Escuela Politécnica Superior de Linares

Grado en Ingeniería Telemática

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Linares

Trabajo Fin de Grado

ESTUDIO DE LAS TECNOLOGÍAS EMPLEADAS EN EL NEGOCIO DE LAS APLICACIONES MÓVILES

Alumno: **Armando Pedroche Hernán**

Tutor: Prof. D. Juan Carlos Cuevas Martínez
Depto.: Ingeniería de Telecomunicación

Septiembre, 2018

RESUMEN

La principal objetivo de este trabajo fin de grado es la de analizar los distintos sistemas, tecnologías y servicios utilizados en el desarrollo y vida de aplicaciones Android para obtener beneficio económico.

El análisis se ha realizado mediante el estudio de la documentación existente en la plataforma de desarrollo Android, revistas tecnológicas, empresas distribuidoras de servicios y tecnologías, y otras diversas encontradas en Internet. Se ha analizado el mercado en busca de soluciones, tanto de la plataforma Android como de empresas de terceros, para dar una visión actual de los recursos que existen al alcance de desarrolladores independientes o empresas que desean rentabilizar y llevar un control de la monetización de aplicaciones móviles realizadas para el sistema operativo Android.

También se ha diseñado una aplicación móvil Android que permite el estudio funcional y técnico de los servicios expuestos por Google en su plataforma Firebase. Entre estos recursos encontramos servicios de análisis de métricas o servicios de monetización mediante publicidad. También se han integrado varios servicios más de dicha plataforma, como servicios de autenticación, bases de datos, almacenamiento, etc.

ABSTRACT

The main objective of this final degree thesis is to analyze the different systems, technologies and services used in the development and lifetime of Android applications for financial gain.

The analysis was carried out by studying the existing documentation on the Android platform development, technological journals, distributors of services and technologies, and several others sources in the Internet. The app market was analyzed looking for solutions, both the Android platform and third-party companies to give a current view of the resources that are available to independent developers or companies who wish to monetize and track the monetization of applications made for the Android mobile operating system.

Within this work, an Android mobile application was also implemented, which allows the functional and technical study of the services exposed by Google through its Firebase platform. These resources are metric analysis services or services monetization through advertising. Several other services of Firebase platform such as authentication services, databases, storage, etc. have been also implemented in the app.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1 INTRODUCCIÓN

En el presente trabajo fin de grado (TFG) se presenta un estudio sobre el negocio de las aplicaciones móviles, y más en concreto en el ámbito de las aplicaciones móviles destinadas al sistema operativo Android. Este ha sido un estudio amplio dentro del contexto de las aplicaciones móviles y abarca todas las fases con las que se encontraría un desarrollador particular o empresa, que se embarque en el desarrollo de una aplicación móvil.

En un primer punto se ha planteado el tema de los almacenes de aplicaciones, desde donde cada sistema operativo distribuye las aplicaciones compatibles con los dispositivos que utilicen dichos sistemas operativos. Se ha realizado una primera toma de contacto para familiarizarse con estos sistemas.

A continuación, en un punto más técnico del estudio, se han descrito las plataformas de desarrollo de aplicaciones móviles más importantes, junto con las tecnologías empleadas para dicha tarea. En este punto se explica cómo se da solución a problemas comunes en el desarrollo de aplicaciones móviles por medio de lenguajes de programación, marcos de desarrollo, conocidos como *frameworks*[1], o servicios externos.

También se ha tratado la manera en la que se añade y gestiona la publicidad dentro de una aplicación móvil, así como obtener un beneficio económico por medio de esa publicidad o de pagos a través de la propia aplicación.

En un siguiente apartado se ha mencionado las obligaciones administrativas para obtener ingresos legalmente de los beneficios de una aplicación móvil y también la legislación española y europea referente a este tipo de aplicaciones.

Por último, se ha implementado una aplicación móvil Android con ejemplos operativos que dan solución a las funcionalidades básicas que hoy en día cuentan la mayoría de aplicaciones móviles del mercado.

1.1 Objetivos

1.1.1 Objetivos generales

Los objetivos generales consisten en realizar un estudio técnico, preciso y riguroso que muestre las tecnologías, plataformas y aplicaciones más importantes en el negocio de las aplicaciones móviles para el sistema operativo Android.

1.1.2 Objetivos específicos

Los objetivos específicos son los siguientes:

[1] Un *framework* es un conjunto de bibliotecas de funciones e interfaces de programación (API) diseñados para facilitar la implementación de aplicaciones para ciertos propósitos.

- Requisitos de publicación en los almacenes de aplicaciones y marketing básico para aplicaciones móviles.
- Plataformas de desarrollo, servicios adicionales como autenticación, almacenamiento remoto, mensajería instantánea, etc.
- Información y gestión de las estadísticas de uso de las aplicaciones.
- Gestión de la publicidad.
- Gestión económica, coste de aplicación, pagos en la aplicación.
- Obligaciones administrativas: el estudio deberá incluir información sobre todos los requisitos legales en España para que un desarrollador o empresa pueda obtener beneficios de una aplicación móvil.
- Legislación española y europea que puede afectar a las aplicaciones móviles.
- Crear ejemplos operativos para aquellas funciones más significativas a desplegar en un móvil:
    - Estadísticas.
    - Publicidad.
    - Pagos en la aplicación.
    - …

1.2 Estado del arte

Este tipo de estudios sobre el negocio de las aplicaciones móviles existe desde que apareció en el mercado este tipo de productos allá por finales de los 90 (varía la fecha según la fuente) y a medida que el mundo del desarrollo se ha normalizado se han ido ampliando en número. Pero debido a que este sector varía tanto en tan corto espacio de tiempo, se hará mención solamente en los estudios más recientes.

Estos estudios se pueden encontrar en multitud de plataformas digitales ya que en muchas ocasiones estos estudios, aunque no tan exhaustivos y extensos, son redactados por empresas del sector de las tecnologías de la información y comunicación (TIC) como revistas o plataformas de aprendizaje.

Situándose al mismo nivel del presente estudio, que serían Trabajos Fin de Grado (TFG) o Proyecto Fin de Carrera (PFC). Se puede encontrar gran multitud de TFG/PFC que tocan el tema de las aplicaciones móviles, pero están más enfocados a explicar o redactar las funcionalidades específicas de alguna aplicación móvil que se ha implementado como objetivo principal del TFG/PFC. Como se ha visto en los objetivos anteriormente enumerados, el caso de este TFG se basa más en el estudio de las plataformas y tecnologías entorno a las aplicaciones móviles y se apoya en el desarrollo de una aplicación móvil para explicar estos términos de manera más real y concisa.

Por lo tanto, se ha incluido en este perfil de estudio algunos TFG/PFC, como podría ser el PFC de Jaime Aranaz Tudela, “Desarrollo de aplicaciones para dispositivos móviles sobre la plataforma Android de Google” [PFCAranaz], realizado en 2009, donde se mencionan las plataformas de desarrollo anteriores a Android y se desgrana la arquitectura, funcionamiento y ciclo de vida de las aplicaciones Android (entre otros aspectos) aunque de manera algo más breve, dedicándole una sección de unas 40 páginas del Proyecto..

También se ajusta a este perfil de estudio el PFC de Francisco Alarcón Barceló, “Desarrollo de Aplicaciones para la Plataforma Android. Un caso de estudio para el intercambio de libros” [PFCAlarcon], redactado en el años 2014, y que hace una mención similar, aunque de forma más escueta, describiendo también algunas funcionalidades del lenguaje nativo de Android (Java), pero extendiéndose más en los aspectos técnicos de la aplicación móvil desarrollada como objetivo principal del citado proyecto.

Se ha incluido un último estudio de ámbito universitario que también abarca el tema de la plataforma Android, centrándose en ella principalmente como objetivo de su trabajo. Aunque el estudio se encuentra centrado en mayor medida en las distintas tecnologías y *frameworks* que permiten desarrollar aplicaciones para el SO Android. El título de este PFC es “Estudio comparativo de alternativas y frameworks de programación, para el desarrollo de aplicaciones móviles en entorno Android” [PFCIskandar], de Ricardo Jose Iskandar Morine con fecha de 2013.

En las conclusiones emitidas en el primero de los PFC que se ha citado encontramos las ventajas al elegir como plataforma de desarrollo a Android, como son su popular lenguaje nativo, la alta cantidad de documentación que ofrece Google, facilidad de desarrollo con su SDK (*Software Developer Kit*), aunque también se incluyen algunas críticas.

Sin embargo, debido a sus fechas de redacción, que van desde 2009 a 2013, las conclusiones pueden no ser concluyentes debido a que 9 años de diferencia son muchos en términos de tecnología.

Se observa también que es complicado encontrar un documento que trate de manera amplia y exhaustiva el tema que nos ocupa. Documentos como PFCs o estudios que sean públicos se encuentran normalmente desactualizados, y yendo a artículos de revistas digitales u otros medios de información con fechas más actuales, suelen ser poco exhaustivos, a veces inconcretos y con un modo de redacción bastante libre. Con la redacción y elaboración del presente TFG se ha pretendido dar una solución a esos inconvenientes, centralizando y actualizando la información.

2 ESTUDIO DE LAS TECNOLOGÍAS DEL NEGOCIO DE LAS APLICACIONES MÓVILES

Este TFG se ha desarrollado con la idea de redactar un estudio entorno al negocio de las aplicaciones móviles, tal y como se encuentra en la actualidad. Aunque se ha analizado en lo posible el resto de plataformas móviles del mercado, la redacción de esta memoria se ha centrado en la plataforma Android. Se ha pretendido dar un valor añadido a lo que sería un estudio de mercado clásico y ahondar en los aspectos técnicos de las plataformas móviles, para hacer así una combinación de ambos.

El presente estudio trata de manera amplía los principios necesarios que cualquier desarrollador o empresa que pretenda trabajar en el mercado de las aplicaciones móviles debieran conocer. Se ha comenzado por los niveles más altos como metodología para analizar cada extracto del negocio de las aplicaciones móviles, que son los niveles más comunes para el usuario habitual de dispositivos móviles, y a partir de ahí se ha ido profundizando en aspectos y características más técnicas. Esto permitiría conocer todos los puntos que un desarrollador o empresa se podrían encuentran en el proceso de crear y comercializar una aplicación.

Teniendo en cuenta que lo que se analiza al final es un producto, los primeros puntos que se han llevado a estudio son los que más relacionados están con la parte comercial y de venta. Es un apartado importante porque reporta gran variedad de métricas muy interesantes y necesarias para ponerse en el contexto actual y comparar cómo de buenos son unos productos respecto a otros. A este mismo nivel se ha descrito cómo es el *marketing* enfocado a este tipo de producto en concreto, las aplicaciones móviles. Las estrategias seguidas en estos productos guardan similitudes con los productos cotidianos, pero también poseen sus propias estrategias.

Después de conocer el estado actual del mercado, como desarrollador o empresa se tendría que poseer el conocimiento sobre qué plataformas hay disponibles hoy en día. En este punto es importante conocer las características que poseen cada una, e incluso el modelo de negocio que siguen, para poder tomar una decisión acertada y que se ajuste a sus necesidades a la hora de desarrollar una aplicación. Para orientarse en el momento que se tenga que elegir una plataforma en la que desarrollar, también hay que tener en cuenta las herramientas que la propia plataforma facilita, y que sirven de guía en el desarrollo de las aplicaciones y además, qué servicios externos son compatibles con la plataforma que se vaya a elegir. En el desarrollo de este punto se centra el foco en Android, por lo tanto, el grueso de este punto (y el de los demás a partir de aquí) estará dedicado a esta plataforma.

En los siguientes puntos se describe la forma en que se puede comercializar una aplicación, y sacar beneficios por medio del producto que se ha implementado. Se aúna en este capítulo los servicios disponibles para integrar y gestionar en una aplicación móvil la publicidad y sus beneficios económicos, junto aspectos importantes del negocio de aplicaciones.

Además, dentro de la parte del trabajo de documentación, se ha hecho referencia a las distintas obligaciones administrativas que existen dentro del territorio español y también qué legislación a nivel español y europeo puede afectar a la hora de crear una aplicación usando cierta tecnología, dependiendo del contenido que tenga o de la información que recabe de sus usuarios, entre otros aspectos.

Por último, en paralelo al trabajo de investigación y redacción de este estudio se ha implementado una aplicación móvil desarrollada en Java, haciendo uso del SDK de Android para beneficiarse así de la interfaz que expone la API (*Application Programming Interface*) Java en la plataforma Android. En la aplicación se han integrado los servicios que Google ofrece en su plataforma Firebase, que es una infraestructura de servicios. El desarrollo está muy relacionado con la redacción del presente estudio ya que abarca el desarrollo de una aplicación móvil para familiarizarse con la plataforma Android y al mismo tiempo integra distintos servicios externos.

2.1 Almacenes de aplicaciones

Uno de los puntos fuertes que poseen los dispositivos móviles (teléfonos móviles, tabletas, etc.) es su fácil usabilidad y comodidad. Esto ha propiciado que, año a año, el mercado móvil haya ido superando al mercado de los ordenadores en número de usuarios. Un ejemplo de ello es el tipo de dispositivo usado para acceder a Internet, y es que desde hace ya unos años los dispositivos móviles son la herramienta preferida por los usuarios que acceden y consumen servicios de Internet.

Debido a estas características, los dispositivos móviles han ido evolucionando poco a poco y con ellos sus sistemas operativos (SO). Hasta llegar al día de hoy, en el que poca gente carece de uno o varios dispositivos móviles, de uso personal o profesional, con gran cantidad de aplicaciones, lo que permiten a estos dispositivos contar con una serie de funcionalidades, casi ilimitadas, estando cada vez más cerca de las que ofrecen los ordenadores personales. Uno de los conceptos que ha propiciado llegar a este punto es el de los almacenes de aplicaciones.

Los almacenes de aplicaciones (o también llamados tiendas de aplicaciones) son sitios web enfocados a servir de sistema de almacenamiento, publicación, venta y descarga de aplicaciones. Cada SO (Android, IOS, Windows Phone, etc) tiene su propio almacén de aplicaciones en los que solo se pueden publicar y descargar aplicaciones hechas para

dicho SO. Google Play Store de Google, Apple App Store de Apple o Windows Store de Microsoft son los almacenes de aplicaciones con más descargas del mercado.

Por otro lado, de un tiempo a esta parte, existen otras marcas del sector tecnológico que han querido formar parte del negocio de los almacenes de aplicaciones. Dentro de este grupo se encuentran marcas como Amazon, que ofrece un gran catálogo de aplicaciones en Amazon Appstore u Opera Mobile Store, creado por los desarrolladores del navegador Opera.

En el siguiente gráfico se puede observar el número aplicaciones móviles disponibles para su descarga en los principales almacenes de aplicaciones del mercado durante el 2017.

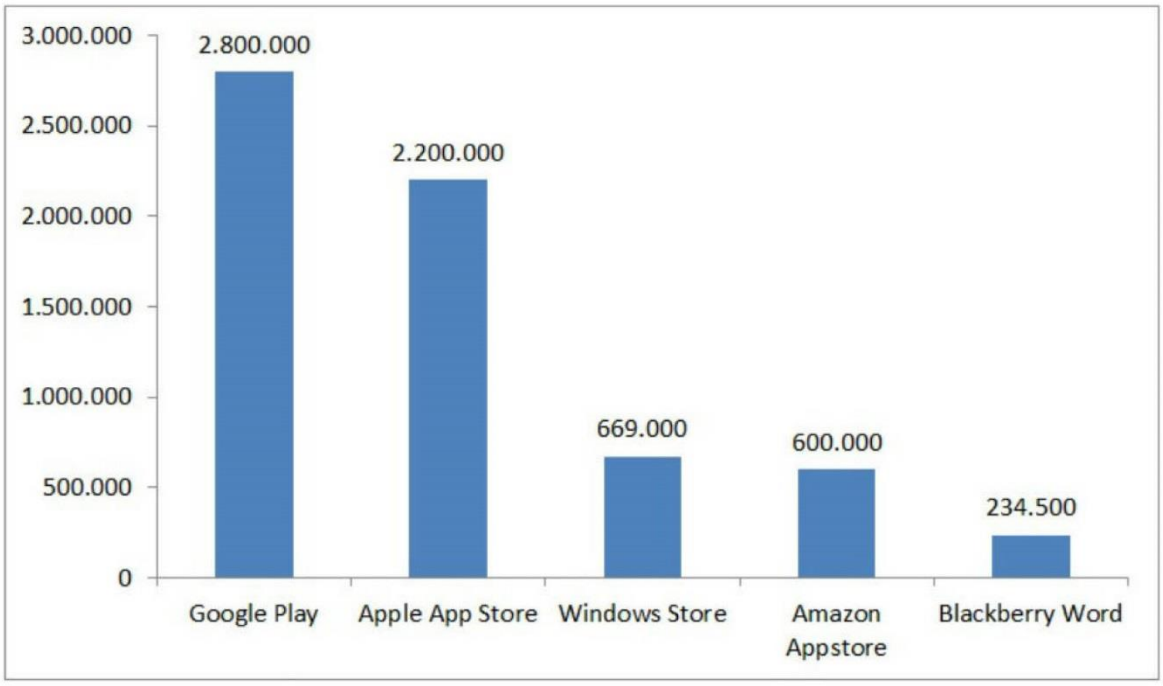

Figura 2.1 Número de aplicaciones disponibles en las principales tiendas de aplicaciones móviles en marzo de 2017. Fuente: [StatistaWeb].

Como se puede ver en la figura anterior, los 3 primeros puestos de este ranking los ocupan los almacenes de aplicaciones de las principales marcas de SO. Esto es debido a que los almacenes de estas marcas no solo tienen las funcionalidades básicas que se han comentado anteriormente, sino que también se encargan de dar un valor añadido a las aplicaciones. Un claro ejemplo de ello pueden ser las políticas de admisión de aplicaciones que llevan a cabo para ofrecer así una grata experiencia de usuario, controlando así la calidad de los productos que van a ofrecer a sus clientes.

A continuación, se han enumerado las principales características comunes que nos ofrecen los almacenes de aplicaciones:

- Centralización desde una misma plataforma de la descarga de aplicaciones.

- Mejora de la seguridad en el momento de la descarga (Debido a las medidas de seguridad en torno al contenido que se publica).
- Organización de aplicaciones por categorías, funcionalidades, temáticas, etc.
- Buscador sobre todo el contenido de la plataforma.
- Personalización del contenido que se muestra mediante el uso de usuarios propios de cada plataforma.
- Información sobre las aplicaciones instaladas en tu dispositivo, como pueden ser actualizaciones.
- Interacción entre usuarios de la plataforma (usuarios finales y desarrolladores de una aplicación) por medio de comentarios, valoraciones, etc.

A parte de estas características que se han citado, que son más o menos comunes en los almacenes de aplicaciones principales, también cabe destacar algunas de las características más relevantes y ventajas de cara a los desarrolladores que ofrece Google Play Store:

- Mejor perspectiva de negocio en el desarrollo de aplicaciones móviles.
- Gran visibilidad a los futuros usuarios de la aplicación.
- Métricas sobre aplicaciones instaladas y número de descargas.
- Sistema de valoraciones de los usuarios.
- Sistema de análisis de errores producidos en la ejecución.
- Cobro de la aplicación gestionado por la propia plataforma.
- Actualización automática de la aplicación para nuevas versiones. [AndroidBook].

2.1.1 Requisitos de publicación en los almacenes de aplicaciones

Como se ha podido observar, a día de hoy y desde hace ya unos cuantos años, tanto Google Play Store como App Store copan los dos primeros puestos a nivel de número de aplicaciones disponibles para sus respectivos sistemas operativos. No es de extrañar, por lo tanto, que estos almacenes de aplicaciones también estén a la cabeza en número de descargas.

En el siguiente gráfico se puede ver cómo ha evolucionado esa disputa en los últimos años.

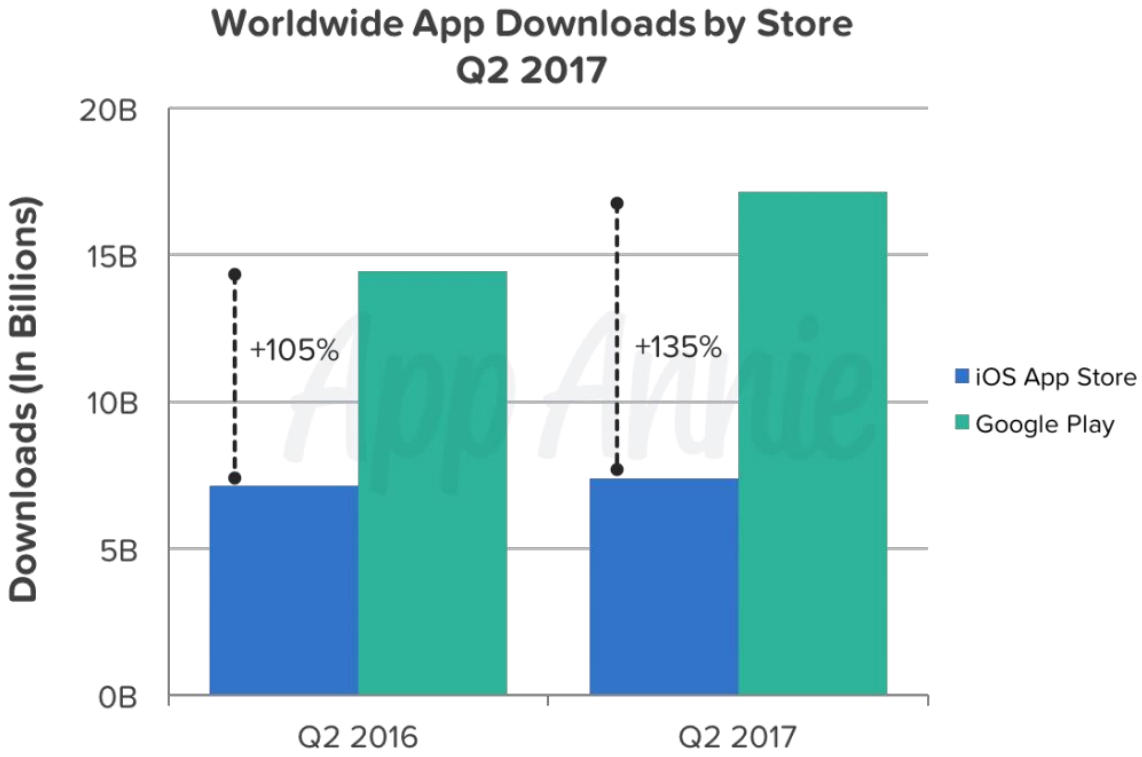

Figura 2.2 Número de descargas en Google Play Store y App Store durante 2016 - 2017. Fuente: App Annie

Según App Annie, importante empresa de estadísticas en el sector de las aplicaciones y que suele realizar estudios de relevancia sobre este tema en concreto, esta disputa lleva ya unos cuantos años y así lo plasmaba en las distintas retrospectivas que ha creado en los últimos años sobre este tema. En su retrospectiva de 2017 (https://www.appannie.com/en/insights/market-data/google-apple-app-stores-q2-2017/) donde podemos ver como Google Play Store sigue creciendo en número de descargas por año, y sigue aumentando su diferencia año a año con su más inmediato y gran competidor.

Pero al contrario de lo que pueda parecer por el gráfico anterior, el número de descargas no es proporcional a la cantidad de beneficios generados por las mismas, ya que, tanto los usuarios que usan los almacenes, tienen perfiles diferentes y no usan de la misma manera estas tiendas de aplicaciones, como las marcas que hay detrás de ellos, tienen una idea de negocio muy diferente. Así, Android apuesta más por numerosos contenidos gratuitos y permite el uso de aplicaciones desde otras fuentes que no sean Google Play Store, frente a Apple, que tiene una política de contenidos más bien de pago (por lo general) y que no permite el uso de aplicaciones con un origen diferente al de su tienda oficial.

En el siguiente gráfico se puede ver la diferencia de ingresos entre Google Play Store y App Store por la cantidad de gastos generados por sus clientes.

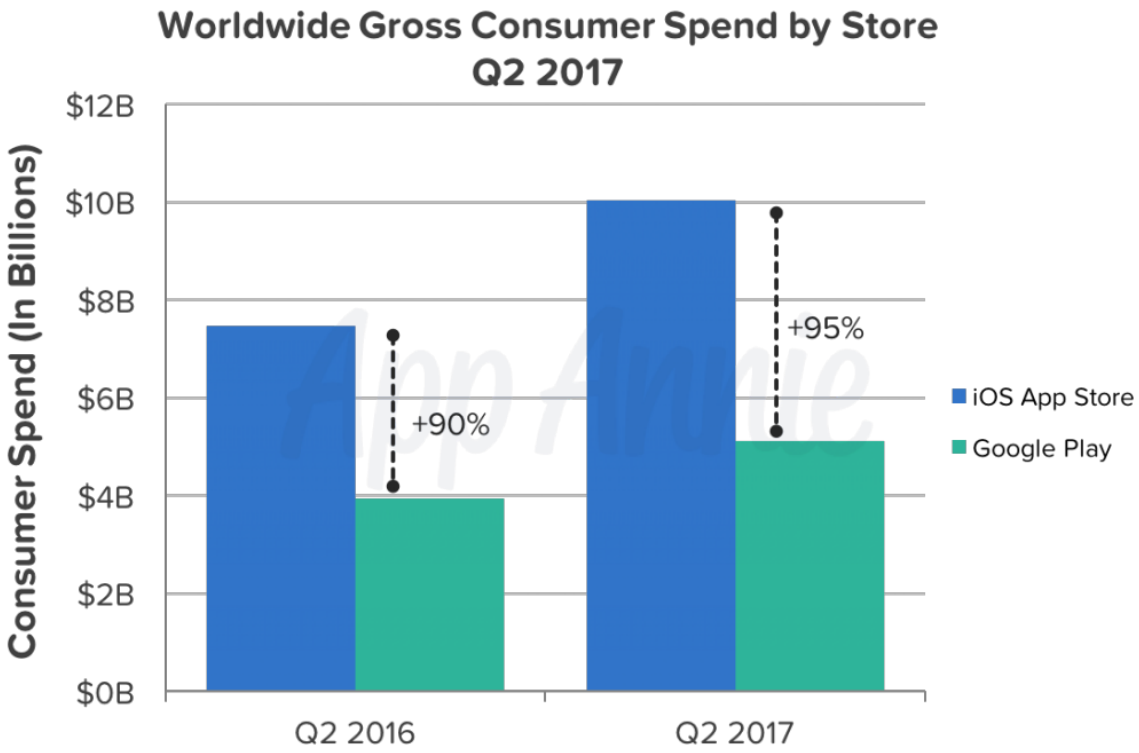

Figura 2.3 Gastos de los consumidores en Google Play Store y App Store durante 2016-2017. Fuente: [AnnieAnalytics].

Este gráfico representa la diferencia de ingresos provenientes de los clientes de cada uno de los almacenes de aplicaciones. Se observa que el modelo de negocio que plantea Apple para su tienda de aplicaciones le resulta más rentable que a Android. Aunque el gráfico podría ser algo diferente en la realidad, ya que las aplicaciones para Android no solo se distribuyen en su tienda de aplicaciones, al contrario de lo que ocurre con Apple.

Aunque tengan modelos de negocio difieren en algunos puntos, sí que tienen una idea en común, y es que las dos marcas consideran que es imprescindible mantener un buen nivel de calidad en las aplicaciones que distribuyen si quieren mantener el liderazgo de poseer las tiendas de aplicaciones donde más descargas se realizan y, por lo tanto, las que más ingresos generan en este ámbito.

Basándose en esta idea de ofrecer un producto de gran calidad, cada uno de los almacenes se marca unos requisitos mínimos de publicación para asegurarse de que las aplicaciones móviles que distribuyen sean de una determinada calidad en cuanto a contenidos o seguridad y relacionar así esta calidad en sus productos como un valor añadido más de su tienda de aplicaciones.

2.1.1.1 Requisitos de publicación en Google Play Store

Se ha comenzado por enumerar los requisitos de Google Play Store ya que es menos restrictivo en sus directrices (https://appyourself.net/es/2016/11/03/cosas-prohibidas-al-crear-una-app/):

- **Cuenta de desarrollador**: Se debe contar con una cuenta de desarrollador para poder publicar tu aplicación.
- **Contenido**: Nada de contenido con connotaciones negativas (sexual, racista, homófobo, violencia, actividades ilegales (como apuestas), etc.
- **Información Personal**: No se permite compartir información sensible (datos fiscales) ni datos de usuarios con terceros.
- **Virus**: Tampoco se permite publicar software malicioso ni obligar a descargar otras App no verificadas.
- **Modificación sin consentimiento**: No se permiten aplicaciones que alteren o modifiquen información del usuario sin su consentimiento.
- **Notificaciones *Push***: Aquellas aplicaciones que manden notificaciones que no tengan que ver con su negocio y tengan como fin la publicidad serán bloqueadas.
- **Ventas**: Debe configurarse si tu App tiene como fin la compra/venta de productos, ya que de lo contrario y en caso de haber transacciones, tu aplicación será bloqueada.
- **Propiedad Intelectual**: No se permite el robo y el lucro de propiedad intelectual.

2.1.1.2 Requisitos de publicación en App Store

Por lo tanto, los requisitos para publicar en App Store son todos los que se han mencionado para Google Play Store junto con los siguientes, que son específicos de su almacén de aplicaciones.

- **Contenido Google**: No se permite contenido relacionado con Google u otras plataformas (Windows, Amazon, etc.).
- **Políticas de privacidad**: Si la App cuenta con inicio de sesión, se debe contar con políticas que aseguren la identidad de los usuarios.
- **Información insuficiente**: Se debe rellenar una gran cantidad de información requerida para el análisis de la funcionalidad de tu App.
- **Capturas de pantalla**: Deben estar relacionadas con el contenido de tu App para pasar los filtros de seguridad.
- **App Demo**: Si es una App de prueba deberá pasar los requisitos establecidos en TestFlight para poder publicarse. [AppStoreTips].

2.2 Marketing básico para aplicaciones móviles

El concepto de marketing no es exclusivo de las aplicaciones móviles, y es que cualquier tipo de producto es susceptible de aplicarle una estrategia de marketing para

mejorar su posicionamiento dentro del mercado y en detrimento de sus competidores. Por lo tanto, veremos en el marketing de las aplicaciones móviles puntos que son genéricos, independientemente del tipo de producto del que hablemos, y otros puntos más específicos, debido a las particularidades del producto que nos atañe en este documento.

Según la Real Academia Española (RAE), concepto de *marketing* se refiere a todas aquellas estrategias empleadas para la comercialización de un producto y para estimular su demanda. [RAEDiccion]. Si esto se extrapola al sector de las aplicaciones móviles, la definición que se puede dar al *marketing* para aplicaciones móviles sería el de todas aquellas estrategias empleadas para adquirir el mayor número de usuarios posibles. [BasicMarkting].

Al final, el objetivo de cualquier producto es el de obtener un beneficio económico mediante el mismo, y dichas estrategias facilitarán al creador y beneficiario de la aplicación móvil a llegar a este objetivo. Para ello, la hoja de ruta más habitual en este tipo de productos es fijar indicadores para poder reconocer el éxito o fracaso que la aplicación está teniendo, y si es en tiempo real mucho mejor, ya que esto permitirá redefinir una estrategia de *marketing* y adaptarla para cumplir los objetivos que se han fijado. A estos indicadores se les conoce por indicadores clave del desempeño (KPI, del inglés *Key Performance Indicator*). Se trata de un valor dado en porcentaje y es un indicador, valga la redundancia, del beneficio y de en qué medida se han cumplido los objetivos con los que se ha creado la aplicación. Un ejemplo de KPI podría ser el del porcentaje de ventas de billetes frente a los billetes ofertados en la aplicación móvil de una compañía aérea.

Entonces, para desarrollar de manera más amplia el concepto de *marketing* de aplicaciones móviles se deben conocer los distintos tipos de estrategias que se pueden aplicar con este fin. Evidentemente en este mundo las estrategias que se emplean podrían dividirse u organizarse de muchísimas maneras, más si cabe el del *marketing* digital, que está tan de moda, es tan cambiante, posee tantas vertientes y puntos de vista desde donde mirarse, sin embargo, generalizando, se puede dividir en dos grupos: las estrategias que se pueden aplicar directamente sobre el producto que se quiere promocionar, en el presente caso una aplicación móvil, y las estrategias aplicadas fuera del producto, básicamente en su entorno.

2.2.1 Estrategias de marketing aplicadas a la aplicación móvil

En este tipo de estrategias entran aquellas que se centran en potenciar la imagen de la aplicación móvil utilizando como medio la propia aplicación móvil.

2.2.1.1 Métrica y analítica de la información generada por el uso de la aplicación

Dentro de este tipo de estrategias una de las técnicas más importantes sería aquella que utiliza medición y análisis de datos relacionados con los perfiles de los usuarios para

saber cuál es el perfil de usuario que atrae tu aplicación o si necesitas enfocarla de otra manera para llegar a un público distinto que te resulte más interesante.

Otra de las técnicas más usadas se trata del registro de eventos generados por los usuarios mediante el normal uso. En este contexto el término de evento se emplea meramente aplicado a su uso en el ámbito del marketing, serviría para registrar el trazo dejado por el usuario en su interacción con la aplicación, lo que en el *marketing* digital se denomina *tracking*, y que posteriormente puede ser analizado y explotado para tomar decisiones que mejorar la experiencia que el usuario tiene con la aplicación móvil (UX).

Existen también otras muchas métricas que dan una información importante y que resulta interesante que sean analizadas como pueden ser el número de descargas e instalaciones de la aplicación.

2.2.1.2 Realizar un desarrollo de calidad para ofrecer un producto de buena calidad

En estas técnicas entran en juego el contexto puramente de desarrollo de la aplicación. Desarrollar la aplicación con buenos patrones y algoritmos de programación, haciendo una aplicación que sea funcional, que reaccione con rapidez y que no se quede bloqueada. Pero no solo un desarrollo de calidad en la parte funcional, sino también en la parte estética, y es que es muy importante una apariencia agradable y unificada dentro de la aplicación, tener un diseño coherente hará sentirse más cómodo al usuario que haya descargado la aplicación.

Para está tarea resulta interesante el *feedback* y revisiones que los propios usuarios pueden ofrecer, y de manera gratuita, por medio de comentarios, valoraciones, aviso de posibles bugs o cualquier otra petición que se quiera valorar.

2.2.1.3 Campañas de publicidad como una funcionalidad más de la propia aplicación

Y es que existen muchas formas en que una aplicación móvil puede realizar campañas de publicidad, como puede ser la captación de más usuarios mediante los usuarios que ya posee, utilizando a estos últimos como bases de datos de cuentas de correo electrónico, a los que poder enviar invitaciones o referencias, abarcando así un número mayor de posibles futuros usuarios. [MarktingApps] [AnalytMarkting].

2.2.2 Estrategias de marketing aplicadas fuera de la aplicación móvil

Fuera del ámbito de la aplicación, existen otros contextos donde se pueden aplicar estrategias que favorezcan la visión de la aplicación de cara a los futuros usuarios.

2.2.2.1 Análisis y estudio de los futuros usuarios

Es importante optimizar las acciones o técnicas que se van a realizar con el objetivo de atraer a cuanto mayor número de usuarios a la aplicación.

Para conseguir este fin es fundamental estudiar hacía que tipo de público se quiere enfocar las técnicas de marketing. Por lo tanto, se debe caracterizar y definir lo mejor

posible al usuario de nuestra aplicación y de esta manera decidir como enfocar el resto de técnicas.

2.2.2.2 Campañas de Email

Después de haber analizado aquellos sectores de usuarios sensibles a utilizar la aplicación, se debe potenciar aquellos puntos clave de la misma que se crean atractivos de cara a los futuros usuarios. La intención de esta técnica es la de llamar la atención.

Mediante el uso del correo electrónico se pueden acercar contenidos de la aplicación, promociones para intentar fidelizar público o cualquier otro contenido que pueda servir para este fin.

Existen varias opciones para realizar este tipo de técnicas, y es que multitud de plataformas o sistemas tienen esta como su exclusiva funcionalidad. Existen aplicaciones web como pueden ser MailChimp o SendWithUs, entre otras tantas, donde puedes configurar envíos, tener infinidad de plantillas en sus respectivos idiomas, listas de usuarios y una gran variedad más de opciones. También se pueden ver sistemas de campañas de email desarrollados e integrados con Google Analytic.

2.2.2.3 Landing Page

Este término, que traducido al español sería algo así como “página de aterrizaje”, hace referencia a la página a la que llega el usuario tras pinchar en algún enlace que se le haya hecho llegar por medio de alguna técnica de *marketing*.

Es importante el uso de esta página, ya que puede tener distintos usos. Se podría utilizar para visibilidad, de cara a los usuarios, o como punto centralizado del flujo de usuarios provenientes de campañas de marketing. Puede ser interesante el uso de esta técnica para ampliar el abanico de usuarios a los que se puede llegar con un sitio web que es totalmente accesible desde Internet.

2.2.2.4 Visibilidad en redes sociales

Este tipo de canales son actualmente los medios de comunicación más usados para el marketing de productos en general, así que no es de extrañar mencionar las redes sociales como un tipo de técnica, si nos fijamos bien actualmente este tipo de aplicaciones tienen reservado bastante espacio para publicidad de sus anunciantes. Por lo tanto, queda claro que es una importante parte de los ingresos de estas empresas.

Pero también existen otras técnicas dentro de las redes sociales, diferentes a pagar a la propia red social por anunciar tu publicidad en esos espacios. Crear perfiles y compartir contenido o el uso de figuras con repercusión pública (*influencers* por ejemplo) para llegar a un mayor número de personas.

2.2.2.5 Posicionamiento dentro de las tiendas de aplicaciones - ASO

El ASO (de App Store Optimization) se puede entender como un término análogo a lo que sería el SEO (Search Engine Optimization).en los buscadores. Así como la finalidad del SEO es la de colocar un sitio web lo más alto posible en los resultados del buscador para una determinada búsqueda (ya que la primera es la página más visitada), la del ASO es tratar de posicionar una aplicación móvil lo más alto posible en los resultados de una determinada búsqueda dentro de las tiendas de aplicaciones.

Las tiendas de aplicaciones móviles marcan unas pautas a seguir y por las cuales penalizarán a aquellas aplicaciones que no se ajusten a estas normas. Las aplicaciones que se mueven dentro de estas pautas son denominadas ASO *friendly*.

Existen empresas que realizan auditorias ASO a aplicaciones móviles para detectar este tipo de “errores” con el objetivo de trabajar posteriormente en un desarrollo lo más ASO *friendly* posible.

2.3 Tecnologías usadas para el desarrollo de aplicaciones móviles

Hoy en día existe una gran variedad de tecnologías disponibles en el mercado, por lo que no es complicado encontrar soluciones para desarrollar una aplicación móvil.

Estás tecnologías se pueden dividir por tipos, agrupándolas por sus características comunes. Esta organización se realiza según lo específica que sea la tecnología que se usa, ya que cuanto más específica sea la tecnología en relación al SO para el que vamos a desarrollar una aplicación móvil, más podremos aprovechar las ventajas que ofrece el SO en el que vivirá esa aplicación.

2.3.1 Aplicaciones web adaptables o “responsive”

Las aplicaciones realizadas utilizando tecnologías adaptables al cliente, o más conocidas por el término en inglés *responsive*, poseen la misma estructura que tendría cualquier aplicación web convencional, y ese que, en esencia pertenecen al mismo tipo de aplicaciones. Se puede decir, que las aplicaciones web *responsive* son un tipo de aplicación web.

Una aplicación web es un sistema con una cierta funcionalidad capaz de mostrar una interfaz de usuario (UI de *User Interface*) por medio de un navegador web. Una aplicación web por norma general se divide en dos partes (a groso modo, participan más sistemas en la arquitectura de una página web), *Front-end* (la parte de código que interpreta el cliente) y *Back-end* (la parte de código que interpreta el servidor). En el caso que se está tratando, la parte importante es la del *Front-end*, ya que es donde se aplican tecnologías que son capaces de hacer un diseño *responsive* de la UI.

Una aplicación web *responsive* será por lo tanto una aplicación web capaz de adaptar su diseño a distintos tamaños de pantalla y manteniendo una estructura coherente, siendo así capaz de visualizarse en un dispositivo móvil.

Existen varías tecnologías capaces modificar el diseño del modelo de objetos del documento (DOM de *Document Object Model*). Una de las más utilizadas, y según los más puristas del sector del diseño web la forma más correcta de convertir en *responsive* una aplicación web, es mediante la aplicación de hojas de estilo en cascada (CSS de *Cascading Style Sheets*). Está tecnología tiene la capacidad de modificar el código del DOM diseñado en lenguaje de marcas de hipertexto (HTML de *HyperText Markup Language*) mediante el uso de hojas de estilos aplicadas al contenido de una vista que vaya a ejecutarse en un navegador. CSS posee unas directivas llamadas *media queries* que al ser interpretadas por el navegador son capaces de establecer que hojas de estilos se aplicaran para un determinado tamaño de ventana.

Otra de las tecnologías más usadas en aplicaciones web *responsives* son los lenguajes de programación interpretados por el cliente. Javascript es un lenguaje interpretado estándar ISO 16262. [ISO16262]. El resto de lenguajes ejecutados en un navegador web están basados o son bibliotecas de Javascript. Hoy en día este tipo de lenguajes se utilizan para muchos fines ya que poseen una serie de recursos bastante amplia, pero una de sus funcionalidades más empleada es la poder interactuar con los elementos del DOM. Al tener una alta dependencia con la UI posee funcionalidades para detectar el tamaño de ventana donde se está ejecutan y a partir de ahí puede modificar el DOM por medio de funciones definidas por el desarrollador.

Un ejemplo de aplicación web *responsive* sería la página corporativa de Acciona, que además utiliza las dos tecnologías que hemos mencionado anteriormente. [AccionaWeb].

Se puede ver entonces lo genérico que es un navegador web al ejecutarse en cualquier dispositivo y sobre cualquier SO, lo que hace que no se pueda aprovechar todas las características que un SO diseñador a propósito para dispositivos móviles ofrece.

2.3.2 Aplicaciones web embebidas

Como se ha podido descubrir en el punto anterior, una aplicación web no es siempre la mejor solución a la hora de crear una aplicación móvil. Una de las razones es que ese tipo de desventajas que se han comentado son palpables de cara al usuario, debilitando esto el peso que ese tipo de productos tienen en el mercado de las aplicaciones móviles, y no en descargas (que es un término que no aplicaría en el caso de las aplicaciones web *responsive*), sino en uso. Para enmendar este inconveniente aparece un nuevo tipo de tecnología que permite convertir aplicaciones web en aplicaciones con un formato y estructura aceptado por los SO móviles. Además, esta tarea se vuelve más sencilla si

tenemos en cuenta que los navegadores de los SO móviles actuales están basados en el mismo motor de navegación, WebKit. Estas aplicaciones se denominan aplicaciones web embebidas.

El termino embeber hace referencia a encapsular o empaquetar un elemento, y en el contexto de desarrollo se refiere a empaquetar y proveer de un formato determinado por medio de herramientas a un sistema que no tiene esa estructura. Esta estructura, en el caso de los SO móviles, será la que permita correr a este tipo de aplicaciones sobre Android, iOS, etc. Además, permite colocar estas aplicaciones en los almacenes oficiales. Con este hito, este tipo de aplicaciones llegan a un mayor número de usuarios, ya que tienen un escaparate más atractivo.

Además de intentar dar solución al tema de la distribución y uso de la aplicación, estas aplicaciones tienen una apariencia “diferente” a una aplicación web, pero únicamente lo que cambia es el elemente donde cae la responsabilidad de interactuar con la aplicación. Pero aun así, este tipo de aplicaciones siguen teniendo mucho en común con las aplicaciones web y con su mala capacidad de relación con los SO móviles.

Un ejemplo de esta tecnología es la aplicación móvil de Iberia Express (aunque desaparecida recientemente de las tiendas de aplicaciones).

Como se puede ver, este tipo de tecnología no consigue dar solución a puntos claves e importantes dentro de la vida de una aplicación móvil. Ni la experiencia de usuario, ni la integración con el SO son óptimas, y esto hace que sea muy complicado obtener un buen resultado en dispositivos móviles, junto con una gran carga de trabajo que muchas empresas o desarrolladores no ven rentable invertir.

2.3.3 Aplicaciones móviles desarrolladas con frameworks multiplataforma

La organización con la que se ha enumerado esta lista de tecnologías disponibles para el desarrollo de aplicaciones móviles tiene una línea clara entorno a las posibilidades que un desarrollador o empresa tendría al hacer uso de ellas. Esto quiere decir que conforme se avance en esta lista de tecnologías con las que desarrollar aplicaciones se obtendrá un mayor rendimiento de las características de los dispositivos móviles y de los SO que en ellos estén integrados.

En este punto se debe definir el concepto de aplicaciones hibridas. Se trata de aplicaciones capaces de ejecutarse en distintos SO con el desarrollo de un “único” proyecto, sin tener que desarrollar una aplicación solo para Android, otra para iOS u otra para Windows Mobile. Normalmente, la definición que se da para este término es que son aplicaciones basadas en tecnologías como HTML, Javascript y CSS, lo que nos recordaría mucho a la definición que hemos dado en las tecnologías que hemos comentado anteriormente. [HybridApps].

La diferencia radica en que para estos desarrollos se utilizan *frameworks* de desarrollo móvil multiplataforma. Estos *frameworks* poseen una arquitectura bien definida, y aunque es difícil dar una definición general, la idea es la de construir aplicaciones que tengan un funcionamiento parecido al que tendía una aplicación nativa.

Para entender mejor el término *frameworks* de desarrollo multiplataforma se han analizado algunos ejemplos. Uno de los *frameworks* que más ha dado que hablar últimamente es Xamarin. Este *framework*, propiedad de Microsoft, da la posibilidad de crear aplicaciones con UI nativos para cada SO compartiendo un proyecto central, denominado *Core* que es escrito en lenguaje C#. Las peculiaridades de cada interfaz están definidas en los proyectos que se crean con plantillas nativas de Android, iOS o Windows. Además, como el proyecto *Core* está desarrollado con librerías .NET Standard, que son librerías creadas en su nuevo *framework* multiplataforma .NET Core, son proyectos reutilizables para aplicaciones web, de escritorio, etc.

Otro *framework* de desarrollo multiplataforma es React Native. Se trata de una versión de React, un *framework* basado en TypeScript, manejado por variables de estado e interpretado en el cliente. Su versión para desarrollo móvil multiplataforma, React Native, permite traducir su código TypeScript (que es en el fondo un lenguaje basado en Javascript, y que al realizar su compilación obtendrá este formato final) al lenguaje nativo de cada plataforma móvil, pudiendo hacer uso así de las funcionalidades nativas de cada SO.

Un ejemplo de este tipo de aplicaciones (desarrollada con React Native) es la aplicación móvil de Facebook. [FacebookApp].

Se puede observar que este tipo de tecnologías cuentan con ventajas muy interesantes, como también es importante el ahorro de coste que puede generar el uso de estos *frameworks*. Acercándose a una experiencia parecida a la que ofrece una aplicación nativa, parece una buena opción a la hora de elegir una tecnología para el desarrollo de aplicaciones móviles.

2.3.4 Aplicaciones móviles desarrolladas por medio de SDK

A medida que se avanza en este listado de tipos de tecnología, encontramos opciones cada vez más potentes y que proveen de una mayor calidad a las aplicaciones desarrolladas con las mismas.

El kit de desarrollo de software (SDK de *Software Development Kit*), se trata de un kit de herramientas que normalmente es distribuido por la propia compañía responsable del SO. Este kit de herramientas suele estar compuesto por herramientas de depuración, de compilación, kit de desarrollo para los lenguajes que soporta el SO, y un variado número de herramientas más que se pueden añadir por medio de la configuración durante el proceso de instalación.

Por ejemplo, analizando lo que ocurre en el caso de Android, vemos que uno de los lenguajes que podemos usar es Java (que no es un lenguaje que podamos considerar nativo de forma estricto, aunque eso es otro tema), que es un lenguaje que puede ser interpretado por el SDK de Android. Aun así, para el uso y la ejecución de programas escritos en Java en una maquina se necesitará de las herramientas que proporcionan el kit de desarrollo de Java (JDK de *Java Development Kit*) y el entorno de ejecución de Java (JRE de *Java Runtime Environment*). Con estas herramientas se puede hacer uso de la API de Android y así tener acceso a las funcionalidades del dispositivo que se exponen desde la misma.

Los entornos de desarrollo integrado (IDE de *Integrated Development Environment*) ya están preparados para integrarse con estos kits de herramientas y complementar las funcionalidades que ya ofrecen de por sí.

Un ejemplo de uso de esta tecnología aplicada a una aplicación móvil real sería la aplicación móvil que se ha realizado como complemento a la redacción del proyecto TutoGuía Firebase. [TutoGuiaApp].

Este tipo de tecnología realiza un aprovechamiento muy óptimo de las características, funcionalidades y hardware del dispositivo donde va a estar instalada la aplicación ya que interactúa directamente con el SO del dispositivo. Aunque pueda parecerlo, acceder a todas las funcionalidades que la plataforma de desarrollo ofrece no es todavía todo lo bueno que podría ser.

2.3.5 Aplicaciones móviles desarrolladas por medio de lenguajes nativos

Para poder tener acceso a todas las funcionalidades que expone un SO móvil debemos hacer uso de un lenguaje nativo de cada plataforma de desarrollo.

Para entender este punto se deben definir antes un par de términos. El lenguaje nativo de una plataforma de desarrollo se refiere a aquel lenguaje que en relación con la arquitectura del SO es un lenguaje de bajo nivel, lo que permite tener un acceso más libre al núcleo de la arquitectura, con las ventajas que ello conlleva. Los lenguajes que no son considerados nativos en una plataforma de desarrollo móvil son aquellos que se quedan en un nivel más alto y necesitan de alguna herramienta más para tener acceso a las funcionalidades del dispositivo, y, por lo tanto, no explotaran al máximo sus posibilidades. [NativOrHybr].

Luego, en el ámbito puramente de las aplicaciones móviles, se considera una aplicación nativa aquella que haya sido implementado por un lenguaje definido como propio de la plataforma de desarrollo.

En el caso de Android, para el desarrollo de una aplicación nativa sería necesario utilizar C/C++, ya que es el lenguaje en que están implementadas las librerías que están compiladas en el código nativo del procesador. Esta capa esta soportada sobre el núcleo

de Android, que es la que maneja los controladores del dispositivo, por lo tanto, acceder a esta capa de una forma tan directa dota de mucha potencia a la aplicación que se desarrolle de esta manera (la arquitectura de Android es un punto que se explica de manera más detallada en puntos posteriores).

Estás librerías nativas que componen la arquitectura también están disponibles para Java por medio de una función que permite cargarlas.

Un ejemplo de este tipo de aplicaciones sería la versión inicial que apareció de WhatsApp era una aplicación nativa de iOS, y que por lo tanto no estaba disponible para el SO Android.

Tras el repaso realizado por todas las tecnologías posibles que están disponibles para el desarrollo móvil se puede destacar esta última como la opción que permite realizar un producto de mayor calidad y que tenga una inmejorable experiencia de usuario (UX de *User Experience*).

Por lo tanto, se puede concluir afirmando que las funcionalidades de las aplicaciones móviles son dadas por (aparte de por las tecnologías usadas para crear las mismas): el conjunto de herramientas ofrecidas por el SDK (kit de herramientas de la plataforma que sea) y el lenguaje nativo de turno, junto con su kit de turno para uso de librerías o escribir programas de dicho lenguaje y tal vez alguno más, para poder ejecutarse el código de ese lenguaje en una máquina; aplicaciones web *responsive*; aplicaciones web embebidas en formatos aceptados por los almacenes de aplicaciones; y aplicaciones móviles creadas con *frameworks* multiplataforma (Xamarin, React Native ,etc.). [DevelopmTec].

2.4 Plataformas de desarrollo de aplicaciones móviles

El concepto de plataforma de desarrollo, en el contexto del desarrollo de aplicaciones móviles, se define como un marco de trabajo que ofrece distintas posibilidades en forma de herramientas, como librerías y paquetes con utilidades o APIs con servicios externos, que permiten montar aplicaciones para dispositivos móviles que funcionen sobre un determinado SO. Normalmente, las empresas responsables de cada SO son las responsables de estas plataformas de desarrollo, entiendo por qué una plataforma de desarrollo suele estar enfocada a un determinado SO. [DevelopPlatf].

Un punto importante a tener en cuenta antes de ponerse a desarrollar una aplicación móvil es la selección de que plataforma de desarrollo vamos a utilizar, ya que existen multitud de diferencias entre cada una de ellas. Además, como ya se ha visto anteriormente, cada empresa tiene un modelo de negocio distinto, lo que se evidencia en sus productos.

En la tabla que se encuentra debajo de este párrafo se puede ver una comparativa que elabora la empresa Statista, que va actualizando paulatinamente, sobre las principales plataformas de desarrollo móvil y se pueden visualizar sus características más importantes.

	Apple iOS 11	**ANDROID 8.1**	**WINDOWS MOBILE 10**
COMPAÑÍA	Apple	Open Handset Alliance	Microsoft
NÚCLEO DEL SO	Mac OS X	Linux	Windows NT
LICENCIA DE SOFTWARE	Propietaria	Abierta	Propietaria
AÑO DE LANZAMIENTO	2007	2008	2003 (Windows phone: 2010)
FABRICANTE ÚNICO	Si	No	No
VARIEDAD DE DISPOSITIVOS	Modelo único	Muy alta	Media
SOPORTE MEMORIA EXTERNA	No	Si	Si
MOTOR DEL NAVEGADOR WEB	WebKit	WebKit/Chromium (Blink)	Trident
TIENDA DE APLICACIONES	App Store	Google Play	Windows Store (antes era Windows Marketplace)
NÚMERO DE APLICACIONES	2.400.000 (sept 2016)	2.000.000 (jun. 2016)	700.000 (oct. 2016)
COSTE PUBLICAR	$99 / año	$25 una vez	$99 / año
OTRAS TIENDAS SIN SUPERVISIÓN	No	Sí	No
FAMILIA CPU SOPORTADA	ARM	ARM, MIPS, Power, x86	ARM
SOPORTE 64 BITS	Sí	Sí	No
MAQUINA VIRTUAL	No	Dalvik/ART	.net

LENGUAJE DE PROGRAMACÓN	Objetive-C, Swift	Java, C++	C#, Visual Basic, C++
PLATAFORMA DE DESARROLLO	Mac	Windows, Mac, Linux	Windows
MULTIUSUARIO	No	Si	No
MODO INVITADO	Si	Si	No

Tabla 2.1 Comparativa de las principales plataformas móviles. Fuente: *[StatistaWeb].*

Para analizar correctamente esta tabla hay que hacerlo con una cierta perspectiva. Para situarse en este contexto se van a matizar algunos puntos a continuación.

Lo primero en lo que hay que fijarse es en la lista de SO que aparecen. Realmente no están todos los que son, ya que también podríamos incluir el SO de BlackBerry pero en la actualizad, y desde hace ya unos cuantos años, no se producen lanzamientos de dispositivos que funcionen sobre este SO. El último terminal con este SO apareció en 2015 llamándose ya por aquel entonces BlackBerry 10, y es que pocos años antes había descontinuado las versiones de su anterior SO denominado BlackBerry OS. A día de hoy, BlackBerry no parece dispuesta a sacar ninguna nueva versión ni realizar ningún mantenimiento de las que ya tiene en el mercado. [BlackBerryD].

Sin dejar de fijar el foco sobre la lista de SO que se enumeran, también resulta importante señalar la situación en la que se encuentra Windows Mobile. El SO que Microsoft utiliza para los dispositivos móviles ha tenido ya algunos altibajos en su corto periodo de vida. Y es que comenzó denominándose Windows Mobile cuando nació, para descontinuarse y pasar a llamarse Windows Phone, y finalmente terminar como Windows Mobile. Además, Microsoft ha dejado de realizar ningún evolutivo más sobre su SO móvil (Dentro de Microsoft, estos paquetes con actualizaciones se denominan *builds*). También ha unificado lo que antes era su tienda de aplicaciones móviles con el almacén de aplicaciones para todas las plataformas, un gesto que denota la postura de Microsoft sobre su línea de desarrollo móvil. [WinMobileD].

Otro punto importante si se entra en las características que se citan, sería el uso de máquinas virtuales en algunas de las plataformas de desarrollo. Se trata de un aspecto a tener en cuenta, ya que va a determinar el tipo de desarrollo que vas a realizar a la hora de implementar aplicaciones para esa plataforma de desarrollo. El uso de la máquina virtual dota de un sistema para traducir las instrucciones escritas en lenguajes que no sean nativos y que necesiten ser pre-compilados para generar un código con un formato compatible con la arquitectura de la unidad de procesamiento central soportada. En este caso, encontramos a iOS, donde se utiliza código nativo a la plataforma de desarrollo, y por otro lado están Android y Windows Mobile que sí utilizan máquinas virtuales. El caso de Android

es algo diferente, ya que también permite desarrollar en un lenguaje nativo aun contando con máquina virtual.

Teniendo en cuenta los datos anteriores, es sencillo deducir las 2 plataformas más significativas en este sector. Su importancia al final se ve reflejada en ingresos para sus respectivas empresas.

La siguiente tabla se encuentra dentro del estudio que realiza anualmente Gartner (empresa consultora del sector TIC) para analizar las ventas de *smartphones*. En dicha tabla se puede ver la cuota de mercado de ventas de *smartphones* organizado por sistemas operativos durante el último cuatrimestre del año 2017 y comparándolo con las ventas del año anterior.

Operating System	2017 Units	2017 Market Share (%)	2016 Units	2016 Market Share (%)
Android	1,320,118.1	85.9	1,268,562.7	84.8
iOS	214,924.4	14.0	216,064.0	14.4
Other OS	1,493.0	0.1	11,332.2	0.8
Total	**1,536,535.5**	**100.0**	**1,495,959.0**	**100.0**

Tabla 2.2 Venta de smartphones en el mundo a usuarios finales por sistema operativo en el último cuatrimestre de 2016 y 2017 (en miles de unidades). Fuente: [Gartner4Qua].

En la tabla anterior se puede destacar como se reparten la cuota de mercado los dos principales SO. Se pueden sacar similitudes comparando estos datos con el número de descargas que poseen las tiendas de aplicaciones propiedad de las empresas responsables de estos SO. Este dibujo del mercado ha ido tendiendo a lo que es hoy en día, donde dos compañías se reparten casi el 100% de la cuota de mercado, teniendo así el monopolio del sector. Es de imaginar lo complicado que es para una empresa entrar en ese sector.

Por lo tanto, las plataformas más interesantes, a nivel de análisis, por todo lo expuesto anteriormente son iOS y Android, ya que son plataformas que se mantienen muy al día de la tecnología y herramientas que utilizan, tienen modelos totalmente distintos y esto se ve marcado al comparar las características de una y otra. De igual manera, el presente estudio tiene como premisa centrarse en los aspectos que tengan que ver con Android, y dejaremos el análisis en profundidad de iOS para otros estudios.

2.4.1 iOS

Apple y su plataforma de desarrollo móvil iOS son el principal competidor de Android en el mercado de las aplicaciones móviles. Posee un modelo de negocio distinto en el que prima la exclusividad y una alta calidad en cada uno de sus productos, tanto software como hardware. Al ser modelos diferentes también resulta interesante para su estudio.

Al igual que Android, iOS también dispone de SDK, que ofrece a terceros para que desarrollen aplicaciones nativas por medio de Swift, uno de los lenguajes oficiales de la plataforma.

La plataforma de desarrollo iOS estructura su arquitectura en una pila dividida en capas o niveles. En la siguiente imagen podemos ver la distribución que tiene la arquitectura de iOS.

Figura 2.4 Arquitectura de iOS. Fuente: [DevelopAndr].

- COCOA TOUCH: Es la capa más importante para el desarrollo de aplicaciones iOS. Posee un conjunto de *frameworks* que proporciona el API de Cocoa para desarrollar aplicaciones. Esta capa está formada por dos *frameworks* fundamentales: UlKit y Foundation.

- MEDIA: Provee los servicios de gráficos y multimedia a la capa superior.

- CORE SERVICES: Contiene los servicios fundamentales del sistema.

- CORE OS: Contiene las características de bajo nivel: ficheros del sistema, manejo de memoria, seguridad, drivers del dispositivo.

Apple da soporte a distintos *frameworks* para poder desarrollar en su plataforma. Organiza estos *frameworks* en capas ordenadas de forma que las capas más bajas actúan más directamente sobre el *hardware*, y a medida que se vaya subiendo se van añadiendo abstracciones construidas sobre las primeras que hace que programar para iOS sea más sencillo.

La arquitectura que presenta es manejada mediante Modelo Vista Controlador (MVC de *Model View Controller*), lo que quiere decir que se hará una petición a un

controlador que tendrá una relación directa con una vista. La respuesta de este controlador será el elemento visual relleno con el modelo de datos solicitado.

2.4.2 Android

La plataforma de desarrollo Android es una pila de software construida con código abierto, lo que quiere decir que es código que está accesible a todo el público y sigue un modelo de colaboración abierta.

Es cierto que en sus inicios Android se autodenominó como una plataforma de *software* abierta y libre. El término de software libre es como se denomina a aquel *software*, que siendo gratuito o no, el usuario que lo adquiere es libre de usarlo, modificarlo y redistribuirlo. Pero dado que Android posee algunos componentes que no siguen está premisa, desde un punto de vista formal no podemos definir con este término la plataforma Android. [FreeVsOpen].

2.4.2.1 Arquitectura de Android

En la siguiente imagen se puede ver como se estructura la arquitectura de la plataforma Android. Esta arquitectura se divide en cinco capas, y como ya se ha comentado anteriormente, se definía como capas basadas en *software* libre.

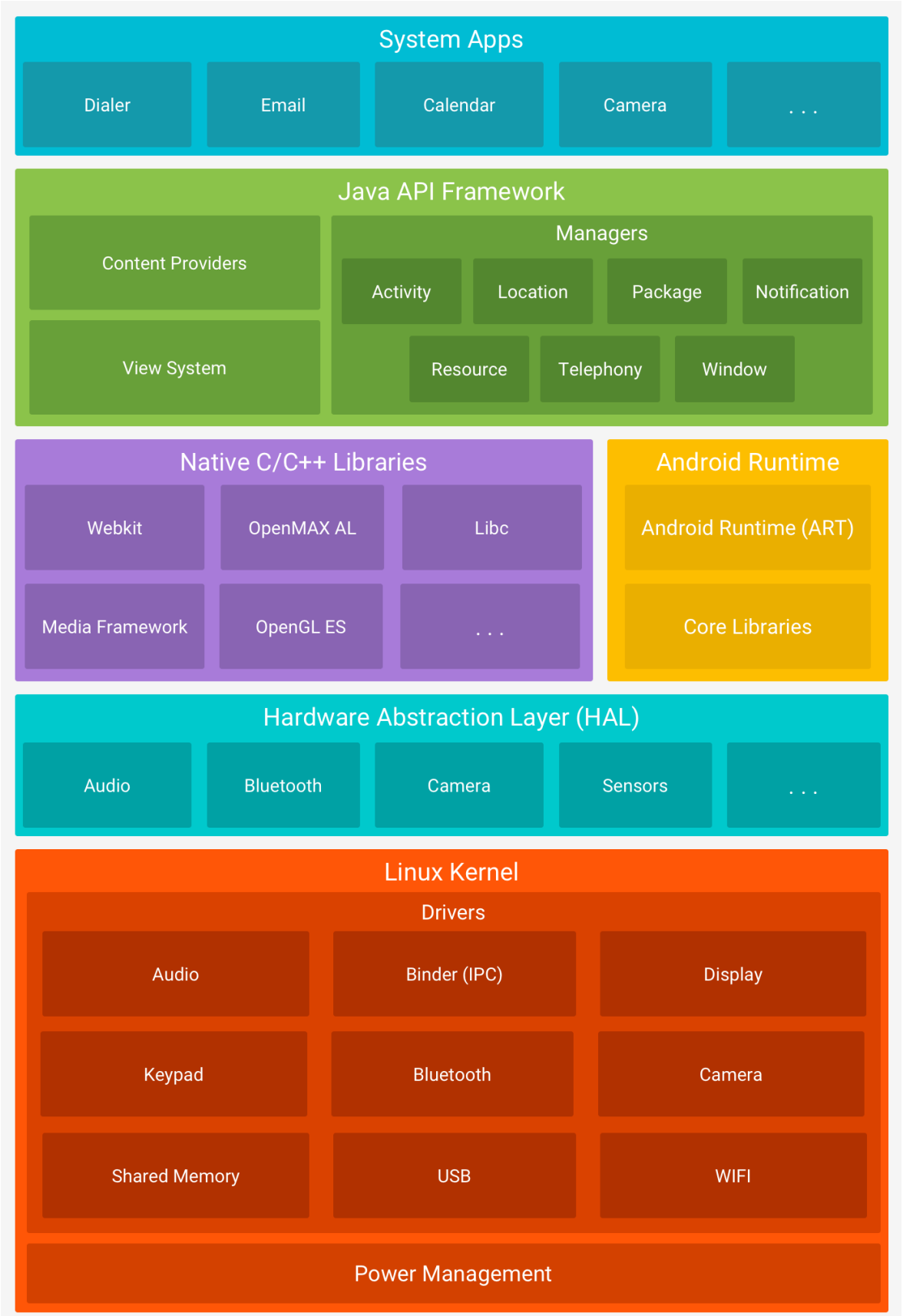

Figura 2.5 Arquitectura de Android. Fuente: [DevelopAndr].

A la hora de analizar la arquitectura de Android para poder entender así el funcionamiento de la plataforma y donde encajaría desarrollada para la misma, se debe conocer el papel que desempaña cada una de las capas. A continuación, se enumerarán y describirán las funciones que realiza cada capa en la arquitectura de Android.

- NÚCLEO: Se trata de la capa más importante, y se encarga de administrar servicios clave como son seguridad, administración de memoria y generación de multiproceso entre otros.
- CAPA DE ABSTRACCIÓN DE HARDWARE: Se encarga de exponer interfaces con las funcionalidades hardware estándares al *framework* de Java situado en el primer nivel de la arquitectura.
- RUNTIME: Se trata del entorno de ejecución que provee Android y se encargan de interpretar lenguaje Java a lenguaje máquina. Está basado en la máquina virtual ART (Dalvik anteriormente). Tiene la capacidad de instanciar varias máquinas virtuales consumiendo pocos recursos. A este mismo nivel también se encuentran librerías del Runtime de Java que serán usadas por el primer nivel. El Runtime se encuentra al mismo nivel que la Capa de Abstracción de Hardware.
- LIBRERIAS NATIVAS: Esta capa incluye un conjunto de librerías en C/C++. Estás bibliotecas son requeridas por capas inferiores y también exponen funcionalidades al primer nivel para usarlas con Java y el SDK de Android, o bien en C/C++ por medio del NDK de Android.
- FRAMEWORK DE LA API JAVA: A este nivel se exponen todas las funcionalidades de Android por medio de un *framework* Java, con un API capaz de gestionar estas funcionalidades. Con el SDK de Android llegamos a disponer de una cantidad de funcionalidades del entorno de ejecución de Java (JRE).
- SISTEMA DE APLICACIONES: Este nivel está formado por las aplicaciones instaladas en el dispositivo Android, las que son propias del SO y las que añada el usuario. Pueden estar escritas en Java o en C/C++. [AndroidArqu].

2.4.2.2 Versiones de Android

Después de haber analizado las plataformas descritas anteriormente, se tiene la información suficiente como para elegir una de ellas. En este punto ya se conoce el lenguaje de programación y tecnología que se va a utilizar para sacar un mayor rendimiento y que cubra las necesidades del desarrollador y también los requerimientos de la aplicación que se va a crear.

Llegados a este punto es importante tener en cuenta a que versión del SO queremos enfocar el desarrollo. Ocurre habitualmente que las empresas poseedoras de un

SO móvil van añadiendo nuevas funcionalidades conforme van actualizando las versiones de su producto.

Al final, el proceso que sigue un SO como producto, no es muy diferente del que sigue otro producto cualquiera dentro del sector del desarrollo. Un producto o solución *software* comienza como una idea que se va plasmando poco a poco. Comienza dibujándose a alto nivel, sin entrar en mucho detalle, utilizando *mockup*[2] y depurando el desarrollo paso a paso. Cuando se llega al hito propuesto para lanzar el producto y se lanza finalmente al mercado, este momento sería el nacimiento de la primera versión del producto.

Si es un producto que se antoja ambicioso, en el que se quiere añadir evolutivos y mejoras, este producto estará sujeto a nuevas mejoras. A medida que vayan añadiéndose estos desarrollos, y llegando a esos hitos para estas nuevas versiones, el producto que se lance al mercado aparecerá con una versión superior a la versión más alta que ya estuviese en el mercado.

Pero no todo es añadir nuevas funcionalidades, sino que también este tipo de productos llevan un gran control y seguimiento a muchos niveles, lo que repercute directamente en que haya un mantenimiento detrás para corregir errores de versiones anteriores, bugs que se detecten por medio de equipos de control o por los mismos usuarios, funcionalidades que se quieran modificar o cambiar para darles un nuevo enfoque, etc.

La siguiente imagen muestra todas las versiones del SO Android hasta la fecha, ordenadas cronológicamente.

Figura 2.6 Historial de versiones de Android. Fuente: [AndroidVers].

[2] *Mockup* es un término con el que denomina a cierta parte de un producto, servicio o funcionalidad, que a la hora de probarlo funciona de manera estática y con datos ficticios, simulando una parte del producto que aún no se ha implementado.

En la figura de arriba se puede ver todas las versiones con sus respectivos nombres comerciales (todos ellos nombres de postres), desde la versión Android 1.0 Apple Pie con nivel de API 1 hasta la última versión que han lanzado, Android 8.1 Oreo con nivel de API 27. [AndroidHist].

En este punto se deben tener varios aspectos en cuenta a la hora de elegir una versión de Android para desarrollar una aplicación.

El primer punto sería entender cómo funcionan los niveles de API en las versiones de Android, ya que una aplicación para móviles Android debe tener establecido por configuración el mínimo nivel de API con el que es compatible. La elección de este valor determina todas las funcionalidades que puede llegar a utilizar una aplicación destinada a la plataforma Android.

Cuanto mayor sea el nivel de API (y, por lo tanto, mayor versión de SO) una aplicación podrá utilizar un mayor número de funcionalidades, ya que para cada nuevo nivel de API que aparece se añaden nuevas funciones y características. Las funcionalidades de niveles inferiores seguirán siendo soportadas, aunque aparecerán marcadas como obsoletas o deprecadas.

Al establecer un nivel mínimo de API para el que se va a desarrollar una aplicación, se limita de manera implícita la versión mínima de SO para la que va a funcionar esa aplicación, ya que los dispositivos que usen una versión por debajo de esa versión mínima que se ha establecido, no podrán instalar la aplicación. La idea que se tiene al realizar esta configuración es la de llegar al mayor número de usuarios posible (cada vez que se suba una versión se estará dejando de llegar al público que utiliza las versiones anteriores) teniendo las mínimas funcionalidades que requiere la aplicación. Esto se conseguiría eligiendo la versión más baja de API que contenga dichas funcionalidades.

En la siguiente imagen se puede observar la pantalla de ayuda que nos proporciona Android Studio previa a la elección del nivel de API mínimo para el que será soportada la aplicación.

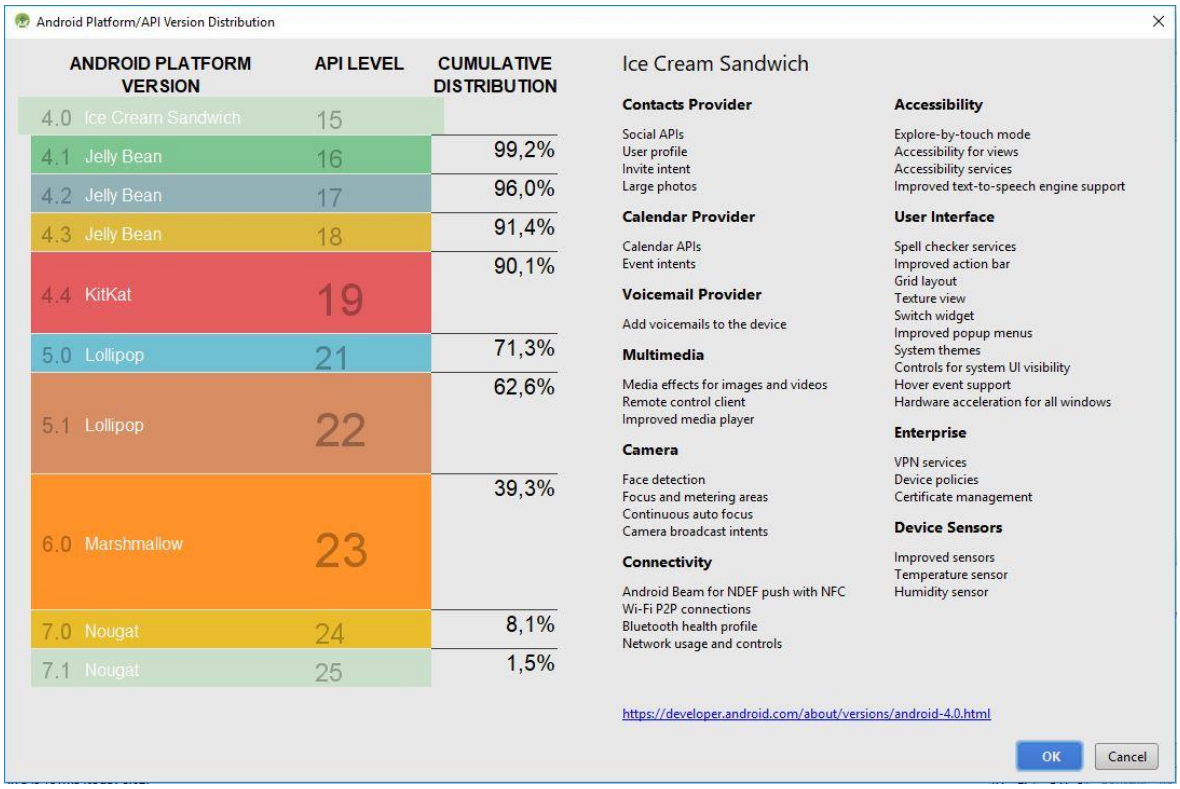

Figura 2.7 Vista de ayuda para configuración del nivel de API en Android Studio. Fuente: [AndroidStudio].

Esta ventana de ayuda sirve de guía para tomar la decisión en función de las características y del tanto por ciento aproximado de usuarios a los que se daría cobertura desde una versión concreta.

El otro punto que se debe tener en cuenta cuando se elige el nivel de API con el que desarrollar son las llamadas librerías de compatibilidad.

Se trata de una característica que apareció en la versión 3.0 de Android y nacen con el fin de incorporar funcionalidades que no trae de serie la versión de API establecida como mínima. De esta manera se compensa que se haya elegido un nivel de API bajo, pensando en la cuota de usuarios. [AndroidBook].

2.4.2.3 Componentes de desarrollo principales de Android

Android provee de una serie de componentes principales que son gestionados y administrados en la capa de la API Java. Entender su funcionamiento es fundamental de cara a una buena utilización de los mismos en el desarrollo de aplicaciones Android.

- VIEW: Hace referencia a los elementos que componen la UI. Todos estos elementos heredarán de la clase View y tendrán su definición mediante código de lenguaje de marcado extensible (XML de *eXtensible Markup Language*) y también su definición como objeto en el lenguaje que estemos desarrollando.

- LAYOUT: Se trata de un conjunto de vistas organizadas de una determinada manera. Existen distintos tipos de *layout* en función de la organización que siguen las vistas que los componen.
- ACTIVITY: Son los elementos básicos de visualización de una aplicación Android. El conjunto de estos elementos conformará la UI de la aplicación. Toda Activity debe pertenecer a una clase que sea descendiente de la clase Activity.
- SERVICE: Son procesos ejecutados en un segundo plano. Estos procesos se ejecutan sin la interacción con el usuario.
- INTENT: Representa la intención de realizar una acción determinada dentro de la ejecución de una aplicación. Los componentes sobre los que se pretende realizar una acción pueden ser internos o externos a la aplicación. También se utiliza para transmitir información entre componentes.
- FRAGMENT: Se trata de componentes visuales que vienen a dar solución al ajuste del diseño a dimensiones de pantalla algo más grandes como pueden ser la de una tablet. Están compuestos por elementos definidos como vistas.
- BROADCAST RECEIVER: Se trata de un receptor de anuncios broadcast que se encuentra escuchando permanentemente este tipo de mensajes. No posee UI, pero puede iniciar una Activity.
- CONTENT PROVIDER: Se trata de un sistema de compartición de información entre las aplicaciones de un dispositivo sin necesidad de comprometer la seguridad del sistema de archivos. [AndroidBook].

2.5 Servicios adicionales para integrar en aplicaciones Android

Después de haber desgranado los aspectos técnicos que ofrece la plataforma Android para desarrollar aplicaciones para su SO, se ha enumerado y analizado en detalle los servicios más comunes e importantes que por norma general caracterizan y dan forma a una aplicación móvil. Siendo este un estudio que trata la forma en que un empresario o desarrollador serían capaces de rentabilizar el ciclo de vida de una aplicación móvil desde que nace hasta que se distribuye para comercializarla, con los costes que ello conlleva. Se ha tratado dar más peso e importancia en la redacción a aquellos servicios que suponen una repercusión económica directa al propietario de la aplicación. Por lo tanto, se ha enumerado esta serie de servicios empezando por aquellos de los que no se obtiene un beneficio directo, y se ha ido profundizando hasta llegar a los servicios imprescindibles para una aplicación móvil a la hora de generar ingresos.

Antes de entrar en detalle sobre cada uno de estos servicios, resulta interesante matizar el concepto de servicios que generan un beneficio económico directo y los que no.

La idea de que un servicio que es capaz de generar ingresos repercute directamente en el beneficio económico del propietario de la aplicación que lo use, es más que evidente. Pero también hay que tener en cuenta los servicios que influyen en estos beneficios de forma indirecta. Por ejemplo, está claro que una aplicación móvil tenga integrado un servicio de autenticación, por poner un ejemplo, no va a repercutir en ingresos a su propietario, ya que esos ingresos se generaran en el momento en que uno de sus usuarios interactúe con publicidad anunciada en la misma aplicación. Pero el servicio de autenticación integrado en una aplicación móvil es un valor añadido de esa aplicación. A la hora de que un usuario decida o no usar esa aplicación, tener un servicio de autenticación fiable y eficiente hará que el usuario instale dicha aplicación en su dispositivo gustosamente. Por lo tanto, se puede decir que estos servicios generan ingresos de forma indirecta, ya que son los que van a atraer a esos usuarios que luego accederán a la publicidad de la aplicación.

2.5.1 Servicios de autenticación

Los servicios de autenticación suelen estar presentes en la mayoría de aplicaciones móviles hoy en día. Es muy importante realizar un sistema de autenticación robusto para que la identidad o datos de nuestros clientes no se vean comprometidos. Esto es especialmente importante si la aplicación móvil maneja datos sensibles como pueden ser datos bancarios.

Para el desarrollo de aplicaciones para Android existen distintas herramientas para realizar esta labor, así que se van a comentar algunas de las más recomendadas.

2.5.1.1 Autenticación haciendo uso del API de Android

Una de las técnicas más recomendadas para realizar la autenticación de usuario en una aplicación Android es utilizar las funcionalidades que expone el API de Android mediante el SDK de turno que vayamos a utilizar.

Android recomienda el uso de su clase AccountManager. Esta funcionalidad apareció en el API 5.0 y se trata de un sistema de gestión de cuentas y creación de las mismas, con la posibilidad de crear tipos de cuenta personalizados.

La manera de trabajar de esta clase es almacenando las credenciales del usuario en el sistema de almacenamiento del dispositivo (la ruta es algo como esto: Environment.getSystemSecureDirectory().getPath()+File.separator+Database_Name).

Evidentemente al pasarle las credenciales de un usuario a esta clase no debe hacerse en claro, ya que se almacenarán en texto plano, deben tener un formato de *token*[3] e ir cifradas de alguna manera para preservar su confidencialidad ante algún posible ataque.

[3] Un *token*, en términos de autenticación, es una firma cifrada que permite identificar un usuario contra un sistema en concreto.

Posteriormente, cuando se realizar el inicio de sesión, se recuperarán las cuentas almacenadas en el dispositivo, que pueden ser tipos de cuentas de entidades como Google, Facebook o Twitter, o de un tipo de cuenta que nosotros creemos.

AccountManager también permite realizar autenticaciones mediante el estándar de autenticación OAuth2 y así permitir a la aplicación comportarse como una entidad de autenticación válida frente a servicios externos que usen este protocolo y autenticarse en nombre de usuario en concreto. [OAuth2.0] [AndAccMang].

La API de Android (desde la 5.0) también expone una clase llamada FingerprintManager. Esta clase, junto con el sistema de almacenamiento de claves, permite realizar autenticaciones mediante el sensor de huellas del dispositivo. [AndroidMarsh].

2.5.1.2 Autenticación por OAuth2 (o OAuth 2.0)

Hoy en día cada usuario posee numerosas cuentas (casi incontables e irrecordables) que utiliza para múltiples fines. Se debe tener también en cuenta que un servicio de autenticación es básico en cualquier servicio proveído por Internet.

Debido a este fenómeno, uno de los métodos de autenticación que más se está usando, y que además es relativamente actual, es el OAuht2.

Se trata de un proceso de autenticación, recomendado por Google en su guía para desarrollo de *software,* que necesita de la intervención de varios actores para generar una relación de confianza y dar por válida la identidad de un usuario. [AuthEndUser].

El flujo de comunicación es sencillo. Desde una aplicación móvil, que funcionará como cliente, deposita la responsabilidad de autenticar a su propio usuario sobre un servicio externo, que hará las veces de servidor de autenticación. El servidor de autenticación (por ejemplo, Facebook, Twitter, etc.) posee su propio servicio de autenticación con las cuentas de sus usuarios. Entonces, estos servidores de autenticación en respuesta a esa petición de validar la identidad de un usuario generan un token con el cual la aplicación cliente, en este caso una aplicación móvil, puede acceder a una determinada información del usuario, además de confirmar que dicho usuario posee una cuenta en un servidor de autenticación en el que confía. [OAuth2Intro].

2.5.1.3 Autenticación de servidor a servidor

Otro de los métodos de autenticación (y autorización) recomendados por Google para el desarrollo de *software* es la autenticación de servidor a servidor. [AuthProduct].

Para esta técnica de autenticación existe un importante cambio en la metodología utilizada. En el uso de esta técnica, no es el usuario de la aplicación móvil el que se somete a la verificación de su identidad, sino que es la propia aplicación móvil la que ocupa su rol en este proceso.

Este mecanismo trata de una comunicación entre dos aplicaciones donde, por norma general, una de ellas provee de ciertos servicios o recursos a la otra, prácticamente

es un patrón cliente-servidor. El primer paso consta de proveer, por un medio seguro, de unas credenciales de autenticación a la aplicación cliente. Estás credenciales son las que usará la aplicación cliente para verificar su identidad. Por lo tanto, en el momento en que la aplicación cliente quiera solicitar dichos servicios o recursos, tendrá que realizar una comunicación previa en la que incluya estás credenciales (cifradas en una cabecera de la petición, por ejemplo) para que la aplicación servidor (que trabajaría como un servidor de identidad) verifique la identidad en su propio sistema. La respuesta de la aplicación servidor incluirá un token de acceso con el que la aplicación cliente podrá hacer uso de los servicios o recursos acordados, ya que también es normal usar este mecanismo como técnica de autorización.

Un ejemplo de uso esta técnica podría ser una aplicación móvil que gestione una serie de documentos nominativos de sus usuarios. La manera de gestionar esos documentos sería a través de un servicio externo y una base de datos documental, por ejemplo. Así que, cuando la aplicación cliente quiera acceder a dichos documentos, tendrá que identificarse antes contra este servicio externo, para poder acceder posteriormente a esos documentos y mostrarlos a cada uno de sus usuarios.

2.5.1.4 Servicio de autenticación externo (Firebase Authentication)

Firebase, la plataforma de servicios que ofrece Google, ofrece un módulo de gestión de distintos tipos de autenticación.

La forma de integrar este servicio en una aplicación móvil Android es haciendo uso del SDK proporcionado por la plataforma. En el caso de este servicio de Firebase, la plataforma ofrece dos opciones a la hora de integrar su servicio en la aplicación móvil. Una de las opciones es por medio del SDK de FirebaseUI. Esta solución se trata de una librería fuertemente ligada con la interfaz de usuario y posee métodos que reaccionan a los eventos generados a la hora de realizar un inicio de sesión desde la UI. Provee de flujos directos de inicio de sesión con distintos proveedores de identidad automatizando en gran medida el proceso de autenticación. La otra opción para integrar este servicio mediante el SDK de Firebase Authentication. Esta librería expone métodos para realizar distintos tipos de autenticación basándose en estándares de la industria (como OAuth 2.0). Permite realizar inicios de sesión con correo electrónico, proveedores de identidad (como Facebook, Twitter, etc.), número de teléfono, entre otros.

Fuera de lo que es la integración con la propia aplicación móvil, Firebase Authentication posee un panel de control con el que llevar una gestión de los usuarios que han realizado el inicio de sesión. Se puede un historial de los inicios de sesión, así como crear, editar y eliminar usuarios. También se puede configurar desde aquí que tipos de autenticación de los que se han comentado anteriormente queremos habilitar o deshabilitar de la aplicación. [FirAuthDoc].

2.5.2 Servicios de mensajería

Una parte importante a tener en cuenta en cualquier aplicación móvil debe ser la comunicación con los usuarios de la misma. De tal forma que, si se desea realizar notificaciones a usuarios en concreto o a segmentos de usuarios que compartan alguna característica en concreto, la aplicación tenga la capacidad de realizar él envió de dicha información.

Los servicios existentes que ofrecen esta funcionalidad suelen utilizar notificaciones *push*. Se trata de una tecnología que permite enviar mensajes de forma directa a los dispositivos que tengan instalada una aplicación móvil. Además, estos servicios de mensajería suelen estar integrados en los servicios de análisis de métricas de las propias plataformas, haciendo uso de ellos para realizar campañas dirigidas a un grupo de usuarios determinado. [PushNotDoc].

2.5.2.1 Servicio externo de mensajería de Amazon (Amazon Pinpoint)

En su plataforma de servicios web (AWS de Amazon Web Services) Amazon cuenta con un servicio de mensajería integrable en todo tipo de aplicaciones.

La forma de tener acceso a este servicio desde una aplicación móvil es por medio de su SDK AWS Pinpoint de Android. Dicho servicio posee varias formas de enviar notificaciones *push* a las aplicaciones móviles que estén integradas con su plataforma de servicios. Una de ellas es a través de la propia aplicación móvil y utilizando el servicio de mensajería de Firebase, Firebase Cloud Messaging. Por lo tanto, además de su SDK, este servicio necesitará del SDK de Firebase Cloud Messaging para utilizar a este último como servidor de mensajería y que el proyecto de la aplicación móvil esté registrado en Firebase. Otra forma de envío de notificaciones que permite este servicio es mediante la consola de la plataforma AWS, donde se pueden crear notificaciones aplicando ciertas configuraciones en la interfaz visual que ofrece, como el sector de usuarios a los que llegará la notificación, entre otras. Como ya se ha señalado anteriormente, este servicio también tiene integrado AWS Analytics, el servicio de Amazon para análisis de métricas, dando la opción a realizar campañas de notificaciones a los usuarios de la aplicación móvil filtrando por las métricas que posee la herramienta.

2.5.2.2 Servicio externo de mensajería de Azure (servicio de notificaciones push de Mobile Apps de Azure)

Microsoft también posee un servicio de notificaciones *push dentro de su plataforma de servicios Azure.*

Su forma de funcionamiento es bastante sencilla. Utiliza otros servicios de mensajería ya integrados en aplicaciones móviles (Firebase Cloud Messaging, Windows Notification Service, etc) para realizar él envió de notificaciones, actuando a como una

aplicación cliente de distintos servidores de mensajería. Posee un panel de control dentro de su sección dedicada a desarrollo móvil, Mobile Apps, donde crear notificaciones y enviarlas a través de los proveedores antes mencionados. [AzurePushN].

2.5.2.3 Servicio externo de mensajería de Google (Firebase Cloud Messaging)

Para implementar esta funcionalidad la plataforma Firebase cuenta con un servicio de mensajería en la nube llamado Cloud Messaging.

Cloud Messaging posee varios métodos para realizar envió de notificaciones a los dispositivos que tengan instalada la aplicación móvil. El primero de ellos se gestiona desde la consola de Firebase, y permite componer mensajes, con y sin datos adicionales, además de otras tantas configuraciones, como establecer los dispositivos de destino de la notificación, fecha de envió, entre otras. La otra forma de envió de notificaciones se realiza por medio del API que Google expone para este servicio. Mediante una petición a su API de tipo POST a través del protocolo de transferencia de hipertexto (HTTP de *Hipertext Transfer Protocol*) de comunicación web, permite realizar él envió de notificaciones desde la aplicación móvil.

Cloud Messaging también posee un apartado para realizar pruebas, con el fin de comprobar el porcentaje de éxito de envió de notificaciones o la versión del SDK de Cloud Messaging (ya que se puede haber usado versiones de SDK distintas para las distintas versiones de la aplicación móvil), entre otros.

Como también ocurría en el servicio de mensajería de Amazon, Cloud Messaging también está integrado con Analytics y permite aplicar distintos filtros a la hora de realizar envió de notificaciones, dando la posibilidad de realizar campañas con distintos fines. [FirClMessag].

2.5.3 Servicios de bases de datos

La mayoría de aplicaciones de hoy en día necesitan almacenar información sobre su negocio, y tener acceso a ella de forma rápida para mostrarla a sus usuarios. Las funcionalidades que requieren las aplicaciones móviles en este aspecto no son diferentes de las que requiere cualquier otra aplicación. Por norma general necesitarán crear, leer, actualizar y borrar esa información (CRUD de *Create, Read, Update and Delete*).

Para solventar estas necesidades los propietarios de las aplicaciones optan por integrar una base de datos para realizar estas funcionalidades.

2.5.3.1 Base de datos SQLite

Se trata de un motor de base de datos pensado para dispositivos móviles. Es ligero y de acceso rápido, características muy valoradas en las aplicaciones móviles.

La base de datos propiamente dicha, se genera en el propio dispositivo de la aplicación. Se trata de un fichero embebido que se almacenará físicamente dentro del

dispositivo móvil. Es una base de datos relacional, lo que quiere decir que sus tablas se definen con un modelo de datos relacional.

Otra característica que se resalta, es que es una base de datos persistente, lo que quiere decir que no se perderán los datos si el dispositivo se apaga en algún momento como ocurriría con la mayoría de procesos en un intento de liberar memoria por parte del dispositivo.

Su rapidez de acceso a los datos se debe a que cuando la aplicación está trabajando con la base de datos, la información que está almacenada en ella está cargada en memoria, lo que posibilita un acceso mucho más rápido para cualquier operación.

La forma de integrarlo en una aplicación Android es por medio de la librería que proporciona la propia plataforma de desarrollo. La librería en concreto se llama "android.database.sqlite" y encontraremos todas las funcionalidades necesarias para trabajar con la base de datos implementadas en la clase SQLiteOpenHelper.

2.5.3.2 Servicio externo de base de datos de Amazon (DynamoDB)

Dentro de los servicios que ofrece Amazon en su plataforma, Amazon Web Services (AWZ), encontramos un servicio de base de datos dedicado a aplicaciones móviles Android, llamado DynamoDB.

La idea de este servicio es ceder la responsabilidad de las tareas de gestión a la propia plataforma de Amazon. Ofrece servicios de mantenimiento, copia de seguridad (*backup*), escalado y cifrado de forma automática o bajo demanda. Todo esto puede gestionarse desde la consola que ofrece la herramienta.

AWS ofrece un servicio de alta disponibilidad debido a los múltiples servidores distribuidos geográficamente. Esto evita posibles fallos de conectividad debido a caídas en servidores de una zona geográfica concreta.

Este servicio consume una base de datos no relacional, o también llamadas *No Only SQL* (NoSQL). Este tipo de base de datos prima por su rapidez de acceso y cortos tiempos de respuesta debido a que usan esquemas de datos poco pesados.

La forma de interactuar con este servicio mediante la aplicación móvil es a través del AWS Mobile SDK de base de datos, que contiene clases para configurar y acceder a la base de datos NoSQL. [AWSNoSQL].

2.5.3.3 Servicio externo de base de datos de Firebase (Realtime Database)

Dentro de la plataforma Firebase, uno de sus servicios más potentes, versátiles y que antes llama la atención es la base de datos en tiempo real.

Se trata de una base de datos no relacional (NoSQL) que trabaja con un esquema de datos parecido al formato JSON. Es una base de datos de acceso rápido debido a su esquema de datos.

Una de sus características más importantes es el concepto de base de datos en tiempo real. Esto quiere decir que cada instancia de la aplicación móvil comparte una instancia de Realtime Database, por lo que la sincronización es automática con la información almacenada en base de datos y cualquier modificación es reportada a cada una de las instancias de la aplicación en tiempo real.

También permite escalado horizontal en varias bases de datos (aunque esta opción no es gratuita) e integración con el servicio de autenticación de Firebase.

Otra de las características interesantes es su modo de trabajo sin conexión, por lo que permite realizar operaciones sobre la base de datos, aunque el dispositivo no esté conectado a Internet, haciendo que los datos persistan en el propio dispositivo a la espera de retomar la conectividad.

Es importante mencionar que este servicio también permite gestionar toda la base de datos desde la consola de Firebase, e incluso añadir reglas de lectura y escritura con expresiones regulares.

La forma de integrar este servicio con una aplicación móvil Android puede ser mediante el uso de su SDK o a través de los métodos que expone la API de Realtime Database. [FirDataBase].

2.5.4 Servicios de análisis de métricas generadas por una aplicación

El análisis de las métricas generadas en una aplicación, tanto si es una aplicación web como una aplicación móvil, es una de las funcionalidades más populares en los últimos años en el mundo del desarrollo de aplicaciones. Solo hay que ver el auge de la llamada minería de datos (Big Data), aunque términos diferentes, son conceptos que están muy relacionados entre sí, ya que los dos buscan la explotación de datos generados por los usuarios de un cierto sistema a mayor o menos escalar.

Lo más normal, es que este tipo de funcionalidades se dejen en manos de servicios externalizados. Estos serán los responsables de representar dichas métricas en gráficas con distintas temáticas, segmentando estás métricas en función de valores como puede ser el sexo, la región, el tipo de dispositivo, etc.

Este tipo de servicios han ido evolucionando y mejorando con los años, y a día de hoy permiten una sencilla integración, una interfaz de usuario muy amena e interactiva y están preparadas para analizar muchas y muy distintas métricas. Es una solución lógica, ya que muchas de ellas permiten un uso gratuito hasta un cierto límite de uso, y cuando el uso que se le quiere dar es el de un flujo de usuarios mayor, adquirir las licencias de estos servicios suele resultar menos costos para una empresa que realizar por ellos mismos o encargar una funcionalidad de este tipo integrada con su aplicación.

En los próximos apartados se enumeran varias de las herramientas de análisis más importantes a día de hoy. [AnalyticTools].

Para comenzar a integrar esta funcionalidad en una aplicación se debe empezar por detectar que indicadores resultan más relevantes por la información que aportan en la interacción de los usuarios con dicha aplicación. Estos indicadores se denominan KPIs, y es un término que ya se ha definido en la sección sobre estrategias de *marketing*, y que también puede reportar información con fines funcionales o técnicos en los que mejorar la aplicación.

2.5.4.1 KPIs relevantes dentro de una aplicación móvil

Según la web Cuatroochenta existen varios indicadores clave a tener en cuenta dentro del análisis de una aplicación móvil:

1. **Usuarios activos diarios**

   Se define como los usuarios que acceden a la aplicación al menos una vez al día. Este indicador puede tener una relevancia variable según la temática y el fin de la aplicación.

2. **Usuarios activos mensuales**

   Indica los usuarios activos al mes para obtener una perspectiva más general.

3. **Duración media de la sesión**

   Este dato indica la duración medía que tiene un usuario en la aplicación y es útil para saber si el usuario está cómodo y la aplicación cumple con su función.

4. **Intervalo de sesión**

   Mide la frecuencia con que un usuario accede a la aplicación.

5. **Dispositivo/Sistema Operativo**

   Indica el dispositivo o sistema operativo que se usa para acceder a la aplicación dando información útil para detectar errores o enfocar desarrollos.

6. **Geo-Segmentación**

   Indicador del porcentaje de usuarios según su región geográfica.

7. **Retención**

   Termino importante tratándose de una aplicación móvil. Indica el porcentaje de usuarios que se fidelizan a la aplicación, quiere decir que habiéndose instalado la aplicación han iniciado sesión en ella.

8. **Valor de tiempo de vida**

   Indica el valor monetario que un usuario ha generado en la aplicación. [480Analytics].

2.5.4.2 Servicio externo de análisis de métricas de App Annie (Intelligence)

La empresa responsable de los análisis estadísticos sobre aplicaciones móviles de forma trimestral durante los últimos años también ha desarrollado un producto de análisis de métricas llamado Intelligence.

Se trata de una solución multiplataforma. Su capacidad de análisis se centra en el rendimiento generado por la aplicación en los almacenes de aplicaciones, por lo tanto, su finalidad es la de ayudar al propietario a convertir su aplicación móvil en lo más ASO *friendly* posible. Para llevar a cabo este propósito se centra en KPIs propios de los almacenes de aplicaciones como el número de descargas, la retención de usuarios, geo-segmentación, entre otros.

Tiene un buen número de herramientas relacionadas con ASO, como pueden ser estrategias con palabras clave, gestión del *feedback* de los usuarios, comparación de estrategias entre aplicaciones rivales y selección de plataformas de publicidad según el perfil de usuario de la aplicación son las características más interesantes que tiene.

Se puede hacer uso de una versión gratuita con datos básicos y también existe otra versión de pago que requiere de una petición individual por cada aplicación donde se desea integrar.

2.5.4.3 Servicio externo de análisis de métricas de Adobe (Analytics)

La solución Adobe Analytics está incluida dentro de la plataforma de servicios enfocados al marketing de Adobe, Adobe Marketing Cloud. Esta herramienta de análisis se gestiona desde la propia plataforma de Adobe, y para hacer uso de ella requiere de la adquisición de una cuenta de pago de Adobe. [AdobeAnalytic].

El panel de control de esta herramienta, denominado Analysis Workspace. En este panel de control el usuario puede monitorizar distintos parámetros de medición, incluidos de forma automática. Las métricas que aparecen de manera inicial en la herramienta son seguimiento de la aplicación por pantallas o secciones, seguimiento de los usuarios (de forma individual), seguimiento de fallos producidos en la aplicación, entre otros. La herramienta es capaz de generar reportes e informes sobre estás métricas en distintos formatos.

Además de las métricas básicas, Adobe Analytics permite generar eventos personalizados y realizar un seguimiento completo de ellos.

También tienen algunas características interesantes, como el hecho de estar integrado con un servicio de aprendizaje automático llamado Adobe Sensei.

Su forma de integrase con una aplicación móvil es similar a la que se ha visto en casos anteriores. Adobe proporciona un SDK para la implementación de sus soluciones en aplicaciones Android. [AdobeRefer].

2.5.4.4 Servicio externo de análisis de métricas de Google (Firebase Analytics)

La solución de Google para análisis de métricas se encuentra dentro del paquete de servicios que ofrece en su plataforma Firebase. Permite ser integrada tanto en aplicaciones móviles como aplicaciones web. Además, es compatible con otros servicios de Google como Admob y Google Tag Manager.

Analytics es uno de los servicios más demandados de la plataforma de Google. Este producto tiene la opción de usar su versión gratuita o la de pago si queremos tener un tráfico de información mejor. Además, es un servicio que permite integrarse con multitud de sistemas tanto de Google como externos. Dentro de Google se integra con AdMob para aportarle métricas interesantes de cara a la monetización de la aplicación.

El módulo que tiene dentro de la plataforma Firebase ofrece una gran variedad de opciones para medir las interacciones de los usuarios en la aplicación donde se integra esta herramienta. La unidad de medida principal de esta herramienta es el evento. Su *dashboard* principal se divide en varios paneles con métricas al detalle sobre diferentes eventos que registra al ser disparados por algún proceso dentro de la aplicación. Estos eventos pueden ser propios de Firebase o personalizados. En estos paneles se pueden aplicar distintos filtros como un perfil concreto del usuario que se quiere mostrar y medir en los paneles según sus propiedades de usuario o bien por audiencias ya definidas.

Los métodos que Analytics propone para caracterizar los usuarios de una aplicación y modelar la misma para que sea más atractiva a usuarios futuros son las propiedades de usuario y las audiencias.

Existen varias formas de caracterizar los usuarios de una aplicación y que esta información sirva para modelar la aplicación a las necesidades de uso que requiere para hacer una experiencia más agradable a los usuarios finales.

Esta solución también se encuentra integrada con el servicio de mensajería descrito anteriormente. Haciendo de más completo este servicio y dotándolo de capacidad de filtrado por audiencias, propiedades de usuario y otras métricas.

2.5.5 Servicios de integración de publicidad

En los apartados anteriores se ha podido analizar distintos servicios que permiten implementar funcionalidades básicas de una aplicación móvil. Servicios de autenticación, base de datos o mensajería proporcionarán funcionalidades que la aplicación va a ofrecer al usuario final de la misma. Sin embargo, estos servicios por sí solos no son capaces de transformar su interacción con los usuarios en un beneficio económico directo para el propietario de la aplicación móvil. Para obtener este beneficio económico las aplicaciones integran dentro de su UI distintos tipos de anuncios que todo tipo de empresas contrata con algún propietario de un servicio de publicidad para que sean expuestos en diferentes sitios.

Estos propietarios de servicios de publicidad (como puede ser Google) facilitan herramientas para la integración de estos anuncios en aplicaciones o sitios web.

El servicio de publicidad de Google, AdSense, permite al desarrollador o propietario de una aplicación gestionar los anuncios que aparezcan en la misma, con un completo sistema de análisis donde poder visualizar el impacto que ha tenido el anuncio, el número

de visualizaciones, los ingresos generados por el mismo, entre otros. También permite llevar un control del beneficio que han generado estos anuncios, además de realizar los cobros correspondientes. [AdMobSite].

Cabe mencionar otro servicio de publicidad propiedad de Google, llamado Google Ad, aunque los dos servicios tengan fines publicitarios, este último está enfocado a crear y colocar anuncios en los resultados de búsqueda de Google, por lo que el contexto es diferente. [AdSenseDoc].

2.5.5.1 Conceptos de monetización de publicidad

Para entender cómo funciona el sistema de anuncios de Google, antes se han definido algunos conceptos para comprender como se mide la monetización de la publicidad.

- Coste por clic (CPC): Es la principal medida a la hora de obtener beneficio económico de un anuncio. Se fija un precio máximo por parte del anunciante y mediante un sistema de pujas se fija este parámetro.
- Coste por mil (CPM): Es el coste que se paga al haberse visualizado mil veces un anuncio de este tipo. El propietario de la aplicación obtiene un pago cada vez que se visualiza este tipo de anuncios. También se fija su precio por sistema de pujas.
- Coste por acción (CPA): El pago de este tipo de anuncios se obtiene cuando el usuario de la aplicación realiza alguna acción más compleja que un clic, como podría ser el registro en un sitio web. También fijado por puja. [CostePClic]

2.5.5.2 Servicio de publicidad externo de Firebase (AdMob)

Como se ha descrito anteriormente, AdSense es el servicio de Google que permite crear anuncios y gestionar sus beneficios mediante la inserción de publicidad en sitios web o aplicaciones.

Pero a la hora de insertar publicidad en aplicaciones móviles, Google provee de un servicio dentro de su plataforma Firebase, llamado AdMob.

AdMob es un servicio que permite integrar anuncios de todo tipo (Banner, Intersticial, Bonificado) en una aplicación móvil. Además, el propio servicio AdMob será el encargado de decidir que anuncio es el que aparece en la aplicación mediante el análisis de las características de los usuarios.

Esta herramienta tiene una sección dentro de la plataforma Firebase, que sirve únicamente para habilitar o deshabilitar las aplicaciones móviles que estén utilizando AdMob, pero se gestiona realmente desde su propio sitio web.

Lo que Google hace con este servicio es sencillo, simplemente encapsula su servicio de publicidad AdSense en un formato que es compatible con aplicaciones móviles. De hecho, la misma cuenta de registro con la que se inicia sesión en AdSense, es compatible con el sitio web de AdMob.

A la hora de integrar los anuncios generados en su plataforma, AdMob provee de un SDK que permite añadir anuncios y proveer del flujo de información a la plataforma sobre la interacción de los usuarios de la aplicación. Estos anuncios se crean desde el propio sitio web de AdMob entre las distintas opciones antes comentadas. Esta acción generará un identificador por cada anuncio que integremos que será el parámetro que utilice AdSense (ya que AdMob usará este servicio) para rellenar las métricas y contabilizar los pagos generados por el anuncio.

3 APLICACIÓN MÓVIL TUTOGUIA FIREBASE INTEGRANDO SERVICIOS DE FIREBASE

De manera paralela al análisis y estudio al que se ha sometido a las plataformas de desarrollo móvil actuales y a los servicios que se pueden integrar en las aplicaciones móviles creadas para las mismas, se ha decido dar una visión más técnica y práctica de lo que se ha plasmado en este documento.

Se ha utilizado la plataforma Android para el desarrollo de la aplicación y así poder aprovechar para plasmar en un proyecto real los conocimientos adquiridos en el estudio de la plataforma.

Como se ha descrito anteriormente, Firebase es una infraestructura de servicios propiedad de Google y hace las funciones de un proveedor de servicios, ofreciendo a los desarrolladores o empresas interesados en el desarrollo de aplicaciones, tanto móviles como web.

Se ha optado por un desarrollo en Java utilizando las herramientas/funcionalidades del SDK de Android que permite interactuar con todas las funcionalidades software y hardware de un dispositivo con SO Android.

A la hora de desarrollar aplicaciones Android en Java existen varios requisitos para poder ejecutarlas correctamente en un ordenador. Los requisitos son tener instalado el SDK de Android para acceder al API de Android, y el JDK y el JRE de Java para poder acceder a las utilidades y herramientas de Java y poder ejecutar aplicaciones en tu ordenador.

Para el desarrollo de esta aplicación se ha utilizado el entorno de desarrollo Android Studio. Existen otros varios entornos de desarrollo que soportan el desarrollo para Android, como pueden ser Eclipse o Visual Studio. Pero ya que Google recomienda Android Studio ha sido este entorno de desarrollo integrado (IDE de *Integrated Development Environment*) el usado.

Por lo tanto, se ha realizado una aplicación Android en la cual se puede interactuar con distintos módulos, donde cada uno de los módulos se corresponde con cada uno de los servicios que se han probado. De esta manera, la aplicación desarrollada externalizará a Firebase la responsabilidad de gestionar estas funcionalidades, haciendo un *backend* más ligero y sencillo, con todas las ventajas que eso ofrece, sobre todo a nivel de desarrollo, ya que permite tener una aplicación muy potente con un desarrollo mucho más sencillo de lo que supondría sin el uso de Firebase.

A su vez, Firebase también tiene unos requisitos previos que se deben cumplir para hacer uso de sus SDK y de las funcionalidades que ofrecen. Los requisitos son los siguientes:

- Un dispositivo con Android 2.3 (Gingerbread) o posterior y servicios de Google Play 9.6.1 o posterior.
- El SDK de los servicios de Google Play proporcionado por el Android SDK Manager.
- Android Studio 1.5 o posterior.
- Un proyecto de Android Studio y su nombre de paquete.

3.1.1 *Primeros pasos con Firebase*

Lo primero que se ha hecho es crear una cuenta de Google. Esto permite tener acceso a la plataforma Firebase para así agregar un proyecto, de forma que quede relacionado de forma única con nuestra cuenta.

Después de haberte logado con dicha cuenta podrás acceder a la página principal de Firebase y crear un nuevo proyecto de la siguiente forma.

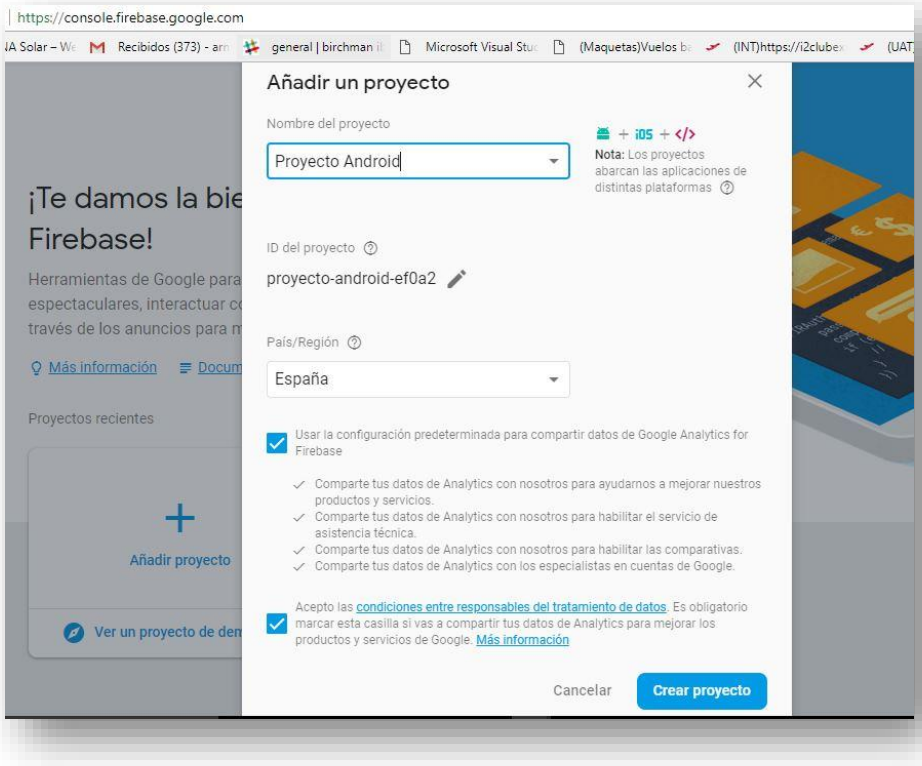

Figura 3.1 Cuadro de diálogo para crear un nuevo proyecto en Firebase

En este punto debemos dar un nombre al proyecto que estamos creando y la plataforma nos generará un identificador único del proyecto. Una vez creado el proyecto, el siguiente paso es crear la configuración necesaria para que nuestra aplicación pueda hacer uso del SDK de Firebase.

En la siguiente imagen se muestra la información necesaria para el archivo de configuración.

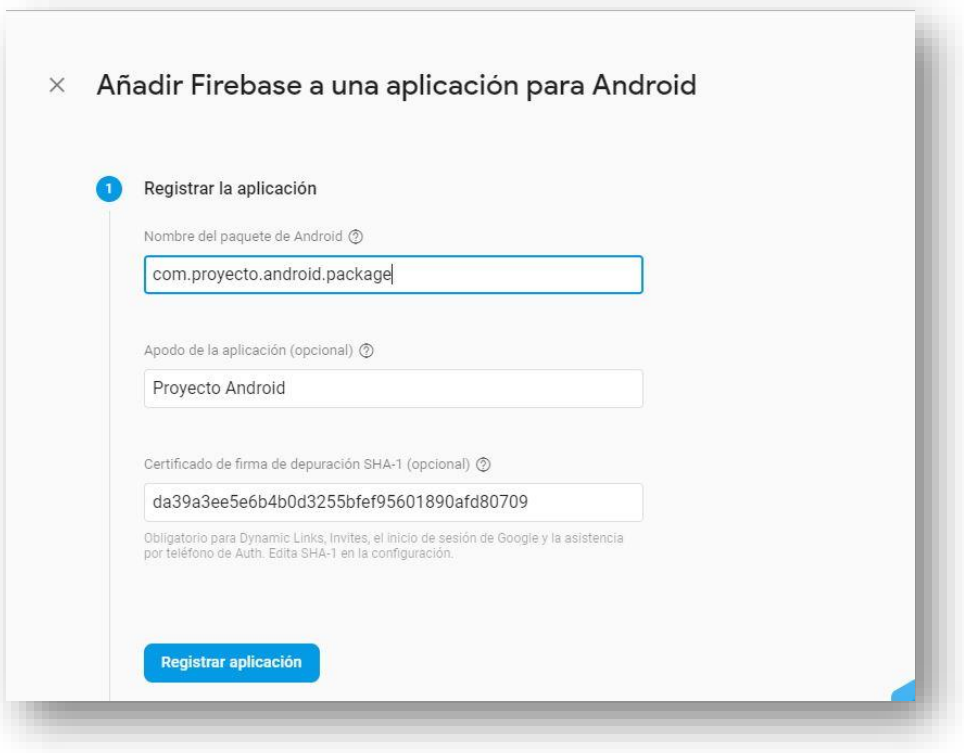

Figura 3.2 Cuadro de diálogo para registro de la aplicación Android donde vamos a integrar los servicios de Firebase.

Lo siguiente, habiendo rellenado los datos anteriores que solicita Firebase, se generará un fichero llamado *google-services.json*. Este archivo en formato JSON le sirve a Firebase para administrar todas las credenciales y configuraciones de su API. En la siguiente imagen se puede ver el cuadro de dialogo donde Firebase solicita incluir ese fichero dentro del directorio del proyecto Android.

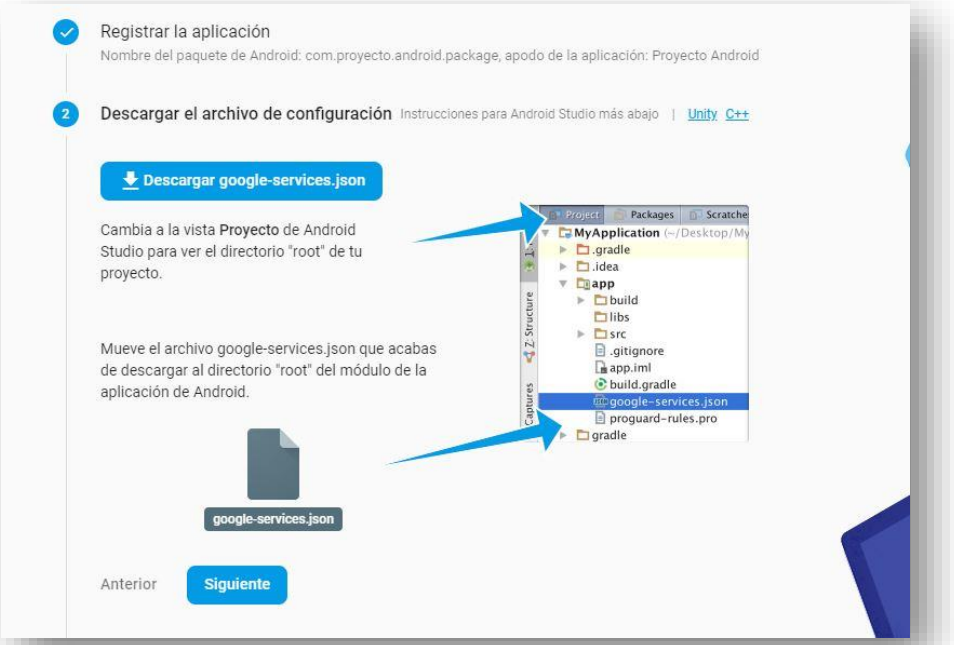

Figura 3.3 Cuadro de dialogo para descargar el fichero google-services.json

El último paso es la importación del SDK de Firebase y el SDK particular de cada uno de los servicios de la plataforma como se muestra en la siguiente imagen.

Figura 3.4 Instrucciones para añadir el SDK de Firebase a un proyecto Android.

Tras haber realizado estos pasos previos, lo siguiente será acceder a la consola de Firebase. Este es el punto principal desde donde se podrá administrar y gestionar todos los servicios disponibles en la plataforma.

En los siguientes puntos se ha descrito algunos de los servicios de la plataforma Firebase que se han integrado en la aplicación TutoGuia Firebase.

3.1.2 Integración del servicio de autenticación

En el ejemplo funcional que se ha desarrollado en paralelo a la redacción de este documento se ha incluido la integración del servicio de autenticación que Google ofrece en la plataforma Firebase.

Posee un *dashboard* donde poder administrar los usuarios creados por medio de la librería que provee de dichos servicios. En la siguiente imagen se puede ver el *dashboard* principal del módulo Authentication de Firebase.

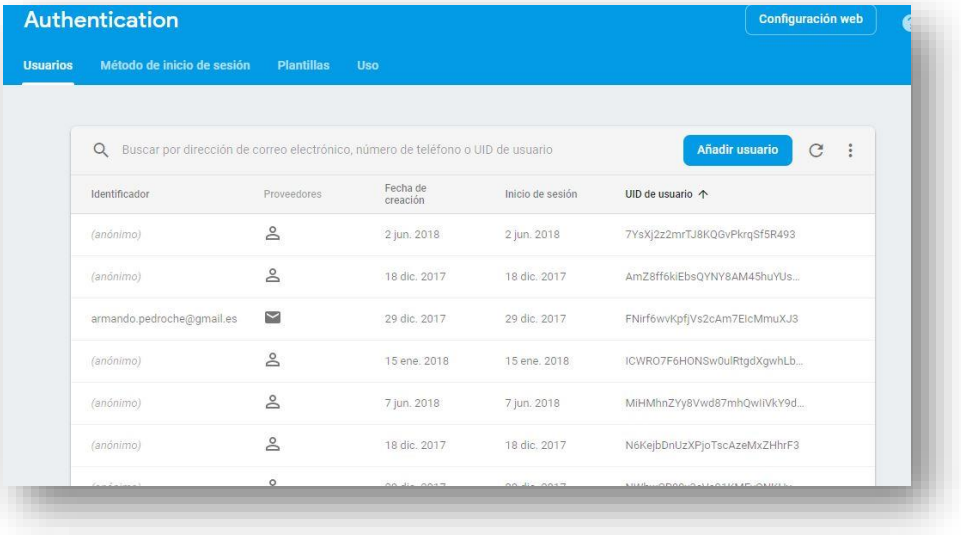

Figura 3.5 Dashboard principal del módulo Authentication. Fuente: [ConsoleFireb].

También ofrece una gran variedad de métodos con los cuales permite dar la opción a los usuarios de iniciar sesión por medio de una aplicación que integre este servicio.

En la siguiente imagen se observan los distintos métodos disponibles de inicio de sesión en la plataforma Firebase.

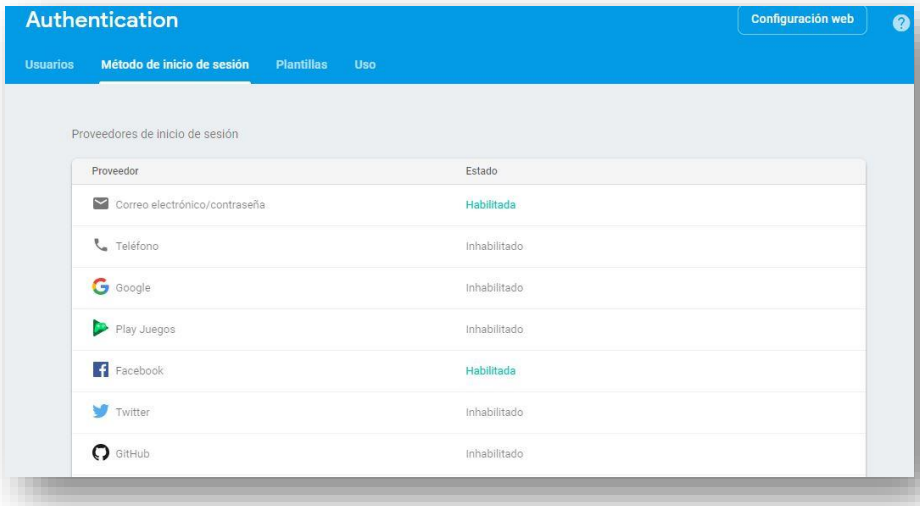

Figura 3.6 Pestaña de selección de métodos de autenticación. Fuente: [ConsoleFireb].

Hacer uso de este servicio consta de importar el SDK de Firebase donde implementa las interfaces con las funcionalidades para crear y verificar usuarios, iniciar sesión con los mismos, etc. El SDK para hacer uso de estas funcionalidades es “com.google.firebase:firebase-auth”.

Este SDK provee de varias clases abstractas, las cuales pretenden ser interfaces de los servicios que exponen ya que se usan de similar manera. Para cargar de dependencias la clase en la que se gestiona la actividad, se ha creado una clase aparte para estas funcionalidades. En esta clase se han inyectado las clases abstractas FirebaseAuth y FirebaseUser.

La siguiente imagen corresponde al fichero Authentication.java, que es la clase que implementa de los servicios de autenticación de Firebase.

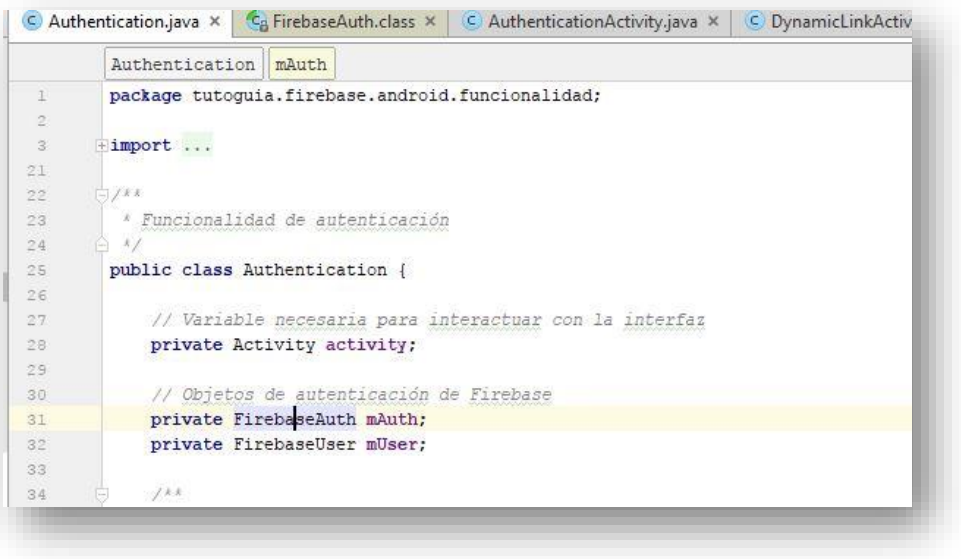

Figura 3.7 Implementación de servicio de autenticación usando el SDK de Firebase. Fuente: Android Studio

A partir de haber inyectado estos servicios se puede hacer uso de los métodos que implementan para crear usuarios y verificar los inicios de sesión de los mismos.

En la siguiente imagen se puede ver el uso del método createUserWithEmailAndPassword() para crear y registrar un usuario en Firebase.

Figura 3.8 Implementación de un servicio propio de creación de usuario usando el SDK de Firebase. Fuente: Android Studio

Este método realiza un registro en la tabla de usuarios que provee Firebase en su sección de Authentication y espera para ver si la acción se ha realizado correctamente para generar en la Activity correspondiente un mensaje de éxito o de error.

Se puede ver en la siguiente imagen la función encargada de iniciar una sesión verificando las credenciales del usuario.

Figura 3.9 Implementación de un servicio propio de creación de usuario usando el SDK de Firebase. Fuente: Android Studio

Para realizar la función del inicio de sesión mediante otro método de autenticación se ha utilizado Facebook como proveedor externo de autenticación, delegando así la responsabilidad de verificar las credenciales a su sistema.

Lo primero que se debe hacer es registrarse como desarrolladores de Facebook (https://developers.facebook.com/). Esto permite hacer uso del servicio de Facebook para hacer un inicio de sesión contra él.

Existen varias formas de integrar este servicio en una aplicación Android. En esta aplicación se ha utilizado el LogginButton que suministra Facebook en su SDK.

Se trata de un botón en la UI con métodos preparados para gestionar los inicios de sesión contra Facebook. Por medio de la clase CallbackManager es capaz de administrar una llamada de vuelta del inicio de sesión de Facebook realizado al pulsar el botón. En la siguiente imagen se puede ver la forma en la que se define este elemento.

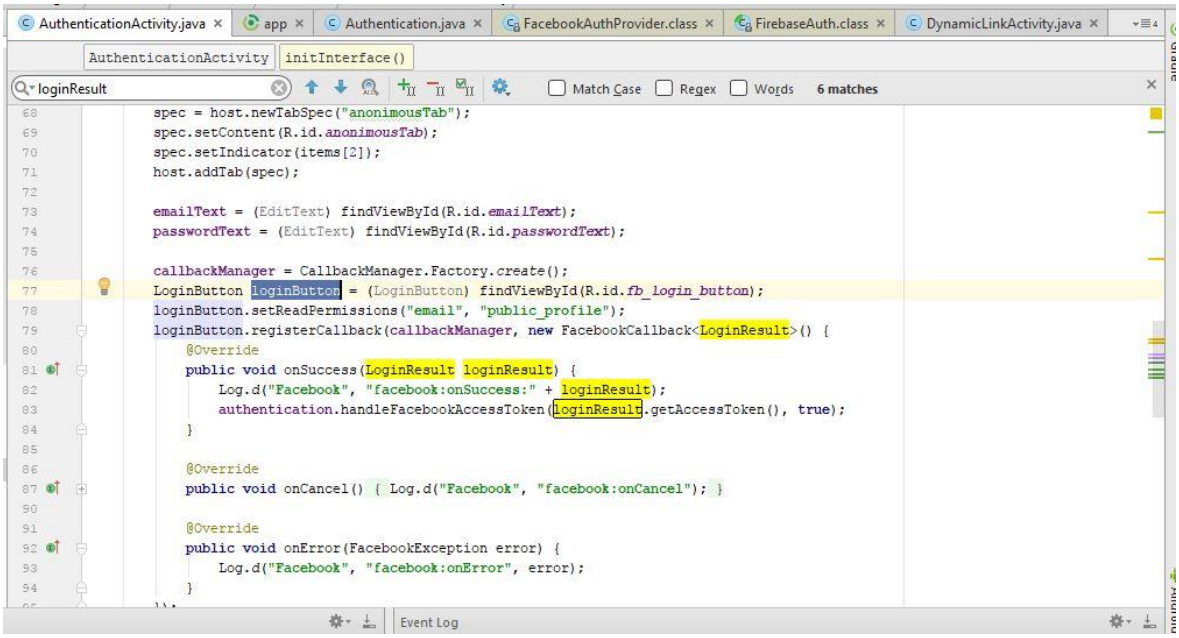

Figura 3.10 Redefinición de los métodos onSucess, onCancel y onError del botón de registro de Facebook.

Por lo tanto, cuando Facebook devuelva la llamada con el resultado del *login* en su sistema llamara a la función RegisterCallBack() y se recogerá la respuesta en alguno de los métodos que han sido sobrescritos según sea un *login* exitoso, un *login* erróneo o se haya producido algún error de otra índole.

En caso de producirse un *login* correcto la llamada entrara por la cláusula onSucces y llamara al manejador para el *login* que se ha creado en la clase Authentication.java.

En la siguiente imagen se puede ver la lógica implementada en el método handleFacebookAccessToken().

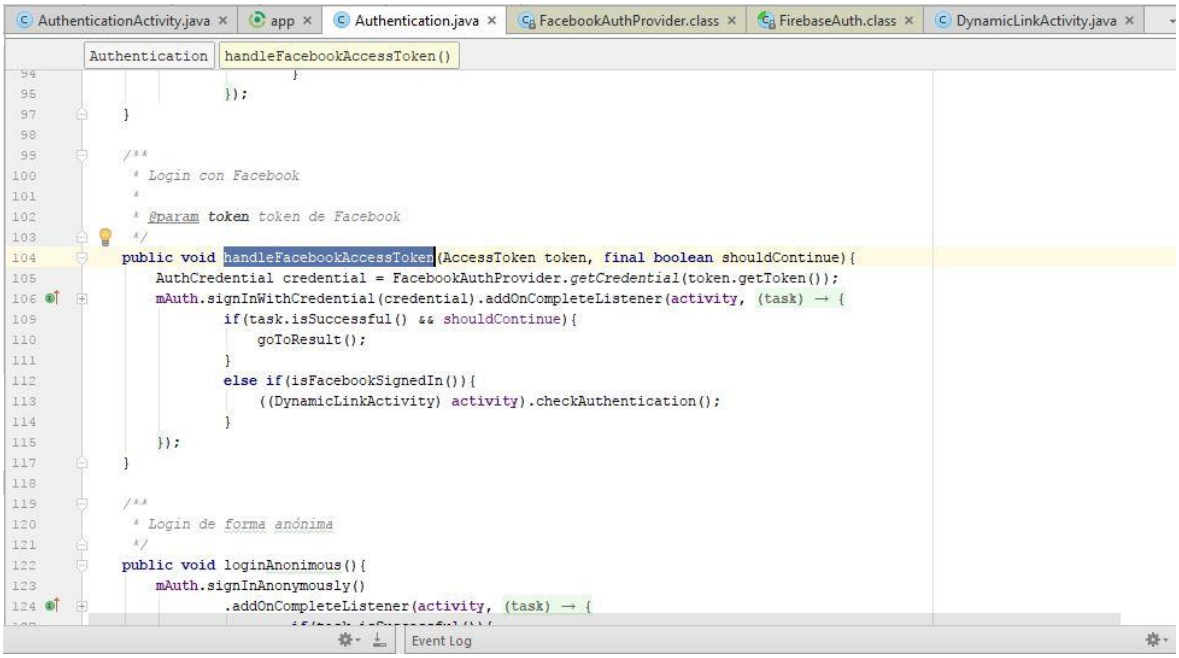

Figura 3.11 Implementación de método para manejar el login a través de Facebook

Mediante el token que genera Facebook se tendrá acceso a las credenciales con las que se hará posteriormente el inicio de sesión como si de un usuario más registrado en Firebase se tratara.

3.1.3 Integración del servicio de base de datos en tiempo real

En la aplicación se ha desarrollado un módulo que hace las funcionalidades de una base de datos, y para ello se ha utilizado el servicio de base de datos en tiempo real de Firebase.

Se han definido un par de tablas de las entidades alumno y profesor, y se han rellenado con datos de prueba. En la siguiente imagen se puede observar la estructura que siguen las tablas en Firebase.

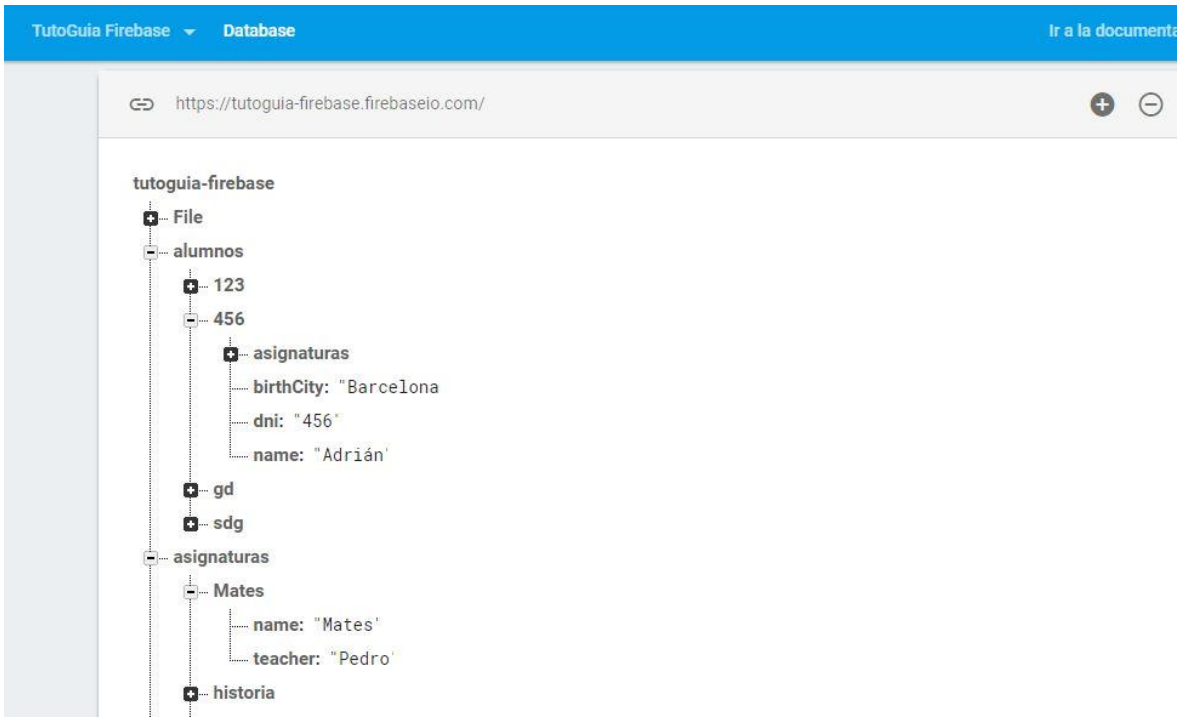

Figura 3.12 Dashboard principal del módulo de Database de Firebase

Después de importar el SDK correspondientes, para hacer uso de este servicio basta con inyectar la clase FirebaseDatabase y obtener una instancia de la misma para poder usar los métodos que expone a través del SDK. Esto se ha hecho en una clase llamada Database.java que es una implementación propia de acceso a la base de datos de Firebase por medio de su SDK.

Entre otras cosas, se puede tener acceso a los datos que se almacenan en las tablas usando el método getChildren() de la instancia de FirebaseDatabase como se ve en la siguiente imagen.

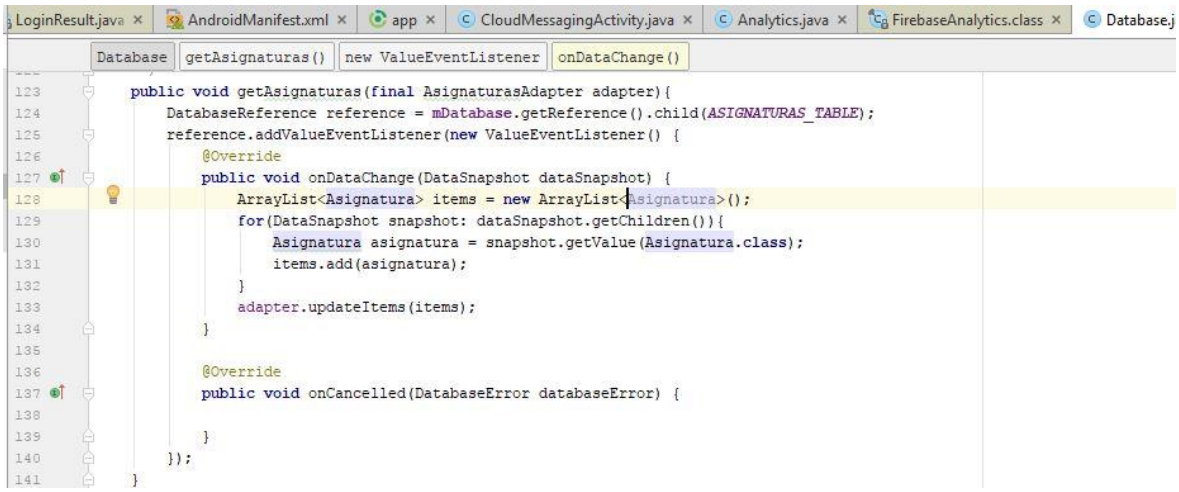

Figura 3.13 Implementación para obtener los elementos de la tabla Asignaturas

También permite crear registros de una entidad ya definida en la base de datos de Firebase creando un DatabaseReference, de la entidad en concreto, con el método child() y pasándole como argumento un nuevo identificador. Esto devolverá una referencia de base de datos con la cual se puede establecer los valores de la entidad creada. A continuación, se observa la implementación del método encargado de crear un nuevo elemento en la tabla de Asignaturas.

Figura 3.14 Implementación para crear un elemento en la tabla Asignaturas

Otra de las características que ofrece la clase DatabaseFirebase es la capacidad de construir consultas mediante métodos de ordenación de elementos, comparación, etc. En la siguiente imagen se puede ver la sintaxis de las consultas.

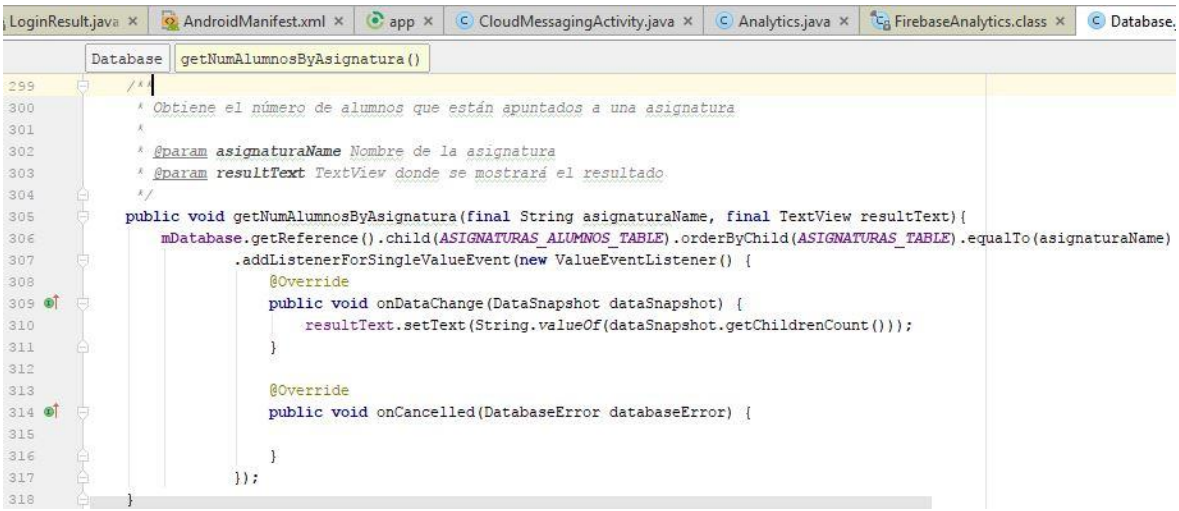

Figura 3.15 Implementación para realizar una consulta contra la tabla que relaciona asignaturas y alumnas

3.1.4 Integración del servicio de almacenamiento

Otro servicio bastante interesante que ofrece Firebase es una cuenta de almacenamiento donde se pueden alojar ficheros de una gran variedad de formatos.

Firebase cuenta en el SDK correspondiente con varias clases para administrar la subida y bajada de archivos desde su plataforma. Estas funcionalidades se implementan en las clases FirebaseStorage y ReferenceStorage. Esto es debido a que tiene un funcionamiento parecido al del servicio de base de datos, que debe generar una referencia a la entidad que quiere crear o actualizar para ejecutar la acción en concreto.

En la siguiente imagen se puede observar la forma de inyectar las clases del SDK para el servicio de almacenamiento de Firebase.

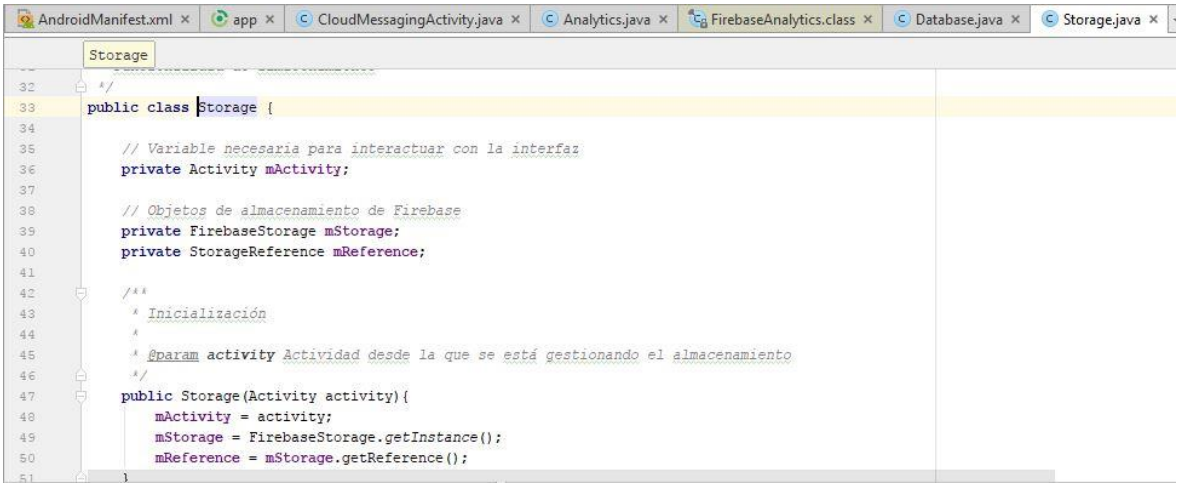

Figura 3.16 Definición de la clase Storage encargada de la gestión de la sección de almacenamiento

Después de obtener una referencia a la cuenta de almacenamiento de Firebase, ese objeto de StorageReference que se ha inyectado tiene la capacidad de actualizar el registro de archivos de la cuenta de almacenamiento y subir un archivo referenciándolo por la ruta donde está alojado. En la siguiente imagen se muestra el método encargado del proceso de subida de un fichero.

Figura 3.17 Implementación para realizar la carga de un fichero

Los ficheros pueden cargarse a la plataforma Firebase por medio del SDK o también en la interfaz de usuario que tiene la consola de Firebase.

A continuación, con la clase FirebaseStorage se tendrá la funcionalidad de descargar un fichero en concreto por medio de su ruta en Firebase, que actuará como identificador único. El siguiente método implementa la lógica para descargar un archivo.

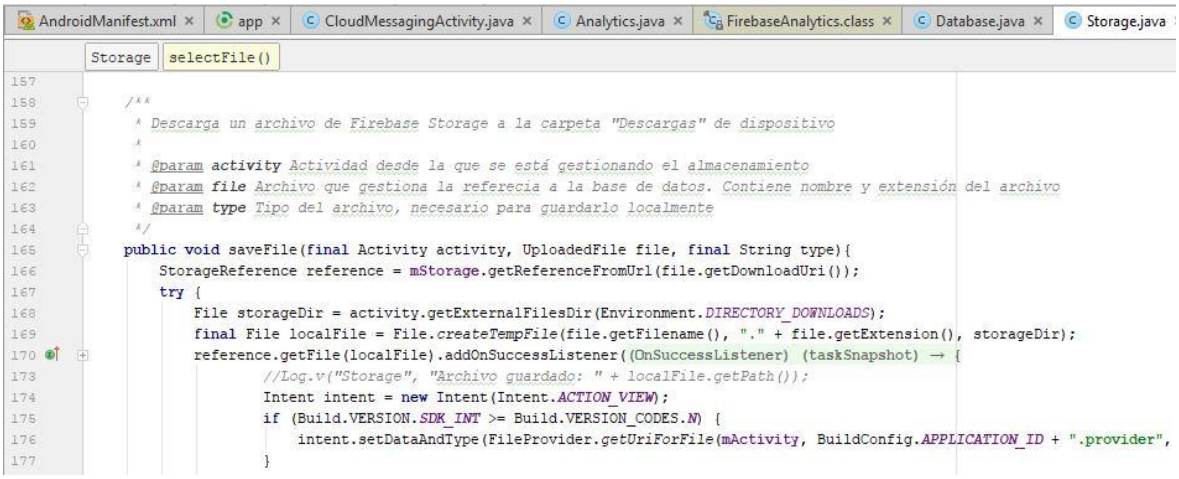

Figura 3.18 Implementación para guardar un fichero al realizar la descarga desde TutoGuiaFirebase.

También tiene otras funcionalidades, como la de obtener los metadatos de un archivo almacenado en la plataforma.

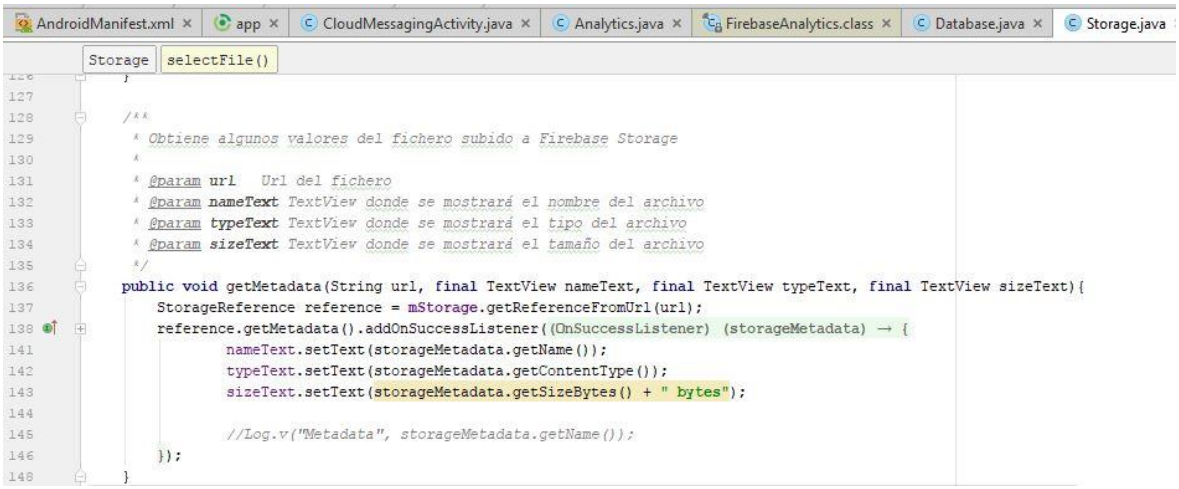

Figura 3.19 Obtener información de los metadatos de un archivo. Fuente: [AndroidStudio].

3.1.5 Integración del servicio de mensajería en la nube

Se ha realizado la integración con Cloud Messaging, un servicio de mensajería en la nube que ofrece Firebase.

Ofrece distintos métodos de uso para el envío de notificaciones. Abarca las plataformas de desarrollo móvil iOS y Android, y también permite su utilización mediante estándares web HTTP y JSON por medio de la API que expone.

Con el fin de poder enviar notificaciones desde la propia aplicación Android, se ha desarrollado una pequeña aplicación web a modo de API para permitir la comunicación con la API propia de Cloud Messaging.

Para él envió de notificaciones desde la API de Cloud Messaging es necesario añadir en la petición el token que identifica de forma única cada instancia de la aplicación móvil.

Haciendo uso de la clase FirebaseInstanceId se puede obtener el token de cada instancia y recibir uno nuevo en caso de que este se haya quedado obsoleto.

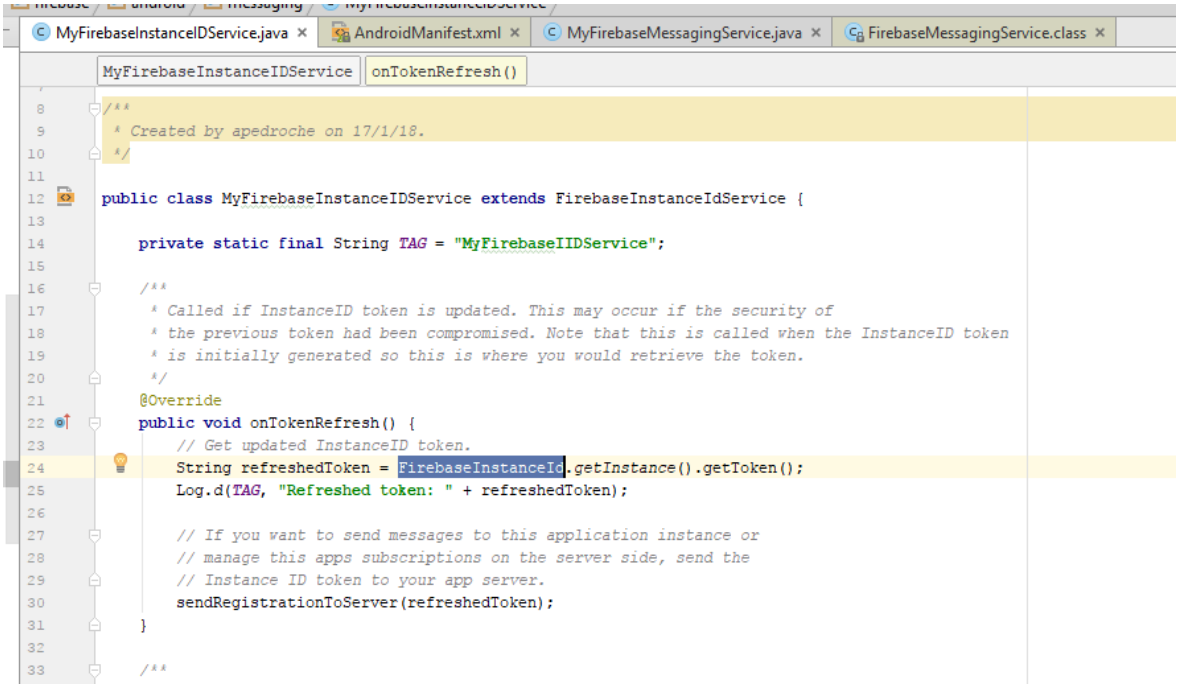

Figura 3.20 Método responsable de recoger el token cada vez que se genera o actualiza.

De esta forma, la aplicación construirá una petición con los parámetros básicos de una notificación, y seleccionando como destino la aplicación identificada por su token.

En la siguiente imagen, se puede observar los parámetros que componen la petición a la API de Cloud Messaging.

JSON representation

```
{
  "name": string,
  "data": {
    string: string,
    ...
  },
  "notification": {
    object(Notification)
  },
  "android": {
    object(AndroidConfig)
  },
  "webpush": {
    object(WebpushConfig)
  },
  "apns": {
    object(ApnsConfig)
  },

  // Union field target can be only one of the following:
  "token": string,
  "topic": string,
  "condition": string
  // End of list of possible types for union field target.
}
```

Figura 3.21 Parámetros de una petición a la API de Cloud Messaging. Fuente: [ConsoleFireb].

El módulo en la consola de Firebase de Cloud Messaging también posee una interfaz visual con la que permite configurar y componer notificaciones. Tiene 2 tipos de notificaciones, las que poseen una configuración básica y no permiten enviar datos, y las notificaciones avanzadas que poseen mayor variedad de posibles configuraciones y además permiten el envío de datos en un formato parecido a un diccionario clave valor.

En la siguiente imagen se observa la pestaña de configuración de las opciones de una notificación básica.

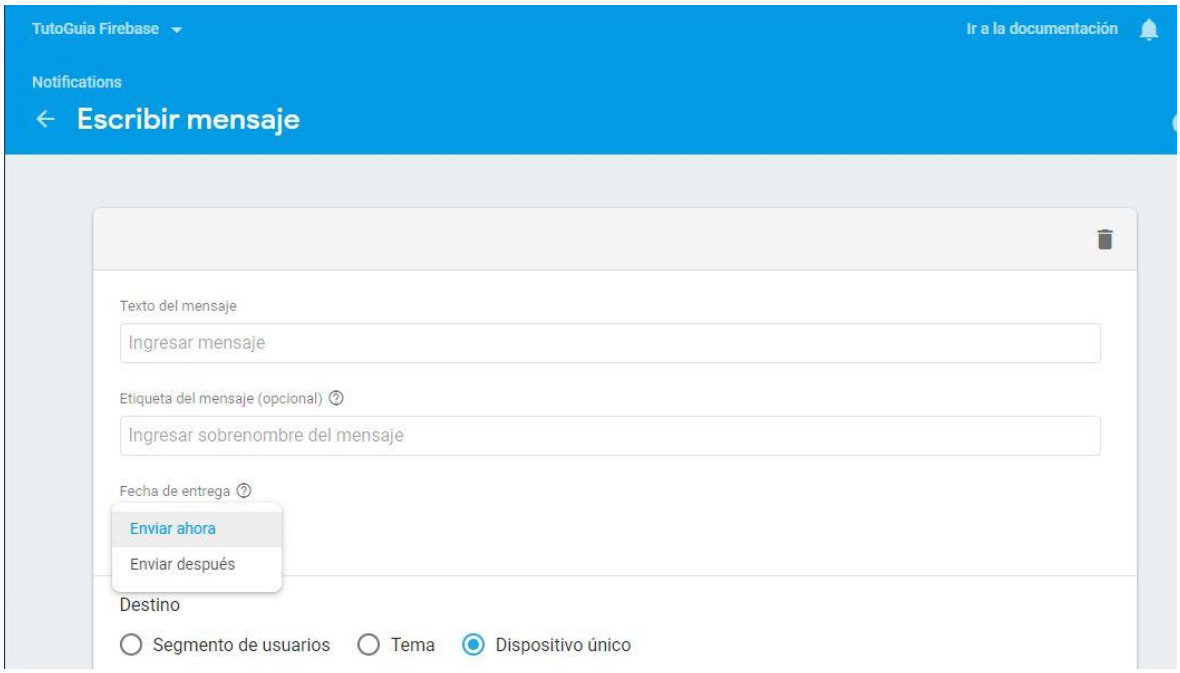

Figura 3.22 Vista de página de configuración de una notificación básica desde Firebase. Fuente: [ConsoleFirebase].

Se puede componer un mensaje sencillo y seleccionar el público de destino de las opciones disponibles. Puede seleccionarse un segmento de usuarios, un tema al que estén suscritos por medio de la aplicación o a un dispositivo único identificado por un token, que se genera automáticamente en cada instancia de la aplicación, y que es posible recoger mediante el uso del SDK de Cloud Messaging.

En una zona plegada, debajo de las opciones de configuración de las notificaciones básicas, se puede encontrar las opciones de las notificaciones avanzadas. En la siguiente foto se observa el panel de configuración disponible.

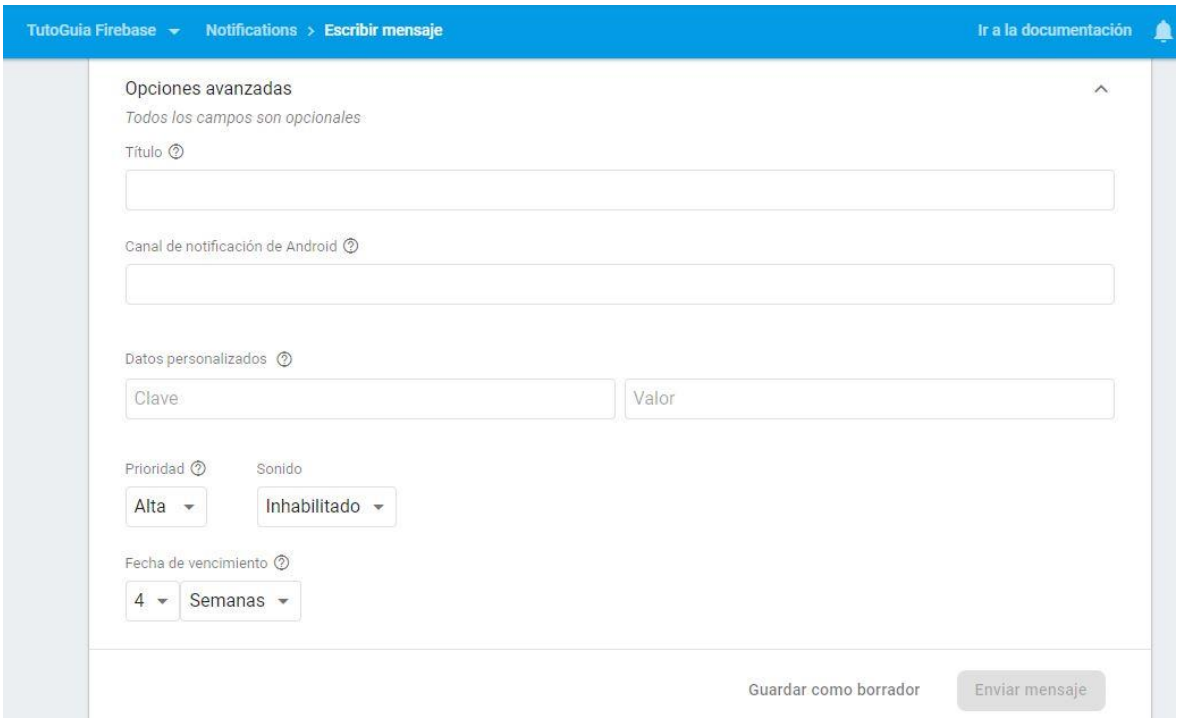

Figura 3.23 Vista de página de configuración de una notificación avanzada en Firebase. Fuente: [ConsoleFirebase].

En el panel se observan las posibles configuraciones y la información que permite adjuntar a la notificación que se envíe.

Además, Firebase guarda un registro de las notificaciones que has mandado a modo de log para poder llevar un seguimiento de las notificaciones enviadas. Este historial de notificaciones es posible verlo en la pantalla principal de la sección Cloud Messaging. Registra junto a las notificaciones algunos detalles de interés como el estado o si ha sido abierta.

En la siguiente imagen se puede ver un registro de notificaciones en el dashboard de Cloud Messaging.

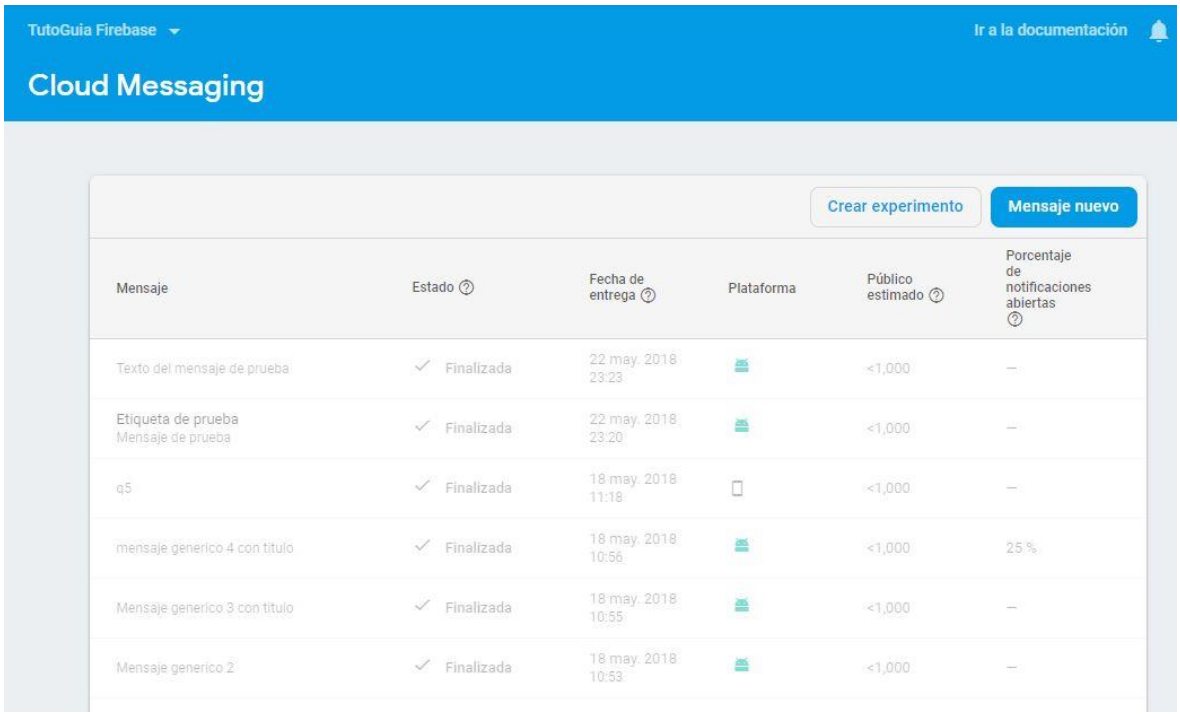

Figura 3.24 Dashboard principal del módulo de notificaciones con el registro de las notificaciones enviadas. Fuente: [ConsoleFirebase].

3.1.6 Integración del servicio de análisis de métricas

Estos servicios que tan de moda se encuentran en la actualidad son imprescindibles a la hora de optar a que la aplicación desarrollada se perfile y modele conforme la información que va recibiendo de sus propios usuarios rellene las métricas de este tipo de servicios.

Como ya se ha comentado anteriormente, esta herramienta posee un *dashboard* a modo de panel de control, donde encontramos los paneles con las métricas principales.

En la siguiente imagen se ve uno de los paneles en el *dashboard* principal donde registra los usuarios activos de la aplicación frente a una línea de tiempo.

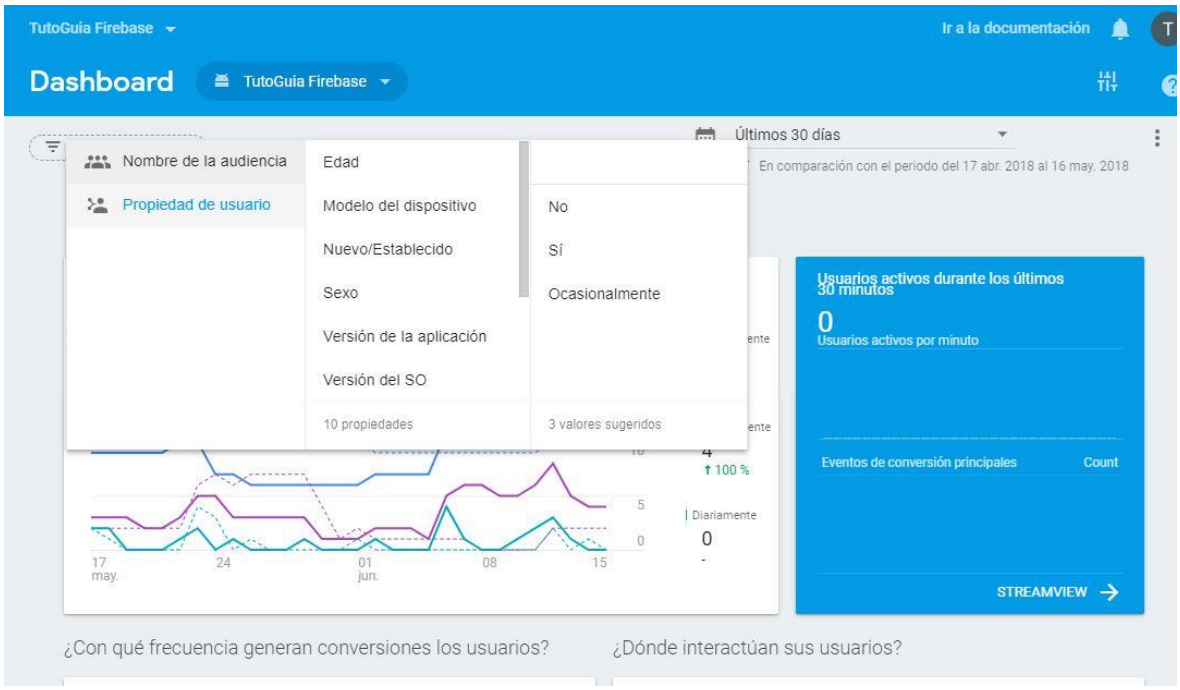

Figura 3.25 Dashboard principal del módulo Google Analytics. Fuente: [ConsoleFirebase].

Existe también un submenú dentro del módulo de Analytics que muestra una lista de los eventos que se encuentran en uso en la aplicación con datos de usuarios y eventos lanzados desde la aplicación. La siguiente imagen muestra el registro de eventos dentro de Analytics.

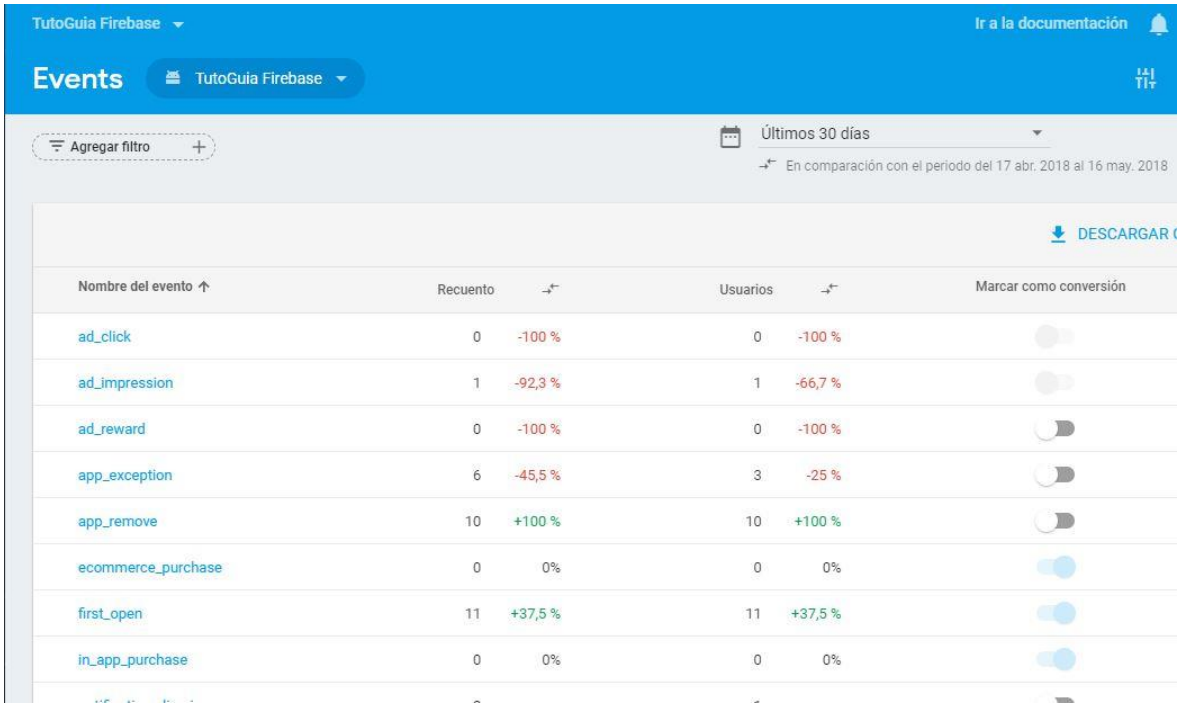

Figura 3.26 Pantalla de resumen del registro de eventos de Google Analytics. Fuente: [ConsoleFirebase].

Como se ha descrito en apartados anteriores, en Analytics existen varias formas de caracterizar los usuarios de una aplicación y que esta información sirva para modelar la aplicación a las necesidades de uso que requiere para hacer una experiencia más agradable a los usuarios finales.

Las propiedades de usuario permiten crear características que puedan perfilar a los usuarios de la aplicación con una sería de valores a elegir, lo que da una segmentación según estás condiciones de los usuarios. La siguiente imagen muestra el panel de configuración de las propiedades de usuario.

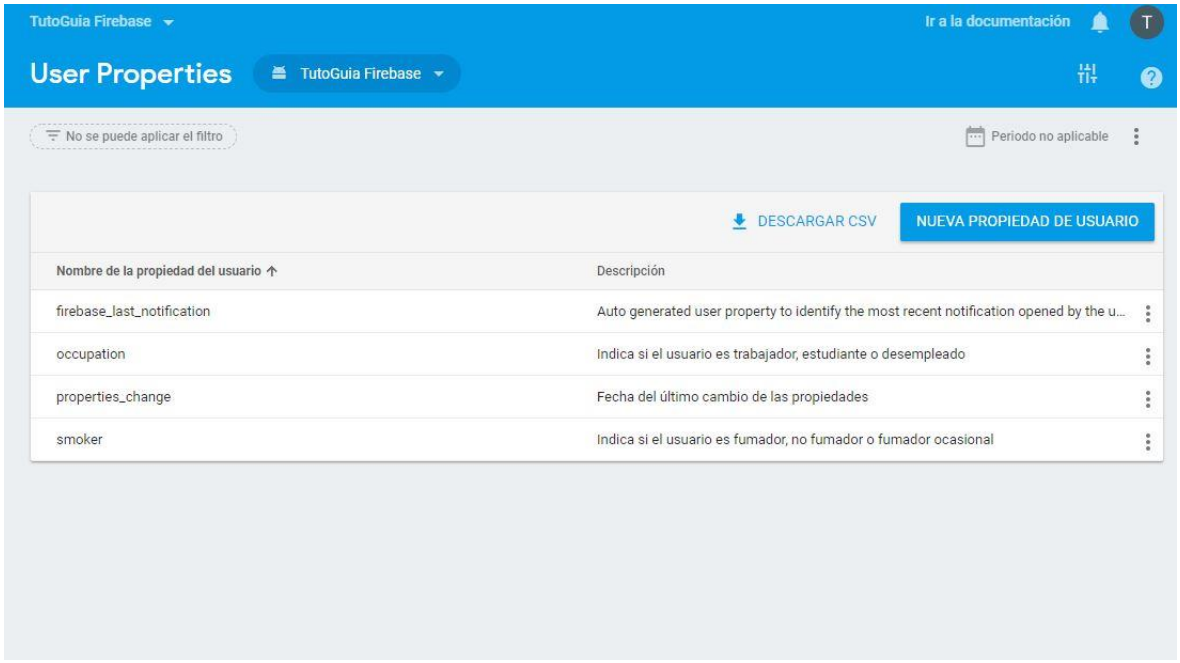

Figura 3.27 Panel de configuración de propiedades de usuario en Google Analytics. Fuente: [ConsoleFirebase].

Las audiencias tratan de definir estos segmentos por medio de eventos o por las mismas propiedades de usuario. Definen audiencias con unas características en concreto medidas por estos valores. Se puede observar varios tipos de audiencias definidos en el panel de Firebase en la imagen a continuación.

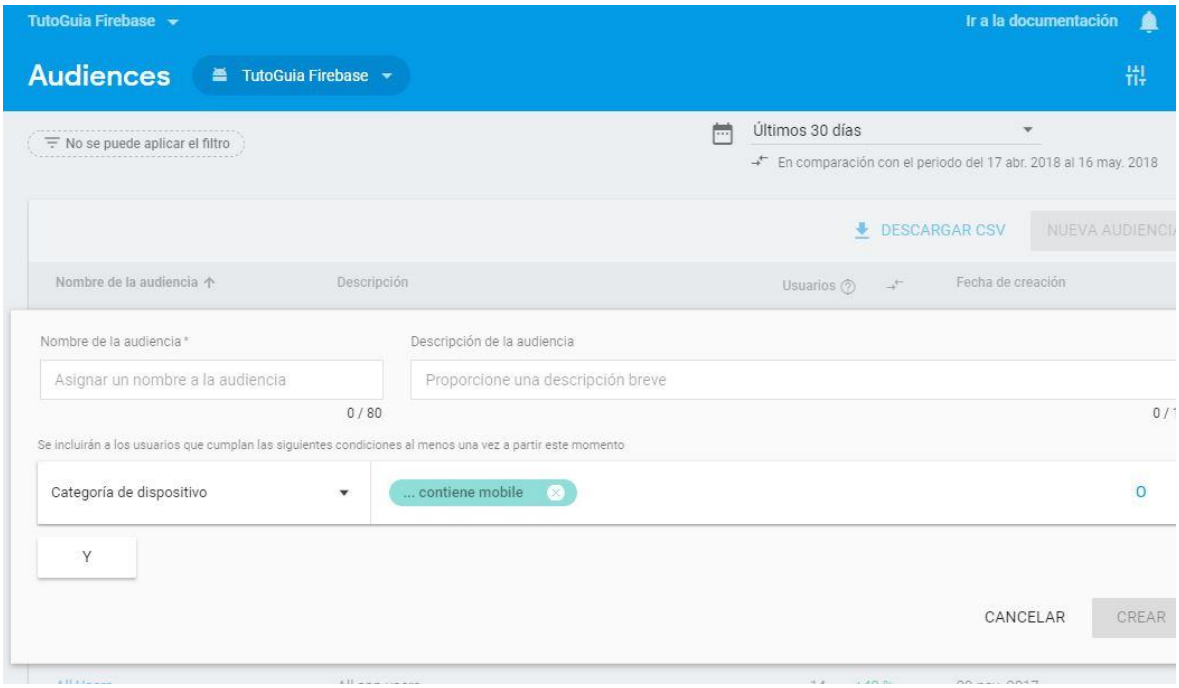

Figura 3.28 Panel de configuración de audiencias en Google Analytics. Fuente: [ConsoleFirebase].

El panel de control de Analytics hay se debe nutrir de información proveniente de la aplicación. La forma de hacerlo es por medio del SDK de Firebase encargado de este asunto. La mayoría de servicios de Firebase necesitan del uso de su propio SDK, así que se pueden importar todos de una vez o cuando se vayan a usar.

La siguiente imagen muestra cómo se referencia el SDK de Firebase, el cual incluye las funcionalidades de Analytics, en el Gradle de la aplicación para importarlo y poder

usarlo.

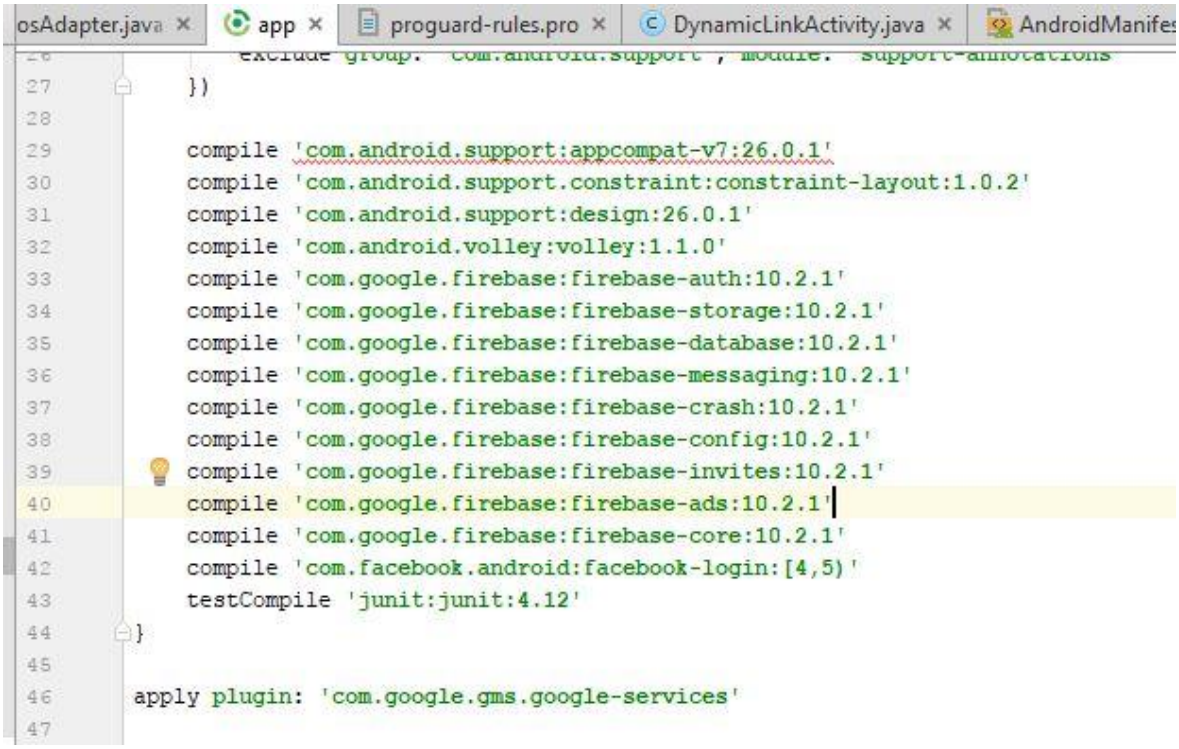

Figura 3.29 Importación de SDK de Firebase.

Para no tener demasiadas dependencias en las Activity, se ha decido aislar la implementación del servicio de Analytics en una clase a modo de servicio, llamada Analytic.java. Así, si el día de mañana cambia el contexto por usar otro servicio externo de análisis en vez de Analytics, solo haría falta cambiar la referencia de una implementación a otra y estaría listo.

A continuación, se puede ver como se inyecta la clase FirebaseAnalytics para utilizarla de interfaz de servicios.

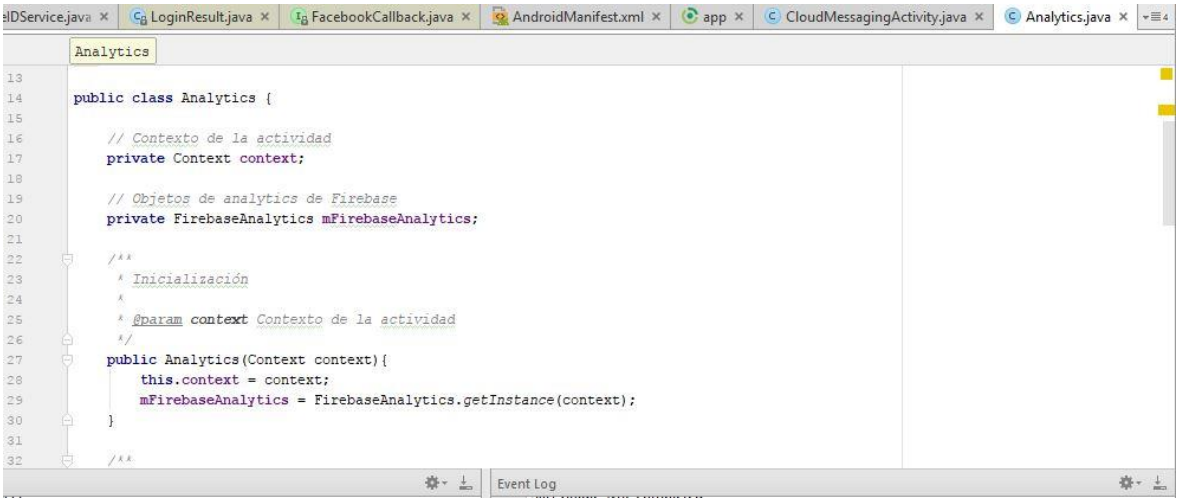

Figura 3.30 Inyección de FirebaseAnalytics.

Con la nueva instancia de un objeto Bundle se puede componer un evento personalizado o no de Firebase y prepararlo para su registro mediante el objeto de la clase FirebaseAnalytics.

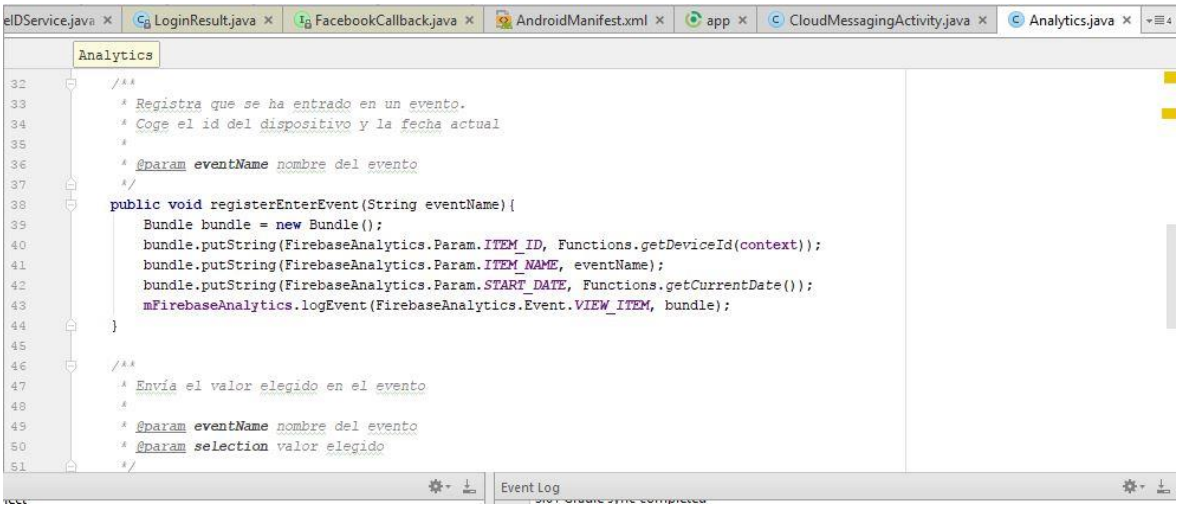

Figura 3.31 Creación de evento y registro en el log de Analytics.

Podría ser que se quisiera hacer el registro de un valor seleccionado por el usuario de una propiedad de usuario (de las que se han creado por la consola de Firebase). Para ello se utiliza el método setUserProperty(), identificando la propiedad a la que se quiere registrar un evento con el valor establecido.

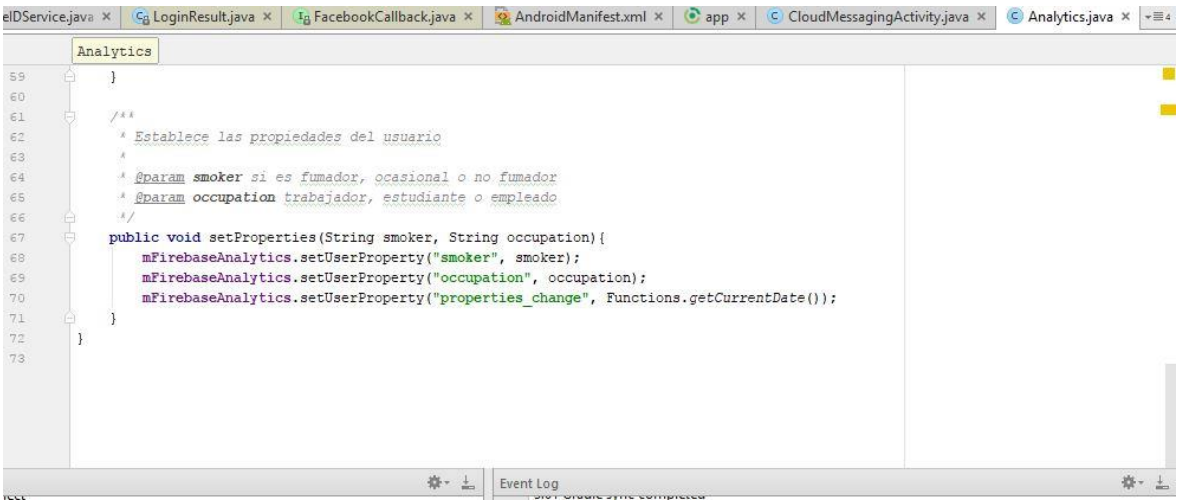

Figura 3.32 Creación de evento con las propiedades de usuario mediante la instancia de FirebaseAnalytics.

3.1.7 Integración del servicio de gestión y monetización de publicidad

Para la integración de anuncios en la aplicación y su posterior monetización, se ha utilizado el servicio de publicidad que ofrece Firebase llamado AdMob.

Como se ha descrito anteriormente, esta herramienta trabaja sobre el servicio de AdSense de Google. Por lo tanto, tendremos todas las funcionalidades que ofrece AdSense disponibles para su uso mediante el SDK de AdMob.

Lo primero será incluir el SDK en el proyecto Android para así poder hacer uso de las clases que proporciona. Después de obtener una cuenta de usuario de AdSense/AdMob, el siguiente paso sería iniciar sesión en el sitio web de AdMob.

A continuación, se puede ver el panel de control principal del sitio web de AdMob.

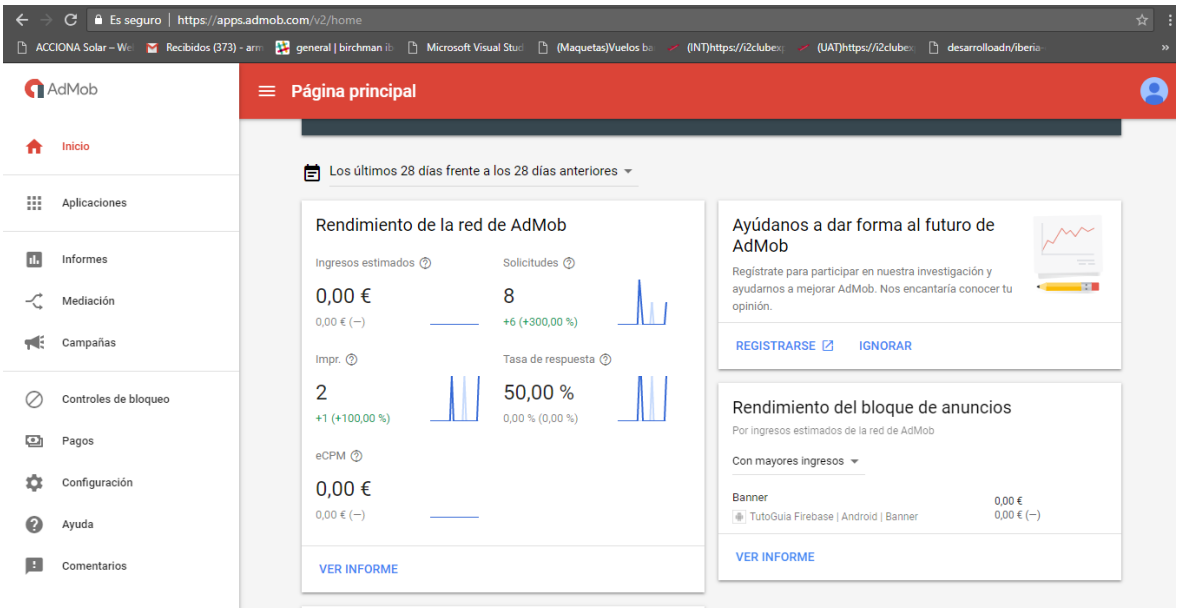

Figura 3.33 Vista del panel de control de AdMob. Fuente: [AdMob].

Desde el panel de control se puede tener acceso a todas las herramientas de AdMob. Sin salirnos del contexto del aprovechamiento de las métricas que recoge AdMob, este servicio posee herramientas para generar informes y exportarlos posteriormente, también posee una herramienta para realizar campañas de *marketing* sobre los usuarios de la propia aplicación, entre otras.

A continuación, se puede observar el panel principal de la sección de Informes de AdMob.

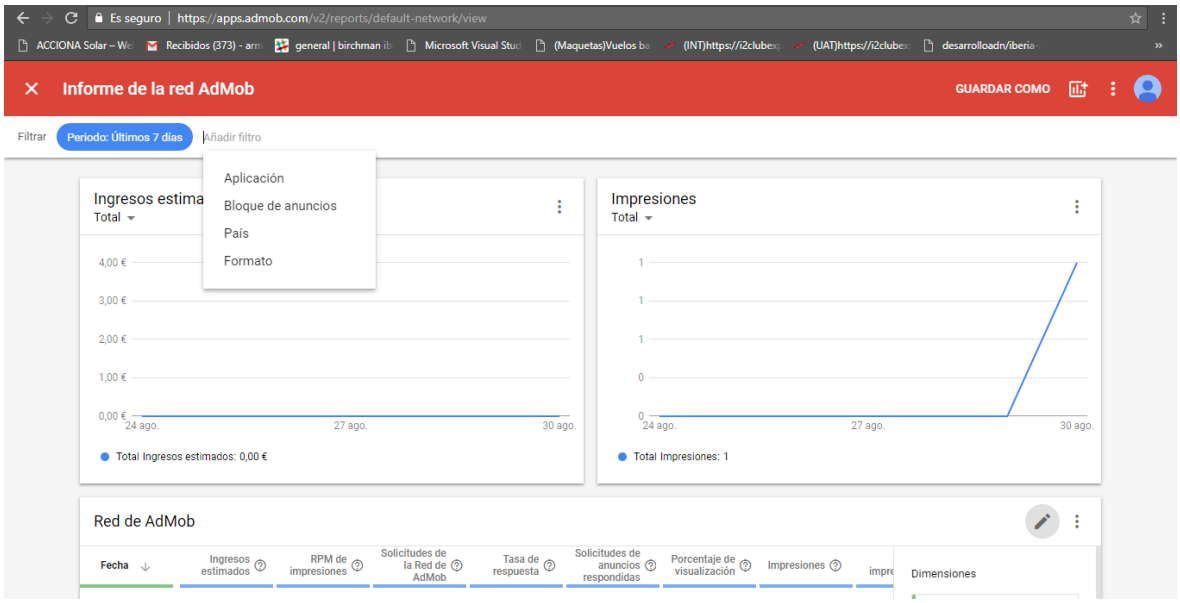

Figura 3.34 Panel principal de la herramienta de informes de AdMob. Fuente: [AdMob].

Desde esta herramienta se pueden generar informes, aplicando distintos tipos de filtros, donde podremos medir parámetros como la tasa de respuesta de los anuncios, el número de impresiones, etc.

Al ser el propio AdMob quien lleve el control de los anuncios que emite en la aplicación móvil (genero, categoría, anunciante, …) parece que el usuario de la plataforma carezca de control sobre este hecho. Pero AdMob posee una herramienta con la que permite configurar al usuario el tipo de anuncio que se va a emitir en la aplicación.

En la siguiente imagen se puede observar el panel de configuración para habilitar o deshabilitar los distintos tipos de anuncio.

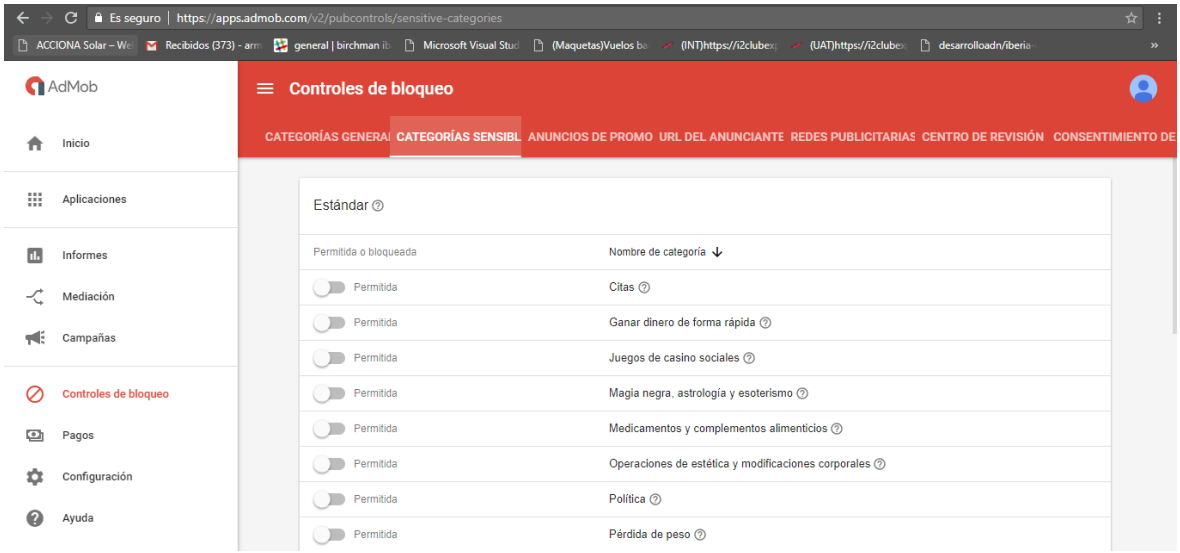

Figura 3.35 Panel de configuración para el control de anuncios. Fuente: [AdMob].

Otro de los apartados importantes dentro de la plataforma de AdMob es la herramienta de Pagos. Con esta herramienta el usuario será capaz de gestionar los pagos

fruto del beneficio de las interacciones de los usuarios de la aplicación móvil. En esta sección, AdMob permite configurar el método de pago y llevar un control de las transacciones que se han realizado.

En esta imagen podemos observar el panel de Pagos de AdMob.

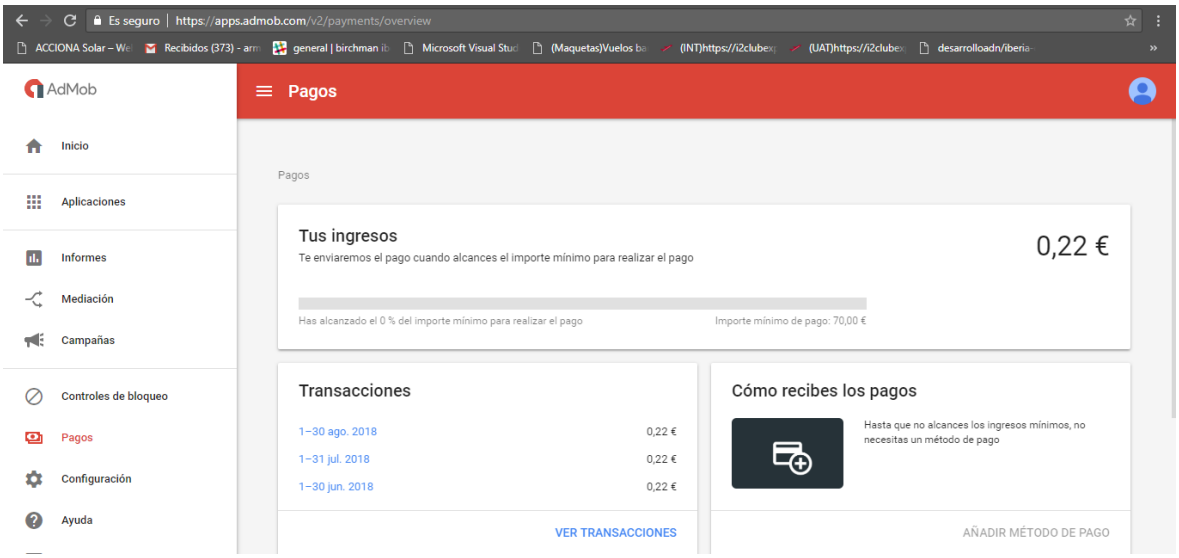

Figura 3.36 Panel de configuración para los pagos al propietario de la aplicación móvil. Fuente: [AdMob].

Desde la propia plataforma, AdMob lleva un control de los anuncios que están creados para ser usados en una aplicación móvil en concreto. Permite registrar diferentes aplicaciones móviles y gestionar los anuncios que se han creado para las mismas. Desde la misma plataforma se pueden crear y borrar anuncios.

A continuación, se puede ver el panel de control para gestionar los anuncios existentes de una aplicación móvil registrada en AdMob.

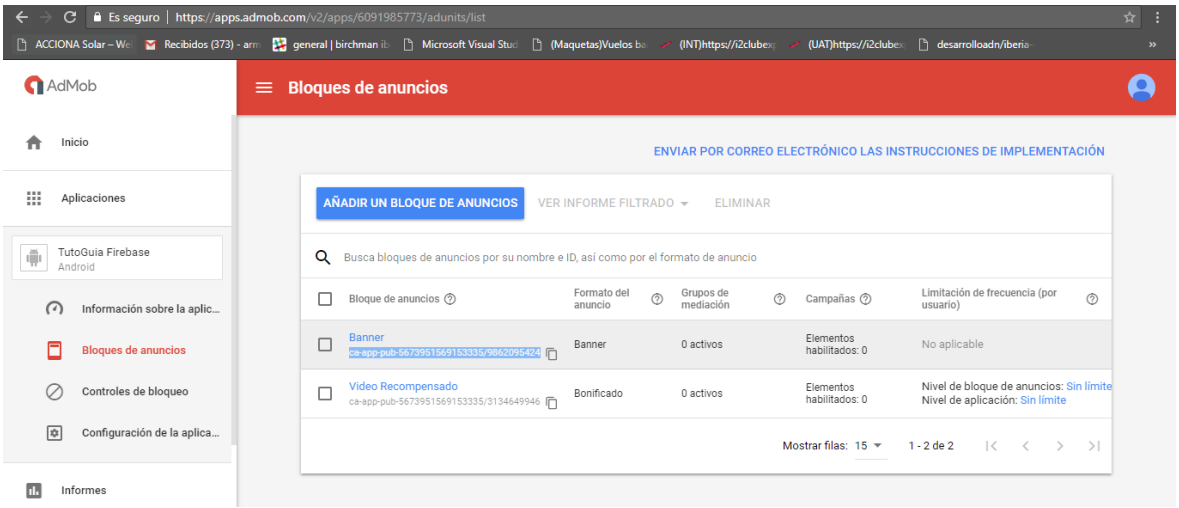

Figura 3.37 Panel de gestión de anuncios de una aplicación móvil. Fuente: [AdMob].

Podemos observar en la imagen anterior que cada uno de los anuncios tiene una cadena de texto que lo identifica de forma única. De esta forma será como AdMob lleve el control de cada uno de los anuncios.

La integración de este servicio en la aplicación Android, y como hemos visto en los anteriores servicios de Firebase, se realiza a través del SDK de AdMob.

Como este servicio tiene una alta relación con la UI, se ha realizado la implementación del servicio en la propia clase de la Activity que va a mostrar los anuncios. Posee clases para gestionar los eventos producidos por cada uno de sus tipos de anuncio.

A continuación, se puede ver la implementación de la clase AdMobActivity.java.

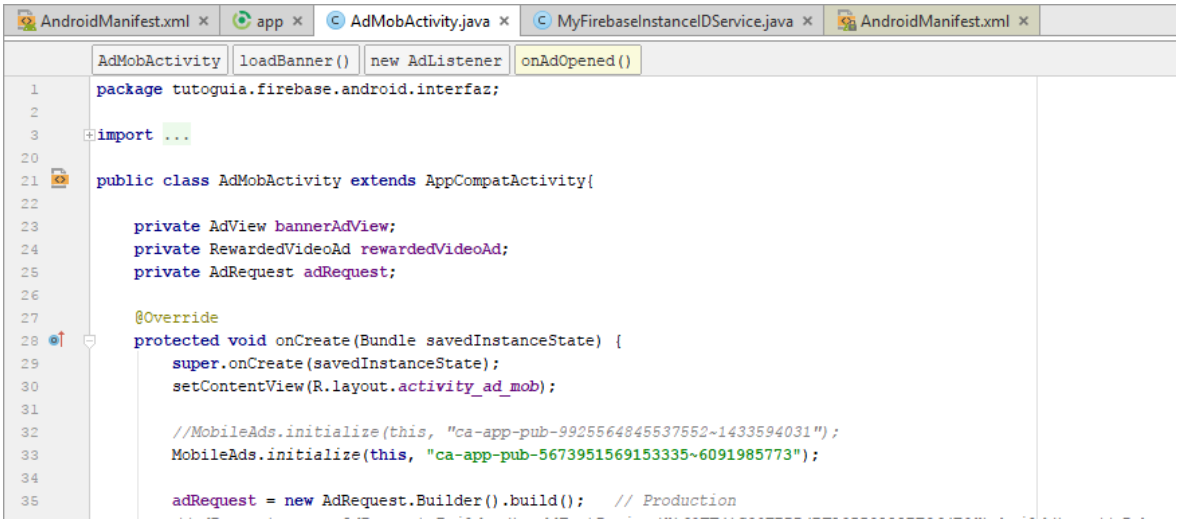

Figura 3.38 Implementación de AdMobActivity haciendo uso de las clases del SDK de AdMob.

En la imagen anterior se puede ver como se inyectan las clases responsables de la gestión de los anuncios de tipo Banner y Bonificado. Además, se puede observar cómo se inicializa el servicio de AdMob, indicando el identificador del anuncio que se va a mostrar.

En el constructor de la Activity también se encuentra la llamada a dos métodos, loadBanner() y loadRewardedVideo() que serán los encargados de cargar los 2 anuncios que han sido creados desde la plataforma AdMob.

En la siguiente imagen se puede observar la metodología seguida para cargar el anuncio de tipo Banner al inicializarse la Activity.

Figura 3.39 Método responsable de cargar el anuncio y gestionar la interacción con el mismo.

3.1.8 Otros servicios de Firebase que también se han integrado

Además de los servicios explicados con detalle anteriormente, también se han integrado algunos más en la aplicación TutoGuia Firebase. Se ha intentado que todos aquellos servicios compatibles con la plataforma Android estén integrados dentro de la aplicación móvil. Y aunque no tengan una fuerte relación con el contexto económico que se trata en este estudio, merece la pena mencionarlos y comentar algunos detalles de las funcionalidades que ofrecen.

3.1.8.1 Firebase Crash Reporting

Este servicio provee desde la consola de Firebase de un panel de control que permite monitorizar las excepciones y fallos generados dentro de la propia aplicación. Resulta una herramienta muy útil a la hora de depurar una aplicación.

3.1.8.2 Firebase Dynamic Links

Este servicio permite crear y distribuir *links.* Estos no son *links* al uso, sino que pueden ser configurados para reaccionar de maneras distintas según el tipo de dispositivo donde se abran o pueden servir para la propia aplicación móvil y responder de una manera u otra en función de que el receptor del *link* tenga la aplicación instalada o no.

3.1.8.3 Firebase Remote Config

Este servicio permite disponer de diccionarios clave-valor a modo de propiedades de configuración. Es muy útil de cara a tener distintas versiones de una misma aplicación,

o tener valores de configuración distintos si es una aplicación de prueba o una aplicación en un entorno productivo.

3.1.8.4 Firebase Cloud Functions

Este servicio permite tener almacenadas funcionalidades escritas en lenguaje JavaScript en un entorno propio de Node.js. Además, posee de distintos disparadores de autenticación, almacenamiento, etc. Que activan la ejecución de estos *scripts.*

3.2 Obligaciones administrativas obtener beneficios de una aplicación móvil en España

Los trámites administrativos que se deben cumplir para que un desarrollador o empresa pueda obtener beneficios de la aplicación móvil son los siguientes.

3.2.1 Solicitud del código de identificación fiscal (CIF)

A través de esta solicitud, la empresa quedará inequívocamente identificada como sociedad y permitirá a la misma la facturación por la venta de productos, habilitándonos en este caso para la distribución y obtención de beneficios por medio de la aplicación móvil. Está se tramita a nivel provincial en la Delegación de Hacienda. Los documentos necesarios para esta solicitud son los siguientes:

- Impreso oficial cumplimentado Modelo 036.
- Fotocopia de la Escritura de constitución de la sociedad.
- Fotocopia del D.N.I, N.I.E o C.I.F. de los socios, así como el D.N.I del administrador o apoderado (adjuntando copia del poder notarial), si fuera persona distinta.

Cumplimentados estos requisitos se le entregará a la sociedad un Código de 6 meses de validez, tras lo cual deberán retirar la Tarjeta de Identificación Fiscal definitiva. [CIF].

3.2.2 Alta en el Impuesto de Actividades Económicas (IAE)

Es un tributo de carácter local, que grava el ejercicio de actividades empresariales, profesionales o artísticas, se ejerzan o no en local. Es obligatorio para toda sociedad, empresario o profesional. Es obligatorio el alta en el impuesto y se presentarán tantas altas como actividades se vayan a ejercer. La cantidad a pagar para darse de alta en el IAE variará en función de las actividades económicas. Los documentos que se deben presentar son los siguientes:

- Si se está exento de pagos (importe neto de la cifra de negocios inferior a 1.000.000 de euros) se debe presentar el Modelo 036, en caso contrario, se debe presentar el Modelo 840.
- CIF de la sociedad.
- NIF del apoderado. [IAE].

3.2.3 *Declaración censal (IVA)*

Antes de poner en marcha una actividad económica se realizar este trámite, que sirve tanto para notificar el inicio de una actividad económica, como su modificación o cese. Se ha de presentar a efectos fiscales por los empresarios, los profesionales y otros obligados tributarios. Los documentos necesarios para realizar este trámite son los siguientes:

- Modelo 036 o 037.
- CIF de la sociedad.
- Alta en el IAE. [BOE036037].

3.2.4 *Registrar la marca*

Debemos registrar la marca y nombre comercial de la aplicación móvil para evitar el plagio. Una marca es un título que concede el derecho exclusivo a la utilización de un signo para la identificación de un producto o un servicio en el mercado. Y el nombre comercial es un título que concede el derecho exclusivo a la utilización de cualquier signo o denominación como identificador de una empresa en el tráfico mercantil. Los nombres comerciales, como títulos de propiedad industrial, son independientes de los nombres de las sociedades inscritos en los Registros Mercantiles. [OEPM].

3.3 Legislación de España/Europa que puede afectar a las aplicaciones móviles

A continuación, se enumeran las normativas a nivel legislativo que aplican a los distintos usos de una aplicación móvil.

3.3.1 *Aspectos legales y jurídicos*

Los diferentes puntos legales y jurídicos que debe cumplir una aplicación móvil son los siguientes:

- **Derechos propios y de terceros**

La aplicación que se lleve a cabo debe cumplir los requerimientos legales de los servicios que integre. La aplicación desarrollada utiliza diferentes servicios de la plataforma Firebase de Google, por tanto, debemos cumplir las condiciones de uso y la política de cada uno de los servicios implementados. Ha de cumplirse la política de datos de los usuarios de los servicios de Google, las condiciones del servicio de las API de Google y algunas licencias y condiciones concretas sobre los servicios de Google AdMob, Google Analytics y Google Cloud.

También debe realizarse el registro de marca y aplicación móvil en registro de patentes.

- **Licencia y condiciones de uso**

Se debe desarrollar la licencia y condiciones de uso de la aplicación móvil. Las licencias de software pueden establecer: la cesión de determinados derechos del propietario al usuario final sobre una o varias copias del software, los límites en la responsabilidad por fallos, el plazo de cesión de derechos, el ámbito geográfico de validez del contrato.

- **Distribución digital de aplicaciones móviles**

Para distribuir una aplicación móvil se necesita una o varias plataformas de distribución digital de aplicaciones móviles. Cada plataforma exige aceptar unos términos y condiciones para distribuir nuestra aplicación móvil.

- **Política de cookies**

La aplicación debe cumplir la ley de política de cookies de España. Una cookie (o galleta informática) es una pequeña información enviada por un sitio web y almacenada en el navegador del usuario, de manera que el sitio web puede consultar la actividad previa del usuario.

Sus principales funciones son las siguientes:

- Llevar el control de usuarios: cuando un usuario introduce su nombre de usuario y contraseña, se almacena una cookie para que no tenga que estar introduciéndolas para cada página del servidor. Sin embargo, una cookie no identifica solo a una persona, sino a una combinación de computador-navegador-usuario.
- Conseguir información sobre los hábitos de navegación del usuario, e intentos de spyware (programas espía), por parte de agencias de publicidad y otros. Esto puede causar problemas de privacidad y es una de las razones por la que las cookies tienen sus detractores.

La nota informativa que se debe incluir en una aplicación móvil para cumplir con la ley de política de cookies de España es la siguiente:

“Solicitamos su permiso para obtener datos estadísticos de su navegación en esta web, en cumplimiento del Real Decreto-ley 13/2012. Si continúa navegando consideramos que acepta el uso de cookies”.

- **Información para el usuario**

La aplicación móvil desarrollada entra dentro Ley de Servicios de la Sociedad de la Información y del Comercio Electrónico, por tanto, debe cumplir unas obligaciones.

Los datos de información general que deberán incluir son los siguientes:

1. Información general:

- Nombre o Denominación Social.
- Residencia o Domicilio (o dirección de establecimiento permanente).
- Cualquier otro dato para establecer comunicación directa y efectiva (Formulario de contacto, teléfono).

2. Datos de inscripción en el Registro Mercantil u otro, si corresponde.

3. Autorización por parte de la administración para realizar la actividad económica (En este caso aparecerán los datos de autorización y órgano competente de la supervisión).

4. Si la profesión es regulada:

- Los datos del Colegio profesional al que, en su caso, pertenezca y número de colegiado.
- El título académico oficial o profesional con el que cuente.
- El Estado de la Unión Europea o del Espacio Económico Europeo en el que se expidió dicho título y, en su caso, la correspondiente homologación o reconocimiento.
- Las normas profesionales aplicables al ejercicio de su profesión y los medios a través de los cuales se puedan conocer, incluidos los electrónicos.

5. Número de identificación fiscal que corresponda.

6. Se facilitará información clara y exacta sobre el precio del producto o servicio, indicando si incluye o no los impuestos aplicables y, en su caso, sobre los gastos de envío.

7. Adherirte a un código de conducta es seguir un código de buenas prácticas que otros promocionan, por ejemplo: Confianza On-line.

8. Y, si ofrecen acceso telefónico a servicios de tarificación adicional (utilizan programas informáticos que efectúan funciones de marcación y que el usuario descarga), debe contar con su consentimiento previo, informado y expreso: características, funciones, procedimiento …

- **Publicidad**

La monetización de la mayoría de las aplicaciones gratuitas proviene de aspectos relacionados con la publicidad. Si bien es totalmente lícito que una App incluya publicidad, ésta deberá aparecer siempre identificada como tal.

- **Legislación española**

La aplicación móvil debe cumplir la Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal. He adjuntado en la carpeta Legislación Española la ley orgánica.

- **Legislación Europea**

La aplicación móvil también debe cumplir con el Reglamente General de Protección de Datos (RGPD) emitido por UE. [Companys].

4 ESTUDIO ECONÓMICO

4.1 Coste asociado al personal

Para la realización del presente proyecto se ha necesitado de los siguientes perfiles profesionales:

- Analista.
- Programador.

Según la remuneración de cada uno de los perfiles profesionales involucrados y al número total de horas trabajadas por cada uno de ellos se han estimado los costes asociados que se ven a continuación.

Actividad Desarrollada	€/hora	Horas	Coste (€)
Estudio y redacción del trabajo teórico experimental	40,00	150	6.000,00
Diseño de software	30,00	150	4.500,00
	Total	**300**	**10.500,00**

Tabla 4.1 Costes asociados al personal

4.2 Coste asociado al trabajo fin de grado

Los costes asociados a la realización del trabajo fin de grado se pueden ver a continuación.

Concepto	Presupuesto (€)
Coste asociado al personal	10.500,00
Total (sin I.V.A)	10.500,00
Total I.V.A (21%)	2.205,00
Total (I.V.A incluido)	12.705,00

Tabla 4.2 Costes asociados al trabajo fin de grado

Asciende el presupuesto del presente Trabajo de Fin de Grado “Estudio de las tecnologías empleadas en el negocio de las aplicaciones móviles” a la cantidad de **doce mil setecientos cinco euros (12.705,00 €)**.

5 CONCLUSIONES Y LÍNEAS FUTURAS

A día de hoy, las soluciones en el sector del desarrollo de aplicaciones móviles son muy variadas. La mayoría de las empresas ligadas a este sector poseen múltiples servicios que ofrecen a sus usuarios para que estos puedan integrarlos en las aplicaciones móviles que vayan a desarrollar.

Empresas como Amazon, Microsoft o Google poseen plataformas de servicios con multitud de herramientas enfocadas a complementar y a añadir valor a las aplicaciones que se desarrollan hoy en día. Otras empresas (App Annie por ejemplo) simplemente realizan una actividad comercial en la que son muy competentes y desarrollan una solución software relacionada con dicha actividad.

Sea como fuere, ocurre que se produce una problemática parecida a la que sucede con Internet y el exceso de información. Existe tanta, que lo complicado es diferenciar cual es la información correcta de la que no lo es.

Seguramente existen servicios externos capaces de integrarse con una aplicación móvil de buena calidad por parte de muchas empresas. Aunque a la hora de llevar a la práctica una integración de varios de estos servicios resulta muy cómodo tener todos ellos centralizados en una sola plataforma. Hoy en día, se tiende a delegar responsabilidades de mantenimiento, seguridad, etc., a plataformas de servicios, sobre todo desde que aparecieron los servicios virtualizados en la nube.

Por la experiencia cosechada en la realización del presente estudio y en el desarrollo en paralelo de la aplicación Android TutoGuia Firebase, la opción que ofrece Google en su plataforma Firebase resulta muy válida ya que permite esa centralización de todos los servicios que antes comentábamos.

Además, siendo Google el propietario de la plataforma Android, es de esperar que ponga especial interés en que los desarrolladores de aplicaciones utilicen su plataforma. De hecho, durante la realización del presente estudio, se ha podido observar como muchos de los servicios que se han integrado ya cuentan con versiones beta que incorporaran distintas mejoras, además de la incorporación de otros tantos nuevos servicios.

También cabe destacar que los servicios de publicidad estén integrados con Google Analytics, uno de los mejores productos en el contexto de los servicios de análisis de métricas, es un punto muy fuerte a su favor, ya que dota a estos servicios de muchas posibilidades.

Con este trabajo se han cumplido los objetivos propuestos al principio de su redacción, además de realizar un análisis de varios servicios más de terceros y de contextos diferentes.

5.1 Líneas futuras

Siendo un proyecto enfocado al desarrollo en aplicaciones móviles tiene muchas posibles ramas en las que encaminarse.

El desarrollo de aplicaciones hoy en día está en auge, y es un negocio que mueve millones de euros. Viendo el beneficio económico que repercute este sector, no es de extrañar que dentro de poco las aplicaciones sigan con este ritmo de crecimiento.

Además, es un estudio que se enfoca en dispositivos móviles, el cual es un mercado, que como hemos visto, no para de crecer y que actualmente ya consigue mayor número de ventas que los ordenadores personales.

Una posible línea de futuro sería realizar un estudio sobre el beneficio económico generado a través de comercializar productos o servicios por medio de aplicaciones móviles, analizar pasarelas de pago, etc.

También, dado que este estudio se ha centrado en el sistema operativo Android de Google, otra posibilidad sería el de personalizarlo para las aplicaciones IOS de Apple.

6 ANEXOS

6.1 ANEXO I: Manual de usuario de aplicación

La aplicación móvil TutoGuia Firebase es una aplicación desarrollada en Android con la que se pretende realizar una guía por los distintos servicios de Firebase, tanto a nivel funcional como a nivel técnico.

El presente manual pretende servir de tutorial de uso de la aplicación, y además resaltar la integración de cada servicio de Firebase con su funcionalidad correspondiente dentro de la aplicación móvil.

Tras la instalación y lanzado de la aplicación en el dispositivo móvil, nos encontraremos con el menú principal de la aplicación, que se divide modularmente en todos los servicios que se han integrado de la plataforma Firebase. También se ha añadido un último apartado titulado “Acerca de” donde se muestra la información básica de la aplicación.

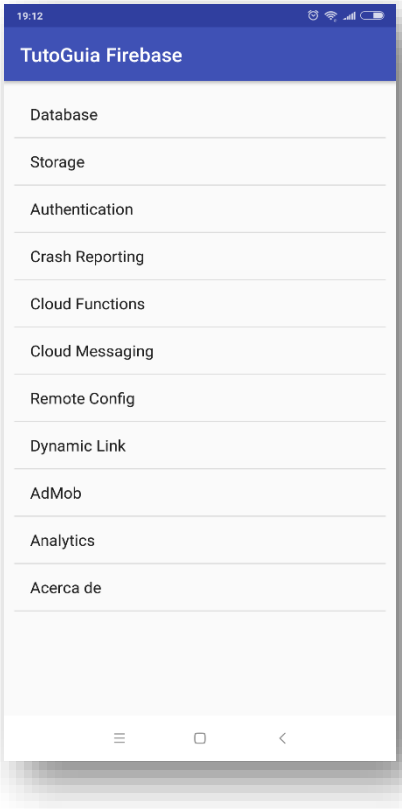

Figura 6.1 Menú principal aplicación TutoGuia Firebase

6.1.1 Módulo Database

En este módulo se ha implementado un sistema de consultas a base de datos mediante el servicio Realtime Database de Firebase.

Existen 3 tablas creadas en la herramienta Realtime Database (aunque en la aplicación solo aparezcan 2). Se trata de 2 tablas, una de alumnos y otra de profesores. La tercera tabla en cuestión se encarga de relacionar las asignaturas que tienen cada uno de los alumnos.

En la primera y segunda pestaña del módulo de Database se pueden observar las 2 tablas que se han comentado anteriormente.

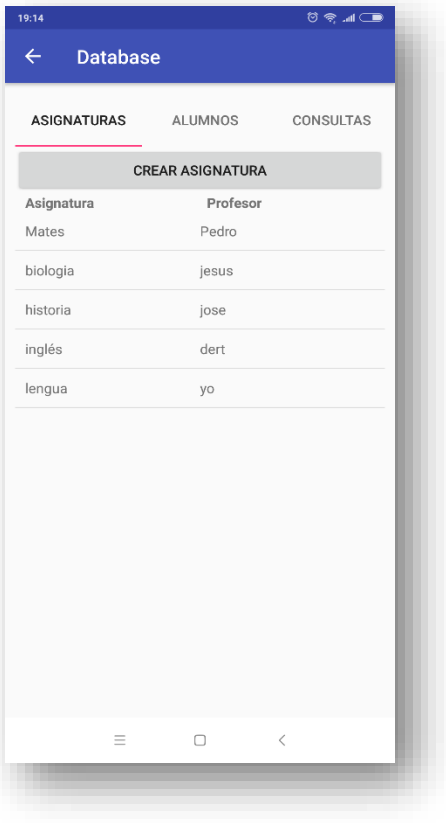

Figura 6.2 Módulo Database. Visualización de tabla Asignaturas.

Por cada una de estas tablas, la aplicación ofrece la posibilidad de crear de nuevos elementos, así como también de realizar acciones de modificación y borrado sobre los que ya existen.

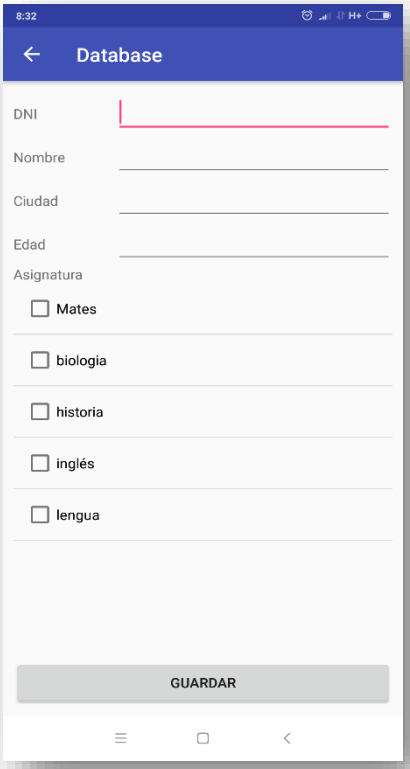

Figura 6.3 Cuadro de dialogo para la creación de un nuevo registro en la tabla Alumnos.

La tercera pestaña del módulo de Database se ha dedicado a un panel de consultas sobre las 3 tablas. Existen varias consultas con resultado numérico y otras cuyo resultado son elementos de una u otra tabla, para lo cual se ha implementado una sección en la parte de abajo de la pestaña para poder visualizar el resultado de dichas consultas.

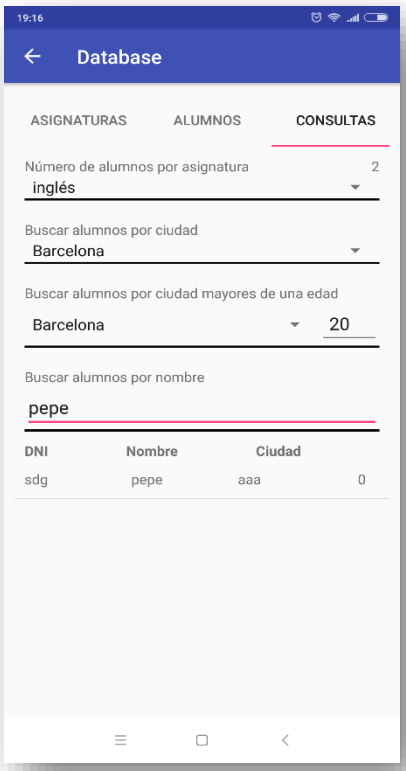

Figura 6.4 Módulo Database. Visualización de sección de consultas.

6.1.2 Módulo Storage

En este módulo se ha implementado un sistema de almacenamiento haciendo uso del servicio Storage de Firebase.

Posee funciones muy similares a las vistas anteriormente en el módulo de Database. Se ha implementado las funcionalidades de subir archivos, guardarlos en el dispositivo móvil y borrarlos del almacenamiento remoto que provee el servicio Storage.

Además, permite almacenar archivos de múltiples formatos: pdf, texto plano, png, jpg, entre otros.

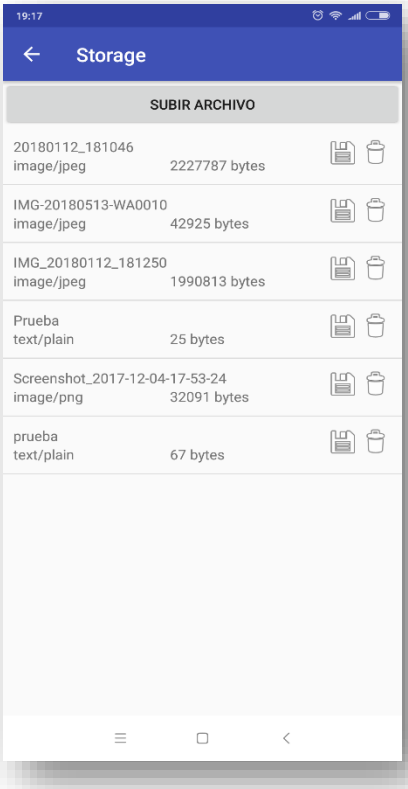

Figura 6.5 Pantalla principal del módulo Storage

6.1.3 Módulo Authentication

Este módulo implementa varias funcionalidades ofrecidas por el servicio Authentication de Firebase.

Se ha pretendido probar una muestra significativa de las distintas formas de inicio de sesión que ofrece esta herramienta. En concreto se han habilitado 3 métodos de inicio de sesión desde el panel de control de Firebase.

Se ha implementado un inicio de sesión por verificación de credenciales simplemente, con correo electrónico y contraseña.

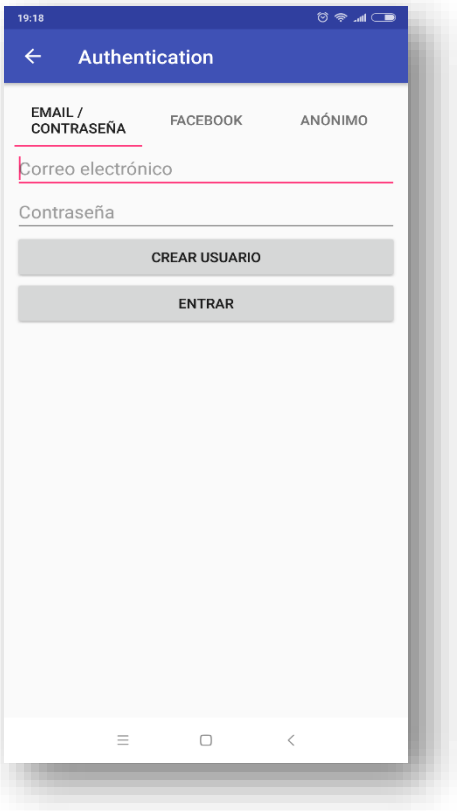

Figura 6.6 Pestaña de inicio de sesión con correo y contraseña del módulo Authentication.

También, se ha implementado un inicio de sesión siguiendo el estándar OAuth 2, utilizando Facebook para delegar la responsabilidad de verificar la identidad del usuario de la aplicación.

Y, además, se ha integrado también un método de autenticación de forma anónima. Con el que se genera un número identificación única y aleatoria que bien podría usarse para gestionar la sesión del usuario.

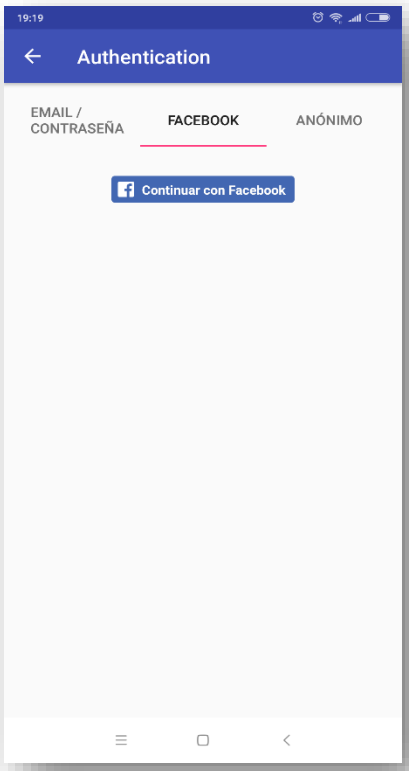

Figura 6.7 Pestaña de inicio de sesión con Facebook.

6.1.4 Módulo Crash Reporting

En este módulo se ha implementado una calculadora con las funcionalidades básicas, y con varios errores que no han sido controlados por código, sino que se ha permitido que se generen para probar la monitorización que posee este servicio.

Uno de esos errores que genera una excepción no controlada es la división de cualquier número por 0, lo cual hará que la aplicación lance un mensaje de error y cierre la aplicación. Dejando un registro en el panel de monitorización de Crash Reporting.

Figura 6.8 Mensaje de error de excepción no controlada en el módulo Crash Reporting.

6.1.5 Módulo Cloud Functions

Este módulo cuenta con 2 pestañas con las que se ha probado 2 de los distintos disparadores que posee esta herramienta.

En la primera pestaña se ha implementado un disparador de tipo autenticación, que permite ejecutar una función alojada en el entorno Node.js mediante la cual enviará un correo de bienvenida al usuario en concreto.

En la segunda pestaña se ha implementado un disparador de tipo almacenamiento, que hará ejecutarse una función que leerá un archivo de texto y lo almacenará línea a línea en la base de datos de Firebase.

Figura 6.9 Pestaña del disparador de autenticación en el módulo Cloud Functions.

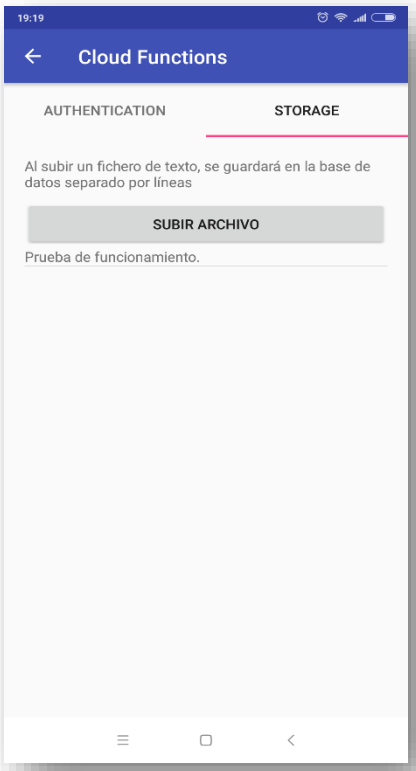

Figura 6.10 Pestaña del disparador de almacenamiento en el módulo Cloud Functions.

6.1.6 Módulo Cloud Messaging

Para la integración de este servicio en la aplicación móvil se ha utilizado la funcionalidad ofrecida por la API de Cloud Messaging.

Desde el módulo Cloud Messaging podemos componer una notificación con un título y un mensaje, y seleccionar al usuario de la aplicación que será el destinatario de dicha notificación. Para esta funcionalidad, es necesario utilizar el token que identifica cada instancia de la aplicación.

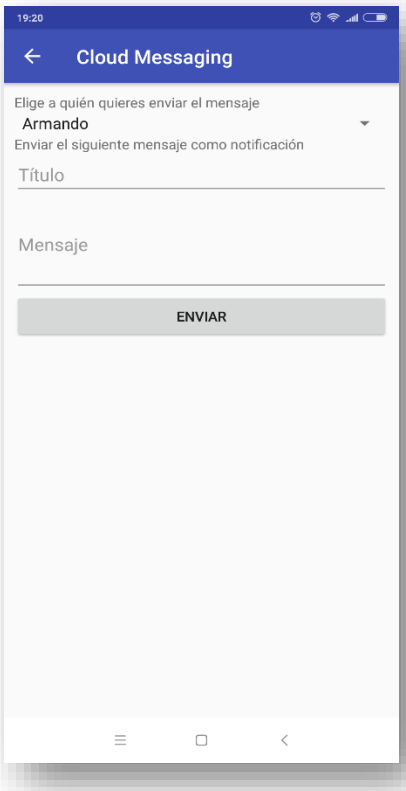

Figura 6.11 Pantalla principal del módulo Cloud Messaging.

6.1.7 Módulo Remote Config

Este servicio se ha integrado en la aplicación móvil de forma que se simulen varios entornos en los que pueda encontrarse una aplicación. Como pueden ser los entornos de desarrollo, *User Acceptance Testing* (UAT) y producción.

Por lo tanto, se puede visualizar en este módulo, los distintos valores de configuración que se han establecido en la plataforma Firebase para simular estos entornos.

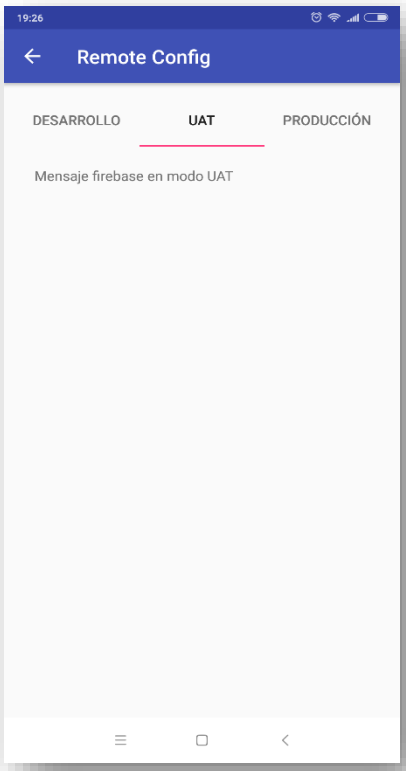

Figura 6.12 Pestaña de entorno de UAT en el módulo Remote Config.

6.1.8 Módulo Dynamic Link

La implementación de este servicio ha consistido en la creación de un *link* con la información del perfil de un usuario de Facebook.

La forma en que funciona es mediante un inicio de sesión de Facebook para recoger los datos necesarios del perfil en cuestión.

Después de haber iniciado sesión exitosamente, se tiene la posibilidad de enviar ese *link* por distintos métodos de envió: mensaje de texto, correo electrónico, etc.

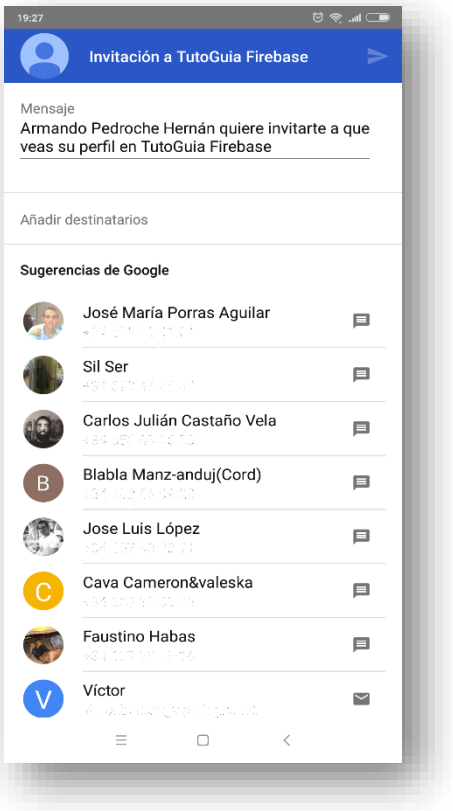

Figura 6.13 Cuadro de dialogo para envió de link en el módulo Dynamic Link

6.1.9 Módulo AdMob

Para la integración de este servicio en la aplicación móvil se ha querido probar algunos de los tipos de anuncio de los que dispone AdMob.

Se ha implementado una Activity en la que se inicializan 2 anuncios por medio de los identificadores generados en la plataforma de AdMob (un banner y un video con recompensa).

El usuario de la aplicación puede visualizar los 2 anuncios e interactuar con ellos mediante la acción de pulsarlos.

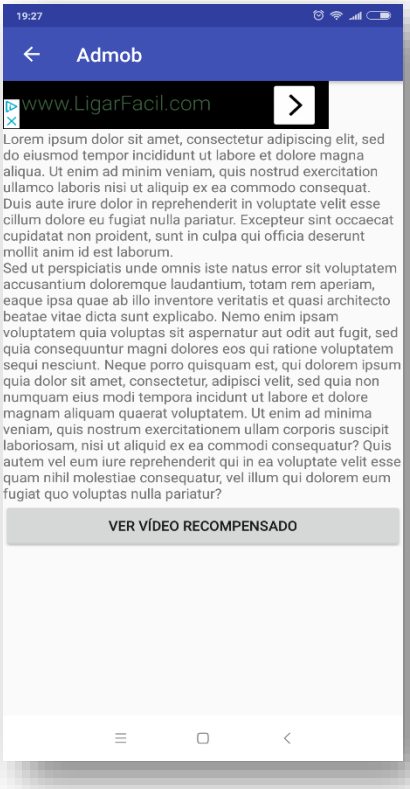

Figura 6.14 Pantalla principal del módulo AdMob.

6.1.10 Módulo Analytics

En este módulo se han integrado distintas funcionalidades que ofrece el servicio de Analytics.

En una primera pestaña, aparece un cuestionario con 3 posibles opciones. Cada una de las opciones genera un evento diferente que será enviado a Analytics, identificando al usuario que los ha emitido en dicha comunicación.

En la segunda pestaña, aparece otro cuestionario, pero en este caso se trata de propiedades del usuario. Se trata de una opción que ofrece Analytics con el fin de caracterizar los usuarios de una aplicación.

Figura 6.15 Respuesta de éxito tras él envió del evento de la primera pestaña del módulo Analytics.

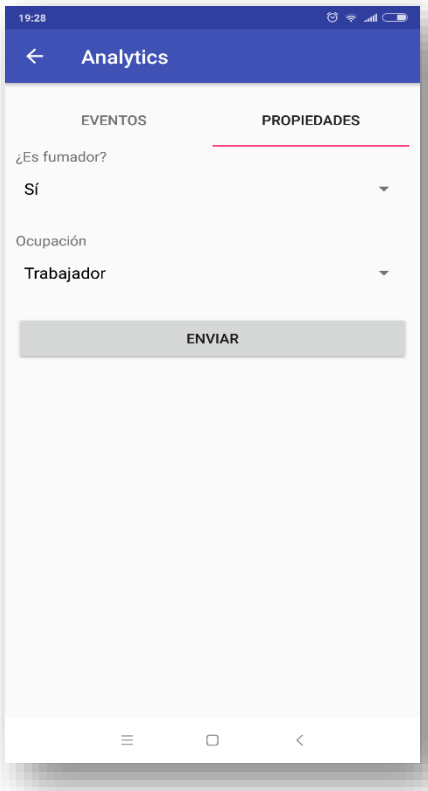

Figura 6.16 Segunda pestaña del módulo de Analytics con las posibles propiedades de usuario a elegir.

7 REFERENCIAS BIBLIOGRÁFICAS

[StatistaWeb] Statista: el portal de estadísticas para datos de mercado, investigaciones de mercado y estudios de mercado. [en línea], [sin fecha]. [Consulta: 28 abril 2018]. Disponible en: https://es.statista.com/.

[AdobeRefer] Página de inicio de la ayuda de Adobe Analytics - Ayuda y referencia de Analytics. [en línea], [sin fecha]. [Consulta: 2 agosto 2018]. Disponible en: https://marketing.adobe.com/resources/help/es_ES/reference/.

[IAE] Impuesto de Actividades Económicas (I.A.E) | Ser autónomo. [en línea], [sin fecha]. [Consulta: 3 agosto 2018]. Disponible en: https://www.serautonomo.net/impuesto-de-actividades-economicas-iae.html.

[Companys] Empresas - LSSI. [en línea], [sin fecha]. [Consulta: 20 julio 2018]. Disponible en: http://www.lssi.gob.es/empresas/Paginas/empresas.aspx.

[OEPM] Oficina Española de Patentes y Marcas - Propiedad industrial. [en línea], [sin fecha]. [Consulta: 13 mayo 2018]. Disponible en: https://www.oepm.es/es/propiedad_industrial/propiedad_industrial/que_se_puede_proteger_y_como/marcas_y_nombres_comerciales/index.html.

[CIF] Solicitud del Código de Identificación Fiscal (CIF). [en línea], [sin fecha]. [Consulta: 4 junio 2018]. Disponible en: http://www.creacionempresas.com/tramites-para-crear-la-empresa/tramites-de-constitucion-de-las-sociedades/solicitud-del-codigo-de-identificacion-fiscal-cif.

[OAuth2Intro] Una introducción a OAuth 2. *DigitalOcean* [en línea], [sin fecha]. [Consulta: 22 mayo 2018]. Disponible en: https://www.digitalocean.com/community/tutorials/una-introduccion-a-oauth-2-es.

[AndroidVers] Todas las versiones de Android » ¡No te lo pierdas! [en línea], [sin fecha]. [Consulta: 5 agosto 2018]. Disponible en: https://actualizar-android.com/versiones/.

[PushNotDoc] Push Notifications: Conociendo la tecnología. [en línea], [sin fecha]. [Consulta: 18 julio 2018]. Disponible en: http://www.innovaage.com/innovaportal/v/627/1/innova.front/push-notifications:-conociendo-la-tecnologia.

[AuthProduct] Setting Up Authentication for Server to Server Production Applications | Authentication. *Google Cloud* [en línea], [sin fecha]. [Consulta: 24 mayo 2018]. Disponible en: https://cloud.google.com/docs/authentication/production.

[AnalyticTools] 10 herramientas de analítica para apps - La Métrica. [en línea], [sin fecha]. [Consulta: 12 mayo 2018]. Disponible en: http://lametrica.com/10-herramientas-de-analitica-para-apps/.

[AppStoreTips] 14 tips para publicar una App en las App Stores. *AppYourself* [en línea], 2016. [Consulta: 19 julio 2018]. Disponible en: https://appyourself.net/es/blog/cosas-prohibidas-al-crear-una-app/.

[AccionaWeb] ACCIONA: Infraestructuras Sostenibles y Energías Renovables. [en línea], [sin fecha]. [Consulta: 20 agosto 2018]. Disponible en: https://www.acciona.com/es/.

[AndAccMang] AccountManager. *Android Developers* [en línea], [sin fecha]. [Consulta: 23 abril 2018]. Disponible en: https://developer.android.com/reference/android/accounts/AccountManager.

[AWSNoSQL] Adición de NoSQL Database a una aplicación móvil con Amazon DynamoDB - AWS Mobile. [en línea], [sin fecha]. [Consulta: 12 mayo 2018]. Disponible en: https://docs.aws.amazon.com/es_es/aws-mobile/latest/developerguide/add-aws-mobile-nosql-database.html.

[AdMobSite] AdMob. [en línea], [sin fecha]. [Consulta: 4 agosto 2018]. Disponible en: https://apps.admob.com/v2/home?pli=1.

[AdobeAnalytic] Adobe Analytics – now part of Adobe Analytics Cloud. [en línea], [sin fecha]. [Consulta: 15 agosto 2018]. Disponible en: https://www.adobe.com/es/data-analytics-cloud/analytics.html.

[Gartner4Qua] Gartner Says Worldwide Sales of Smartphones Recorded First Ever Decline During the Fourth Quarter of 2017. [en línea], [sin fecha]. [Consulta: 28 junio 2018]. Disponible en: https://www.gartner.com/en/newsroom/press-releases/2018-02-22-gartner-says-worldwide-sales-of-smartphones-recorded-first-ever-decline-during-the-fourth-quarter-of-2017.

[480Analytics] Analíticas y métricas para mejorar una app - 480 Cuatroochenta. *480* [en línea], [sin fecha]. [Consulta: 4 agosto 2018]. Disponible en: https://cuatroochenta.com/analiticas-y-metricas-para-mejorar-una-app/.

[AndroidHist] Android - Historia. [en línea], [sin fecha]. [Consulta: 20 julio 2018]. Disponible en: https://www.android.com/intl/es_es/history/#/marshmallow.

[AndroidMarsh] Android 6.0 API. *Android Developers* [en línea], [sin fecha]. [Consulta: 7 julio 2018]. Disponible en: https://developer.android.com/about/versions/marshmallow/android-6.0?hl=es-419.

[DevelopAndr] Android Developers. [en línea], [sin fecha]. [Consulta: 26 mayo 2018]. Disponible en: https://developer.android.com/.

[TutoGuiaApp] APEDROCHE, 2018. *Contribute to apedroche/TutoGuiaFirebase development by creating an account on GitHub* [en línea]. Java. S.l.: s.n. [Consulta: 29 julio 2018]. Disponible en: https://github.com/apedroche/TutoGuiaFirebase.

[HybridApps] Aplicaciones móviles híbridas: la solución más eficiente para el desarrollo multiplataforma | Consultoría y Servicios IT para empresas. *Profile Software Services* [en línea], 2017. [Consulta: 28 agosto 2018]. Disponible en: https://profile.es/blog/aplicaciones-moviles-hibridas-la-solucion-mas-eficiente-para-el-desarrollo-multiplataforma/.

[NativOrHybr] ¿App híbrida o App Nativa? *480* [en línea], [sin fecha]. [Consulta: 12 agosto 2018]. Disponible en: https://cuatroochenta.com/app-hibrida-o-app-nativa-segun-para-que/.

[AndroidArqu] Arquitectura de la plataforma | Android Developers. [en línea], [sin fecha]. [Consulta: 24 agosto 2018]. Disponible en: https://developer.android.com/guide/platform/?hl=es-419.

[AuthEndUser] Authenticating as an End User | Authentication. *Google Cloud* [en línea], [sin fecha]. [Consulta: 14 agosto 2018]. Disponible en: https://cloud.google.com/docs/authentication/end-user.

[AzurePushN] CONCEPTDEV, [sin fecha]. Adición de notificaciones de inserción a una aplicación Android con Mobile Apps. [en línea]. [Consulta: 12 junio 2018]. Disponible en: https://docs.microsoft.com/es-es/azure/app-service-mobile/app-service-mobile-android-get-started-push.

[CostePClic] Coste por clic: el modelo más utilizado de publicidad online. [en línea], [sin fecha]. [Consulta: 29 agosto 2018]. Disponible en: https://www.inboundcycle.com/diccionario-marketing-online/coste-por-clic.

[AdSenseDoc] Diferencia entre Google Ads y AdSense - Ayuda de AdSense. [en línea], [sin fecha]. [Consulta: 20 agosto 2018]. Disponible en: https://support.google.com/adsense/answer/76231?hl=es&ref_topic=1319753.

[FreeVsOpen] Diferencias entre Software Libre y Open Source. [en línea], [sin fecha]. [Consulta: 15 agosto 2018]. Disponible en: https://hipertextual.com/archivo/2014/05/diferencias-software-libre-y-open-source/.

[OAuth2.0] OAuth 2.0 — OAuth. [en línea], [sin fecha]. [Consulta: 30 agosto 2018]. Disponible en: https://oauth.net/2/.

Download Android Studio and SDK tools. *Android Developers* [en línea], [sin fecha]. [Consulta: 21 agosto 2018]. Disponible en: https://developer.android.com/studio/.

[BOE036037] El BOE publica los cambios en los modelos 036, 037, 303 y 322 | Fiscal Impuestos. *fiscal-impuestos.com* [en línea], [sin fecha]. [Consulta: 15 agosto 2018]. Disponible en: https://www.fiscal-impuestos.com/modificaciones-modelos-036-037-303-322-declaraciones-censales-iva.html.

[FacebookApp] Facebook - Aplicaciones en Google Play. [en línea], [sin fecha]. [Consulta: 14 abril 2018]. Disponible en: https://play.google.com/store/apps/details?id=com.facebook.katana&hl=es. Facebook para desarrolladores | Hacemos del mundo un lugar más conectado. *Facebook para desarrolladores* [en línea], [sin fecha]. [Consulta: 23 mayo 2018]. Disponible en: https://developers.facebook.com/.

[BlackBerryD] FERNÁNDEZ, S., 2017. La muerte de BlackBerry OS se consumará a finales de 2019 con el cierre de su tienda. *Xataka Móvil* [en línea]. [Consulta: 20 mayo 2018]. Disponible en: https://www.xatakamovil.com/blackberry/la-muerte-de-blackberry-os-se-consumara-a-finales-de-2019-con-el-cierre-de-su-tienda.

[FirAuthDoc] Firebase Authentication. *Firebase* [en línea], [sin fecha]. [Consulta: 15 junio 2018]. Disponible en: https://firebase.google.com/docs/auth/?hl=es-419.

[FirClMessag] Firebase Cloud Messaging. *Firebase* [en línea], [sin fecha]. [Consulta: 14 mayo 2018]. Disponible en: https://firebase.google.com/docs/cloud-messaging/?hl=es-419.

[ConsoleFireb] Firebase console. [en línea], [sin fecha]. [Consulta: 20 abril 2018]. Disponible en: https://console.firebase.google.com/.

[FirDataBase] Firebase Realtime Database | Firebase Realtime Database. *Firebase* [en línea], [sin fecha]. [Consulta: 10 abril 2018]. Disponible en: https://firebase.google.com/docs/database/?hl=es-419.

[PFCAranaz] JAIME ARANAZ TUDELA, 2014. *Desarrollo de Aplicaciones para la Plataforma Android. Un caso de estudio para el intercambio de libros*. S.l.: s.n.

[PFCAlarcon] FRANCISCO ALARCÓN BARCELÓ, 2009. *Desarrollo de aplicaciones para dispositivos móviles sobre la plataforma Android de Google*. S.l.: s.n.

[PFCIskandar] RICARDO JOSE ISKANDAR MORINE, 2013. *Estudio comparativo de alternativas y frameworks de programación, para el desarrollo de aplicaciones móviles en entorno Android*. S.l.: s.n.

[AndroidBook] JESÚS TOMÁS GIRONÉS, 2014. *El gran libro de Android*. 4ªEdición. S.l.: s.n.

[AnnieAnalytics] App Annie - The App Analytics and App Data Industry Standard. [en línea], [sin fecha]. [Consulta: 2 mayo 2018]. Disponible en: https://www.appannie.com/en/.

[RAEDiccion] Real Academia Española. [en línea], [sin fecha]. [Consulta: 3 mayo 2018]. Disponible en: http://www.rae.es/.

[WinMobileD] Windows 10 Mobile está muerto y ni en el Programa Insider se acuerdan de la plataforma móvil de Microsoft. [en línea], [sin fecha]. [Consulta: 30 agosto 2018]. Disponible en: https://www.xatakawindows.com/windows-phone/windows-10-mobile-esta-muerto-y-ni-en-el-programa-insider-se-acuerdan-de-la-plataforma-movil-de-microsoft.

[BasicMarkting] Marketing de aplicaciones móviles - Marketing Analítico. [en línea], [sin fecha]. [Consulta: 3 mayo 2018]. Disponible en: https://www.marketing-analitico.com/pln/marketing-de-aplicaciones-moviles/#Puntos_basicos_en_un_plan_de_marketing_de_aplicaciones_moviles.

[MarktingApps] ¿Qué es el marketing para apps? Estrategias posibles. [en línea], [sin fecha]. [Consulta: 5 mayo 2018]. Disponible en: https://somechat.es/marketing-para-apps/.

[AnalytMarkting] Marketing de aplicaciones móviles - Marketing Analítico. [en línea], [sin fecha]. [Consulta: 10 julio 2018]. Disponible en: https://www.marketing-analitico.com/pln/marketing-de-aplicaciones-moviles/.

[DevelopPlatf] Plataformas de desarrollo móvil. [en línea], [sin fecha]. [Consulta: 11 agosto 2018]. Disponible en: https://code.tutsplus.com/es/articles/mobile-development-platforms--cms-28944.

[ISO16262] ISO/IEC 16262:2011 - Information technology -- Programming languages, their environments and system software interfaces -- ECMAScript language specification. [en línea], [sin fecha]. [Consulta: 5 agosto 2018]. Disponible en: https://www.iso.org/standard/55755.html.

[DevelopmTec] Tecnologías de desarrollo para móviles | Programming and So. [en línea], [sin fecha]. [Consulta: 10 agosto 2018]. Disponible en: https://angelborroy.wordpress.com/2012/10/31/tecnologias-de-desarrollo-para-moviles/.