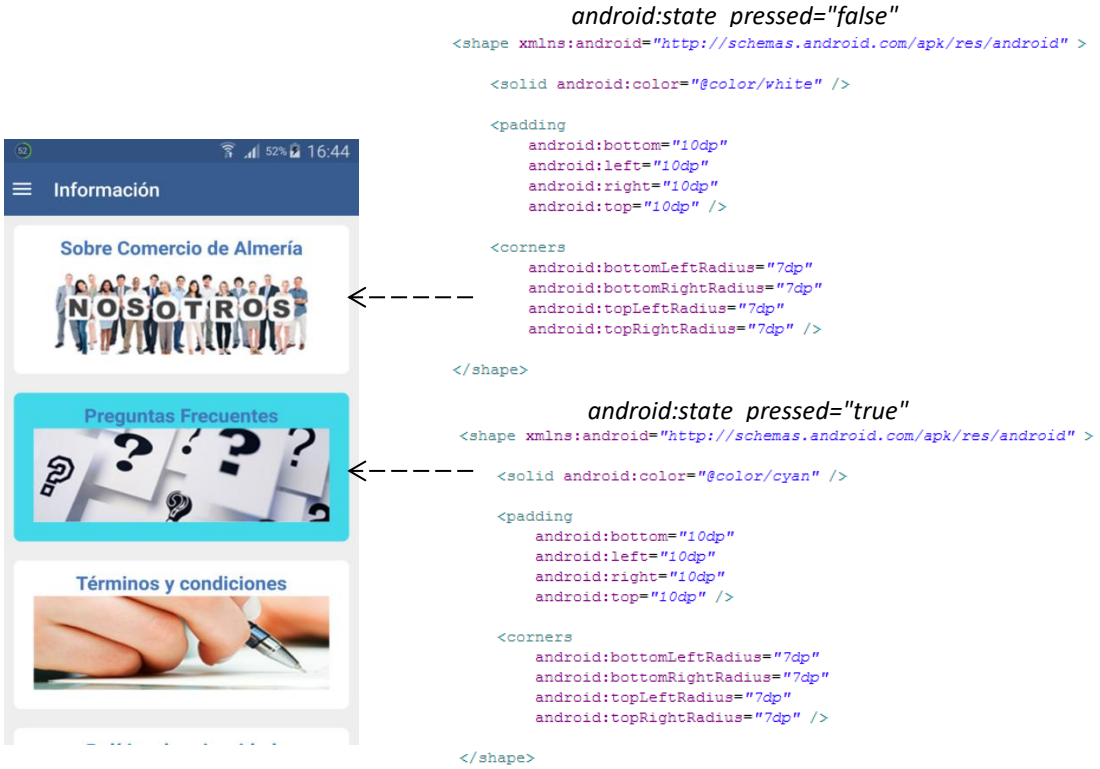

Figura 50. Ejemplo 2 de aplicación de drawables como recursos de estilo

Capítulo 7:
Desarrollo

7. DESARROLLO

El desarrollo de esta aplicación abarca desde el diseño de la base de datos hasta la codificación de cada una de las funcionalidades, que permiten por ejemplo, trazar una ruta entre nuestro destino y un negocio de interés.

7.1. Diseño de la base de datos

El programa se servirá de dos bases de datos, una en el servidor y otra local. La base de datos del servidor está hospedada en Hostinger, un servicio de hosting gratuito en el utilizaremos la herramienta PHPMyAdmin para su administración. Dicha base de datos es de tipo MySQL y consta de la siguiente tabla:

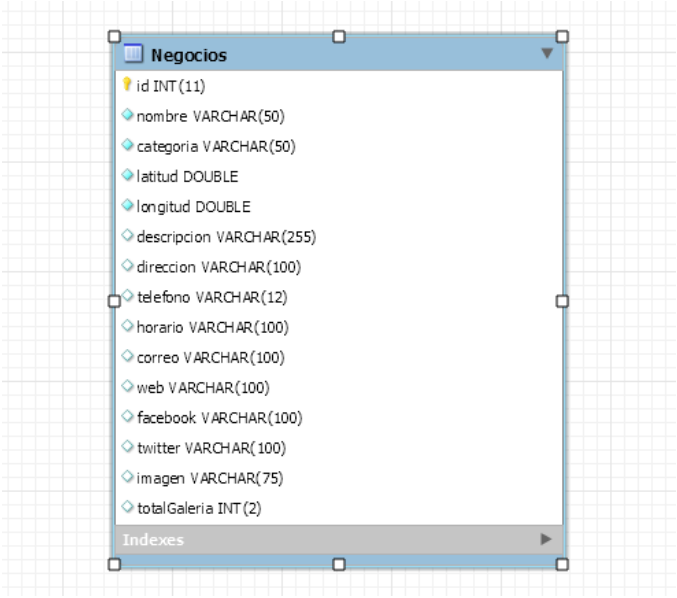

Figura 51. Diseño de la base de datos remota

-id: clave primaria de tipo entero, no nulo.
-nombre: string de tipo VARCHAR con longitud 50 y no nulo para almacenar el nombre del negocio.
-categoria: string de tipo VARCHAR con longitud 50 y no nulo para almacenar la categoría del negocio.
-latitud: double no nulo para almacenar la latitud del negocio.
-longitud: double no nulo para almacenar la longitud del negocio.
-descripcion: string de tipo VARCHAR con longitud 255 para almacenar la descripción del negocio.
-direccion: string de tipo VARCHAR con longitud 100 para almacenar la dirección del negocio.
-telefono: string de tipo VARCHAR con longitud 12 para almacenar el teléfono del negocio.
-horario: string de tipo VARCHAR con longitud 100 para almacenar el horario del negocio.
-correo: string de tipo VARCHAR con longitud 100 para almacenar el correo del negocio.
-web: string de tipo VARCHAR con longitud 100 para almacenar la url de la web del negocio.

-facebook: string de tipo VARCHAR con longitud 100 para almacenar la url del facebook del negocio.
-twitter: string de tipo VARCHAR con longitud 100 para almacenar la url del twitter del negocio.
-imagen: string de tipo VARCHAR con longitud de 75 para almacenar la url de la carpeta con las imágenes del negocio.
-totalGaleria: valor de tipo entero que indica el número de imágenes de la galería del negocio.

Además de utilizar la base de datos remota, la aplicación contará con una base de datos local para no depender de servicios externos en la medida de lo posible y así evitar estar expuesta a errores de conexión o de otro tipo. Esta base de datos local es de tipo SQLite, que es el motor de base de datos que usa Android y consta de las siguientes tablas:

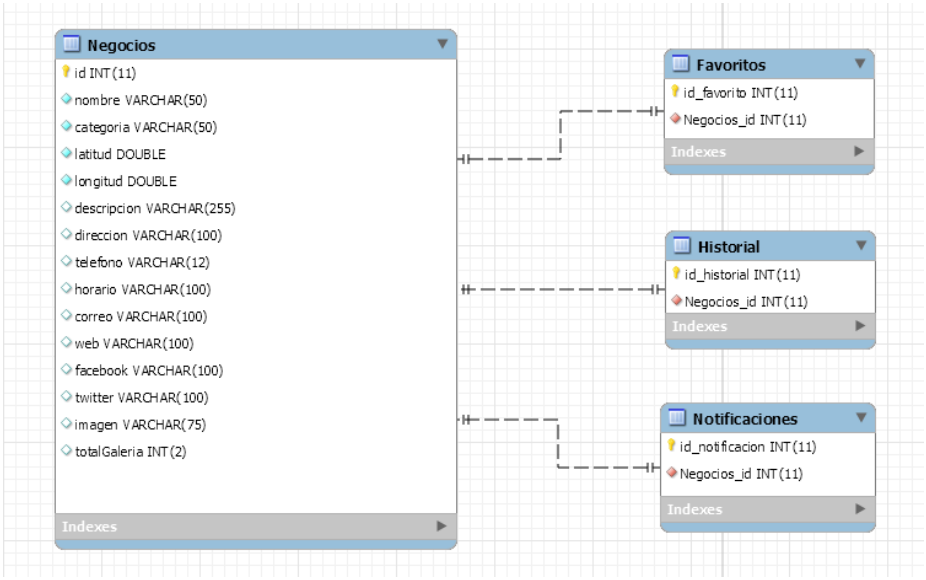

Figura 52. Diseño de la base de datos local

Las tablas Historial, Favoritos y Notificaciones contienen dos campos:

-id_favorito / id_historial / id_notificacion: clave primaria de tipo entero autoincremental, único y no nulo.
-id_negocio: clave foránea de tipo entero, único y no nulo que referencia al id de la tabla Negocios.

La base de datos local almacenará en la tabla Negocios los registros presentes en la tabla del servidor y además tendrá las tablas Historial, Favoritos y Notificaciones con las cuales mantiene una relación de 1:1. Es decir, para cada registro de la tabla Negocios podrá haber un único registro coincidente en las otras tablas.

El hecho de haber definido esta estructura se debe a que las tablas Historial, Favoritos y Notificaciones guardarán un solo un subconjunto de los registros de la tabla principal.

La tabla Historial almacenará los negocios visualizados si la preferencia "Guardar Historial" está seleccionada. Estos negocios se devolverán al usuario al seleccionar la opción Historial del menú principal. El campo id_historial se utilizará para devolver dichos registros por orden descendente, es decir, el usuario verá los negocios visitados más recientemente en primer lugar y viceversa.
La tabla Favoritos guarda los negocios que el usuario haya marcado con la opción "Añadir a favoritos". Estos negocios pueden enviar notificaciones de proximidad si la preferencia "Alertas de favoritos" está seleccionada.
En la tabla Notificaciones se guardan los negocios que el usuario determina presionando la opción "Recordar". Se alertará al usuario si alguno de estos negocios se encuentra dentro de un radio de proximidad determinado mediante una notificación.

Para almacenar datos de ubicación geográfica de un punto del mapa se suele hacer uso de bases de datos geoespaciales, almacenándose éstos en campos de tipo POINT. Existen varios motivos por los que no se ha utilizado una base de datos geográfica ni se han almacenado las coordenadas de los negocios como tipo POINT:

-La funcionalidad de nuestra aplicación no necesitar hacer uso del potencial de las herramientas de estos sistemas y de sus funciones.
-El mapa de Google que usa nuestra aplicación trabaja con coordenadas de tipo LatLng (double latitud, double longitud) para posicionar los marcadores, y por lo tanto en nuestra aplicación era suficiente utilizar dos columnas de una tabla para guardar dicha información.
-La tabla de la base de datos local (SQLite) no ofrece soporte para el almacenamiento del tipo de datos POINT.

7.3. Clases .java

Son muchas las clases java que se incluyen en el proyecto y que representan activities, fragments, adaptadores, servicios y otros. A continuación se describen algunas de las más importantes junto con su código más representativo.

7.3.1. StartScreenActivity.java

La pantalla de inicio de la aplicación muestra al usuario el logotipo de la aplicación junto con un ProgressBar que simula un proceso de carga. A su vez, comprueba si el dispositivo tiene y conexión a Internet y si es así, se ocupa de cargar los negocios presentes en la base de datos remota por medio de una clase que extiende de AsyncTask, que permite realizar tareas asíncronas en segundo plano. Esta clase se denomina CargarNegocios.java y funciona de la siguiente manera:

1. Realiza una petición HTTP por medio de las librerías de apache a la URL del fichero .php del servidor.
2. El fichero .php ejecuta una consulta SELECT sobre la base de datos para seguidamente devolver los resultados en formato JSON.
3. A través de la clase JSONParser.java el resultado devuelto se almacena en una variable de tipo JSONArray.

4. Se extraen todos los campos de cada negocio elemento del JSONArray, se crea un objeto de tipo Negocio y se añade al ArrayList<Negocio> resultado.

```
//Obtenemos el JSON de la URL
JSONObject json = new JSONObject();
json = jParser.makeHttpRequest(url_negocios, "GET", params);
```

Figura 53. Obtención de datos en formato JSON a partir de una petición HTTP

```
// Guardamos cada item del JSON
int id = 0, totalGaleria = 0;
String nombre = "", categoria = "", descripcion = "", direccion = "", horario = "",
        telefono = "", correo = "", web = "", facebook = "", twitter = "", imagen = "";
id = c.getInt(TAG_ID);
nombre = c.getString(TAG_NOMBRE);
categoria = c.getString(TAG_CATEGORIA);
double latitud = c.getDouble(TAG_LATITUD);
double longitud = c.getDouble(TAG_LONGITUD);
if(!c.isNull("descripcion")) descripcion = c.getString(TAG_DESCRIPCION);
if(!c.isNull("direccion")) direccion = c.getString(TAG_DIRECCION);
if(!c.isNull("horario")) horario = c.getString(TAG_HORARIO);
if(!c.isNull("telefono")) telefono = c.getString(TAG_TELEFONO);
if(!c.isNull("correo")) correo = c.getString(TAG_CORREO);
if(!c.isNull("web")) web = c.getString(TAG_WEB);
if(!c.isNull("facebook")) facebook = c.getString(TAG_FACEBOOK);
if(!c.isNull("twitter")) twitter = c.getString(TAG_TWITTER);
if(!c.isNull("imagen")) imagen = c.getString(TAG_IMAGEN);
if(!c.isNull("totalGaleria")) totalGaleria = c.getInt(TAG_GALERIA);

Negocio negocio = new Negocio(id, nombre, categoria, latitud, longitud, descripcion,
        direccion, horario, telefono, web, correo, facebook, twitter,
        imagen, totalGaleria);

negociosList.add(negocio);
```

Figura 54. Creación de un objeto de tipo Negocio a partir de los datos del JSON

Una vez disponemos de nuestro ArrayList de negocios, en el caso de que la carga haya sido exitosa, insertamos los negocios en la base de datos local por medio de la clase ComercioBD.java. Esta clase abre la base de datos local en modo lectura / escritura por medio de SQLiteOpenHelper e inserta los negocios a través del método insertTablaNegocios(ArrayList<Negocio> negocios), borrando previamente los registros existentes;

Resumiendo, nuestra app actualiza el listado de negocios cada vez que se inicializa, si dispone de Internet.
Si no es así y la carga no tiene éxito por no tener conexión o por otro tipo de error externo, simplemente cargamos los negocios que ya existían en la base de datos local, que en una primera carga se vuelcan desde un fichero .sqlite presente en la carpeta assets.

7.3.2. MainActivity.java

El MainActivity.java consta de un menú lateral desplegable de 9 opciones. Cada opción ejecuta el método selectItem(int posicion) de la clase MainActivity, al ser seleccionada. En dicho método se comprueba que la opción elegida sea distinta de la actual y si es así se realiza una transacción en la que

se remplaza el fragment actual por el fragment elegido mediante el método replace. La opción por defecto es el Mapa.

Ejemplo:

```
if (opciones[position].equals("Mapa")) {
    Fragment fragment = new MapaFragment();
    getFragmentManager().beginTransaction().replace(R.id.content_frame, fragment,
            "MainActivity").commit();
```

Figura 55. Captura del método selectItem() de MainActivity.java

En MainActivity también encontramos otro método sobrescrito de utilidad, onBackPressed(). Este método es utilizado para gestionar las pulsaciones en el botón de Atrás, por parte del usuario. Su funcionamiento se basa en que si la pila de fragments contiene elementos para desapilar, hacemos pop, y si no llamamos a doubleBackPressed() para mostrar al usuario un Toast con el mensaje "Presiona otra vez Atrás para salir".

En la siguiente captura podemos ver un ejemplo cuando el fragment actual (f) es el fragment de Información.

```
//Si el Fragment actual es InformacionFragment
}else if(f instanceof InformacionFragment) {
    int count = getFragmentManager().getBackStackEntryCount();
    if (count == 0) {
        doubleBackPressed();
    } else {
        getFragmentManager().popBackStack();
    }
```

Figura 56. Captura del método onBackPressed() de MainActivity.java

7.3.3. MapaFragment.java:

Haciendo uso de la librería de google cargamos el mapa de tipo MapView, un tipo de vista de la clase SupportMapFragment recomendado para la utilización de mapas dentro de fragments. El mapa se carga dentro de la pestaña "Mapa" haciendo uso del método getMapAsync() que implementa la interfaz OnMapReadyCallBack(), la cual garantiza que siempre tendremos disponible una instancia de dicho mapa.

```
mapView = (MapView) v.findViewById(R.id.mapView);
mapView.onCreate(savedInstanceState);
mapView.getMapAsync(this);
```

Figura 57. Código para cargar el MapView en MapaFragment.java

A continuación se ejecuta el método onMapReady(GoogleMap map), el cual mostrará el mapa en modo normal, satélite o híbrido dependiendo de la preferencia seleccionada en Ajustes.

```
@Override
public void onMapReady(GoogleMap map) {
    mapaGoogle = map;
    if (Utilidades.obtenerTipoMapaSharedPreferences(getActivity().getApplicationContext())
            .compareTo(AjustesFragment.MAPA_NORMAL) == 0){
        mapaGoogle.setMapType(GoogleMap.MAP_TYPE_NORMAL);
    }
    if (Utilidades.obtenerTipoMapaSharedPreferences(getActivity().getApplicationContext())
            .compareTo(AjustesFragment.MAPA_SATELITE) == 0){
        mapaGoogle.setMapType(GoogleMap.MAP_TYPE_SATELLITE);
    }
    if (Utilidades.obtenerTipoMapaSharedPreferences(getActivity().getApplicationContext())
            .compareTo(AjustesFragment.MAPA_HIBRIDO) == 0){
        mapaGoogle.setMapType(GoogleMap.MAP_TYPE_HYBRID);
    }
```

Figura 58. Captura del método onMapReady() de MapaFragment.java

Por medio de una consulta a la base de datos local, obtenemos los negocios a partir de la categoría seleccionada en el Spinner del Toolbar. Por defecto, la categoría seleccionada es "Todas las categorías".

```
spinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
        public void onItemSelected(AdapterView<?> adapterView, View view, int i, long l) {
            if(spinnerTouched) {
                categoria = adapterView.getItemAtPosition(i).toString();
                negociosList = comercioBD.getNegociosPorCategoria(categoria);
                addMarkers(negociosList);
            }
            spinnerTouched = false;
        }

        public void onNothingSelected(AdapterView<?> adapterView) {
            return;
        }
});
```

Figura 59. Captura de setOnItemSelectedListenter() para cargar los negocios por categoría

```
public ArrayList<Negocio> getNegociosPorCategoria(String categoria) {
    ArrayList<Negocio> negocios = new ArrayList<Negocio>();
    String query = "";
    if (categoria.equals("Todas las categorías")) {
        query = "SELECT * FROM negocios";
    } else {
        query = "SELECT * FROM negocios WHERE categoria='" + categoria + "'";
    }
    query += " ORDER BY categoria, nombre";
    Cursor c = mDatabaseOpenHelper.rawQuery(query, null);
```

Figura 60. Captura de getNegociosPorCategoria() para obtener los negocios de la BD

Una vez disponemos del ArrayList con los negocios, queda añadir los marcadores al mapa. Esto lo hacemos por medio del método addMarkers(ArrayList<Negocio> negociosList), que atribuirá una posición, un título, un snippet y un icono a cada negocio.

```
public void addMarkers(ArrayList<Negocio> negociosList){
    mapaGoogle.clear();
    for (int j = 0; j < negociosList.size(); j++) {
        mapaGoogle.addMarker(new MarkerOptions()
                .position(new LatLng(negociosList.get(j).getLatitud(),
                        negociosList.get(j).getLongitud()))
                .title(negociosList.get(j).getNombre())
                .snippet(negociosList.get(j).getDireccion())
                .icon(seleccionarDescriptor(negociosList.get(j).getCategoria())));

    }
```

Figura 61. Captura del método addMarkers() de MapaFragment.java

Por medio del widget SearchView, el usuario puede realizar la búsqueda de un comercio en concreto introduciendo su nombre o parte de él. Durante el proceso de inserción de este texto, se le mostrará al usuario una lista de sugerencias con los nombres de los negocios que concuerden con el texto introducido. Esto se consigue ejecutando una consulta a la base de datos desde el método onQueryTextChange(String newText), propio de la clase SearchView.

Otro de los métodos de SearchView es onQueryTextSubmit(String query), llamado al presionar Enter tras haber introducido un nombre, que también ejecuta una consulta sobre la base de datos para extraer el negocio deseado.

Tanto si el usuario presiona uno de los negocios de la lista de sugerencias o bien inserta el nombre completo del negocio y pulsa Enter, se añadirá al mapa por medio addMarkers() el marcador del negocio en cuestión. También aparecerá seleccionada en el Spinner la categoría a la que pertenece el negocio.

```
@Override
public boolean onQueryTextSubmit(String query) {
    String queryFormateado = Utilidades.capitalizarCadena(query);
    Log.d(TAG, "queryFormateado: "+queryFormateado);
    Negocio negocio = comercioBD.getNegocioPorNombre(queryFormateado);
        if(negocio == null){
            searchView.clearFocus();
            searchView.setFocusableInTouchMode(false);
            listaBusqueda.setAdapter(null);
            listaBusqueda.setVisibility(View.GONE);
            Toast.makeText(getActivity(), "No existe ningún negocio con ese nombre",
                    Toast.LENGTH_SHORT).show();
            return false;
        }else{
            spinnerTouched = false;
            final String catNegocioBuscado = comercioBD.getCategoriaNegocio(queryFormateado);
            negociosList.clear();
            negociosList.add(negocio);
            addMarkers(negociosList);
            //Asignamos al spinner la categoria del negocio buscado
            spinner.setSelection(adapter.getPosition(catNegocioBuscado));
            searchView.setQueryHint("Buscar Negocio");
            searchView.clearFocus();
            searchView.setFocusableInTouchMode(false);
            searchView.onActionViewCollapsed();
            listaBusqueda.setAdapter(null);
            listaBusqueda.setVisibility(View.GONE);
            return true;
        }

}
```

Figura 62. Captura del método onQueryTextSubmit() de MapaFragment.java

Ambos widgets, Spinner y SearchView, han sido declarados como ítems de menú en el fichero /res/menú/mapa e inicializados en el método onCreateOptionsMenu(Menu menu, MenuInflater inflater).

Se ha implementado además una característica existente en la página web que muestra un listado con los negocios visibles en el mapa. Esta función se lleva a cabo en la aplicación mediante el método generarListaNegocios(), que es lanzado cuando el usuario presiona la pestaña Lista y muestra en pantalla un ListView con los negocios correspondientes. Dichos negocios han sido guardados previamente en el ArrayList<Negocio> negociosVisibles y se rellena por medio del método generarNegociosVisibles() cada vez que el usuario desplaza la posición de la cámara, evento captado a través de la interfaz OnCameraChangeListener.

```
private void generarNegociosVisibles(){
    //Region visible del mapa
    LatLngBounds bounds = mapaGoogle.getProjection().getVisibleRegion().latLngBounds;
    negociosVisibles = new ArrayList<Negocio>();
    for(int i=0; i<negociosList.size(); i++){
        LatLng point = new LatLng(negociosList.get(i).getLatitud(), negociosList.get(i).getLongitud());
        if(bounds.contains(point)){
            negociosVisibles.add(negociosList.get(i));
        }else{
            if(negociosVisibles.contains(negociosList.get(i))){
                negociosVisibles.remove(negociosList.get(i));
            }
        }
    }
    cont = negociosVisibles.size();
    notificacion.setText(String.valueOf(cont));
}
```

Figura 63. Captura del método generarNegociosVisbles() de MapaFragment.java

7.3.4. DetalleNegocioActivity.java

Hacer click sobre las ventanas de información de cada marcador, nos llevará a DetalleNegocioActivity.java.

Aquí se mostrará al usuario un nuevo mapa con el marcador del negocio, además de los valores de cada propiedad del negocio, como el nombre, descripción, dirección…

Un aspecto importante en este activity es la propiedad clickable que implementan algunos de los TextViews mostrados, permitiendo lanzar un Intent a un servicio en concreto. Es decir, si por ejemplo el usuario presiona sobre el número de teléfono, se abrirá automáticamente el servicio de llamadas con el teléfono escrito en pantalla. Esto también es válido para el correo electrónico, página web, facebook o twitter, que actuarán de forma pertinente.

```
public void onClickTelefono(View v) {
    TextView numTlf = (TextView) v;
    Intent i = new Intent(android.content.Intent.ACTION_DIAL,
            Uri.parse("tel:" + numTlf.getText().toString()));
    try {
        startActivity(i);
    } catch (android.content.ActivityNotFoundException ex) {
        Toast.makeText(this, "No es posible realizar la llamada",
                Toast.LENGTH_LONG).show();
    }
}
```

Figura 64. Captura del método onClickTelefono() de DetalleNegocioActivity.java

Otro apartado importante es el menú de opciones incluido, que entre otras cosas permite añadir el negocio a favoritos, compartir la ubicación del negocio, enviar notificaciones o trazar la ruta más corta (a pie o en coche) entre la ubicación del dispositivo y el negocio. Esta última función se consigue por medio de la clase GMapV2Direction.java, que devuelve la ruta entre origen y destino indicados, calculada a través de la api directions de Google y representada en el mapa a través de un objeto de tipo Polyline. La función se llama a través del método route.

```
private void route(final LatLng sourcePosition, final LatLng destPosition, final String mode) {
    final Handler handler = new Handler() {
        public void handleMessage(Message msg) {
            try {
                Document doc = (Document) msg.obj;
                GMapV2Direction md = new GMapV2Direction(sourcePosition, destPosition, mode);
                ArrayList<LatLng> directionPoint = md.getDirection(doc);
                PolylineOptions rectLine =  new PolylineOptions().width(10).geodesic(true);
                if(mode.equals(GMapV2Direction.MODE_WALKING)){
                    rectLine.color(Color.RED);
                }else{
                    rectLine.color(Color.BLUE);
                }

                for (int i = 0; i < directionPoint.size(); i++) {
                    rectLine.add(directionPoint.get(i));
                }
                if(polyline != null) polyline.remove();
                polyline = mapaGoogle.addPolyline(rectLine);
                md.getDurationText(doc);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }

        ;
    };
    new GMapV2DirectionAsyncTask(handler, sourcePosition, destPosition, mode).execute();
}
```

Figura 65. Captura del método route() de DetalleNegocioActivity.java

7.3.5. NoticiasFragment.java

En este fragment cargamos la sección de Noticias. Dado que las noticias pueden ser variables y es un contenido que depende de la página web, lo mostramos a través de un WebView, es decir, visualizamos el apartado de noticias de la web dentro de nuestro fragment. Para ello cargamos la url de la página web correspondiente y le aplicamos los métodos de navegación propios cómo canGoBack() o canGoForward().

```
webView = (WebView) v.findViewById(R.id.webView);
webView.setVisibility(View.INVISIBLE);

//Habilitamos JavaScript
webView.getSettings().setJavaScriptEnabled(true);
webView.setWebViewClient(new MyWebViewClient());
webView.canGoBack();
webView.canGoForward();
webView.loadUrl(urlNoticias);
```

Figura 66. Captura de declaración del WebView de NoticiasFragment.java

Además, inyectamos código JavaScript al html devuelto para ocultar la cabecera y pie de página. Se mostrará al usuario un ProgressDialog mientras dure este proceso de carga.

```
@Override
public void onPageFinished(WebView view, String url) {
    StringBuilder builder = new StringBuilder("");
    builder.append("javascript:(function() {");
    //Comprobamos la url para ocultar contenidos
    if(url.equals(urlNoticias)){
        builder.append("document.getElementById('headercomercios').style.display='none';");
        builder.append("document.getElementById('footer').style.display='none';");
        builder.append("document.getElementsByClassName('row')[0].style.display='none';");
    }else{
        builder.append("document.getElementById('headercomercios').style.display='none';");
        builder.append("document.getElementById('footer').style.display='none';");
    }

    builder.append("})()");
    webView.loadUrl(builder.toString());
    webView.setVisibility(View.VISIBLE);
    progressDialog.dismiss();
}
```

Figura 67. Captura de onPageFinished() de NoticiasFragment.java

Otros elementos de la aplicación que utilizan WebViews son los apartados de "Ayuda al comerciante" y "Emprendedores", ambos ubicados en ZonaComercianteFragment.java. Puesto que el contenido de estos apartados es muy extenso y no vamos a modificar su estructura, resulta más viable visualizar por medio de un WebView lo que ya existe en la página web que declarar multitud de layouts y drawables junto con sus clases .java correspondientes, pues sólo conseguiríamos ampliar notablemente el tamaño de la aplicación.

7.3.6. MercadosYMercadillosFragment.java:

Este fragment representa la sección de Mercados / Mercadillos de nuestro menú. Contiene 3 opciones para las secciones Mercado central, Otros mercados y Mercadillos. Estas opciones se seleccionan a través de layouts clickables y ejecutan el método replace() de la clase FragmentTransaction para cargar CategoriaMercadoFragment.java pasándole un TAG como parámetro. Esta clase analiza el TAG recibido e infla el layout correspondiente a la opción elegida. Veamos un ejemplo para cargar la opción de Mercado central, haciendo uso del TAG "Central" en el método replace():

```
public void onClick(View v) {
    getFragmentManager().beginTransaction().commit();
    FragmentTransaction ft = getFragmentManager().beginTransaction();
    switch(v.getId()) {
    case R.id.layout_mercado_central:
        ft.replace(R.id.frame_mercados_y_mercadillos,
            new CategoriaMercadoFragment(), "Central").addToBackStack(null);
        break;
```

Figura 68. Captura del método onClick() de MercadosYMercadillosFragment.java

```
View v = null;
if(this.getTag().equals("Central")) {

    v = inflater.inflate(R.layout.fragment_mercado_central, container, false);

    toolbar.setTitle("Mercado central");
```

Figura 69. Captura de código del método onCreateView() de CategoriaMercadoFragment.java

Las galerías existentes en Mercados / Mercadillos se han llevado a cabo a través del uso de modelos, adapters y GridViews. Para tratar por ejemplo, la galería de imágenes de Mercado central, utilizamos la clase GaleriaAdapter.java que extiende de ArrayAdapter:

```
GaleriaAdapter adapter = new GaleriaAdapter(this.getActivity(),
        R.layout.item_empresa_galeria, getData(8, urlImagen));
gridView.setAdapter(adapter);
```

Figura 70. Captura de asignación de una Adapter al GridView de Mercado central

Donde getData() se corresponde con un método que devuelve un ArrayList con las urls de cada item que conforma la galería, partiendo del número de imágenes presentes en la galería y de la url de dicha galería:

```
private ArrayList getData(int totalGaleria, String url) {
    ArrayList<ItemGaleria> items = new ArrayList<ItemGaleria>();
    String urlImagen = url + "galeria/0";
    for (int i = 1; i <= totalGaleria; i++) {
        items.add(new ItemGaleria(urlImagen + i + ".jpg"));
    }
    return items;
}
```

Figura 71. Captura del método getData() para generar las urls de las imágenes

Las urls obtenidas serán asignadas en el Adapter a la librería Picasso, la cual se utiliza a lo largo de todo el proyecto para cargar las imágenes del servidor:

```
Picasso.with(context)
.load(item.getUrl())
.placeholder(R.drawable.progress_animation)
.memoryPolicy(MemoryPolicy.NO_CACHE, MemoryPolicy.NO_STORE)
.error(R.drawable.sin_imagen)
.into(holder.imagen);
```

Figura 72. Captura del uso de Picasso para cargar imágenes vía Internet

En la mayoría de los casos, se han utilizado los siguientes métodos de Picasso para el proceso de carga de imágenes:

-with: recibe un contexto determinado como parámetro.
-load: url de la imagen que queremos cargar. También se le puede asignar un recurso drawable.
-placeholder: muestra la imagen asignada mientras tiene lugar el proceso de carga, en este caso, una animación que gira una imagen estática para dar el efecto de un .gif.
-memoryPolicy: establece una política de memoria. Por defecto, Picasso guarda las imágenes en caché a menos que indiquemos lo contrario.
-error: para mostrar una imagen de error de carga.
-into: para establecer el ImageView donde será mostrada la imagen.

La aplicación también dispone de un ViewPager para ver las imágenes ampliadas y desplazarse entre una y otra.
Para conseguir esto, se envía un Intent a la clase DetalleGaleriaActivity.java, con la url de la galería de la imagen presionada, la posición en la galería de dicha imagen y el total de imágenes en la galería. Una vez obtenidos los extras del Intent, se asigna al ViewPager su Adapter correspondiente como podemos observar en la siguiente captura:

```
if(getIntent().getExtras() != null){

    url = getIntent().getStringExtra("url");
    posicion = getIntent().getIntExtra("posicion", 0);
    total = getIntent().getIntExtra("total", 0);

    viewPager = (ViewPager) findViewById(R.id.view_pager);
    viewPagerAdapter = new ViewPagerAdapter(this, getData(total));
    viewPager.setAdapter(viewPagerAdapter);
    viewPager.setCurrentItem(posicion);
}
```

Figura 73. Código de onCreate() de DetalleGaleriaActivity.java para el uso de ViewPager

7.3.7. ProductosLocalesFragment.java

En este fragment se implementa la sección de Productos locales de nuestro menú. Contiene también 3 opciones seleccionables a partir de layouts clickables para los apartados de Alimentación, Artesanía y Bebidas. Aquí también se hará uso del método replace() de FragmentTransaction para cargar el nuevo

fragment CategoriaProductoFragment.java pasándole de nuevo un TAG como parámetro. Esta clase analizará dicho TAG para cargar el fragment final de ProductoFragment.java en base al tipo de producto seleccionado:

```
if(this.getTag().equals("Alimentación")){
    for (String tab_nombre : tabs_alimentacion) {
        Bundle b = new Bundle();
        b.putString("categoria", "Alimentación");
        b.putString("producto", tab_nombre);
        b.putStringArray("productos", tabs_alimentacion);
        tabs.addTab(tabs.newTabSpec(tab_nombre).setIndicator(tab_nombre),
                ProductoFragment.class, b);
    }
```

Figura 74. Captura de código del método onCreateView() de CategoriaProductoFragment.java

Las empresas que serán mostradas en ProductoFragment.java representan a empresas de distintos ámbitos localizadas en Almería y precursoras de productos de la región. Se obtienen de un fichero .xml localizado en el directorio assets. Este tipo de implementación se debe a que las empresas de dicho fragment también son visualizadas en la página web a partir de un .xml, debido a que son un elemento estático, no cambiante, que no necesita de actualizaciones. Para la carga de este fichero y su tratamiento se hará uso de la clase XMLParser.java, que se encarga de parsear el contenido etiquetado de las empresas, obteniendo posteriormente los ítems de empresas que se añaden a su contenedor, un GridView:

```
InputStream in = getActivity().getApplicationContext()
        .getAssets().open("empresas.xml");
XMLParser parser = new XMLParser();
empresas = parser.parsear(in);
```

Figura 75. Código de ProductoFragment.java para parsear "empresas.xml"

Una vez disponemos del ArrayList, creamos un GridView de forma programática, al que añadir los ítems.

```
//Creamos el GridView con las empresas
GridView gridView = new GridView(this.getActivity());
gridView.setLayoutParams(
        new GridView.LayoutParams(LayoutParams.WRAP_CONTENT,
                LayoutParams.WRAP_CONTENT));
gridView.setNumColumns(GridView.AUTO_FIT);
gridView.setColumnWidth(Utilidades.dpToPx(getActivity(), 125));
gridView.setVerticalSpacing(60);
gridView.setHorizontalSpacing(50);
gridView.setStretchMode(GridView.STRETCH_COLUMN_WIDTH);
gridView.setVerticalScrollBarEnabled(false);
gridView.setAdapter(new EmpresasAdapter(this.getActivity(),
        R.layout.item_empresa, empresasResultado));
gridView.setOnItemClickListener(getItemClickListener());
layout_empresas.addView(gridView);
```

Figura 76. Código de ProductoFragment.java para crear el GridView con las empresas

7.3.8. HistorialFragment.java

El fragment de Historial muestra los negocios que han sido visualizados en DetalleNegocioActivity.java. Cada vez que se visita un negocio y si la preferencia "Guardar historial" está habilitada, se almacena en la tabla historial de la base de datos local el id del negocio correspondiente, si no ha sido almacenado antes.

Para mostrar los negocios de dicha tabla al usuario, se ejecuta una consulta SQL que devuelve los negocios por orden inverso al insertado. De esta forma, los negocios presentados de nuevo en un GridView, aparecerán por orden de visita.

```
public ArrayList<ItemGrid> getNegociosHistorial(){
    ArrayList<ItemGrid> items = new ArrayList<ItemGrid>();
    String query = "SELECT id, nombre, categoria, imagen FROM negocios "
            + "INNER JOIN historial ON negocios.id=historial.id_negocio "
            + "ORDER BY historial.id_historial DESC";
    Cursor c = mDatabaseOpenHelper.rawQuery(query, null);
    if (c.getCount() > 0) {
        c.moveToFirst();
        for(int i=0; i<c.getCount(); i++){
            ItemGrid item = new ItemGrid(c.getInt(c.getColumnIndexOrThrow("id")),
                    c.getString(c.getColumnIndexOrThrow("nombre")),
                    c.getString(c.getColumnIndexOrThrow("categoria")),
                    c.getString(c.getColumnIndexOrThrow("imagen")));
            items.add(item);
            c.moveToNext();
        }
        c.close();
    } else {
        items = null;
        c.close();
    }
    return items;
}
```

Figura 77. Método getNegociosHistorial() para devolver los negocios de historial

El usuario también dispondrá de una opción en los ajustes donde poder limpiar el historial, si así lo desea. Esta opción realizara un borrado de la tabla historial, como se puede observar en la siguiente captura:

```
public boolean deleteTablaHistorial() {
    mDatabaseOpenHelper.comercioBD.beginTransaction();
    try {
        mDatabaseOpenHelper.comercioBD.delete("historial", null, null);
        mDatabaseOpenHelper.comercioBD.setTransactionSuccessful();
    } catch (Exception e){
        mDatabaseOpenHelper.comercioBD.endTransaction();
        return false;
    } finally {
        mDatabaseOpenHelper.comercioBD.endTransaction();
    }
    return true;
}
```

Figura 78. Método deleteTablaHistorial() para borrar los registros de la tabla historial

7.3.9. FavoritosFragment.java

La sección de Favoritos tiene una estructura similar al de Historial, es decir, aparecerán los ítems de los negocios compuestos por una imagen y texto, dentro de un GridView.

En este caso los negocios se visualizan en el fragment tras haber sido añadidos a la tabla Favoritos al presionar la opción "Añadir a favoritos" de DetalleNegocioActivity.java. Para eliminar el favorito, se puede volver a presionar la opción mencionada anteriormente, en cuyo texto ahora aparecerá "Borrar favorito". También se puede eliminar el favorito a través del menú Popup de FavoritosFragment.java, que tiene las opciones "Ver negocios" para ir a DetalleNegocioActivity y "Borrar favorito" para eliminar el favorito de la tabla y del GridView.

```
if (item.getTitle().equals("Borrar favorito")) {
    comercioBD.deleteFromFavoritos(items.get(position).getId());
    items.remove(position);
    gridViewAdapter.notifyDataSetChanged();
    Toast.makeText(getActivity(), "Favorito eliminado",
            Toast.LENGTH_SHORT).show();
```

Figura 79. Captura de FavoritosFragment.java para eliminar un negocio de Favoritos

Sin embargo, puede darse el caso de que el usuario se encuentre en Favoritos, pulse un negocio para ir a DetalleNegocioAcitivity, borre allí el favorito y después vuelva a Favoritos. En este caso deberíamos también eliminar el negocio del GridView si fue eliminado en DetalleNegocioAcitivty. Esto lo conseguimos iniciando DetalleNegocioActivity mediante startActivityForResult() y posteriormente eliminando el negocio en onActivityResult() si es preciso.

```
//Para  eliminar el favorito del gridview si es eliminado en DetalleNegocioActivity
@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == 1) {
        if(resultCode == Activity.RESULT_OK){
            int result=data.getIntExtra("result", 0);
            if(result == 1){
                items.remove(item_selected);
                gridViewAdapter.notifyDataSetChanged();
                if (items.size() == 0) {
                    gridView.setVisibility(View.GONE);
                    tv_historial.setVisibility(View.VISIBLE);
                    tv_historial.setText("No hay negocios en Favoritos");
                }
            }
        }
    }
}//onActivityResult
```

Figura 80. Captura de onActivityResult() de FavoritosFragment.java

7.3.10. InformacionFragment

Este fragment muestra toda la información referente a la aplicación de Comercio Almería, dividida en los apartados “Sobre Comercio de Almería”, “Preguntas Frecuentes”, “Términos y condiciones”, “Política de privacidad” y “Contacto”. También hace uso de transacciones de fragments a través de una clase intermedia (CategoriaInformacionFragment.java) , que en función del TAG aplicado como parámetro en el remplazamiento, infla el layout pertinente.

```
View v;
if(this.getTag().equals("SobreComercio")) {
    v = inflater.inflate(R.layout.fragment_sobrecomercio, container, false);
}else if(this.getTag().equals("PreguntasFrecuentes")){
    v = inflater.inflate(R.layout.fragment_preguntas, container, false);
}else if(this.getTag().equals("TerminosYCondiciones")){
    v = inflater.inflate(R.layout.fragment_terminos, container, false);
}else if(this.getTag().equals("PoliticaPrivacidad")){
    v = inflater.inflate(R.layout.fragment_politica, container, false);
}else{
    v = inflater.inflate(R.layout.fragment_contacto, container, false);
```

Figura 81. Captura del método onCreateView() de CategoriaInformacionFragment.java

Los layouts mencionados constan únicamente de TextViews de contenido estático declarado en el fichero /res/values/strings.xml.

```
<!-- InformacionFragment -->
<!-- SobreComercio -->
<string name="titulo_sobrecomercio">SOBRE COMERCIO DE ALMERÍA</string>
<string name="texto_sobrecomercio">Comercio de Almería es un proyecto que persigue <b>impulsar</b>
    así como los <b>productos locales</b> elaborados en nuestra tierra.\n\nAdemás ofrecemos <b>infc
    dispone el ayuntamiento si eres un emprendedor.\n\nCualquier tipo de negocio de la capital pod
</string>
```

Figura 82. Captura del texto de InformacionFragment almacenado en el fichero strings.xml

7.3.11. LocationService.java

Nuestra app también utiliza servicios para llevar a cabo el envío de notificaciones. En primer lugar disponemos de la clase AutoStartReceiver.java que es lanzada cuando ha sido habilitado el GPS. Esta clase extiende de BroadcastReceiver, que permite a la aplicación escuchar en segundo plano las acciones que tienen lugar en todo el dispositivo.
En otras palabras, cuando el usuario activa el GPS del teléfono, Android envía un Intent con la acción etiquetada de "android.location.PROVIDERS_CHANGED". Nuestra clase AutoStartReceiver.java analiza si el Intent enviado coincide con dicha acción, en cuyo caso inicializa el servicio de LocationService.java.

LocationService.java extiende de Service y se utiliza para comprobar que negocios son objeto de envío de notificaciones, para añadir su alerta de proximidad correspondiente. Estos negocios son extraídos de la tabla Notificaciones o de mediante una consulta de unión entre las tablas Notificaciones y Favoritos, si es que la preferencia Alertas de Favoritos está habilitada. Dichos registros son almacenados en un ArrayList<ItemNotificacion> y añadida su alerta correspondiente mediante el método addProximityAlert().

```
private void addProximityAlert(double latitude, double longitude, int id_negocio, String nombre){
    Bundle extras = new Bundle();
    extras.putInt("id_negocio", id_negocio);
    extras.putString("nombre_negocio", nombre);
    intent.putExtra(BROADCAST_ACTION, extras);
    PendingIntent proximityIntent = PendingIntent.getBroadcast(mContext, id_negocio, intent,
            PendingIntent.FLAG_UPDATE_CURRENT);
    locationManager.addProximityAlert(latitude,longitude, radius, -1, proximityIntent);
    IntentFilter filter = new IntentFilter(BROADCAST_ACTION);
    mContext.registerReceiver(receiver, filter);
}
```

Figura 83. Método addProximityAlert() de LocationService.java para añadir alertas

Si queremos dejar de enviar alertas para un determinado negocio, por ejemplo cuando el usuario pulsa el botón “Dejar de recordar”, además de eliminar dicho negocio de la tabla Notificaciones debemos eliminar la alerta creada. Esto lo hacemos mediante el método deleteProximityAlert().

```
PendingIntent proximityIntent = PendingIntent.getBroadcast(this, id_negocio, intent,
        PendingIntent.FLAG_CANCEL_CURRENT);
proximityIntent.cancel();
locationManager.removeProximityAlert(proximityIntent);
```

Figura 84. Código de deleteProximityAlert() de LocationService.java para eliminar alertas

Añadidas las alertas, necesitamos una clase que reciba los Intents generados en LocationService.java para crear las notificaciones de forma adecuada. Esta clase se denomina ProximityAlertReceiver.java y extiende también de BroadcastReceiver. Hace uso de NotificationCompat.Builder y tiene en cuenta las preferencias marcadas en Ajustes para construir la notificación de la siguiente manera:

```
NotificationCompat.Builder notificationBuilder = new NotificationCompat.Builder(context)
        .setWhen(System.currentTimeMillis())
        .setContentText(contentText)
        .setContentTitle("Alerta de proximidad")
        .setSmallIcon(R.drawable.ic_launcher)
        .setAutoCancel(true)
        .setContentIntent(pendingIntent)
        .setPriority(Notification.PRIORITY_DEFAULT);

Uri sound = Uri.parse(Utilidades.obtenerTonoSharedPreferences(mContext));
notificationBuilder.setSound(sound);

if (Utilidades.obtenerLedSharedPreferences(mContext)) {
    notificationBuilder.setLights(0xFF0000FF, 100, 3000);
}
if (Utilidades.obtenerVibracionSharedPreferences(mContext)) {
    notificationBuilder.setVibrate(new long[] { 1000, 1000 });
}

Notification notification = notificationBuilder.build();

notificationManager.notify(NOTIFICATION_ID, notification);
```

Figura 85. Código de onReceive() de ProximityAlertReceiver.java para generar una notificación

Tanto los receiver como sus acciones, "android.location.PROVIDERS_CHANGED" y de "ACTION_PROXIMITY_ALERT", deben ser declaradas en el manifest de la siguiente manera:

```
<receiver android:name="com.example.comercioalmeria.service.AutoStartReceiver">
     <intent-filter>
         <action android:name="android.location.PROVIDERS_CHANGED" />
         <category android:name="android.intent.category.DEFAULT" />
     </intent-filter>
 </receiver>
```

Figura 86. Declaración del receiver en AndroidManifest.xml

7.3.12. AjustesFragment.java

AjustesFragment.java extiende de la clase PreferenceFragment y hace uso de SharedPreferences para el almacenamiento de preferencias en ficheros .xml. Dichos ficheros son almacenados en el directorio */data/data/paquete.java/shared_prefs/nombre_fichero.xml* del dispositivo.
En Ajustes, el usuario dispone de diversas opciones divididas en categorías, todas ellas declaradas en el fichero /res/xml/preferences.xml. Las categorías y opciones son las siguientes:

-General:

-Tipo de mapa: permite cambiar el tipo de mapa (Normal, Satélite o Híbrido).
-Guardar historial: habilita o deshabilita el almacenamiento de negocios visitados en la tabla historial de la base de datos.
-Limpiar historial: permite vaciar la tabla historial de la base de datos.

-Notificaciones:

-Radio de alertas: establece el radio en el cuál será enviada la alerta de proximidad (25 m, 50 m, 100 m).
-Recordar: permite establecer la frecuencia en la que se enviarán las alertas sobre un negocio determinado (Sólo una vez, Siempre).
-Alertas de favoritos: si se selecciona esta opción, se enviará una notificación sobre cualquier negocio añadido a favoritos. La frecuencia de envío de notificaciones de favoritos será "Siempre", aunque en la preferencia Recordar esté seleccionada la opción "Sólo una vez". Esto se debe a querer enfocar de una forma diferente los negocios favoritos.
-Tono: establece un tono para la alerta. Los tonos disponibles se extraen del propio Android.
-Vibración: habilita o deshabilita la vibración para el envío de notificaciones.
-LED: habilita o deshabilita el LED del teléfono para el envío de notificaciones.

-Recomendar:

-Enviar correo/whatsapp/line…: permite el envío a los contactos de un enlace al store donde se haya publicado la aplicación a modo publicitario.

-Puntuar:

-Puntuar en Google Play: enlaza con el store en el que se encuentra publicada la aplicación para darle una puntuación.

-Sobre:

-Versión: muestra el número de versión de la aplicación extraído del fichero AndroidManifest.xml.

Como podemos ver, Comercio Almería cuenta con algunos ajustes que dotan de una funcionalidad extra a la aplicación. Otros ajustes típicos como Recomendar o Puntuar, podemos encontrarlos en cualquier aplicación Android por lo que no es inapropiado que nuestra app también cuente con ellos.

En la siguiente captura podemos apreciar cómo se asignan los valores a una preferencia determinada:

```
// Actualizamos la lista de radios
alertas = (ListPreference) findPreference("key_alertas");
if (alertas != null) {
    CharSequence entries[] = {ALERTAS_25, ALERTAS_50, ALERTAS_100};
    CharSequence entryValues[] = {ALERTAS_25, ALERTAS_50, ALERTAS_100};
    alertas.setEntries(entries);
    alertas.setEntryValues(entryValues);
}
```

Figura 87. Captura de AjustesFragment.java para asignar la lista de radios a su preferencia

7.4. Compatibilidad

En esta sección analizaremos la compatibilidad de la app en diferentes dispositivos y pantallas.

En primer lugar, cabe destacar que esta aplicación hace uso de los servicios de Google Play por lo que si no están instalados en el dispositivo, no podrá funcionar. Estos servicios se utilizan para actualizar las aplicaciones de Google y de Google Play. Proporcionan funciones principales, como la autenticación para servicios de Google, contactos sincronizados, acceso a la última configuración de privacidad del usuario, así como servicios basados en la ubicación de mayor calidad y menor potencia. Asimismo, los servicios de Google Play mejoran la experiencia de una aplicación. Permiten agilizar las búsquedas sin conexión, proporcionan mapas más envolventes y mejoran la experiencia de usuario.

En el caso de que el dispositivo en el que se instala la aplicación no disponga de estos servicios, se mostrará una advertencia al usuario y se cerrará la aplicación. La siguiente captura está tomada de un dispositivo emulado que no dispone de Google Play services:

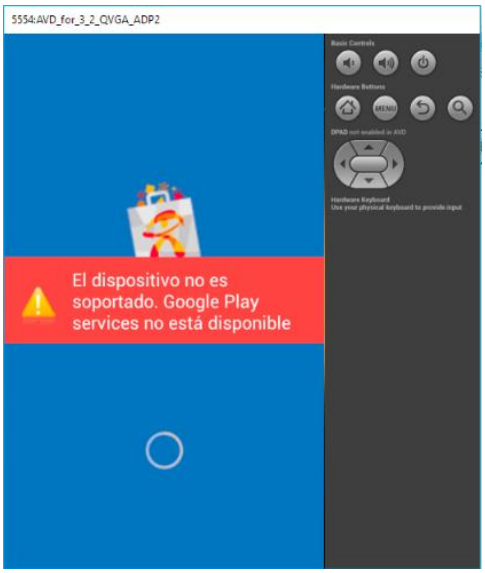

Figura 88. Captura de un dispositivo sin Google Play services

Como ya se ha mencionado anteriormente en este documento, la aplicación requiere disponer de la versión 4.0 como mínimo para funcionar correctamente, que se corresponde con la API 14. Esto se debe a que existen funcionalidades en la aplicación que exigen el uso de un cierto nivel de API. Algunos ejemplos son los siguientes:

-El uso de Fragments requiere de la API 11.
-El uso de *Theme.Holo.Light.NoActionBar* como theme principal de la app, requiere de la API 13.
-Los métodos *setImeOptions()* y *onActionViewColapsed()* requieren la API 14.

Por lo tanto, el valor minSdk de AndroidManifest.xml está establecido a 14. Teniendo esto en cuenta, si observamos la siguiente figura podremos hacernos una idea de la amplitud de mercado que podremos abarcar:

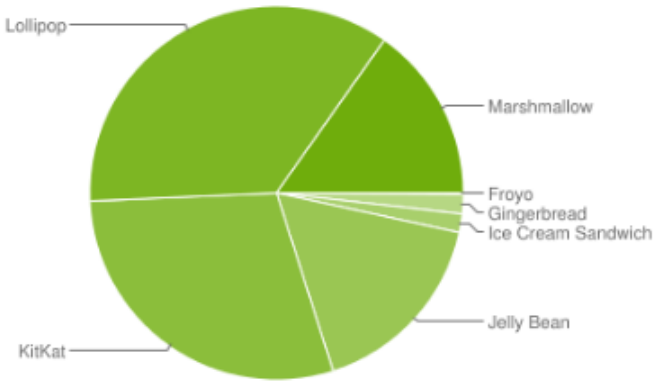

Figura 89. Gráfica de la distribución de versiones de Android en el mercado

En esta otra imagen se muestran los datos de versiones por porcentajes recopilados hasta el 1 de agosto de 2016 (no se tienen en cuenta las versiones con un porcentaje inferior al 0,1%):

Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	1.7%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.6%
4.1.x	Jelly Bean	16	6.0%
4.2.x		17	8.3%
4.3		18	2.4%
4.4	KitKat	19	29.2%
5.0	Lollipop	21	14.1%
5.1		22	21.4%
6.0	Marshmallow	23	15.2%

Figura 90. Datos recopilados sobre las versiones de Android

Como podemos ver el uso de la API 14 como SDK mínimo no supone un gran impacto a la hora de llegar al mayor número posible de dispositivos pues sólo alrededor del 2 % de los dispositivos existentes están por debajo de la versión Ice Cream Sandwich.

La aplicación es adaptable a pantallas en modo portrait y landscape, es decir, se posibilita la visualización de la app con la pantalla del dispositivo orientada ya sea horizontal o verticalmente. Esto se consigue mediante la declaración de los atributos *android:configChanges = "orientation| keyboardHidden | screenSize"* o *android:screenOrientation="fullSensor"* dependiendo de si queremos recrear el layout por completo o solamente rotar las vistas, respectivamente.

```
<!-- MainActivity -->
<activity
    android:name="com.example.comercioalmeria.MainActivity"
    android:label="@string/app_name"
    android:configChanges="orientation|keyboardHidden|screenSize" >
</activity>
```

Figura 91. Declaración del MainActivity.java en AndroidManifest.xml

En el caso de utilizar *android:configChanges*, necesitamos sobrescribir el método onConfigurationChanged() para aplicar los respectivos cambios al Activity en cuestión.

```
@Override
public void onConfigurationChanged(Configuration newConfig) {
    super.onConfigurationChanged(newConfig);
    toggle.onConfigurationChanged(newConfig);
}
```

Figura 92. Sobreescritura del método onConfigurationChanged() de MainActivity.java

Por último, la aplicación también es compatible con dispositivos de diferentes tamaños de pantalla (small, normal, large, xlarge).

Se han incluido en la carpeta /res/ del proyecto los directorios /layout-large y /layout-xlarge, destinadas a dispositivos de tamaños de pantalla de 7” y 10” respectivamente, además de la ya existente /layout, para pantallas de tamaño normal. En estas carpetas se han añadido aquellos layouts que requerían de alguna modificación de diseño para poder ser visualizados correctamente en pantallas de mayor tamaño.

En la siguiente imagen podemos apreciar la correspondencia entre pulgadas y densidades de pixeles y sus tamaños generalizados:

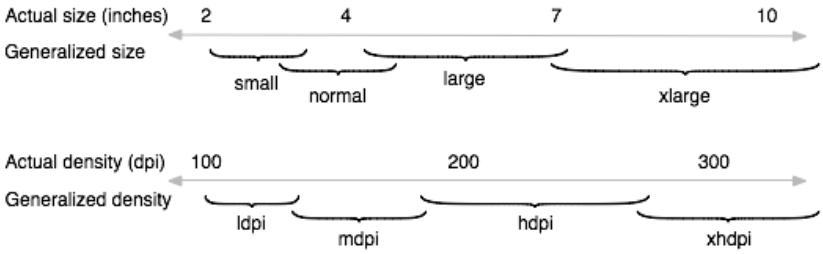

Figura 93. Captura de asignación de densidades y tamaños reales a densidades y tamaños generalizados

Los layouts mencionados se correponden con los fragments principales de las secciones de "Mercados/Mercadillos", "Productos locales", "Zona comerciante" e "Información".
La siguiente captura muestra el contenido de estas carpetas en el proyecto de Eclipse:

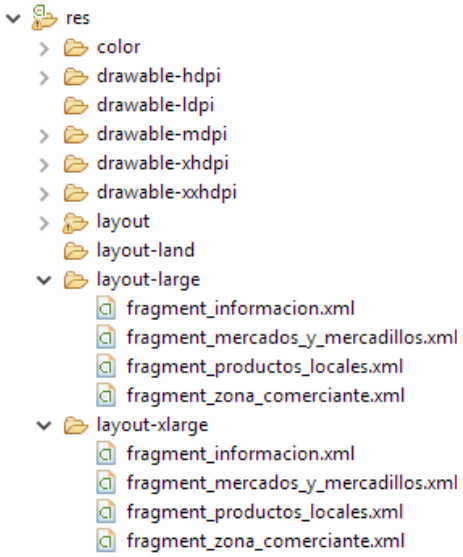

Figura 94. Captura de las carpetas layout-large y layout-xlarge

A estos nuevos layouts se le han asignado tamaños de fuente e imagen diferentes a los de la carpeta /res/layout/ para que el contenido se ajuste adecuadamente a los nuevos tamaños de pantalla. Por ejemplo, el layout de Mercados/Mercadillos (fragment_mercados_y_mercadillos.xml) de la carpeta /res/layout/, tiene las siguientes propiedades:

Figura 95. Configuración del layout de Mercados/Mercadillos para pantallas normales

Y el layout de la carpeta /res/layout-xlarge/:

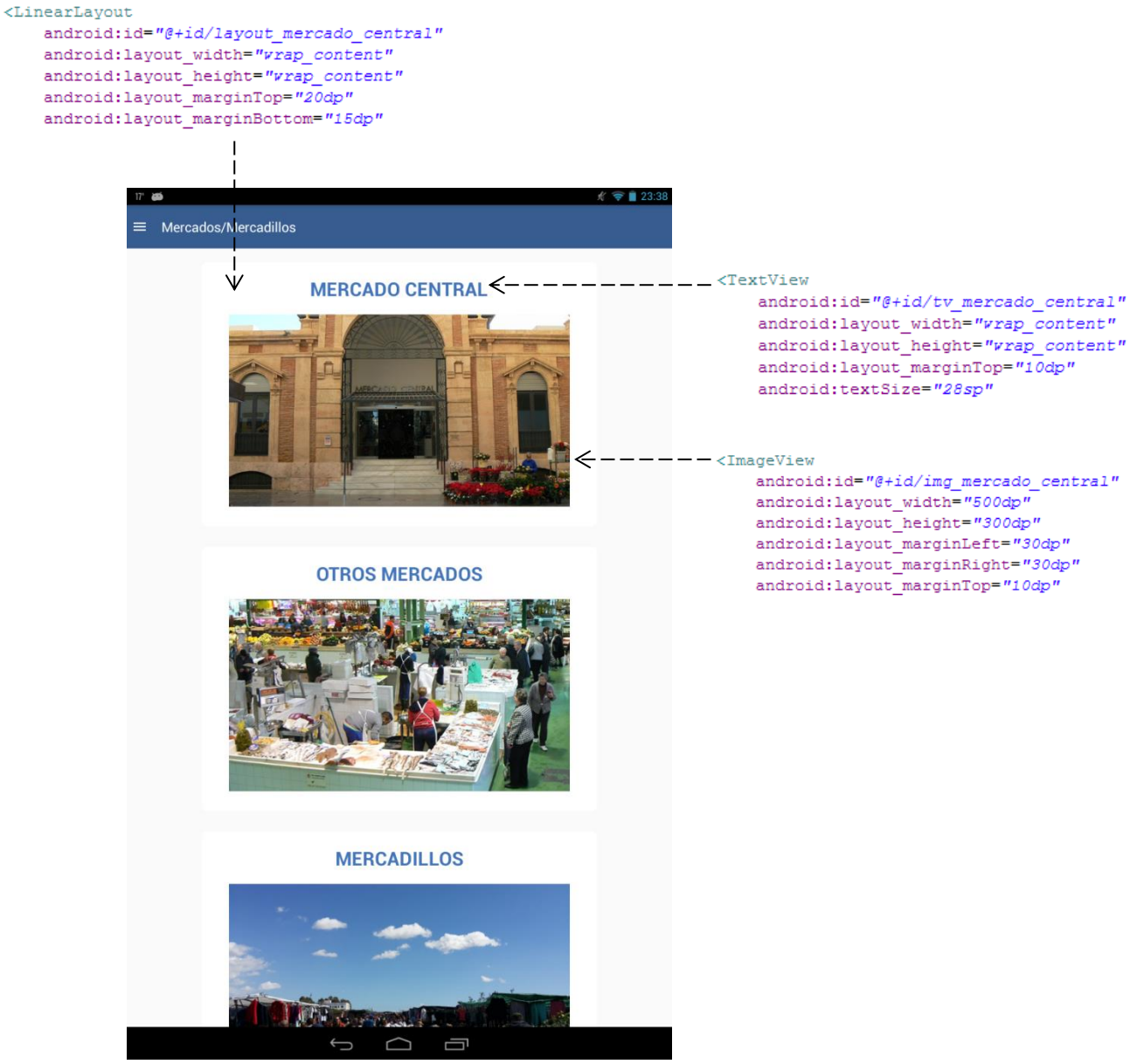

Figura 96. Configuración del layout de Mercados/Mercadillos para pantallas de 10”

7.5. Depuración

Durante el desarrollo del proyecto se han realizado tareas de depuración con bastante frecuencia para comprobar si el código que se iba implementando estaba libre de errores. En la mayoría de los casos se ha depurado mediante el uso de sentencias Log visualizadas en el LogCat de Eclipse, como podemos observar a continuación:

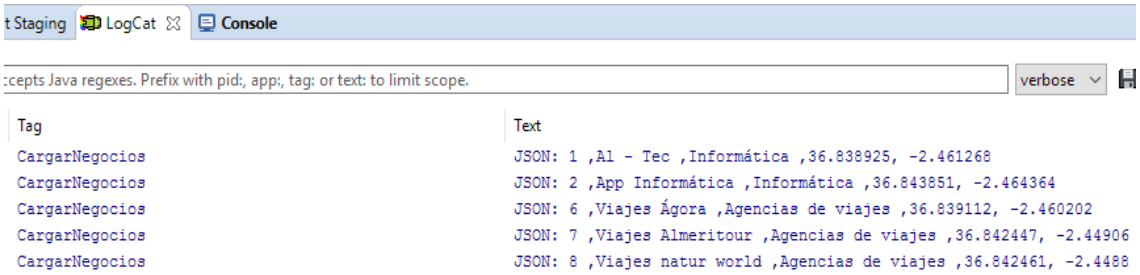

Figura 97. Captura del uso del LogCat para las tareas de depuración

Dado que esta tarea puede resultar lenta y tediosa utilizando dispositivos emulados, casi todo el proceso de depuración se ha llevado a cabo en el propio teléfono móvil del que se dispone, un Samsung Galaxy S4. Para ello nos hemos servido de la herramienta ADB, que es el acrónimo de "Android Debug Bridge". Esta herramienta que se instala junto con el SDK de Android, nos permite acceder y controlar un dispositivo Android desde el ordenador, ya sea a través de USB o por Wifi.

7.5.1. Depuración USB

En este modo de depuración conectamos el dispositivo al ordenador mediante un cable USB. Si el servicio adb no está ejecutándose, nos situamos sobre el directorio /Android/android-sdk/platform-tools/ e introducimos el comando adb start-server:

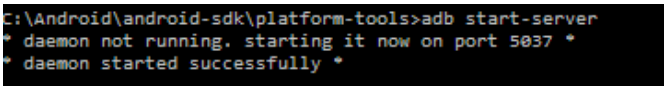

Figura 98. Captura de la inicialización del servicio adb

Si lo deseamos, también podemos configurar adb cómo una variable de entorno para poder acceder a ella desde cualquier directorio. Para ello, sólo hay que dirigirse a la Configuración Avanzada del Sistema del PC, Variables de entorno y en Variables de Sistema, buscamos PATH y lo editamos añadiendo la ruta donde está el emulador de adb dentro de la carpeta platform-tools (En nuestro caso C:\Android\android-sdk\platform-tools\).

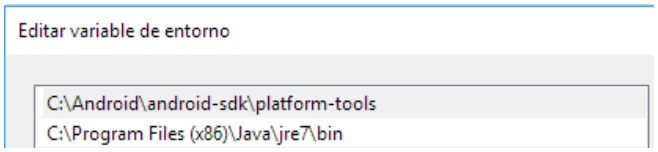

Figura 99. Captura de configuración de adb como variable de entorno

Tras introducir el comando adb start-server, es posible que en nuestro dispositivo aparezca un diálogo solicitándonos aceptar la huella de clave RSA del PC, la aceptamos y ya habremos creado una conexión vía USB entre nuestro ordenador y nuestro dispositivo. Lo único que queda es ejecutar el proyecto como Aplicación Android seleccionando como Target el dispositivo conectado.

7.5.2. Depuración Wifi

En ocasiones puede ser preferible que la conexión sea mediante Wifi para así evitar el uso de cables y tener mayor movilidad. Para este propósito se utilizará el protocolo TCP/IP para poder acceder al dispositivo a distancia, utilizando la misma conexión LAN. Crear una conexión entre dispositivo y ordenador vía TCP/IP, exige la inserción de otros comandos adb. Para no depender de esta inserción de comandos cada vez que se quiera depurar por wifi, el estudiante ha creado un script de Windows dentro de un fichero .bat para automatizar todo el proceso:

```
cd \Android\android-sdk\platform-tools
adb shell setprop service.adb.tcp.port 5555
adb tcpip 5555
pause
adb shell getprop | findstr ipaddress>ip_debug.txt
for /F "tokens=2 delims=: " %%i in (ip_debug.txt) do set ip=%%i
echo %ip%
pause
adb connect %ip:~1,13%
del ip_debug.txt
```

Figura 100. Captura del script creado para la automatización del proceso de depuración wifi

Este script realiza lo siguiente:

-Nos situamos sobre el directorio \Android\android-sdk\platform-tools\
-Establecemos el tipo de conexión tcp a través del puerto 5555.
-Esperamos a que el usuario acepte la conexión en su dispositivo.
-Volcamos al fichero ip_debug.txt la IP del dispositivo dentro de la LAN.
-Leemos el valor de la IP en el fichero ip_debug.txt y lo asignamos a la variable ip.
-Esperamos a que el usuario desconecte el cable USB.
-Usamos la variable ip para conectarnos a la IP del dispositivo por medio del comando adb.
-Eliminamos el fichero ip_debug.txt

7.5.3. Depuración de la base de datos

La base de datos local creada en el dispositivo también requiere de un proceso de depuración, ya sea para consultar los registros que se añaden o se eliminan como para modificar los campos de las tablas. Este proceso se lleva a cabo mediante el DDMS (Dalvik Debug Monitor Service), una utilidad de depuración integrada en Eclipse. A través de esta utilidad, en la pestaña File Explorer podemos acceder a los directorios y ficheros del dispositivo conectado. En nuestro caso nos interesan los archivos .sqlite de la carpeta /comercioalmeria/sqldatabases/:

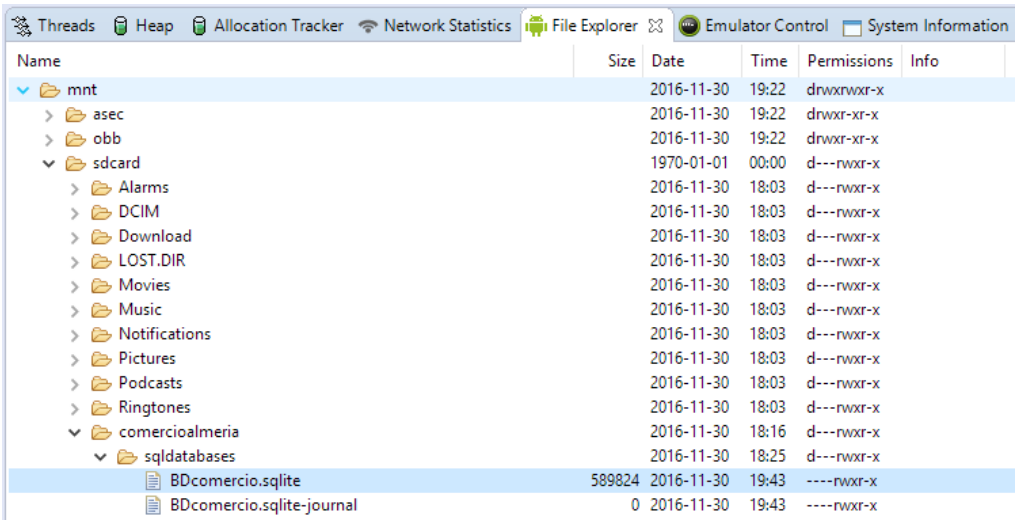

Figura 101. Captura del explorador de archivos del DDMS de Eclipse

La depuración de la bases de datos local en este fichero se ha realizado mediante la herramienta SQLite Manger. SQLite Manager es un plugin para el navegador Mozilla Firefox, que permite la administración y consulta de bases de datos SQLite.

Podremos crear y modificar tablas, realizar consultas y ejecutar sentencias SQL. En la siguiente captura se muestra una consulta del contenido de la tabla “negocios”:

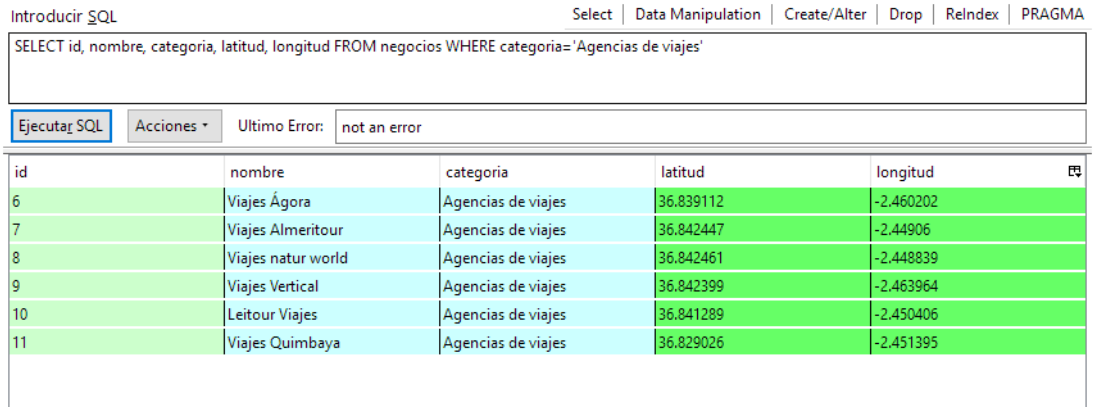

Figura 102. Captura de la herramienta SQLite Manager para administrar bases de datos locales

7.5.4. Android Lint

Android Lint es una herramienta que se ha utilizado para analizar nuestro código en busca de posibles errores comunes.
Algunos de ejemplos de estos errores son:

- Traducciones que faltan y traducciones sin usar.
- Recursos no utilizados.
- Problemas con los iconos (como no considerar densidades, iconos duplicados, tamaños erróneos, etc.).
- Problemas de usabilidad (como no especificar un tipo de entrada en un campo de texto).
- Errores en el Manifest.
- Problemas de accesibilidad e internacionalización (olvidar contentDescription, strings incrustados en el código, etc.).

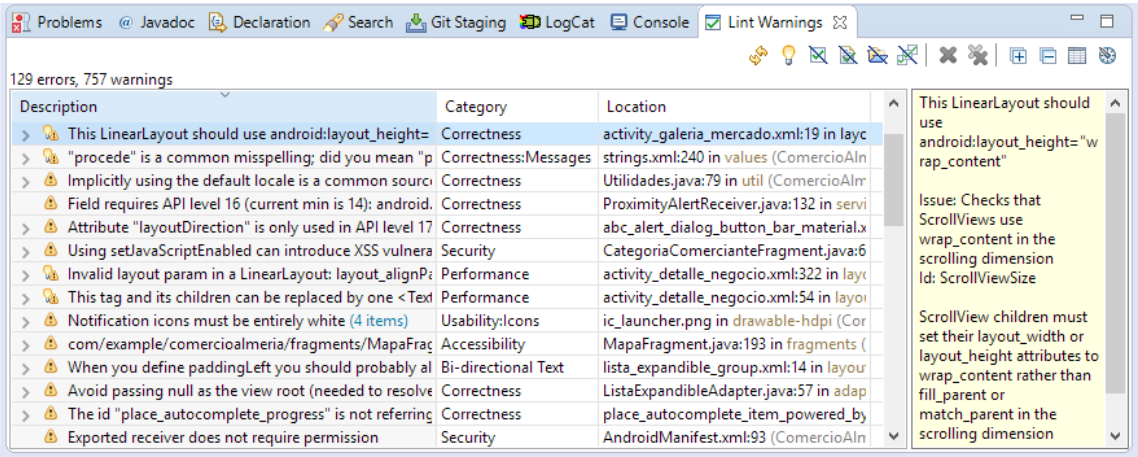

Figura 103. Captura de Android Lint para la búsqueda de posibles errores

7.6. Errores

A lo largo del proceso de depuración se han detectado multitud de errores que resultaban en detenciones de la aplicación o en problemas de funcionalidad, algunos relacionados con la codificación y otros debidos a bugs del propio Android. Los errores más destacables son los siguientes:

7.6.1. Errores de codificación

-Errores con el teclado de Android: Este error se producía al navegar entre diferentes pantallas de la aplicación sin haber cerrado previamente el teclado de Android, que podía haberse abierto por ejemplo para introducir el nombre de un negocio en el SearchView. Se soluciona mediante el uso del método

cerrarTeclado() y llamándolo desde el método onDestroyView() del Fragment para así cerrar el teclado cuando las vistas del Fragment se destruyan.

```
//Para cerrar el teclado de Android si está abierto
private void cerrarTeclado(){
    InputMethodManager input = (InputMethodManager) getActivity()
            .getSystemService(Activity.INPUT_METHOD_SERVICE);
    input.hideSoftInputFromWindow(this.getView().getWindowToken(), 0);
}
```

Figura 104. Captura del método cerrarTeclado() para cerrar el teclado de Android

-Errores con el Spinner del Mapa: el Spinner detectaba la selección errónea de una categoría si por ejemplo el usuario realizaba la búsqueda de un negocio en el SearchView. Puesto que cuando se ejecuta esta acción le asignamos al Spinner la categoría del negocio buscado, lo que ocurría es que el Spinner detectaba que se estaba seleccionando dicha categoría y se cargaban todos los negocios de ésta en lugar de sólo el negocio buscado.
Para la resolución de este error se ha sobrescrito el método onTouch() de la clase Spinner y se ha hecho uso del booleano spinnerTouched:

```
spinner.setOnTouchListener(new View.OnTouchListener() {
    //Para cargar los negocios solamente cuando el spinner se haya tocado y no
    //cuando asignamos al spinner la categoria del negocio buscado
    @Override
    public boolean onTouch(View v, MotionEvent event) {
        spinnerTouched = true;
        return false;
    }
});
```

Figura 105. Captura del método onTouch() de Spinner

-Errores con el SearchView del Mapa: El SearchView del mapa ha dado lugar a errores de distinto tipo, como por ejemplo, permanecer abierto o retener el foco de manera no deseada. Estos problemas se solucionaron con el uso de métodos de la clase SearchView cómo *setFocusableInTouchMode(false)* o *onActionViewCollapsed().*

-Errores en la visualización de imágenes: para la visualización de imágenes ha sido necesaria todo un proceso de análisis de propiedades de ImageViews o GridViews. Dependiendo de los parámetros asignados a dichas vistas, las imágenes se mostraban con tamaños y escalas inapropiadas. Algunas de las estrategias llevadas a cabo para evitar esto, han sido aplicar anchos de columna fijos a algunos de los GridView o el uso de propiedades de ImageView como *adjustViewBounds="true"* y *scaleType="fitCenter"*.

-Errores en la carga de galerías de imágenes: la carga de galerías de imágenes resultaba en errores de detención del programa por falta de RAM. Esto se producía porque dichas imágenes alojadas en el servidor poseían unas resoluciones y tamaños inadecuados. De modo que cuando la librería Picasso

intentaba cargar muchas imágenes al mismo tiempo, el LogCat mostraba advertencias del uso de RAM y la aplicación se detenía.
Se solucionó cambiando la resolución a dichas imágenes para que ocuparan menor tamaño, intentando mantener una calidad de imagen adecuada a los distintos dispositivos que puedan visualizarlas. También se estableció una política de memoria para el uso de Picasso, en la cual se especifica no guardar en la caché de la aplicación las imágenes menos importantes y más pesadas.

7.6.2. Bugs

-Errores con el servicio de notificaciones: por algún motivo, el BroadcastReceiver que detecta la activación del GPS es lanzado dos veces. Es decir, cuando el GPS era activado o desactivado, Android informaba al resto de aplicaciones de dicha acción enviando dos Intents en lugar de uno. Esto se traducía en que el servicio de localización que añade las alertas era iniciado dos veces y las alertas se agregaban erróneamente de manera duplicada. Para resolver este problema se incorporó al código el booleano GPSOn, al que se le asigna el valor true o false si el GPS está encendido o no. El siguiente código muestra el uso de GPSOn en un condicional para lanzar el servicio LocationService una única vez, en concepto de inicialización:

```
if (intent.getAction().matches("android.location.PROVIDERS_CHANGED")) {
    LocationManager manager = (LocationManager) context.getSystemService(Context.LOCATION_SERVICE );
    if( manager.isProviderEnabled(LocationManager.GPS_PROVIDER) && (GPSOn == false)){
        GPSOn = true;
        Intent pushIntent = new Intent(context, LocationService.class);
        context.startService(pushIntent);
    }
```

Figura 106. Captura del booleano GPSOn del servicio de localización

-Errores del mapa: Tras la incorporación de la librería Android-support-v7-appcompat, el mapa junto con los markers presentes desaparecía y se volvía negro al realizar el remplazamiento de fragments. Este es un bug conocido que ocurría cuando se pretendía remplazar el fragment del mapa por cualquier otro, pulsando una opción del menú. Fue solventado añadiendo un efecto de transición en dicho remplazamiento, que tiene lugar en el método selectItem() del MainActivity.java. Mediante el uso del método setTransition(FragmentTransaction.TRANSIT_FRAGMENT_FADE) conseguimos que el mapa no se desvanezca:

```
Fragment fActual = getFragmentManager().findFragmentById(R.id.content_frame);
//TRANSIT_FRAGMENT_FADE soluciona el problema del mapa en negro al hacer el replace
if(fActual instanceof MapaFragment) {
    getFragmentManager().beginTransaction().replace(R.id.content_frame, fragment, "MainActivity")
        .setTransition(FragmentTransaction.TRANSIT_FRAGMENT_FADE).commit();
```

Figura 107. Captura de setTransition() para evitar la desaparición del mapa

7.8. Implantación

Para distribuir la aplicación a los terminales Android, ya sea mediante Google Play o de otra manera, debemos exportar el proyecto de Eclipse a un archivo .apk. Este formato es una variante del formato JAR de Java y se usa para distribuir e instalar componentes empaquetados para la plataforma Android. Básicamente es un archivo comprimido ZIP con diferente extensión por lo cual pueden ser abiertos e inspeccionados usando un software como 7-Zip, Winzip, WinRAR o Ark. El tipo MIME definido para .apk es *application/vnd.android.package-archive*.

Un archivo .apk normalmente contiene lo siguiente:

- AndroidManifest.xml
- classes.dex
- resources.arsc
- res (carpeta)
- META-INF (carpeta)
- lib (carpeta)

Para generar un paquete .apk que esté firmado correctamente se han seguido los siguientes pasos:

1. Seleccionamos la opción Exportar Aplicación Android del menú de Eclipse y el proyecto que nuestra aplicación.
2. Introducimos un nombre para el almacén de claves, por ejemplo "claves_android" y seleccionamos la ubicación donde se guardarán los ficheros de clave.
3. Establecemos la contraseña para el almacén de claves. Si es la primera vez que generamos un certificado marcamos la opción "Create new keystore", y si ya disponemos de un almacén de claves seleccionamos "Use existing keystore":
4. Introducimos los datos de la clave que crearemos para certificar nuestra aplicación.
5. A continuación se indica la carpeta y el nombre del paquete APK compilado y certificado, que será el fichero que posteriormente se subirá y publicará en Android Market, si lo deseamos.

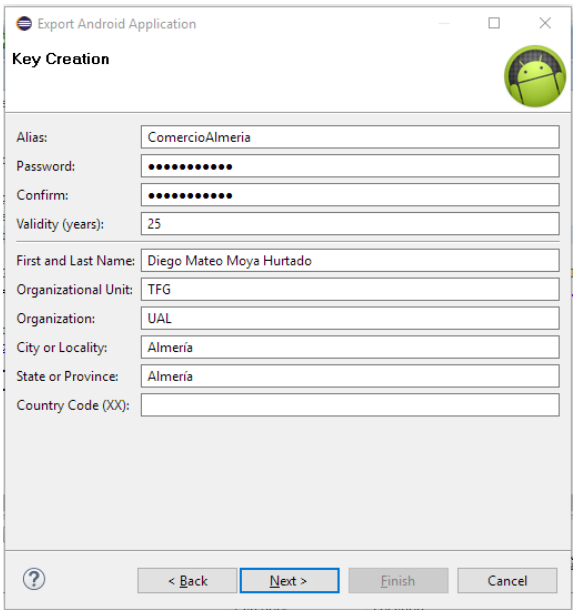

Figura 108. Ejemplo de creación de una clave de certificación para la app

Si lo deseamos, también podemos hacer uso de la herramienta Proguard antes de generar la .apk. Nos permite ofuscar, optimizar y reducir el código en tiempo de compilación. Una de las funciones que realiza, precisamente, es detectar los fragmentos de código que no se utilizan gracias al recorrido que hace en árbol del código, de forma que todo el código no alcanzado (o innecesario) será extraido del .apk. Además, nos añadirá funciones de seguridad, pues cambiará el nombre de campos, clases e interfaces, haciendo que el código sea más ligero e ilegible ante ingeniería inversa.

7.9. Mejoras

Son muchas las mejoras que se le pueden incluir a nuestra aplicación para hacerla más competitiva. Algunas de estas mejoras son las siguientes:

-Uso de base de datos geo-espaciales para el almacenamiento de los negocios y de sus coordenadas.
-Búsqueda de los negocios por zonas y barrios.
-Realidad Aumentada para visualizar la localización de los negocios a través de la cámara del dispositivo.
-Inicio de sesión y registro de usuarios.
-Permitir al comerciante incluir ofertas a modo publicitario.
-Buscador de ofertas de los negocios.
-Notificaciones de ofertas.
-Permitir a los usuarios puntuar y escribir comentarios de un negocio concreto.
-Ordenar la lista de negocios por nombre, categoría, distancia, valoración…
-Mejorar el estilo de la aplicación por medio de la librería Material Design.

-Ocultar el Toolbar en pantallas menores a 7” al hacer scroll hacia abajo en determinados layouts.
-Hacer que la aplicación sea compatible con dispositivos que usan diferente sistema operativo, como iOS o Windows Phone.
-Incluir otros idiomas como el inglés y una opción en Ajustes que permita cambiar entre uno y otro.

Capítulo 8:
Conclusión

8. CONCLUSIÓN

A lo largo de todo este trabajo se han analizado distintas herramientas y procedimientos para el desarrollo de una aplicación en Android. Nuestra app es un instrumento de gran utilidad para la búsqueda de negocios de la región, así como fácil de usar. Favorece el comercio local en contraposición a muchas otras aplicaciones de este tipo, permitiendo al usuario obtener información sobre empresas de la provincia, mercados locales o incluso ofrecer consejo a comerciantes emprendedores.
Los ajustes incluidos permiten al usuario configurar a su gusto muchos de los aspectos de la app, haciendo de ella una herramienta muy versátil. Otro aspecto no menos importante es que ofrece un buen rendimiento y adaptabilidad a todo tipo de pantallas.

Por contrapartida, el desarrollo de esta aplicación ha sido una tarea exigente. La programación de aplicaciones, por sencillas que sean, requieren de mucho tiempo y esfuerzo. Por ello, ciertas funcionalidades que se tenían en mente no se han podido incluir. Algunas de estas funcionalidades aparecen reflejadas en el apartado de "Mejoras", como la visualización de ofertas mediante un buscador o permitir a los usuarios escribir valoraciones y puntuar un negocio en concreto.
Sin embargo, todas estas funciones requieren de un previo sistema de inicio de sesión y registro, el cual no está implementado en la página web original. Es por ello que esta difícil tarea que engloba desde un diseño de bases de datos hasta la implementación de nuevos menús, no ha llegado a realizarse.

En conclusión, podríamos decir que el desarrollo de este proyecto ha servido al estudiante para desarrollar sus habilidades como programador de Android. Además, se ha cumplido con el objetivo principal que era el de adaptar una página web ya existente a una aplicación, así como incluir opciones que hicieran de ella más competitiva en comparación con otras aplicaciones de este tipo que podemos encontrar en el mercado.

Capítulo 9:
Referencias

9. REFERENCIAS

Versión web de Comercio Almería: http://150.214.150.173/index.html

Hostinger: https://cpanel.hostinger.es/index/index

Stackoverflow: http://stackoverflow.com/

Actualizar el IDE y las herramientas del SDK: https://developer.android.com/studio/intro/update.html

Versiones de la plataforma de Android: https://developer.android.com/about/dashboards/index.html

Compatibilidad con diferentes pantallas:
https://developer.android.com/guide/practices/screens_support.html

Notificaciones: https://developer.android.com/guide/topics/ui/notifiers/notifications.html

Objetos de Mapas: https://developers.google.com/maps/documentation/android-api/map?hl=es

Controles y gestos del mapa: https://developers.google.com/maps/documentation/android-api/controls

Configuración de proyectos:
https://developers.google.com/maps/documentation/android-api/config#specify_android_permissions

LocationManager:
https://developer.android.com/reference/android/location/LocationManager.html

GPS Coordinates - Latitude and Longitude:
http://www.gps-coordinates.net/

Geolocation Techniques: Principles and Applications:
https://books.google.es/books?id=tvCJLFLRwzcC&pg=PA17&dq=geolocation&hl=es&sa=X&ved=0ahUKEwi05ZDsj8TQAhUDuBQKHXdqD98Q6AEIOTAE#v=onepage&q=geolocation&f=false

Capítulo 10:
Anexo A

10. ANEXO A

En este anexo se detallan cada uno de los requisitos funcionales y no funcionales enumerados en los apartados 5.8. Lista de requisitos funcionales y 5.9. Lista de requisitos no funcionales.

10.1. Requisitos funcionales

<table>
<tr><td>RF-01</td><td>Actualizar negocios</td></tr>
<tr><td>Versión</td><td>1.0</td></tr>
<tr><td>Dependencias</td><td>- Dispositivo compatible
- Conexión a Internet
- Bases de datos disponibles</td></tr>
<tr><td>Descripción</td><td>El sistema actualizará los negocios de la base de datos local con los negocios de la base de datos del servidor.</td></tr>
<tr><td>Precondición</td><td>El usuario inicializa la aplicación</td></tr>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><th>Paso</th><th>Acción</th></tr>
<tr><td>1</td><td>El usuario inicia la aplicación</td></tr>
<tr><td>2</td><td>El sistema muestra al usuario un proceso de carga</td></tr>
<tr><td>3</td><td>El sistema comprueba si el dispositivo dispone de conexión a Internet</td></tr>
<tr><td>4</td><td>El sistema obtiene los negocios de la base de datos remota</td></tr>
<tr><td>5</td><td>El sistema inserta los negocios en la base de datos local</td></tr>
</table>
</td></tr>
<tr><td>Postcondición</td><td>La base de datos local es actualizada</td></tr>
<tr><td>Excepciones</td><td>P1 - El usuario no tiene un dispositivo compatible con la aplicación o no tiene instalados los servicios de Google Play.
P4 - Sin conexión a Internet no se accederá al servidor
P5 - La base de datos no ha sido creada</td></tr>
<tr><td>Importancia</td><td>Media</td></tr>
<tr><td>Comentarios</td><td>Ninguno</td></tr>
</table>

Tabla 7. Especificación del Requisito Funcional 1

RF-02	Comprobar tablas

<table>
<tr><td>Versión</td><td>1.0</td></tr>
<tr><td>Dependencias</td><td>- RF-01</td></tr>
<tr><td>Descripción</td><td>El sistema comprobará las tablas de la base de datos local para eliminar algún registro correspondiente a negocios eliminados del servidor</td></tr>
<tr><td>Precondición</td><td>El usuario inicializa la aplicación</td></tr>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><th>Paso</th><th>Acción</th></tr>
<tr><td>1</td><td>El sistema comprueba si las tablas Historial, Favoritos y Notificaciones tienen registros con id de negocio que no está en la tabla Negocios</td></tr>
<tr><td>2</td><td>El sistema elimina los registros correspondientes</td></tr>
</table>
</td></tr>
<tr><td>Postcondición</td><td>La base de datos local es actualizada</td></tr>
<tr><td>Excepciones</td><td>Ninguna</td></tr>
<tr><td>Importancia</td><td>Alta</td></tr>
<tr><td>Comentarios</td><td>Ninguno</td></tr>
</table>

Tabla 8. Especificación del Requisito Funcional 2

<table>
<tr><td>RF-03</td><td>Cargar mapa</td></tr>
<tr><td>Versión</td><td>1.0</td></tr>
<tr><td>Dependencias</td><td>- Dispositivo compatible
- Conexión a Internet
- Google Play Services instalado
- Base de datos local disponible
- RF-41</td></tr>
<tr><td>Descripción</td><td>El sistema cargará los negocios de la base de datos local y los situará en el mapa</td></tr>
<tr><td>Precondición</td><td>El usuario selecciona el mapa del menú o inicializa la aplicación</td></tr>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><th>Paso</th><th>Acción</th></tr>
<tr><td>1</td><td>El sistema carga el mapa del tipo seleccionado en Ajustes por el usuario</td></tr>
<tr><td>2</td><td>El sistema obtiene los negocios de la base de datos local</td></tr>
<tr><td>3</td><td>El sistema añade los negocios al mapa</td></tr>
</table>
</td></tr>
</table>

Postcondición	Se carga el mapa y se añaden los negocios
Excepciones	P1 - El mapa puede no cargarse sin conexión a Internet
Importancia	Muy Alta
Comentarios	Ninguno

Tabla 9. Especificación del Requisito Funcional 3

RF-04	Mostrar contador de negocios
Versión	1.0
Dependencias	- RF-03
Descripción	El sistema mostrará un contador con los negocios visibles en el mapa
Precondición	El usuario selecciona el mapa del menú
Flujo de eventos	Paso: 1 — Acción: El usuario selecciona el mapa del menú Paso: 2 — Acción: El sistema obtiene el número de negocios visibles del mapa Paso: 3 — Acción: El sistema actualiza el contador
Postcondición	Se añaden los negocios a la lista
Excepciones	Ninguna
Importancia	Baja
Comentarios	Ninguno

Tabla 10. Especificación del Requisito Funcional 4

RF-05	Actualizar contador de negocios
Versión	1.0
Dependencias	- RF-03 - RF-04
Descripción	El sistema mostrará un contador con los negocios visibles en el mapa
Precondición	El usuario desplaza la posición de la cámara del mapa

Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario desplaza la posición de la cámara del mapa</td></tr><tr><td>2</td><td>El sistema obtiene el número de negocios visibles del mapa</td></tr><tr><td>3</td><td>El sistema actualiza el contador</td></tr></table>
Postcondición	Se añaden los negocios a la lista
Excepciones	Ninguna
Importancia	Baja
Comentarios	Ninguno

Tabla 11. Especificación del Requisito Funcional 5

RF-06	Generar lista de negocios
Versión	1.0
Dependencias	-RF-03 - Conexión a Internet
Descripción	El sistema mostrará los negocios del mapa a través de una lista
Precondición	El usuario selecciona la lista de negocios del mapa
Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario selecciona la lista de negocios del mapa</td></tr><tr><td>2</td><td>El sistema obtiene los negocios visibles del mapa</td></tr><tr><td>3</td><td>El sistema genera la lista con los negocios</td></tr></table>
Postcondición	Se añaden los negocios a la lista
Excepciones	P3 - Sin conexión a Internet no se descargarán las imágenes del servidor
Importancia	Alta
Comentarios	Ninguno

Tabla 12. Especificación del Requisito Funcional 6

RF-07	Cargar negocios por categoría

<table>
<tr><td>Versión</td><td>1.0</td></tr>
<tr><td>Dependencias</td><td>-RF-03</td></tr>
<tr><td>Descripción</td><td>El sistema mostrará los negocios asociados a la categoría seleccionada</td></tr>
<tr><td>Precondición</td><td>El usuario selecciona una categoría en la lista desplegable</td></tr>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><th>Paso</th><th>Acción</th></tr>
<tr><td>1</td><td>El usuario selecciona una categoría en la lista desplegable</td></tr>
<tr><td>2</td><td>Se obtienen los negocios de la base de datos local en base a la categoría</td></tr>
<tr><td>3</td><td>El sistema limpia el mapa de negocios</td></tr>
<tr><td>4</td><td>El sistema añade los nuevos negocios</td></tr>
<tr><td>5</td><td>El sistema actualiza el contador de negocios visibles</td></tr>
<tr><td>6</td><td>El sistema genera la lista de negocios si la vista actual es la lista</td></tr>
</table>
</td></tr>
<tr><td>Postcondición</td><td>Se añaden los negocios al mapa y puede que a la lista</td></tr>
<tr><td>Excepciones</td><td>P6 - La lista no es generada si la vista actual es el mapa</td></tr>
<tr><td>Importancia</td><td>Alta</td></tr>
<tr><td>Comentarios</td><td>Ninguno</td></tr>
</table>

Tabla 13. Especificación del Requisito Funcional 7

RF-08	Buscar negocio por nombre
Versión	1.0
Dependencias	- RF-03
Descripción	El sistema mostrará el negocio buscado en el campo de búsqueda
Precondición	- El usuario introduce un nombre en el campo de búsqueda

<table>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><th>Paso</th><th>Acción</th></tr>
<tr><td>1</td><td>El usuario introduce el nombre del negocio en el campo de búsqueda</td></tr>
<tr><td>2</td><td>El sistema muestra una lista de sugerencias de la base de datos local en base al nombre</td></tr>
<tr><td>3</td><td>El usuario pulsa Intro</td></tr>
<tr><td>4</td><td>El sistema obtiene el negocio de la base de datos local</td></tr>
<tr><td>5</td><td>El sistema limpia el mapa de negocios</td></tr>
<tr><td>6</td><td>El sistema añade el nuevo negocio</td></tr>
<tr><td>7</td><td>El sistema actualiza el contador de negocios visibles</td></tr>
<tr><td>8</td><td>El sistema genera la lista con el negocio si la vista actual es la lista</td></tr>
</table>
</td></tr>
<tr><td>Postcondición</td><td>Se añade el negocio al mapa</td></tr>
<tr><td>Excepciones</td><td>P2 - El texto introducido no coincide con el nombre de ningún negocio de la base de datos
P4 - No existe ningún negocio con el nombre introducido
P6 - No hay negocio que añadir</td></tr>
<tr><td>Importancia</td><td>Alta</td></tr>
<tr><td>Comentarios</td><td>El usuario puede seleccionar un elemento de la lista de sugerencias en lugar de hacer Intro. Véase RF-09 para analizar este requisito.</td></tr>
</table>

Tabla 14. Especificación del Requisito Funcional 8

RF-09	Mostrar negocio de la lista de sugerencias
Versión	1.0
Dependencias	- RF-03
Descripción	El sistema mostrará el negocio pulsado de la lista de sugerencias
Precondición	- El usuario introduce un nombre en el campo de búsqueda

<table>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><th>Paso</th><th>Acción</th></tr>
<tr><td>1</td><td>El usuario introduce un nombre en el campo de búsqueda</td></tr>
<tr><td>2</td><td>El sistema muestra una lista de sugerencias de la base de datos local en base al nombre</td></tr>
<tr><td>3</td><td>El usuario pulsa sobre un elemento de la lista</td></tr>
<tr><td>4</td><td>El sistema obtiene el negocio de la base de datos local</td></tr>
<tr><td>5</td><td>El sistema limpia el mapa de negocios</td></tr>
<tr><td>6</td><td>El sistema añade el nuevo negocio</td></tr>
<tr><td>7</td><td>El sistema actualiza el contador de negocios visibles</td></tr>
<tr><td>8</td><td>El sistema genera la lista con el negocio si la vista actual es la lista</td></tr>
</table>
</td></tr>
<tr><td>Postcondición</td><td>Se añade el negocio al mapa</td></tr>
<tr><td>Excepciones</td><td>P2 - El texto introducido no coincide con el nombre de ningún negocio de la base de datos</td></tr>
<tr><td>Importancia</td><td>Alta</td></tr>
<tr><td>Comentarios</td><td>El usuario puede presionar Intro en lugar de seleccionar un elemento de la lista. Véase RF-08 para analizar este requisito.</td></tr>
</table>

Tabla 15. Especificación del Requisito Funcional 9

<table>
<tr><td>RF-10</td><td>Mostrar información del negocio</td></tr>
<tr><td>Versión</td><td>1.0</td></tr>
<tr><td>Dependencias</td><td>- RF-03</td></tr>
<tr><td>Descripción</td><td>El sistema mostrará una ventana emergente con el nombre y dirección del negocio clicado</td></tr>
<tr><td>Precondición</td><td>El usuario pulsa sobre el marcador de un negocio del mapa</td></tr>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><th>Paso</th><th>Acción</th></tr>
<tr><td>1</td><td>El usuario pulsa sobre el marcador de un negocio del mapa</td></tr>
<tr><td>2</td><td>El sistema muestra una ventana emergente con la información del negocio</td></tr>
</table>
</td></tr>
<tr><td>Postcondición</td><td>Se muestra una ventana emergente con la información del negocio</td></tr>
<tr><td>Excepciones</td><td>Ninguna</td></tr>
</table>

Importancia	Alta
Comentarios	Ninguno

Tabla 16. Especificación del Requisito Funcional 10

RF-11	Ver detalles del negocio del mapa
Versión	1.0
Dependencias	- RF-10
Descripción	El sistema mostrará la información detallada del negocio
Precondición	El usuario pulsa sobre la ventana emergente con la información del negocio en la vista del mapa
Flujo de eventos	Paso: 1 — El usuario pulsa sobre la ventana emergente con la información del negocio en la vista del mapa Paso: 2 — El sistema muestra toda la información sobre el negocio
Postcondición	Se muestra la información detallada del negocio
Excepciones	P2 - Sin conexión a Internet no se descargarán las imágenes del servidor
Importancia	Alta
Comentarios	El usuario también puede mostrar la información detallada del negocio pulsando sobre un elemento de la lista en la vista de lista, pulsando sobre un negocio de las secciones de favoritos e historial o pulsando sobre la opción de ver negocio del menú emergente de favoritos. Para analizar estos casos véanse RF-12, RF-13 y RF-14

Tabla 17. Especificación del Requisito Funcional 11

RF-12	Ver detalles del negocio de la lista
Versión	1.0
Dependencias	- RF-06
Descripción	El sistema mostrará la información detallada del negocio
Precondición	El usuario pulsa sobre un negocio de la lista del mapa

Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario pulsa sobre un negocio de la lista del mapa</td></tr><tr><td>2</td><td>El sistema muestra toda la información sobre el negocio</td></tr></table>
Postcondición	Se muestra la información detallada del negocio
Excepciones	P2 - Sin conexión a Internet no se descargarán las imágenes del servidor
Importancia	Alta
Comentarios	El usuario también puede mostrar la información detallada del negocio pulsando sobre la ventana de información del mapa, pulsando sobre un negocio de las secciones de favoritos e historial o pulsando sobre la opción de ver negocio del menú emergente de favoritos. Para analizar estos casos véanse RF-11, RF-13, RF-14 y RF-15

Tabla 18. Especificación del Requisito Funcional 12

RF-13	Ver detalles del negocio del historial
Versión	1.0
Dependencias	- Conexión a Internet - RF-20 - RF-21
Descripción	El sistema mostrará la información detallada del negocio
Precondición	El usuario pulsa sobre un negocio de la sección del historial
Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario pulsa sobre un negocio de la sección del historial</td></tr><tr><td>2</td><td>El sistema muestra toda la información sobre el negocio</td></tr></table>
Postcondición	Se muestra la información detallada del negocio
Excepciones	P2 - Sin conexión a Internet no se descargarán las imágenes del servidor
Importancia	Alta
Comentarios	El usuario también puede mostrar la información detallada del negocio pulsando sobre un elemento de la lista en la vista de lista, sobre la ventana de información en el mapa o pulsando sobre un negocio de la sección de favoritos o pulsando sobre la opción de ver negocio del menú emergente de favoritos. Para analizar estos casos véanse RF-11, RF-12, RF-14 y RF-15

Tabla 19. Especificación del Requisito Funcional 13

<table>
<tr><td>RF-14</td><td>Ver detalles del negocio de favoritos</td></tr>
<tr><td>Versión</td><td>1.0</td></tr>
<tr><td>Dependencias</td><td>- Conexión a Internet
- RF-16
- RF-19</td></tr>
<tr><td>Descripción</td><td>El sistema mostrará la información detallada del negocio</td></tr>
<tr><td>Precondición</td><td>El usuario pulsa sobre un negocio de la lista de la sección de favoritos</td></tr>
<tr><td>Flujo de eventos</td><td><table><tr><td>Paso</td><td>Acción</td></tr><tr><td>1</td><td>El usuario pulsa sobre un negocio de la sección de favoritos</td></tr><tr><td>2</td><td>El sistema muestra toda la información sobre el negocio</td></tr></table></td></tr>
<tr><td>Postcondición</td><td>Se muestra la información detallada del negocio</td></tr>
<tr><td>Excepciones</td><td>P2 - Sin conexión a Internet no se descargarán las imágenes del servidor</td></tr>
<tr><td>Importancia</td><td>Alta</td></tr>
<tr><td>Comentarios</td><td>El usuario también puede mostrar la información detallada del negocio pulsando sobre un negocio de la lista en la vista de lista, sobre la ventana de información en el mapa, pulsando sobre un negocio de la sección de historial o pulsando sobre la opción de ver negocio del menú emergente de favoritos. Para analizar estos casos véanse RF-11, RF-12, RF-13 y RF-15</td></tr>
</table>

Tabla 20. Especificación del Requisito Funcional 14

RF-15	Ver detalles del negocio del menú de favoritos
Versión	1.0
Dependencias	- Conexión a Internet - RF-14
Descripción	El sistema mostrará la información detallada del negocio
Precondición	El usuario pulsa sobre la opción de ver negocio en el menú de favoritos

Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario pulsa sobre un negocio de la lista de la vista de lista del mapa</td></tr><tr><td>2</td><td>El sistema muestra toda la información sobre el negocio</td></tr></table>
Postcondición	Se muestra la información detallada del negocio
Excepciones	P2 - Sin conexión a Internet no se descargarán las imágenes del servidor
Importancia	Alta
Comentarios	El usuario también puede mostrar la información detallada del negocio pulsando sobre un negocio de la lista en la vista de lista, sobre la ventana de información en la vista de mapa, pulsando sobre un negocio de la sección de historial o pulsando sobre un negocio de la sección de favoritos. Para analizar estos casos véanse RF-11, RF-12, RF-13 y RF-14

Tabla 21. Especificación del Requisito Funcional 15

RF-16	Añadir negocio a favoritos
Versión	1.0
Dependencias	- RF-11 / RF-12 / RF-13
Descripción	El sistema añadirá a favoritos el negocio seleccionado
Precondición	El usuario indica que quiere añadir un negocio a favoritos
Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario indica que quiere añadir un negocio a favoritos</td></tr><tr><td>2</td><td>El sistema inserta el negocio en la tabla Favoritos</td></tr></table>
Postcondición	Se añade el negocio a favoritos
Excepciones	Ninguna
Importancia	Alta
Comentarios	Ninguno

Tabla 22. Especificación del Requisito Funcional 16

RF-17	Borrar favorito

Versión	1.0
Dependencias	- RF-16
Descripción	El sistema borrará de favoritos el negocio seleccionado
Precondición	El usuario indica que quiere eliminar un favorito
Flujo de eventos	Paso: 1 — El usuario indica que quiere eliminar un favorito Paso: 2 — El sistema borra el negocio de la tabla Favoritos
Postcondición	Se borra el negocio a favoritos y el botón cambia su función
Excepciones	Ninguna
Importancia	Alta
Comentarios	El usuario también puede eliminar el negocio de favoritos a través del menú de la sección “Favoritos”. Véase RF-18

Tabla 23. Especificación del Requisito Funcional 17

RF-18	Borrar favorito del menú de favoritos
Versión	1.0
Dependencias	- RF-16 - RF-19
Descripción	El sistema borrará de favoritos el negocio seleccionado
Precondición	El usuario selecciona la opción de eliminar favorito en el menú de favoritos
Flujo de eventos	Paso: 1 — El usuario selecciona la opción de eliminar favorito en el menú de favoritos Paso: 2 — El sistema borra el negocio de la tabla Favoritos Paso: 3 — El sistema elimina el negocio de la vista de Favoritos
Postcondición	Se borra el negocio a favoritos y se elimina de la vista actual
Excepciones	Ninguna

Importancia	Alta
Comentarios	El usuario también puede eliminar el negocio de favoritos a través de la pantalla de detalles del negocio. Véase RF-16

Tabla 24. Especificación del Requisito Funcional 18

RF-19	Ver favoritos
Versión	1.0
Dependencias	- RF-16
Descripción	El sistema mostrará los negocios de la sección de favoritos
Precondición	El usuario selecciona la opción Favoritos del menú principal
Flujo de eventos	Paso: 1 — El usuario selecciona la opción Favoritos del menú principal Paso: 2 — El sistema obtiene los negocios de la tabla favoritos Paso: 3 — El sistema genera una vista a modo de rejilla con los negocios
Postcondición	Se muestran los negocios favoritos
Excepciones	P2 - La tabla favoritos no tiene ningún registro P3 - La imagen del negocio puede no ser visualizada sin conexión
Importancia	Alta
Comentarios	Ninguno

Tabla 25. Especificación del Requisito Funcional 19

RF-20	Ver historial
Versión	1.0
Dependencias	- RF-11 / RF-12 / RF -13 / RF-14 / RF-15 - RF-21
Descripción	El sistema mostrará los negocios de la sección de historial
Precondición	El usuario selecciona la opción Historial del menú principal

Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario selecciona la opción Historial del menú principal</td></tr><tr><td>2</td><td>El sistema obtiene los negocios de la tabla historial</td></tr><tr><td>3</td><td>El sistema genera una vista a modo de rejilla con los negocios</td></tr></table>
Postcondición	Se muestran los negocios guardados en el historial
Excepciones	P2 - La tabla historial no tiene ningún registro P3 - La imagen del negocio puede no ser visualizada sin conexión
Importancia	Alta
Comentarios	Ninguno

Tabla 26. Especificación del Requisito Funcional 20

RF-21	Añadir negocio al historial
Versión	1.0
Dependencias	- RF-11 / RF-12 / RF -13 / RF-14 / RF-15 - RF-42
Descripción	El sistema guardará un historial de negocios visualizados
Precondición	El usuario visualiza un negocio en la pantalla de detalle de negocio
Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario visualiza un negocio en la pantalla de detalle de negocio</td></tr><tr><td>2</td><td>El sistema añade el negocio a la tabla historial</td></tr></table>
Postcondición	El negocio es añadido al historial
Excepciones	P2 - El negocio ya se encuentra en la tabla / La preferencia “Guardar historial” está deshabilitada
Importancia	Alta
Comentarios	Ninguno

Tabla 27. Especificación del Requisito Funcional 21

RF-22	Trazar ruta hasta el negocio

<table>
<tr><td>Versión</td><td>1.0</td></tr>
<tr><td>Dependencias</td><td>- Conexión a Internet
- Habilitar GPS
- RF-11 / RF-12 / RF -13 / RF-14 / RF-15</td></tr>
<tr><td>Descripción</td><td>El sistema mostrará una ruta en el mapa de detalle del negocio, para llegar desde la ubicación del usuario hasta el negocio</td></tr>
<tr><td>Precondición</td><td>El usuario selecciona la opción “Cómo llegar”</td></tr>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><th>Paso</th><th>Acción</th></tr>
<tr><td>1</td><td>El usuario pulsa sobre el botón Cómo llegar</td></tr>
<tr><td>2</td><td>El sistema muestra un menú emergente con las opciones “A pie” y “En coche”</td></tr>
<tr><td>3</td><td>El usuario selecciona una opción</td></tr>
<tr><td>4</td><td>El sistema muestra un diálogo de acceso a ajustes del GPS</td></tr>
<tr><td>5</td><td>El usuario activa el GPS en Ajustes de ubicación del dispositivo</td></tr>
<tr><td>6</td><td>El sistema muestra la ruta en base a la opción seleccionada en P-3</td></tr>
</table>
</td></tr>
<tr><td>Postcondición</td><td>Se muestra la ruta en el mapa</td></tr>
<tr><td>Excepciones</td><td>P4 - No se muestra el diálogo si el GPS ya está activado
P5 - Puede que el usuario no lo active o que ya esté activado
P6 - No se muestra la ruta sin conexión a Internet ni GPS</td></tr>
<tr><td>Importancia</td><td>Alta</td></tr>
<tr><td>Comentarios</td><td>Ninguno</td></tr>
</table>

Tabla 28. Especificación del Requisito Funcional 22

RF-23	Compartir ubicación del negocio
Versión	1.0
Dependencias	- Aplicación de mensajería instalada - Conexión a Internet - RF-11 / RF-12 / RF -13 / RF-14 / RF-15
Descripción	El sistema enviará un mensaje con la ubicación del negocio

Precondición	El usuario indica que quiere compartir la ubicación de un negocio
Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario indica que quiere compartir la ubicación de un negocio</td></tr><tr><td>2</td><td>El sistema muestra una ventana de selección de aplicaciones para llevar a cabo dicha acción (Whatsapp, correo electrónico…)</td></tr><tr><td>3</td><td>El usuario selecciona una aplicación</td></tr><tr><td>4</td><td>El sistema crea un mensaje con la ubicación en la aplicación seleccionada</td></tr><tr><td>5</td><td>El usuario envía el mensaje</td></tr></table>
Postcondición	Se envía un mensaje con la ubicación del negocio
Excepciones	P3 - El dispositivo no dispone de aplicaciones de mensajería P5 - El mensaje no puede ser enviado por falta de conexión
Importancia	Media
Comentarios	Ninguno

Tabla 29. Especificación del Requisito Funcional 23

RF-24	Notificar negocio
Versión	1.0
Dependencias	- RF-11 / RF-12 / RF -13 / RF-14 / RF-15
Descripción	El sistema añadirá el negocio a notificaciones
Precondición	El usuario indica que quiere añadir un negocio a notificaciones
Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario indica que quiere añadir un negocio a notificaciones</td></tr><tr><td>2</td><td>El sistema añade el negocio a la tabla Notificaciones</td></tr></table>
Postcondición	Se añade el negocio a notificaciones
Excepciones	Ninguna
Importancia	Alta
Comentarios	Ninguno

Tabla 30. Especificación del Requisito Funcional 24

<table>
<tr><td>RF-25</td><td>Dejar de notificar negocio</td></tr>
<tr><td>Versión</td><td>1.0</td></tr>
<tr><td>Dependencias</td><td>- RF-11 / RF-12 / RF -13 / RF-14 / RF-15</td></tr>
<tr><td>Descripción</td><td>El sistema eliminará el negocio de notificaciones</td></tr>
<tr><td>Precondición</td><td>El usuario indica que quiere eliminar un negocio de las notificaciones</td></tr>
<tr><td>Flujo de eventos</td><td><table><tr><td>Paso</td><td>Acción</td></tr><tr><td>1</td><td>El usuario indica que quiere eliminar un negocio de las notificaciones</td></tr><tr><td>2</td><td>El sistema elimina el negocio de la tabla Notificaciones</td></tr></table></td></tr>
<tr><td>Postcondición</td><td>Se elimina el negocio de notificaciones</td></tr>
<tr><td>Excepciones</td><td>Ninguna</td></tr>
<tr><td>Importancia</td><td>Alta</td></tr>
<tr><td>Comentarios</td><td>Ninguno</td></tr>
</table>

Tabla 31. Especificación del Requisito Funcional 25

<table>
<tr><td>RF-26</td><td>Ver galería del negocio</td></tr>
<tr><td>Versión</td><td>1.0</td></tr>
<tr><td>Dependencias</td><td>- Conexión a Internet
- RF-11 / RF-12 / RF -13 / RF-14 / RF-15</td></tr>
<tr><td>Descripción</td><td>El sistema mostrará la galería de imágenes del negocio</td></tr>
<tr><td>Precondición</td><td>El usuario se desplaza a través de la galería del negocio</td></tr>
<tr><td>Flujo de eventos</td><td><table><tr><td>Paso</td><td>Acción</td></tr><tr><td>1</td><td>El usuario se desplaza a través de la galería del negocio</td></tr><tr><td>2</td><td>El sistema carga la imagen solicitada de la galería</td></tr><tr><td>3</td><td>El sistema muestra la imagen de la posición requerida</td></tr></table></td></tr>
<tr><td>Postcondición</td><td>Se muestran las imágenes de la galería del negocio</td></tr>
</table>

Excepciones	P3 - La imagen no puede ser visualizada por falta de conexión o problemas del servidor
Importancia	Media
Comentarios	El usuario también puede ver la galería ampliada. Ver RF-27

Tabla 32. Especificación del Requisito Funcional 26

RF-27	Ver galería ampliada del negocio
Versión	1.0
Dependencias	- RF-26
Descripción	El sistema mostrará la galería de imágenes ampliada del negocio
Precondición	El usuario selecciona una imagen de la galería del negocio
Flujo de eventos	Paso / Acción P1 – El usuario selecciona una imagen de la galería del negocio P2 – El sistema carga la imagen ampliada de la galería P3 – El usuario se desplaza a través de la galería ampliada del negocio P4 – El sistema carga la imagen solicitada de la galería P5 – El sistema muestra la imagen ampliada
Postcondición	Se muestran las imágenes de la galería ampliada del negocio
Excepciones	P2 / P5 - La imagen no puede ser visualizada por falta de conexión o problemas del servidor
Importancia	Media
Comentarios	El usuario también puede ver la galería de imágenes sin ampliar. Ver RF-26

Tabla 33. Especificación del Requisito Funcional 27

RF-28	Mostrar versión web de la aplicación
Versión	1.0

Dependencias	- Dispositivo compatible - Conexión a Internet
Descripción	El sistema mostrará la versión web de la aplicación
Precondición	El usuario pulsa sobre el logotipo de la aplicación situado en el menú
Flujo de eventos	Paso: 1 — El usuario pulsa sobre el logotipo de la aplicación situado en el menú Paso: 2 — El sistema muestra una ventana de selección de navegador para abrir la página Paso: 3 — El usuario selecciona un navegador Paso: 4 — El sistema carga la página web por medio del navegador
Postcondición	Se muestra la versión web de la aplicación
Excepciones	P2 - No hay navegadores instalados / Existe un navegador por defecto P4 - No se dispone de conexión a Internet
Importancia	Baja
Comentarios	Ninguno

Tabla 34. Especificación del Requisito Funcional 28

RF-29	Mostrar noticias
Versión	1.0
Dependencias	- Dispositivo compatible - Conexión a Internet
Descripción	El sistema cargará las noticias presentes en la versión web
Precondición	El usuario selecciona la sección de Noticias del menú
Flujo de eventos	Paso: 1 — El usuario selecciona la sección de Noticias del menú Paso: 2 — El sistema carga las noticias de la web dentro de la aplicación
Postcondición	Se muestran las noticias presentes en la versión web
Excepciones	P2 - No se dispone de conexión a Internet

Importancia	Alta
Comentarios	Ninguno

Tabla 35. Especificación del Requisito Funcional 29

RF-30	Mostrar mercados/mercadillos
Versión	1.0
Dependencias	- Dispositivo compatible - Conexión a Internet
Descripción	El sistema mostrará información e imágenes de los mercados y mercadillos
Precondición	El usuario selecciona la sección Mercados / Mercadillos del menú principal
Flujo de eventos	Paso: Acción 1: El usuario selecciona la sección Mercados / Mercadillos del menú principal 2: El sistema carga una serie de opciones para visualizar "Mercado central", "Otros mercados" y "Mercadillos" 3: El usuario selecciona una opción del P2 4: El sistema muestra información e imágenes en base a la opción elegida en P3
Postcondición	Se muestra información e imágenes de los mercados y mercadillos
Excepciones	P4 - No se dispone de conexión a Internet para cargar las imágenes
Importancia	Media
Comentarios	Ninguno

Tabla 36. Especificación del Requisito Funcional 30

RF-31	Ver galería ampliada de mercados/mercadillos
Versión	1.0
Dependencias	- RF-30
Descripción	El sistema mostrará la galería de imágenes ampliada de los mercados/mercadillos
Precondición	El usuario pulsa sobre una imagen de la galería de mercados / mercadillos

Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario pulsa sobre una imagen de la galería de mercados / mercadillos</td></tr><tr><td>2</td><td>El sistema carga la imagen ampliada de la galería</td></tr><tr><td>3</td><td>El usuario se desplaza a través de la galería ampliada</td></tr><tr><td>4</td><td>El sistema carga la imagen solicitada de la galería</td></tr><tr><td>5</td><td>El sistema muestra la imagen ampliada de la posición requerida</td></tr></table>
Postcondición	Se muestra la galería ampliada de imágenes de mercados/mercadillos
Excepciones	P2 / P5 - La imagen no puede ser visualizada por falta de conexión o problemas del servidor
Importancia	Media
Comentarios	Ninguno

Tabla 37. Especificación del Requisito Funcional 31

RF-32	Mostrar productos locales
Versión	1.0
Dependencias	- Dispositivo compatible - Conexión a Internet - Fichero “empresas.xml”
Descripción	El sistema mostrará los productos locales y empresas que los trabajan
Precondición	El usuario selecciona la opción de Productos locales del menú principal
Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario selecciona la opción de Productos locales del menú principal</td></tr><tr><td>2</td><td>El sistema carga una serie de opciones para visualizar “Alimentación”, “Artesanía” y “Bebidas”</td></tr><tr><td>3</td><td>El usuario selecciona una opción del P2</td></tr><tr><td>4</td><td>El sistema carga los productos y empresas del fichero .xml en base a la opción elegida en P3</td></tr></table>
Postcondición	Se muestran los productos locales y empresas que lo trabajan

Excepciones	P4 - No se dispone de conexión a Internet para cargar las imágenes
Importancia	Alta
Comentarios	Ninguno

Tabla 38. Especificación del Requisito Funcional 32

RF-33	Ver mapa de productos	
Versión	1.0	
Dependencias	- RF-32	
Descripción	El sistema mostrará una imagen ampliada del mapa	
Precondición	El usuario pulsa sobre el mapa de productos	
Flujo de eventos	Paso	Acción
	1	El usuario pulsa sobre el mapa de productos
	2	El sistema muestra la imagen del mapa ampliada
Postcondición	Se muestra la imagen ampliada del mapa de productos	
Excepciones	Ninguna	
Importancia	Baja	
Comentarios	Ninguno	

Tabla 39. Especificación del Requisito Funcional 33

RF-34	Ver detalles de empresa
Versión	1.0
Dependencias	- RF-32
Descripción	El sistema mostrará los detalles de la empresa seleccionada
Precondición	El usuario pulsa sobre una de las empresas

Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario pulsa sobre una de las empresas</td></tr><tr><td>2</td><td>El sistema obtiene los datos de la empresa presentes en el fichero .xml</td></tr><tr><td>3</td><td>El sistema muestra la información de la empresa</td></tr></table>
Postcondición	Se muestra información detallada de una empresa
Excepciones	Ninguna
Importancia	Alta
Comentarios	Ninguno

Tabla 40. Especificación del Requisito Funcional 34

RF-35	Contactar con negocio
Versión	1.0
Dependencias	- Aplicación de mensajería/navegador instalada - Conexión a Internet - RF-11 / RF-12 / RF -13 / RF-14 / RF-15
Descripción	El sistema ejecutará el servicio de contacto indicado
Precondición	El usuario pulsa sobre el teléfono, e-mail, Facebook o twitter del negocio
Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario pulsa sobre el teléfono, e-mail, facebook o twitter del negocio</td></tr><tr><td>2</td><td>El sistema lanza el servicio correspondiente al elemento pulsado en P1</td></tr><tr><td>3</td><td>El usuario contacta con el negocio a través del servicio de P2</td></tr></table>
Postcondición	El usuario contacta con el negocio
Excepciones	P2 - Requiere de las aplicaciones necesarias para lanzar dichos servicios
Importancia	Media
Comentarios	Ninguno

Tabla 41. Especificación del Requisito Funcional 35

<table>
<tr><td>RF-36</td><td>Contactar con empresa</td></tr>
<tr><td>Versión</td><td>1.0</td></tr>
<tr><td>Dependencias</td><td>- RF-34
- Aplicación de mensajería/navegador instalada</td></tr>
<tr><td>Descripción</td><td>El sistema ejecutará el servicio de contacto indicado</td></tr>
<tr><td>Precondición</td><td>El usuario pulsa sobre el teléfono, e-mail, Facebook o twitter de la empresa</td></tr>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><td>Paso</td><td>Acción</td></tr>
<tr><td>1</td><td>El usuario pulsa sobre el teléfono, e-mail, facebook o twitter de la empresa</td></tr>
<tr><td>2</td><td>El sistema lanza el servicio correspondiente al elemento pulsado en P1</td></tr>
<tr><td>3</td><td>El usuario contacta con la empresa a través del servicio de P2</td></tr>
</table>
</td></tr>
<tr><td>Postcondición</td><td>El usuario contacta con la empresa</td></tr>
<tr><td>Excepciones</td><td>P2 - Requiere de las aplicaciones necesarias para lanzar dichos servicios</td></tr>
<tr><td>Importancia</td><td>Media</td></tr>
<tr><td>Comentarios</td><td>Ninguno</td></tr>
</table>

Tabla 42. Especificación del Requisito Funcional 36

RF-37	Ver galería ampliada de empresa
Versión	1.0
Dependencias	- RF-34
Descripción	El sistema mostrará la galería de imágenes ampliada de una empresa
Precondición	El usuario pulsa sobre una imagen de la galería

Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario pulsa sobre una imagen de la galería</td></tr><tr><td>2</td><td>El sistema carga la imagen ampliada de la galería</td></tr><tr><td>3</td><td>El usuario se desplaza a través de la galería ampliada</td></tr><tr><td>4</td><td>El sistema carga la imagen solicitada de la galería</td></tr><tr><td>5</td><td>Se muestra la imagen ampliada de la posición requerida</td></tr></table>
Postcondición	Se muestra la galería ampliada de imágenes de la empresa
Excepciones	P2 / P5 - La imagen no puede ser visualizada por falta de conexión o problemas del servidor
Importancia	Media
Comentarios	Ninguno

Tabla 43. Especificación del Requisito Funcional 37

RF-38	Ver video de empresa
Versión	1.0
Dependencias	- RF-34
Descripción	El sistema reproducirá un video de la empresa
Precondición	Ninguna
Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario pulsa sobre el botón Play del recuadro del video</td></tr><tr><td>2</td><td>El sistema reproduce el video de la empresa</td></tr></table>
Postcondición	Se reproduce el video de la empresa
Excepciones	P2 - El video ha sido eliminado de Youtube
Importancia	Baja
Comentarios	Ninguno

Tabla 44. Especificación del Requisito Funcional 38

<table>
<tr><td>RF-39</td><td>Mostrar ayuda al comerciante</td></tr>
<tr><td>Versión</td><td>1.0</td></tr>
<tr><td>Dependencias</td><td>- Dispositivo compatible
- Conexión a Internet</td></tr>
<tr><td>Descripción</td><td>El sistema mostrará las secciones de Ayuda al comerciante disponibles en la web</td></tr>
<tr><td>Precondición</td><td>El usuario pulsa sobre la opción de zona comerciante del menú principal</td></tr>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><th>Paso</th><th>Acción</th></tr>
<tr><td>1</td><td>El usuario pulsa sobre la opción de zona comerciante del menú principal</td></tr>
<tr><td>2</td><td>El sistema carga una serie de opciones para visualizar “Ayuda al comerciante” y “Emprendedores”</td></tr>
<tr><td>3</td><td>El usuario selecciona una opción del P2</td></tr>
<tr><td>4</td><td>El sistema carga los contenidos de la página web seleccionados en P3</td></tr>
</table>
</td></tr>
<tr><td>Postcondición</td><td>Se muestra información de utilidad para comerciantes y emprendedores</td></tr>
<tr><td>Excepciones</td><td>P4 - No se dispone de conexión a Internet</td></tr>
<tr><td>Importancia</td><td>Media</td></tr>
<tr><td>Comentarios</td><td>Ninguno</td></tr>
</table>

Tabla 45. Especificación del Requisito Funcional 39

RF-40	Ver información sobre la aplicación
Versión	1.0
Dependencias	- Dispositivo compatible
Descripción	El sistema mostrará informativa relativa al uso de la aplicación
Precondición	El usuario pulsa la opción de información del menú principal

<table>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><th>Paso</th><th>Acción</th></tr>
<tr><td>1</td><td>El usuario pulsa la opción de información del menú principal</td></tr>
<tr><td>2</td><td>El sistema carga una serie de opciones para visualizar “Sobre Comercio Almería”, “Preguntas frecuentes”, “Términos y condiciones” y “Contacto”</td></tr>
<tr><td>3</td><td>El usuario selecciona una opción del P2</td></tr>
<tr><td>4</td><td>El sistema carga los contenidos de la opción seleccionada en P3</td></tr>
</table>
</td></tr>
<tr><td>Postcondición</td><td>Se muestra información relativa a la aplicación</td></tr>
<tr><td>Excepciones</td><td>Ninguna</td></tr>
<tr><td>Importancia</td><td>Media</td></tr>
<tr><td>Comentarios</td><td>Ninguno</td></tr>
</table>

Tabla 46. Especificación del Requisito Funcional 40

<table>
<tr><td>RF-41</td><td>Cambiar tipo de mapa</td></tr>
<tr><td>Versión</td><td>1.0</td></tr>
<tr><td>Dependencias</td><td>- Dispositivo compatible</td></tr>
<tr><td>Descripción</td><td>El sistema mostrará el mapa del tipo seleccionado</td></tr>
<tr><td>Precondición</td><td>El usuario cambia el tipo de mapa en la preferencia de Ajustes</td></tr>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><th>Paso</th><th>Acción</th></tr>
<tr><td>1</td><td>El usuario pulsa sobre la preferencia del tipo de mapa</td></tr>
<tr><td>2</td><td>El sistema muestra un cuadro diálogo con los tipo de mapas (Normal, Satélite, Híbrido)</td></tr>
<tr><td>3</td><td>El usuario selecciona un tipo de mapa distinto al actual</td></tr>
<tr><td>4</td><td>El sistema guarda la nueva preferencia</td></tr>
</table>
</td></tr>
<tr><td>Postcondición</td><td>Se cambia el tipo de mapa</td></tr>
<tr><td>Excepciones</td><td>Ninguna</td></tr>
<tr><td>Importancia</td><td>Alta</td></tr>
<tr><td>Comentarios</td><td>Ninguno</td></tr>
</table>

Tabla 47. Especificación del Requisito Funcional 41

<table>
<tr><td>RF-42</td><td>Habilitar/Deshabilitar guardar historial</td></tr>
<tr><td>Versión</td><td>1.0</td></tr>
<tr><td>Dependencias</td><td>- Dispositivo compatible</td></tr>
<tr><td>Descripción</td><td>El sistema almacenará/no almacenará los negocios vistos en la pantalla de detalles del negocio en la tabla historial</td></tr>
<tr><td>Precondición</td><td>El usuario pulsa sobre la preferencia de guardar el historial</td></tr>
<tr><td>Flujo de eventos</td><td><table><tr><td>Paso</td><td>Acción</td></tr><tr><td>1</td><td>El usuario habilita o deshabilita la preferencia de guardar el historial</td></tr><tr><td>2</td><td>El sistema guarda la nueva preferencia</td></tr></table></td></tr>
<tr><td>Postcondición</td><td>Se almacenan o dejan de almacenar los negocios vistos</td></tr>
<tr><td>Excepciones</td><td>Ninguna</td></tr>
<tr><td>Importancia</td><td>Alta</td></tr>
<tr><td>Comentarios</td><td>Ninguno</td></tr>
</table>

Tabla 48. Especificación del Requisito Funcional 42

<table>
<tr><td>RF-43</td><td>Limpiar historial</td></tr>
<tr><td>Versión</td><td>1.0</td></tr>
<tr><td>Dependencias</td><td>- Dispositivo compatible</td></tr>
<tr><td>Descripción</td><td>El sistema eliminará los registros de la tabla historial</td></tr>
<tr><td>Precondición</td><td>El usuario pulsa sobre la preferencia de limpiar el historial</td></tr>
<tr><td>Flujo de eventos</td><td><table><tr><td>Paso</td><td>Acción</td></tr><tr><td>1</td><td>El usuario pulsa la preferencia de limpiar el historial</td></tr><tr><td>2</td><td>El sistema muestra un cuadro de diálogo con las opciones (Cancelar, Aceptar)</td></tr><tr><td>3</td><td>El usuario pulsa Aceptar en el cuadro de diálogo</td></tr><tr><td>4</td><td>El sistema elimina los registros de la tabla historial</td></tr></table></td></tr>
<tr><td>Postcondición</td><td>Se eliminan los negocios vistos de la tabla historial</td></tr>
<tr><td>Excepciones</td><td>P4 - No hay registros que eliminar</td></tr>
</table>

Importancia	Media
Comentarios	Ninguno

Tabla 49. Especificación del Requisito Funcional 43

RF-44	Elegir radio de alertas
Versión	1.0
Dependencias	- Dispositivo compatible
Descripción	El sistema cambiará el radio de proximidad de las alertas
Precondición	El usuario presiona la preferencia del radio de alertas
Flujo de eventos	Paso / Acción 1 / El usuario pulsa la preferencia del radio de alertas 2 / El sistema muestra un cuadro de diálogo con los radios 3 / El usuario selecciona un radio distinto del actual 4 / El sistema guarda la nueva preferencia
Postcondición	Se cambia el radio de proximidad de las alertas
Excepciones	Ninguna
Importancia	Alta
Comentarios	Ninguno

Tabla 50. Especificación del Requisito Funcional 44

RF-45	Elegir frecuencia de alertas
Versión	1.0
Dependencias	- Dispositivo compatible
Descripción	El sistema cambiará la frecuencia con que se notifican las alertas
Precondición	El usuario pulsa la preferencia de frecuencia de notificaciones

Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario pulsa la preferencia de frecuencia de notificaciones</td></tr><tr><td>2</td><td>El sistema muestra un cuadro de diálogo con las frecuencias</td></tr><tr><td>3</td><td>El usuario selecciona un valor distinto del actual</td></tr><tr><td>4</td><td>El sistema guarda la nueva preferencia</td></tr></table>
Postcondición	Se cambia la frecuencia con que se notifican las alertas
Excepciones	Ninguna
Importancia	Alta
Comentarios	Ninguno

Tabla 51. Especificación del Requisito Funcional 45

RF-46	Habilitar alertas de favoritos
Versión	1.0
Dependencias	- Dispositivo compatible
Descripción	El sistema enviará/no enviará alertas de proximidad de los negocios favoritos
Precondición	El usuario pulsa sobre la preferencia de alertas de favoritos
Flujo de eventos	<table><tr><th>Paso</th><th>Acción</th></tr><tr><td>1</td><td>El usuario habilita o deshabilita la preferencia de alertas de favoritos</td></tr><tr><td>2</td><td>El sistema guarda la nueva preferencia</td></tr></table>
Postcondición	Se habilitan/deshabilitan las alertas de favoritos
Excepciones	Ninguna
Importancia	Alta
Comentarios	Ninguno

Tabla 52. Especificación del Requisito Funcional 46

RF-47	Seleccionar tono de alerta
Versión	1.0

<table>
<tr><td>Dependencias</td><td>- Dispositivo compatible</td></tr>
<tr><td>Descripción</td><td>El sistema asignará un tono a las alertas de proximidad</td></tr>
<tr><td>Precondición</td><td>El usuario pulsa sobre la preferencia de tono</td></tr>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><th>Paso</th><th>Acción</th></tr>
<tr><td>1</td><td>El usuario pulsa sobre la preferencia de tono</td></tr>
<tr><td>2</td><td>El sistema abre un cuadro de diálogo con los tonos disponibles en el dispositivo</td></tr>
<tr><td>3</td><td>El usuario selecciona un tono distinto al actual</td></tr>
<tr><td>4</td><td>El sistema guarda la nueva preferencia</td></tr>
</table>
</td></tr>
<tr><td>Postcondición</td><td>Se asigna un nuevo tono a las alertas</td></tr>
<tr><td>Excepciones</td><td>Ninguna</td></tr>
<tr><td>Importancia</td><td>Baja</td></tr>
<tr><td>Comentarios</td><td>Ninguno</td></tr>
</table>

Tabla 53. Especificación del Requisito Funcional 47

<table>
<tr><td>RF-48</td><td>Habilitar vibración para alertas</td></tr>
<tr><td>Versión</td><td>1.0</td></tr>
<tr><td>Dependencias</td><td>- Dispositivo compatible
- Dispositivo con vibración disponible</td></tr>
<tr><td>Descripción</td><td>El sistema habilitará / deshabilitará la vibración del dispositivo cuando se notifique la alerta</td></tr>
<tr><td>Precondición</td><td>El usuario pulsa sobre la preferencia de vibración</td></tr>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><th>Paso</th><th>Acción</th></tr>
<tr><td>1</td><td>El usuario habilita o deshabilita la preferencia de vibración</td></tr>
<tr><td>2</td><td>El sistema guarda la nueva preferencia</td></tr>
</table>
</td></tr>
<tr><td>Postcondición</td><td>Se habilita / deshabilita la vibración para las notificaciones</td></tr>
<tr><td>Excepciones</td><td>Ninguna</td></tr>
<tr><td>Importancia</td><td>Baja</td></tr>
</table>

Comentarios	Ninguno

Tabla 54. Especificación del Requisito Funcional 48

RF-49	Habilitar LED para alertas
Versión	1.0
Dependencias	- Dispositivo compatible - Dispositivo con LED disponible
Descripción	El sistema habilitará / deshabilitará el LED en las notificaciones
Precondición	Ninguna
Flujo de eventos	Paso: 1 — Acción: El usuario habilita o deshabilita la preferencia del LED Paso: 2 — Acción: El sistema guarda la nueva preferencia
Postcondición	Se habilita / deshabilita el LED para las notificaciones
Excepciones	Ninguna
Importancia	Baja
Comentarios	Ninguno

Tabla 55. Especificación del Requisito Funcional 49

RF-50	Recomendar aplicación
Versión	1.0
Dependencias	- Dispositivo compatible - Conexión a Internet - Aplicación de mensajería instalada - Estar la aplicación subida a Google Play
Descripción	El sistema enviará un mensaje al contacto elegido publicitando la aplicación
Precondición	El usuario pulsa sobre la preferencia de recomendar

<table>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><th>Paso</th><th>Acción</th></tr>
<tr><td>1</td><td>El usuario pulsa sobre la preferencia de recomendar</td></tr>
<tr><td>2</td><td>El sistema muestra una ventana de selección de aplicaciones para llevar a cabo dicha acción (Whatsapp, correo electrónico…)</td></tr>
<tr><td>3</td><td>El usuario selecciona una aplicación</td></tr>
<tr><td>4</td><td>El sistema crea un mensaje con el enlace al store de la aplicación</td></tr>
<tr><td>5</td><td>El usuario envía el mensaje</td></tr>
</table>
</td></tr>
<tr><td>Postcondición</td><td>Se crea una mensaje publicitario con un enlace a la aplicación</td></tr>
<tr><td>Excepciones</td><td>P3 - El dispositivo no dispone de aplicaciones de mensajería
P4 - El enlace no es válido si la aplicación no está subida a Google Play
P5 - El mensaje no puede ser enviado por falta de conexión</td></tr>
<tr><td>Importancia</td><td>Baja</td></tr>
<tr><td>Comentarios</td><td>Ninguno</td></tr>
</table>

Tabla 56. Especificación del Requisito Funcional 50

<table>
<tr><td>RF-51</td><td>Puntuar aplicación</td></tr>
<tr><td>Versión</td><td>1.0</td></tr>
<tr><td>Dependencias</td><td>- Dispositivo compatible
- Conexión a Internet
- Aplicación de Google Play / Navegador instalado
- Estar la aplicación subida a Google Play</td></tr>
<tr><td>Descripción</td><td>El sistema mostrará el store de la aplicación para que pueda ser puntuada</td></tr>
<tr><td>Precondición</td><td>Ninguna</td></tr>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><th>Paso</th><th>Acción</th></tr>
<tr><td>1</td><td>El usuario pulsa la preferencia de puntuar</td></tr>
<tr><td>2</td><td>El sistema muestra el store de la aplicación</td></tr>
</table>
</td></tr>
<tr><td>Postcondición</td><td>Se muestra el store de la aplicación</td></tr>
<tr><td>Excepciones</td><td>P2 - No se encuentra el elemento porque la aplicación no está subida a Google Play
P2 - No se puede abrir el store porque no hay navegador / aplicación de Google Play instalada, o no se dispone de conexión a Internet</td></tr>
</table>

Importancia	Baja
Comentarios	Ninguno

Tabla 57. Especificación del Requisito Funcional 51

RF-52	Ver versión de la aplicación
Versión	1.0
Dependencias	- Dispositivo compatible
Descripción	El sistema mostrará la versión de la aplicación
Precondición	El usuario se sitúa sobre la preferencia de versión de la aplicación
Flujo de eventos	Paso: 1 — El usuario se sitúa sobre la preferencia de versión de la aplicación Paso: 2 — El sistema muestra la versión de la aplicación
Postcondición	Se muestra la versión de la aplicación
Excepciones	Ninguna
Importancia	Muy Baja
Comentarios	Ninguno

Tabla 58. Especificación del Requisito Funcional 52

RF-53	Notificar alerta de proximidad
Versión	1.0

<table>
<tr><td>Dependencias</td><td>- Dispositivo compatible
- GPS
- RF-16
- RF-24
- RF-44
- RF-45
- RF-46
- RF-47
- RF-48
- RF-49</td></tr>
<tr><td>Descripción</td><td>El sistema mostrará una notificación de proximidad a un negocio de interés</td></tr>
<tr><td>Precondición</td><td>Ninguna</td></tr>
<tr><td>Flujo de eventos</td><td>
<table>
<tr><th>Paso</th><th>Acción</th></tr>
<tr><td>1</td><td>El usuario activa el GPS</td></tr>
<tr><td>2</td><td>El usuario se adentra en el radio de proximidad seleccionado en Ajustes de un negocio recordado o favorito (si las alertas de favoritos están habilitadas)</td></tr>
<tr><td>3</td><td>El sistema envía la notificación de la alerta de proximidad con la configuración seleccionada en Ajustes</td></tr>
<tr><td>4</td><td>El sistema elimina la alerta de proximidad</td></tr>
</table>
</td></tr>
<tr><td>Postcondición</td><td>Se envía la notificación de la alerta proximidad</td></tr>
<tr><td>Excepciones</td><td>P4 - La alerta no se elimina si el negocio es favorito (los favoritos se notificarán siempre)</td></tr>
<tr><td>Importancia</td><td>Muy Alta</td></tr>
<tr><td>Comentarios</td><td>- El radio y frecuencia de alertas depende de los ajustes seleccionados. Véase RF-44, RF-45.
- Para que se envíen alertas de favoritos, la preferencia de alertas de favoritos debe estar habilitada. Véase RF-46
- El tono, vibración, LED de la alerta depende de los ajustes seleccionados. Véase RF-47, RF-48, RF-49.</td></tr>
</table>

Tabla 59. Especificación del Requisito Funcional 53

10.2. Requisitos no funcionales

A continuación se describen los requisitos no funcionales de la aplicación:

RNF-01	Tecnología de la aplicación
Versión	1.0
Dependencias	- Rendimiento - Cuota de mercado
Descripción	La aplicación se desarrollará en Android
Importancia	Alta
Comentarios	Se desarrollará la aplicación en Android por utilizar un lenguaje nativo y por su mayor cuota de mercado respecto a otras tecnologías

Tabla 60. Especificación del Requisito No Funcional 1

RNF-02	Funcionalidades de la aplicación
Versión	1.0
Dependencias	- Funcionalidades de la versión web - Funcionalidades de aplicaciones similares en el mercado - Explotar características del dispositivo
Descripción	El sistema implementará las funcionalidades presentes en la versión y se le incluirán otras nuevas.
Importancia	Alta
Comentarios	Las funcionalidades de la aplicación irán acorde a las de la web y se le añadirán otras que saquen partido de los dispositivos móviles y hagan de ella una herramienta competitiva.

Tabla 61. Especificación del Requisito No Funcional 2

RNF-03	Estilo de la aplicación
Versión	1.0
Dependencias	- Estilo de la versión web

Descripción	El estilo de la aplicación irá en consonancia con el estilo de la versión web
Importancia	Media
Comentarios	Ninguno

Tabla 62. Especificación del Requisito No Funcional 3

UNIVERSIDAD DE ALMERÍA

Resumen/Abstract

Este proyecto ha sido desarrollado con el objetivo de implementar una aplicación complementaria a la web de Comercio Almería, explotando las características que ofrecen los dispositivos móviles e incluyendo funcionalidades que hagan de ella una herramienta competitiva.

De entre sus funciones cabe destacar la posibilidad de localizar comercios de la localidad de Almería, ofrecer información acerca de productos locales, empresas, mercados y mercadillos.
Además, hace uso de utilidades como el GPS y envío de notificaciones.
Es una herramienta versátil, pues posee una sección de ajustes desde donde el usuario es capaz de configurar una serie de opciones y está adaptada para todo tipo de pantallas.

La tecnología para la cual la aplicación está disponible es Android, por ser la plataforma con la mayor cuota de mercado y emplear un lenguaje nativo como lo es Java. Para el completo desarrollo de la aplicación se han utilizado otros lenguajes además de Java, como PHP y MySQL. También herramientas de distinto tipo, desde el propio entorno de Eclipse hasta PHPMyAdmin.

GRADO EN INGENIERÍA INFORMÁTICA, 2016/2017