Escuela Politécnica Superior de Linares

Grado en Ingeniería Telemática

UNIVERSIDAD DE JAÉN

Escuela Politécnica Superior de Linares

Trabajo Fin de Grado

SISTEMA WEB-MÓVIL PARA LA GESTIÓN Y EL CONTROL DE COMUNICACIÓN ENTRE USUARIOS

Alumno: **José Luis Cabeza Díaz**

Tutor: Prof. D. Ángel Inocencio Aguilera García
Depto.: Informática

Octubre, 2017

Resumen

Los smartphones están cada día más presentes en nuestras vidas, tanto para comunicarnos como para hacer muchas otras funciones propias de los PCs, que poco a poco se van delegando en nuestros dispositivos móviles. Dentro del mundo de los smartphones el sistema operativo más utilizado es Android de Google, seguido por IOS de Apple. Como tercer sistema operativo utilizado encontramos Windows Phone, el cual parece condenado a la extinción después de que se confirmara que no se van a seguir desarrollando nuevas funcionalidades para este, y el reciente abandono de HP.

El proyecto trata sobre una aplicación para dispositivos Android, mediante la cual se posibilita la comunicación entre usuarios mediante texto y voz. Para la realización de dicho servicio he utilizado la plataforma Firebase de Google, que nos aporta el backend del servicio, a excepción del soporte para voz, el cual viene dado por una centralita software Asterisk que corre en una máquina Linux.

A lo largo de este documento se explicará en detalle cada una de las partes del proyecto, así como una introducción tanto a la programación Android como a cada uno de los servicios y a cada software utilizado.

Índice

Índice de figuras

Índice de tablas

Índice de acrónimos

GIMP	GNU Image Manipulation Program (Edición de imágen)
GPRS	General Packet Radio Service (Sistema de telecomunicaciones)
IDE	Integrated Development Environment (Entorno de desarrollo)
IP	Internet Protocol (Protocolo de Internet)
PBX	Private Branch Exchange (Centralita telefónica)
PC	Personal Computer (Ordenador personal)
TLS	Transport Layer Security (Protocolo criptográfico)
VoIP	Voice over IP (Transporte de voz sobre el protocolo IP)
Wi-Fi	Wireless Fidelity (Mecanismo de conexión inalámbrica)
XML	Extensible Markup Language (Lenguaje de marcas)

1. Introducción

En esta memoria se va a desarrollar cada uno de los aspectos del proyecto, en el cual se ha desarrollado una aplicación para la gestión de las comunicaciones entre usuarios, basada en la arquitectura cliente-servidor.

Dicha aplicación se ha desarrollado para dispositivos Android, ésta posibilita una comunicación sencilla entre usuarios mediante texto y voz sobre IP. El servicio que da la aplicación se apoya en una base de datos en tiempo real, un servicio de autenticación de usuarios, un almacenamiento cloud y una centralita software. Estos servicios, excepto la centralita, nos los ofrece la plataforma Firebase de Google, la cual nos ofrece amplios servicios de backend a la hora de desarrollar aplicaciones.

El sistema completo tendría una arquitectura como la siguiente:

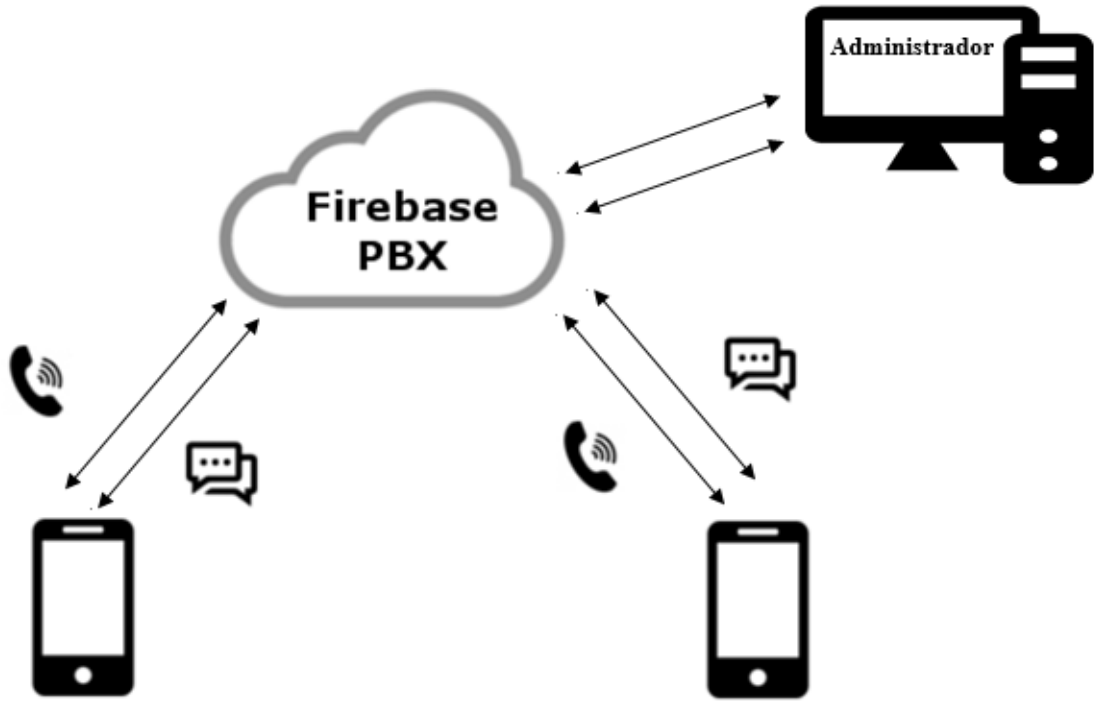

Figura 1. Arquitectura del sistema.

El proyecto engloba tanto la parte del cliente, constituida por la aplicación Android, como la parte del servidor. Ambas partes serán desarrolladas en siguientes apartados.

1.1. Motivación

La principal motivación para la realización de este proyecto ha sido la importancia que tienen los smartphones a día de hoy en nuestras vidas, y la necesidad que tenemos en la actualidad de estar siempre en contacto con nuestros conocidos a través de Internet. Esto hace que, en el ámbito de las comunicaciones, resulte más útil desarrollar software para estos dispositivos que para PCs.

Los smartphones se han convertido en nuestros compañeros inseparables en el día a día, lejos quedan los tiempos en los que si querías utilizar una red social, navegar por internet o simplemente enviar un email no tenías más remedio que recurrir a un PC. Hoy en día todas estas funciones, además de muchas otras, las tenemos en la palma de la mano y, gracias al avance de las redes móviles, podemos tener acceso a Internet casi en cualquier lugar del mundo.

Para el desarrollo de la aplicación se ha elegido el sistema operativo Android, esta elección ha sido motivada por ser el sistema operativo móvil más usado del mundo y por toda la ayuda que Google aporta para hacerles la vida más fácil a los desarrolladores.

Android tiene mucho futuro y ofrece muchas posibilidades. Es un sistema operativo para el que merece la pena desarrollar. Este proyecto es también una oportunidad de aprendizaje para profundizar en el desarrollo Android, y en él se aplicaran los conocimientos adquiridos durante estos años de formación universitaria.

1.2. Objetivos

A continuación se detallan los objetivos del trabajo de fin de grado, tanto a nivel docente como a nivel de proyecto:

- Desarrollo de una aplicación web que lleve el control y seguimiento de las comunicaciones entre usuarios.

- Diseño y puesta en marcha de una base de datos en un sistema gestor de bases de datos.

- Desarrollo de una aplicación móvil sencilla para llevar a cabo la comunicación entre usuarios.

- Posibilitar la comunicación entre usuarios tanto a través de texto como de voz.

- Elaboración de un proyecto de ingeniería de donde estarán incluidas todas las fases: análisis, diseño y planificación, codificación y prueba.

- El aprendizaje de nuevas tecnologías de manera autónoma y la elección de las diferentes estrategias para resolver los problemas que le puedan surgir en el desarrollo del sistema.

- La aplicación práctica de los contenidos vistos en la titulación.

2. Análisis del estado del arte

En este capítulo se va a llevar a cabo un resumen sobre el nacimiento y evolución de los smartphones. También se va a realizar una comparativa de los distintos sistemas operativos que encontramos en este campo, finalizando con un breve resumen sobre la telefonía IP.

2.1. Smartphones. Nacimiento y evolución

Un smartphone es un teléfono móvil que integra muchas de las capacidades de un ordenador, los cuales se pueden considerar ordenadores personales de bolsillo.

Se puede decir que el primer teléfono móvil del mundo fue el Motorola Dyna TAC 8000X, utilizado por primera vez en 1973 por Martín Cooper, directivo de Motorola considerado el padre del teléfono móvil. Este dispositivo se puso a la venta en 1984 a un escalofriante precio de 3.995 dólares, a pesar de esto, muchos usuarios adquirieron el preciado dispositivo.

El Dyna TAC 8000X pesaba casi un kilo y medía 33 centímetros de alto, lejos queda aquel primer teléfono móvil de los dispositivos actuales, desde entonces se ha realizado un crecimiento exponencial en este sector.

- Años más tarde, empezamos a encontrar dispositivos que pueden ser considerados los primeros smartphones:

- En 1994 salió al mercado el que está considerado el primer smartphone de la historia, el IBM Simon Communicator. Este dispositivo permitía tareas como enviar emails y mensajes de texto, también contaba con una pantalla táctil. Así

unía las funciones de una agenda electrónica y de un teléfono móvil en un sólo dispositivo.

- La compañía Nokia lanzó en 1996 el Nokia 9000 Communicator, que disponía de funciones como calendario, calculadora y envío de fax y correos electrónicos.

- El término smartphone fue utilizado por primera vez en 1997 por la compañía Ericsson para describir a su dispositivo GS88 Penélope como un "teléfono inteligente", el cuál incorporaba un teclado QWERTY.

- En 2002 salió al mercado el BlackBerry 5810, un dispositivo que nos permitía incluso acceder a Internet por GPRS.

Todos estos dispositivos anteriores quedan algo lejanos a la idea de Smartphone que tenemos en la actualidad, la cual llegó en el año 2007 cuando Steve Jobs presentó el iPhone, un dispositivo con pantalla táctil, cámara y hasta conexión Wi-Fi. Utilizaba el sistema operativo IOS.

Desde aquellos comienzos los smartphones han crecido a pasos agigantados y, en la actualidad, se han convertido en nuestros compañeros inseparables. Estos dispositivos entán presentes en el día a día y cubren la mayoría de las funciones de los ordenadores personales.

2.2. Sistemas operativos móviles

A continuación se realizará una comparativa diferentes sistemas operativos enfocados a los dispositivos móviles, la cual se centrará en tres plataformas:

2.2.1. iOS

Este sistema operativo pertenece a Apple, ha sido desarrollado para el iPhone, aunque también se utiliza actualmente en el iPad, que es una tableta basada en el iPhone que podríamos considerar como un intermediario entre el iPhone y el Mac. Su primera versión fue presentada en 2007 y actualmente se encuentra en la versión número 11.

Actualmente se encuentra en segundo lugar, por detrás de Android, con respecto a la cuota de mercado mundial.

Ventajas:

- La principal ventaja de este sistema operativo es que está diseñado íntegramente para el iPhone, llevando la compatibilidad software-hardware a un nivel que no encontramos en otras plataformas. Esto también se ve en la sincronización del iPhone con el Mac.

- Su interfaz está diseñada para conseguir una máxima simplicidad y hacerle la vida fácil al usuario.

- Las nuevas actualizaciones del sistema operativo se filtran rápidamente hasta los usuarios.

- Al ser un sistema más cerrado, se ejerce un mayor control sobre las aplicaciones que pueden instalar los usuarios, mejorando la seguridad.

Desventajas:

- Su interfaz es sencilla y fácil de utilizar, pero por el contra no permite mucha personalización.

- El IDE oficial para el desarrollo de aplicaciones iOS sólo está disponible para Mac.

- Apple sólo diseña dispositivos de gama alta, en consecuencia hay menos usuarios que puedan permitírselos.

2.2.2. Android

Es un sistema operativo basado en Linux creado por Android Inc., que posteriormente fue comprada por Google. Está diseñado para smartphones, pero también podemos encontrarlo en otros dispositivos como tabletas, televisores y relojes inteligentes.

Fue presentado en 2007, y actualmente va por la versión 8. Actualmente es el sistema operativo móvil más usado del mundo.

Ventajas:

- Permite una gran personalización, es algo menos sencillo que iOS pero por el contrario tenemos muchas más opciones en este ámbito.
- Es un sistema muy abierto, permitiendo que haya más aplicaciones desarrolladas para esta plataforma y podamos instalarlas desde fuera de la tienda de aplicaciones.
- Su IDE oficial lo encontramos para cualquier plataforma de escritorio.
- Podemos encontrar muchas más opciones a la hora de elegir un dispositivo, pues al ser código abierto, un sinfín de fabricantes pueden incorporarlo a sus dispositivos. Podemos encontrar dispositivos Android desde menos de cien euros, hasta más de mil.

Desventajas:

- Al existir tantos fabricantes distintos, con un hardware muy heterogéneo, no hay tanta integración software-hardware como en los dispositivos de Apple.
- Google desarrolla el sistema operativo, pero depende de cada uno de los fabricantes el hacer llegar las últimas actualizaciones hasta los usuarios, cosa que no siempre se cumple y, cuando lo hace, con algo de retraso.
- Al ser la plataforma más utilizada en el mundo encontramos mayor cantidad de malware, tanto fuera de la tienda de aplicaciones como en la propia tienda, a

causa de unas políticas por parte de Google demasiado permisivas a la hora de filtrar el software que se sube a la tienda.

2.2.3. Windows Phone

Se trata de un sistema operativo desarrollado por Microsoft y basado en Windows, sucesor de Windows Mobile. Como los dos anteriores, está orientado a los dispositivos móviles. Fue presentado en 2010 y, actualmente, va por la versión 10. Este sistema operativo es el que menos cuota de mercado tiene de los tres.

Ventajas:

- Tiene una interfaz muy sencilla.
- Cuenta con sincronización total con los PCs Windows.
- Podemos encontrar dispositivos en diferentes gamas de precios, al igual que en Android.

Desventajas:

- El número de aplicaciones que encontramos en su tienda es muy inferior al de iOS o Android.
- Baja posibilidad de personalización.
- Microsoft ha anunciado que no va a seguir desarrollando nuevas funcionalidades para esta plataforma.

2.2.4. Elección del sistema operativo para el proyecto

El sistema operativo elegido para el trabajo ha sido Android, para la elección se ha tenido en cuenta lo siguiente:

- **Cuota de mercado:** Android es el sistema operativo con mayor cuota de mercado del mundo.

- **IDE:** lo podemos encontrar de forma gratuita para diversos sistemas operativos.
- **Funcionalidad:** posee muchísimas funcionalidades para explotar todo el hardware del dispositivo.

Windows Phone ha quedado descartado en primer momento, no solo por su baja cuota de mercado, sino también porque el hecho de que Microsoft deje de lado el desarrollo para esta plataforma, la condena a la extinción, no teniendo sentido desarrollar para ella.

iOS es un sistema operativo con muchas posibilidades, pero no puede igualarse a Android en cuota de mercado, esto lo podemos ver en la siguiente figura:

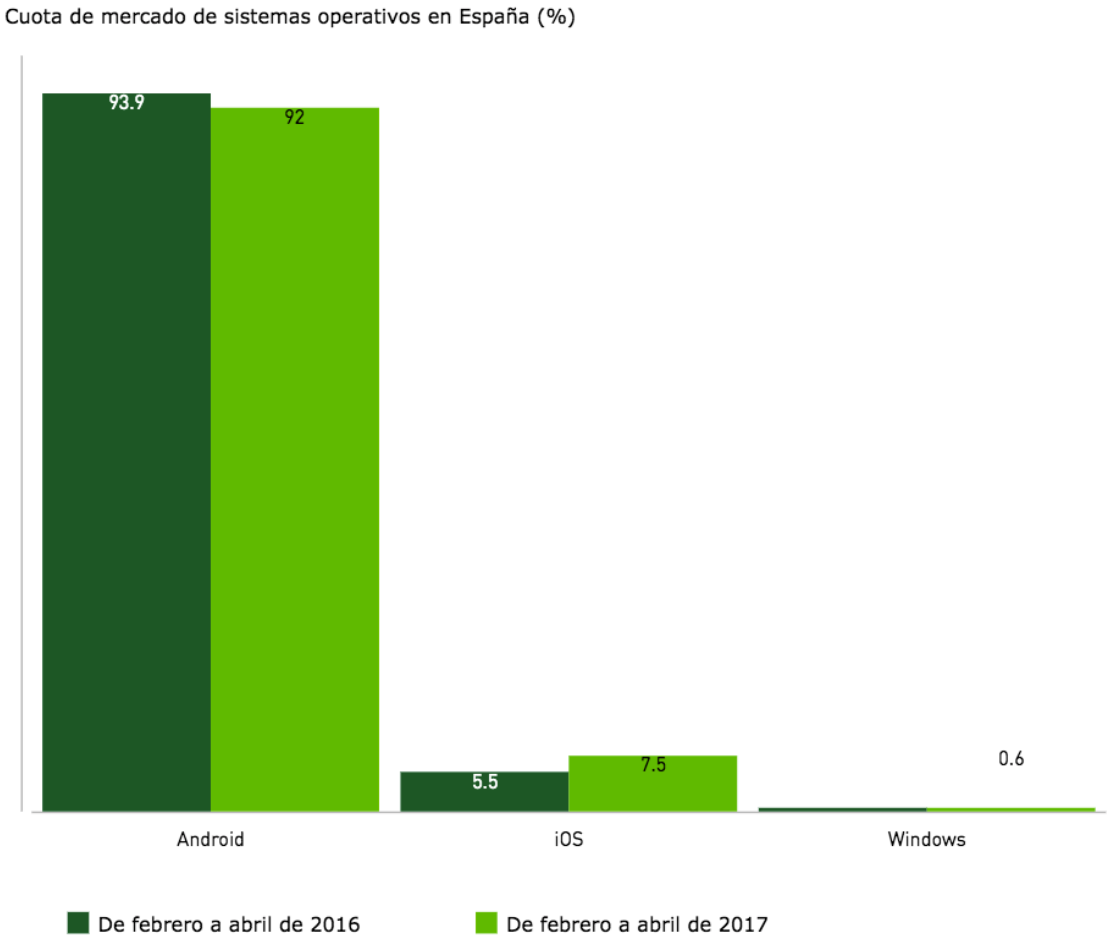

Figura 2. Cuota de mercado de plataformas móviles en España. [1]

2.3. Telefonía IP (VoIP)

Como su nombre indica consiste en transmitir señales de voz a través de Internet, apoyándose en el protocolo IP. Esto hace posible realizar llamadas telefónicas gratuitas a través de Internet. Requiere de una conexión con un buen ancho de banda ya que, si se produjeran retardos, resultaría muy molesto para el usuario o incluso haría imposible una comunicación entendible por voz.

A demás del protocolo IP intervienen algunos otros, los principales a la hora de realizar una llamada de VoIP son:

- **SIP**: Protocolo de iniciación de sesión, es el encargado de iniciar, modificar y finalizar las sesiones multimedia.
- **SDP**: Protocolo de descripción de sesión, es el encargado de describir los parámetros de una sesión multimedia.
- **RTP**: Protocolo de transporte en tiempo real, es el encargado de transportar contenido multimedia en tiempo real, por ejemplo el audio en una llamada IP.
- **RTCP**: Protocolo de control en tiempo real, proporciona información de control asociada a un flujo RTP.

En un sistema VoIP encontramos tres elementos fundamentales como podemos ver en la figura 3:

- **Terminales**: son el equivalente a los teléfonos tradicionales (teléfonos digitales, softphones).
- **Gateways**: son los encargados de enlazar con la red telefónica tradicional.
- **Gatekeepers**: son el equivalente a las centrales telefónicas (PBX).

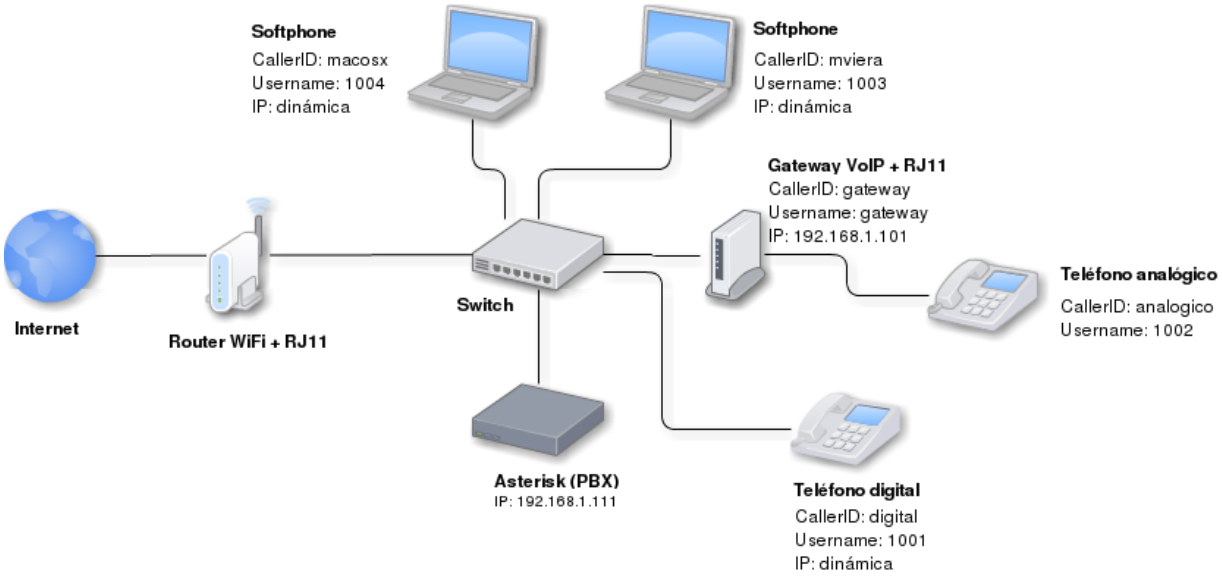

Figura 3. Arquitectura de un sistema VoIP. [2]

Una llamada VoIP requiere un proceso de intercambio de mensajes entre los interlocutores y el servidor, a continuación se muestra un esquema de los intercambios de mensajes en una sesión SIP de una llamada IP:

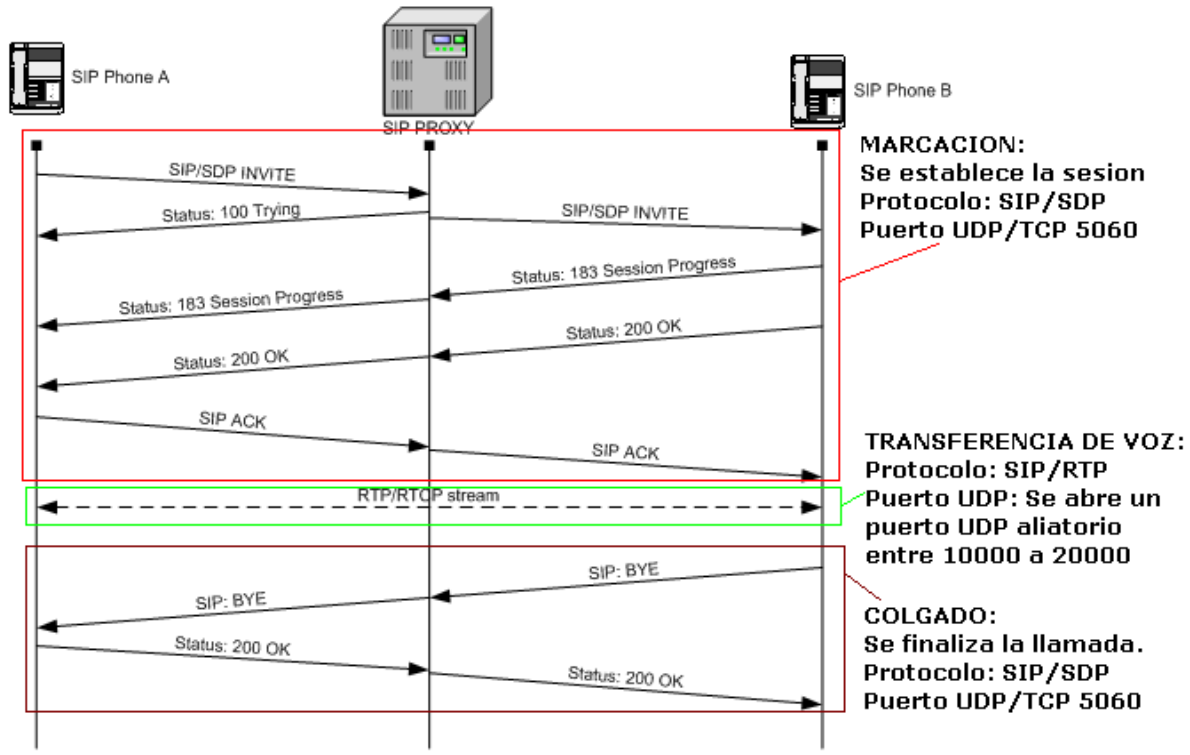

Figura 4. Intercambio de mensajes en una llamada IP. [3]

En conclusión la VoIP presenta una gran oportunidad y es el futuro de la telefonía, ya que un usuario tan sólo necesitaría una conexión a Internet para realizar llamadas de manera gratuita. Este servicio se irá haciendo más notable con el avance de las tecnologías y el aumento de la velocidad en las redes.

3. La plataforma Android

En este capítulo se hablará de la plataforma móvil elegida para la realización del proyecto, Android. Se trata de un sistema operativo desarrollado por Google, está basado en Linux, y pensado para dispositivos móviles con pantalla táctil.

Se presentó en el año 2007 como un competidor para iOS, siendo el HTC Dream el primer Smartphone con Android del mundo. En la actualidad es el sistema operativo más utilizado del mundo.

Como podemos ver en la Figura 6 Android cuenta con numerosas versiones, encontrándose actualmente en su octava versión:

Nombre código	Número de versión	Fecha de lanzamiento	Nivel de API
Android 1.0[1]	1.0	23 de septiembre 2008	1
Android 1.1[1]	1.1	9 de febrero 2009	2
Cupcake	1.5	27 de abril de 2009	3
Donut	1.6	15 de septiembre de 2009	4
Eclair	2.0–2.1	26 de octubre de 2009	5-7
Froyo	2.2–2.2.3	20 de mayo 2010	8
Gingerbread	2.3–2.3.7	6 de diciembre 2010	9–10
Honeycomb[2]	3.0–3.2.6	22 de febrero de 2011	11–13
Ice Cream Sandwich	4.0–4.0.5	18 de octubre 2011	14–15
Jelly Bean	4.1–4.3.1	9 de julio de 2012	16–18
KitKat	4.4–4.4.4, 4.4W–4.4W.2	31 de octubre de 2013	19–20
Lollipop	5.0–5.1.1	12 de noviembre de 2014	21–22
Marshmallow	6.0–6.1	5 de octubre de 2015	23
Nougat	7.0 - 7.1.2	15 de junio de 2016	24-25
Oreo	8.0	21 de agosto de 2017	26

Figura 5. Historial de versiones de Android. [4]

3.1. Arquitectura del sistema

A continuación se muestra una ilustración donde podemos observar las diferentes partes del sistema Android:

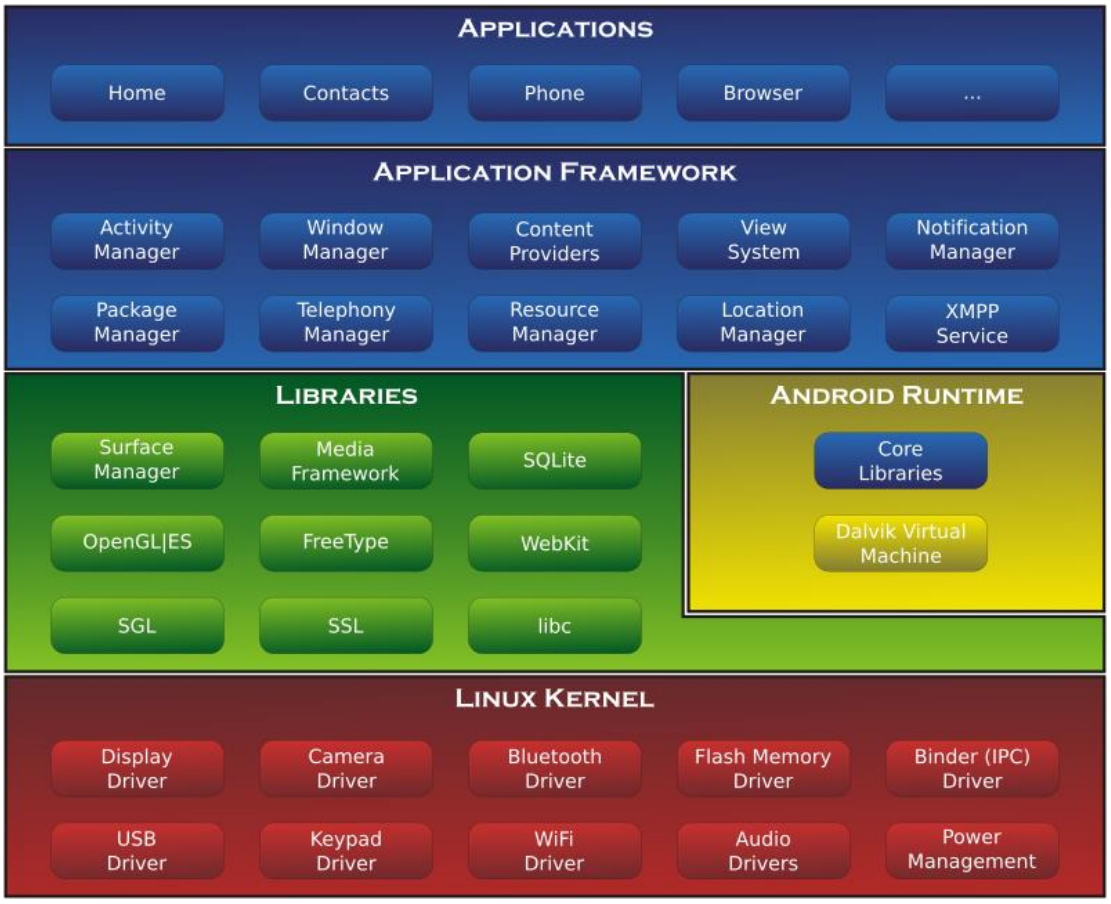

Figura 6. Arquitectura del sistema Android. [4]

A continuación se realizará una breve explicación de cada una de las partes del sistema:

- **Kernel Linux:** se utiliza un kernel Linux 2.6, esta capa contiene los drivers necesarios para la utilización de cada uno de los componentes de hardware del dispositivo.

- **Librerías:** en esta capa encontramos las librerías que utiliza Andoid, escritas en C/C++. Éstas proporcionan a Android la mayor parte de sus características. En este mismo nivel encontramos **Android Runtime**, constituido por librerías Java y una máquina virtual Dalvik.

- **Framework de aplicaciones:** aquí encontramos las herramientas de desarrollo que utilizan las aplicaciones Android.

- **Aplicaciones:** en este nivel encontramos cada una de las aplicaciones del dispositivo.

3.2. Estructura de las aplicaciones Android

Una aplicación Android consta de diferentes partes, a continuación se explican las principales:

- **Actividades:** están compuestas por una parte visual, la cual viene definida por un archivo xml, y otra parte de código Java que controla el funcionamiento de la vista y realiza las acciones pertinentes. Cuentan con el siguiente ciclo de vida:

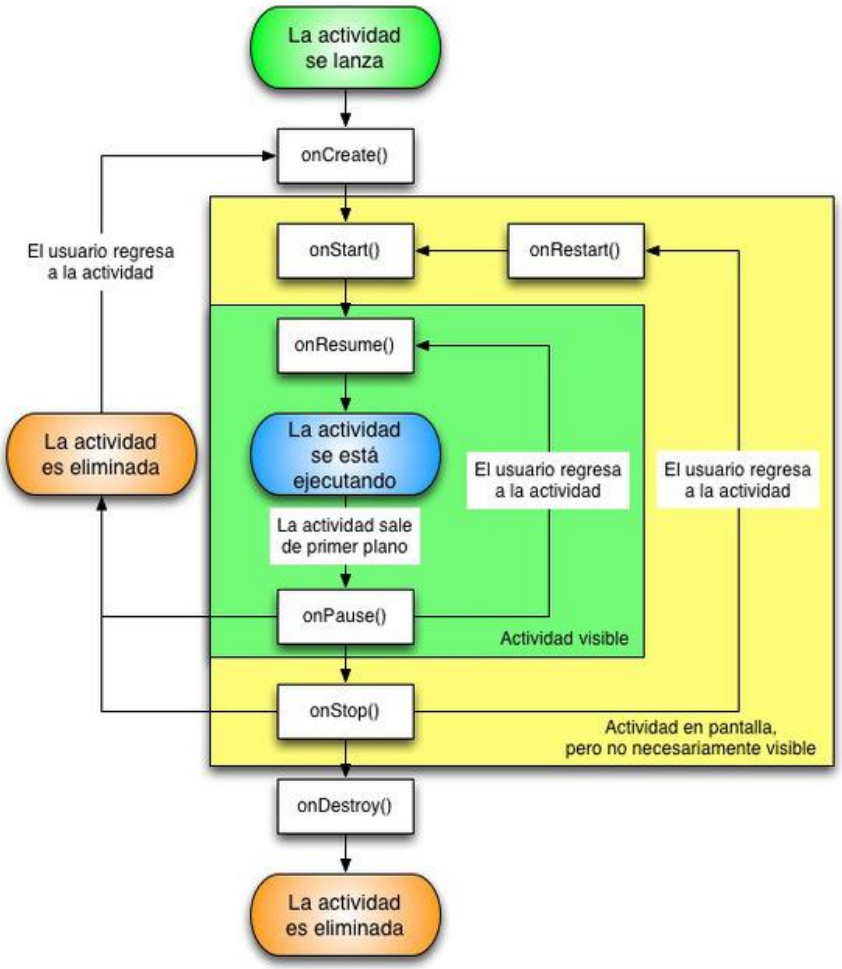

Figura 7. Ciclo de vida de una actividad Android. [5]

- **Servicios:** son fragmentos de código que se ejecutan en segundo plano, en el caso que nos ocupa es lo que hace posible que recibamos notificaciones aunque la aplicación no esté en primer plano.

- **Broadcast Receivers:** no poseen interfaz de usuario, se utilizan para hacer posible la respuesta a eventos, como en este caso, responder a una llamada entrante.

- **Fragmentos:** corresponden a vistas al igual que las actividades, pero estas están enfocadas a formar parte de una actividad. Una actividad puede estar compuesta por varios fragmentos, haciendo así una vista más dinámica, posibilitando cambios de fragmentos de la vista sin tener que cambiar de actividad.

- **Permisos:** las aplicaciones Android se ejecutan en un entorno cerrado, si necesitan hacer una función determinada es necesario que tengan permiso. Si por ejemplo la aplicación necesita hacer uso de la cámara del dispositivo, esta necesitará permiso para ello.

- **Manifiesto:** está compuesto por el archivo AndroidManifest.xml, actúa como medio de control para la aplicación. Contiene la declaración de todos los permisos, versiones, actividades, servicios, etc… que va a necesitar la aplicación para funcionar.

3.3. Android Studio

Este es el IDE oficial para el desarrollo de aplicaciones Android, está basado en IntelliJ IDEA. Con el podemos crear aplicaciones para todo tipo de dispositivos Android, desde smartphones hasta smartwatches. Está disponible para su descarga de forma gratuita en www.developer.android.com, la página web oficial de desarrollo Android. Se puede instalar en Windows, Mac y Linux.

Algunas de sus características son:

- **Instant Run:** nos permite ver en el dispositivo Android los cambios que vallamos haciendo en el código en tiempo real, sin necesidad de volver a compilar el proyecto.

- **Editor de código inteligente:** Android Studio nos ayuda en cada línea de código, dándonos sugerencias, ayudando a corregir errores y analizando el código de manera avanzada.

- **Emulador Android:** este IDE cuenta con un emulador con el que podremos ejecutar nuestra aplicación en todo tipo de dispositivos.

- **Compilación:** su sistema de compilación está basado en Gradle.

- **Integración:** total integración con los servicios de google.

  En la siguiente figura podemos ver la interfaz de usuario de este software:

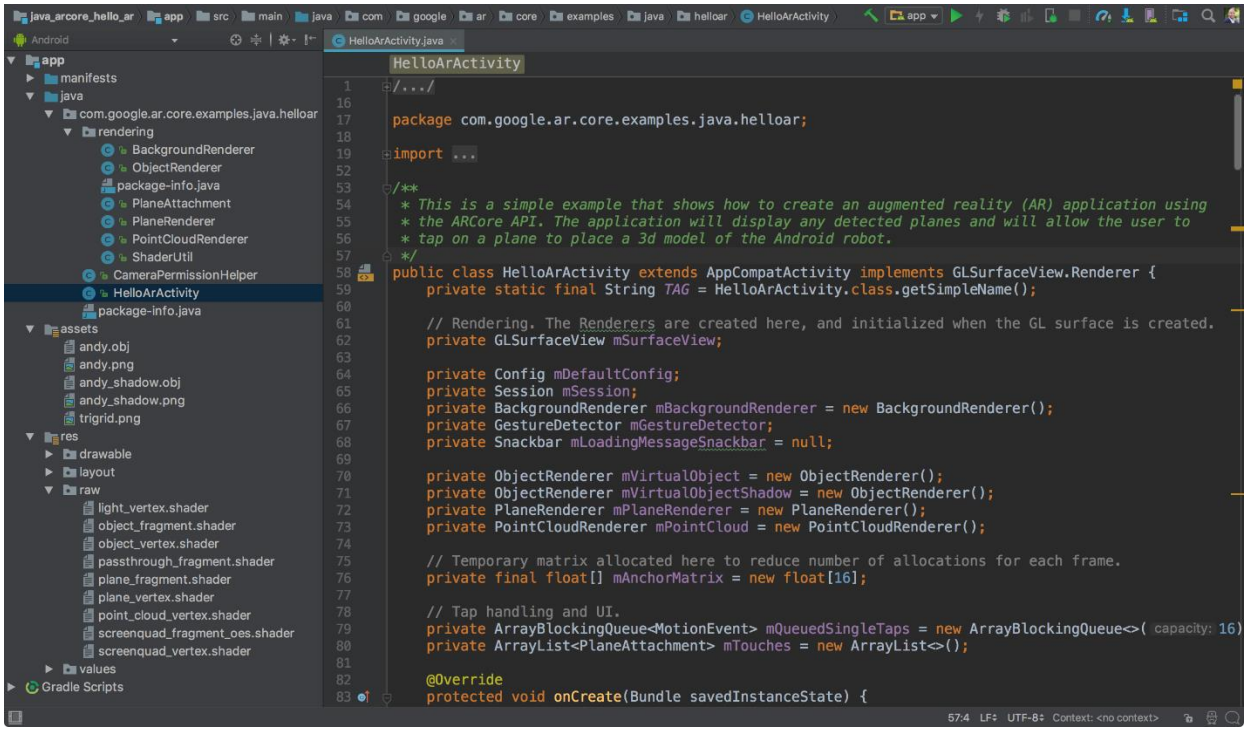

Figura 8. Interfaz de Android Studio. [6]

4. Diseño técnico

En este apartado se planteará la solución técnica seleccionada para conseguir llevar a cabo los objetivos marcados en el proyecto.

4.1. Planteamiento del problema

En la actualidad es importante poder comunicarnos en cualquier momento con otra persona, independientemente de la distancia que nos separe. Esto puede resultar caro si utilizamos los servicios de telefonía tradicionales como llamadas y mensajes sms, con la ayuda de Internet este problema se ha ido solucionando, ya que podemos compartir información con cualquier parte del mundo sin coste adicional por distancia.

Ahora está en manos de los desarrolladores crear servicios que nos aporten las mismas facilidades y simpleza de las comunicaciones tradicionales. Hoy en día existen diversos servicios de este tipo, como por ejemplo WhatsApp, pero que están perdiendo su esencia en un intento de imitar la funcionalidad de las redes sociales.

También cabe destacar el alto coste de mantenimiento que puede tener dar un servicio de mensajería o de voz a un gran número de usuarios. Requiriendo realizar una labor constante en la parte de servidor tanto para mantener el buen funcionamiento del servicio como para proporcionar una seguridad adecuada.

4.2. Planteamiento de la solución

Como solución se ha realizado una aplicación móvil sencilla de utilizar, sin extras innecesarios que estropeen la experiencia de usuario o compliquen el uso de la aplicación. Al estar desarrollada para smartphones Android, es accesible para casi cualquier persona y nos aporta un mayor parentesco con las tecnologías tradicionales.

En la parte de servidor se ha optado por Firebase, un servicio de Google que nos aporta las funcionalidades de servidor, sin necesidad de que el usuario realice ningún

mantenimiento adicional en este más allá de las acciones propias requeridas por su software frontend concreto.

Para hacer posible el servicio de VoIP se ha seleccionado la centralita software Asterisk, funcionando en una máquina con Linux. Estos son productos gratuitos, lo que no encarece el producto final.

4.3. Diseño de la aplicación

La aplicación se llamará “ChatApp” y constará con un total de once vistas, de las cuales nueve son actividades y dos son fragmentos:

- **Pantalla de bienvenida:** es la primera vista que encontraremos al abrir por primera vez la aplicación, en ella se da la bienvenida al usuario.

- **Pantalla de inicio de sesión:** constará de los campos necesarios para que el usuario pueda introducir sus datos e iniciar sesión, así como un acceso a la pantalla de registro.

- **Pantalla de registro:** constará de los campos necesarios para que el usuario pueda introducir sus datos y registrarse en el sistema.

- **Pantalla principal:** es la pantalla central de la aplicación, en ella el usuario podrá ver tanto sus chats iniciados como su lista de contactos. También tendrá acceso a la edición de su perfil, y a la opción de búsqueda de usuarios. Por último encontraremos la opción de cerrar sesión.

    - Dentro de esta vista, tanto la **lista de chats** como la de **contactos** estarán constituidas por fragmentos, los cuales se irán cambiando según la selección del usuario.

- **Pantalla de edición de perfil:** aquí el usuario tendrá una vista de su nombre y su foto de perfil, así como la posibilidad de modificarlos.

- **Pantalla de búsqueda:** en esta vista el usuario podrá buscar a otros usuarios filtrando por el nombre introducido, y tendrá la posibilidad de agregarlos a nuestros contactos.

- **Pantalla de chat:** es la vista de cada uno de los chats, donde podremos ver la lista de mensajes que tenemos con ese usuario, su foto y su nombre. En la parte inferior encontraremos los elementos para escribir y enviar mensajes. En la parte superior de la vista también encontraremos opciones para llamar al usuario, eliminar el chat y eliminar al usuario de nuestros contactos.

- **Pantalla de foto:** esta vista consta de una visualización a pantalla completa de la foto de perfil de un usuario.

- **Pantalla de llamada:** esta pantalla se visualizará tanto cuando iniciemos una llamada como cuándo recibamos una. En ella podremos ver el nombre y la foto de perfil del usuario al que llamamos o que nos llama. En la parte inferior encontraremos los botones necesarios para responder y colgar la llamada.

5. Desarrollo del servicio

En este apartado se va a explicar en detalle el desarrollo tanto de la aplicación móvil cómo de la parte de servidor.

5.1. Servidor

La zona de servidor va a contar con dos partes, una encargada únicamente del servicio de voz sobre IP, y otra encargada del resto del servicio.

Para la parte de voz sobre IP se utilizará una centralita software Asterisk corriendo en una máquina Linux, y para el resto del servicio se utilizará la plataforma Firebase de Google.

5.1.1. Asterisk

Se trata de un software libre que nos proporciona las funcionalidades de una central telefónica, permitiendo que usuarios conectados a él realicen llamadas telefónicas entre ellos.

Este software correrá en una máquina Linux, concretamente Ubuntu 16. Dicho sistema operativo se ha descargado de la página oficial, www.ubuntu.com, y funciona en una máquina virtual en un PC con Windows 10, como máquina virtual se utiliza VirtualBox.

El proceso para la instalación y configuración del software Asteisk es el siguiente:

- Para instalar Asterisk, abrimos el terminal de comandos de Ubuntu y ejecutamos el siguiente comando: “sudo apt-get install asterisk”. Una vez finalizado el proceso, quedaría instalado el software.

- El siguiente paso es crear a los usuarios que usarán los servicios de la centralita, para ello debemos editar dos de los ficheros de configuración de Asterisk: sip.conf y extensions.conf. Estos ficheros se encuentran en la ruta */etc/asterisk*.

    - **sip.conf**: aquí crearemos a cada usuario, añadiendo las siguiente al final del archivo para cada usuario (ejemplo usuario 1 con contraseña 12345 y contexto prueba):

        - [1]

          type=fiend

          secret= 12345

          qualify=200

          context=prueba

          callerid=1<1>

          host=dynamic

    - **extensions.conf**: ahora se crearán las extensiones para cada usuario, se deberán añadir las siguientes líneas para cada usuario al final del archivo (ejemplo usuario 9, configuración para contactar con él al marcar 9, contexto prueba):

        - [prueba]

          exten => 9,1,Dial(SIP/9,30,Ttm);

          exten => 9,2,Hangup;

          exten => 9,102,Voicemail(9);

          exten => 9,103,Hangup;

- Para iniciar el servicio de centralita se utiliza el comando: “sudo service asterisk start”.

- Podemos abrir la consola de Asterisk con el comando “sudo asterisk –r”, para ver los sucesos que valla notificando el software y visualizar los diferentes usuarios y su estado. Dentro de esta consola también podemos realizar configuraciones extra.

Una vez completados estos pasos, quedaría configurado y operativo el servicio de centralita, a la espera de que los usuarios de la aplicación móvil se registren en él. La estrategia elegida para el registro de usuarios será explicada en el siguiente apartado.

5.1.2. Firebase

Es una plataforma de Google donde podemos acceder a diferentes servicios cloud para nuestros proyectos de software en plataformas como Android, iOS o Web. Tan solo necesitaremos una cuenta de Google, el servicio es gratuito con limitaciones y existen otros planes de pago según el uso que vallamos a hacer.

Los servicios ofrecidos son los siguientes:

- **Realtime Database:** una base de datos noSQL del tipo clave-valor, con sincronización de datos en tiempo real.

- **Crash Reporting:** servicio para reportar errores de las aplicaciones.

- **Authentication:** servicio de autenticación de usuarios mediante diversos métodos como email y contraseña, cuenta de Google, Facebook o Twitter.

- **Cloud Functions:** permite alojar tu propio código de backend.

- **Cloud Storage:** servicio de almacenamiento cloud para tus aplicaciones.

- **Cloud Firestone:** nueva versión de la base de datos en tiempo real, actualmente se encuentra en versión beta.

- **Hosting:** alojamiento web con certificado SSL gratuito.

- **Test Lab para Android:** permite ejecutar pruebas automáticas para las aplicaciones Android.
- **Supervisión del rendimiento:** diagnósticos de rendimiento de la aplicación.
- **Google Analytics:** servicio de analíticas para el comportamiento de los usuarios en la aplicación.
- **Cloud Messaging:** envío de notificaciones a los distintos usuarios de tus aplicaciones, en Android, iOS y Web.
- **Invites:** permite que los usuarios compartan invitaciones para la instalación de tu aplicación.
- **App Indexing:** tu aplicación aparecerá cuando los usuarios busquen aplicaciones similares.
- **AdMob:** permite monetizar la aplicación mostrando publicidad.
- **AdWords:** publicación de anuncios orientada a segmentos.

La utilización de estos servicios es gratuita, pero tiene algunas limitaciones como la cantidad de datos almacenados, si queremos sobrepasar estas limitaciones contamos con planes de pago según el uso que vallamos a hacer de los servicios.

A continuación se puede ver un resumen completo de los precios por uso de cada servicio que encontramos para Firebase:

Productos	Plan Spark Límites generosos para aficionados Sin cargo	Plan Flame Precios predecibles para apps en expansión USD 25/mes	Plan Blaze Calcula los precios de las apps a gran escala Pago por uso
Incluido sin cargo Analytics, App Indexing, Dynamic Links, Invites, Notifications, Remote Config, Authentication, Cloud Messaging y Crash Reporting.	✓ Incluidos	✓ Incluidos	✓ Incluidos
Realtime Database			
Conexiones simultáneas	100	100K/instance	100K/instance
GB almacenados	1GB	2.5 GB	$5/GB
GB descargados	10 GB/mes	20 GB/mes	USD 1/GB
Copias de seguridad automáticas	✗	✗	✓
Cloud Firestore			
Stored data	1 GB total	2.5 GB total	$0.18/GB
Bandwidth	10GB/month	20GB/month	Google Cloud Pricing
Document writes	20,000/día	100K/day	$0.18/100K
Document reads	50,000/día	250,000/día	$0.06/100K
Document deletes	20,000/día	100K/day	$0.02/100K
Storage			
GB almacenados	5 GB	50 GB	USD 0.026/GB
GB descargados	1 GB/día	50 GB/día	USD 0.12/GB
Operaciones de carga	20,000/día	100K/day	$0.05/10k
Operaciones de descarga	50,000/día	250,000/día	$0.004/10k
Cloud Functions			
Invocaciones	125,000/mes	2,000,000/mes	USD 0.40/millón
GB-segundo	40,000/mes	400K/month	USD 0.0025/mil
CPU-segundo	40,000/mes	200K/month	USD 0.01/mil
Redes de salida	Solo para Google	5 GB/mes	USD 0.12/GB
Phone Auth			
US, Canada, India	10k/month	10k/month	$0.01/verification
All other countries			$0.06/verification
Hosting			
GB almacenados	1 GB	10 GB	USD 0.026/GB
GB transferred	10 GB/mes	50GB/month	USD 0.15/GB
Dominio personalizado y SSL	✓	✓	✓
Test Lab			
Virtual Device Tests	10 tests/day	10 tests/day	$1/device/hour
Physical Device Tests	5 tests/day	5 tests/day	$5/device/hour

Figura 9. Precio de Firebase [7]

Para el desarrollo del servicio del proyecto se han usado los siguientes servicios:

- Autenticación de usuarios mediante email y contraseña.
- Base de datos en tiempo real para almacenar todos los datos de los usuarios:
    - Cada usuario está identificado en la base de datos por su UID, código que se obtiene al registrarse en el sistema y que es único para cada usuario. Dentro del apartado de cada usuario tenemos tres apartados:
        - **mis_datos**: en este apartado encontramos el nombre del usuario y el usuario que tiene asuignado en Asterisk.
        - **chats**: aquí están cada uno de los chats que el usuario tiene comenzados, representados por el UID de la persona con la que tienen cada chat. Los mensajes se irán guardando en formato: “in”,”out”, representando si es un mensaje entrante o saliente. Cada “in” y “out” ira acompañado de un número que representa el orden de los mensajes en la conversación. Cada mensaje contendrá la fecha y la hora a la que se envió, así como el texto del mensaje.
        - **contactos:** en este apartado está guardada la lista de contactos del usuario, representada por los UID de cada uno de los contactos y su nombre.
    - También hay una lista con todos los usuarios de Asterisk, donde cada uno está marcado como “libre” u “ocupado”, según si está asociado o no a algún usuario.
- Almacenamiento cloud para guardar cada una de las fotos de perfil de los usuarios.
- El apartado de notificaciones para hacer posible el envío de información, como por ejemplo avisos de mantenimiento, a todos los usuarios.

A continuación se muestran algunas capturas de la plataforma:

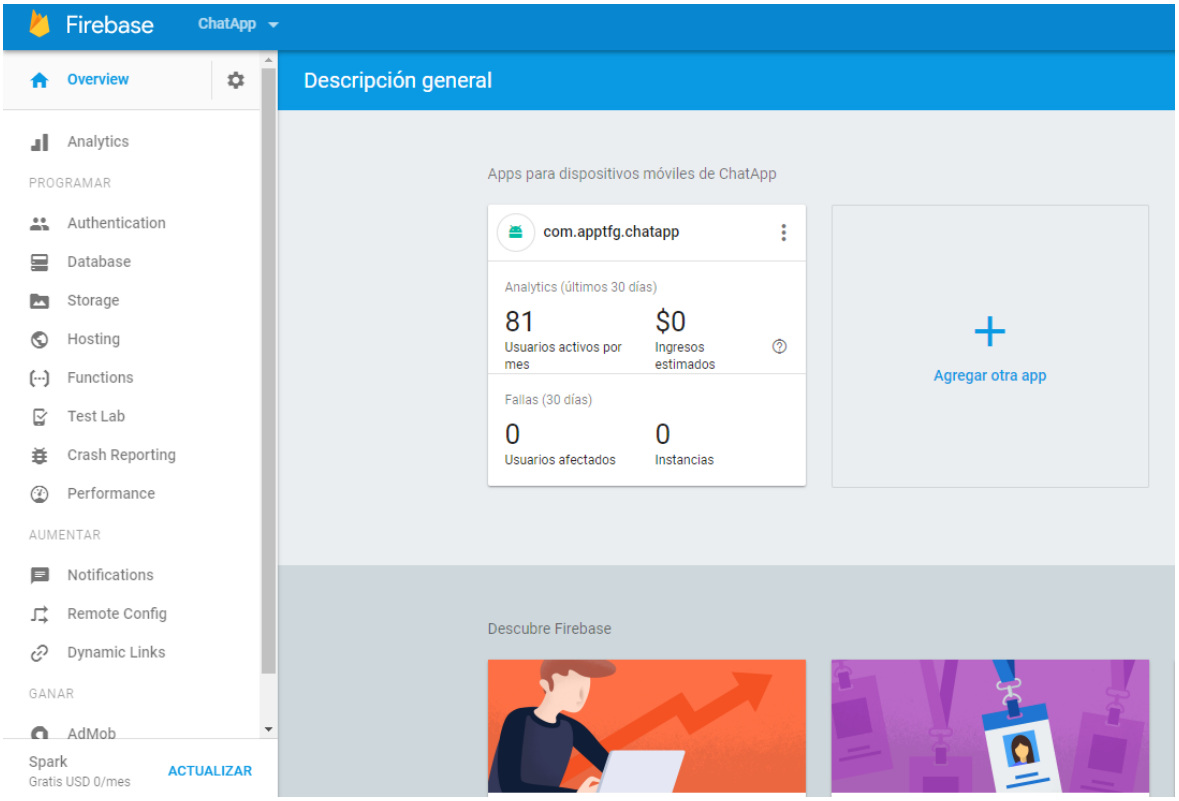

Figura 10. Página principal de Firebase.

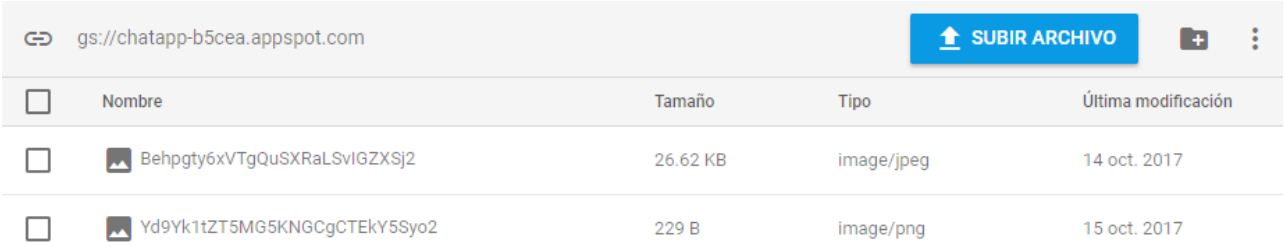

Figura 11. Almacenamiento cloud de fotos de perfil.

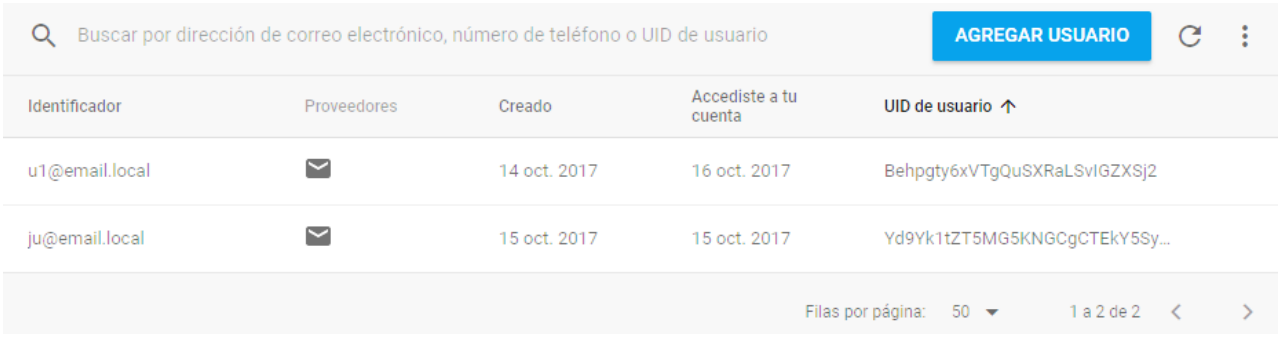

Figura 12. Cuentas de usuarios.

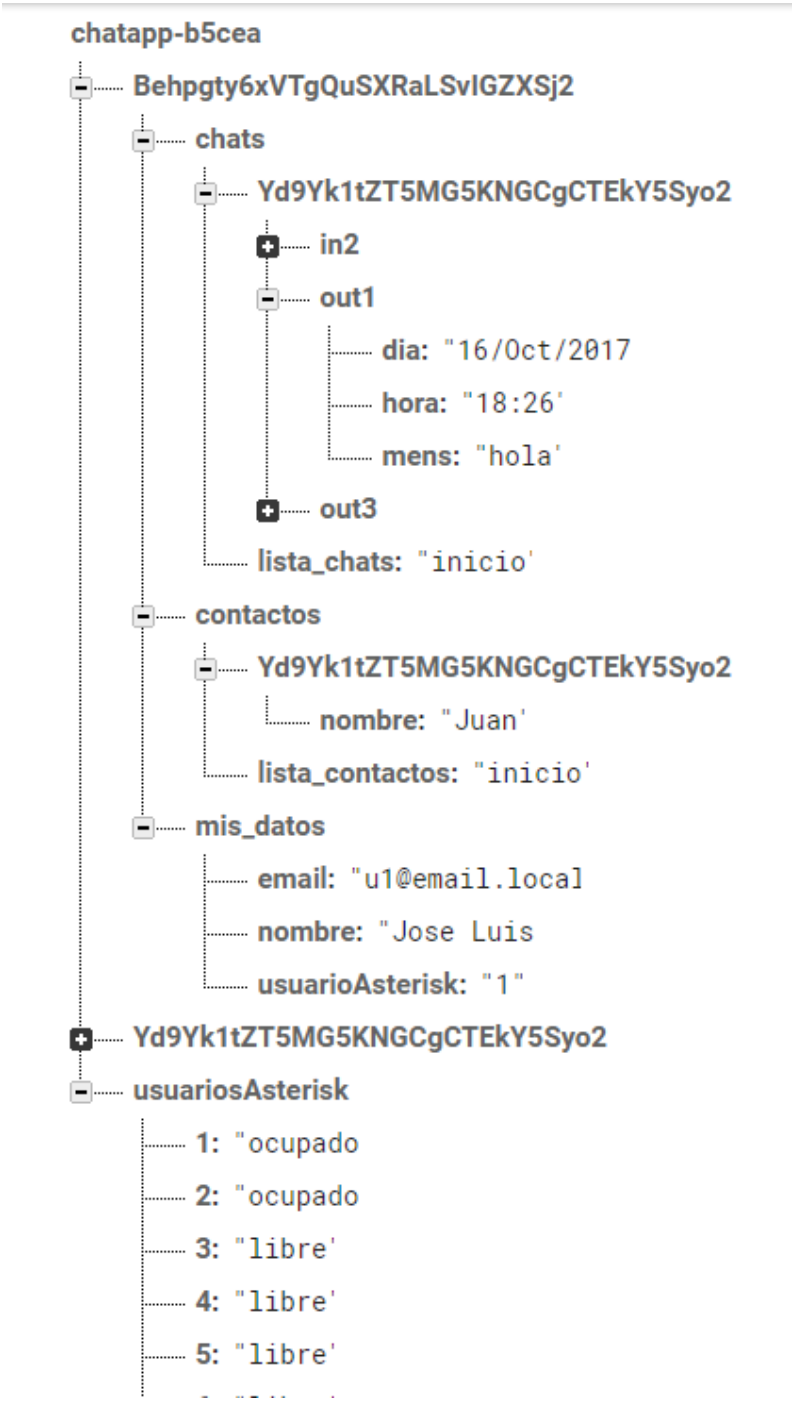

Figura 13. Captura de la base de datos.

Como se puede observar, la plataforma tiene una interfaz sencilla y fácil de manejar, que da muchas posibilidades.

5.2. Aplicación móvil

A continuación se va a explicar el desarrollo de la aplicación Android, la cual ha sido desarrollada con el IDE oficial, Android Studio. La aplicación se llamará "ChatApp". Se explicarán las funciones principales, para más detalle se puede recurrir al código completo del proyecto, que se adjunta con este documento.

5.2.1. Conexión con Firebase

Lo primero que debemos hacer es, una vez instalado Android Studio desde su página oficial de Android Developers, crear un nuevo proyecto. Una vez creado el proyecto "ChatApp", el siguiente paso es conectar dicho proyecto con Firebase, para ello seguiremos los siguientes pasos en orden:

- Se accede a www.console.firebase.google.com donde, tras iniciar sesión con nuestra cuenta de Google, crearemos un nuevo proyecto Firebase pulsando en el "+" que encontraremos en la página.

- En el siguiente campo introducimos el nombre del proyecto y nuestro país, después pulsamos en "crear proyecto".

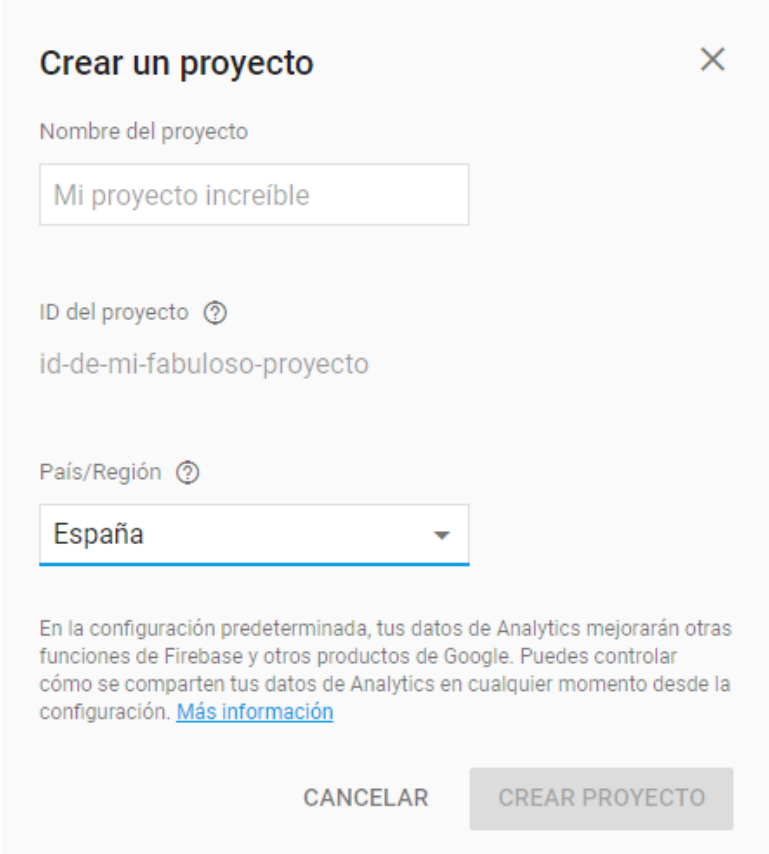

Figura 11. Crear proyecto Firebase

- Se abrirá la pantalla de control de Firebase, donde debemos pulsar en la opción "Agrega Firebase a tu app para Android:

Figura 12. Agregar Firebase a Android

- Se abrirá una ventana en la que sólo debemos seguir los pasos que nos indican:
    - Paso 1, rellenar los campos y pulsar en "registrar app":

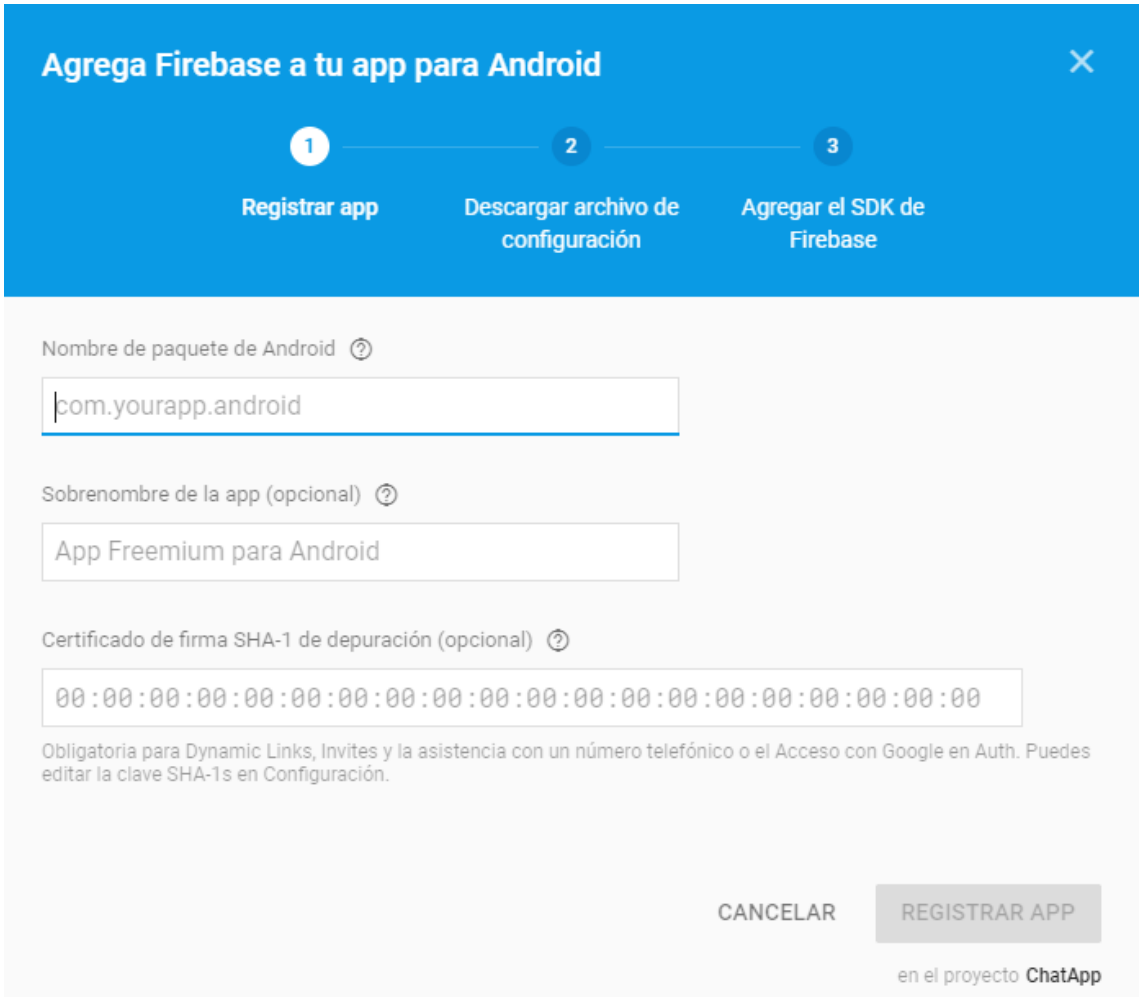

Figura 13. Agregar Firebase a Android, Paso 1.

- Paso 2, descargar el archivo indicado e introducirlo en el directorio que nos indican:

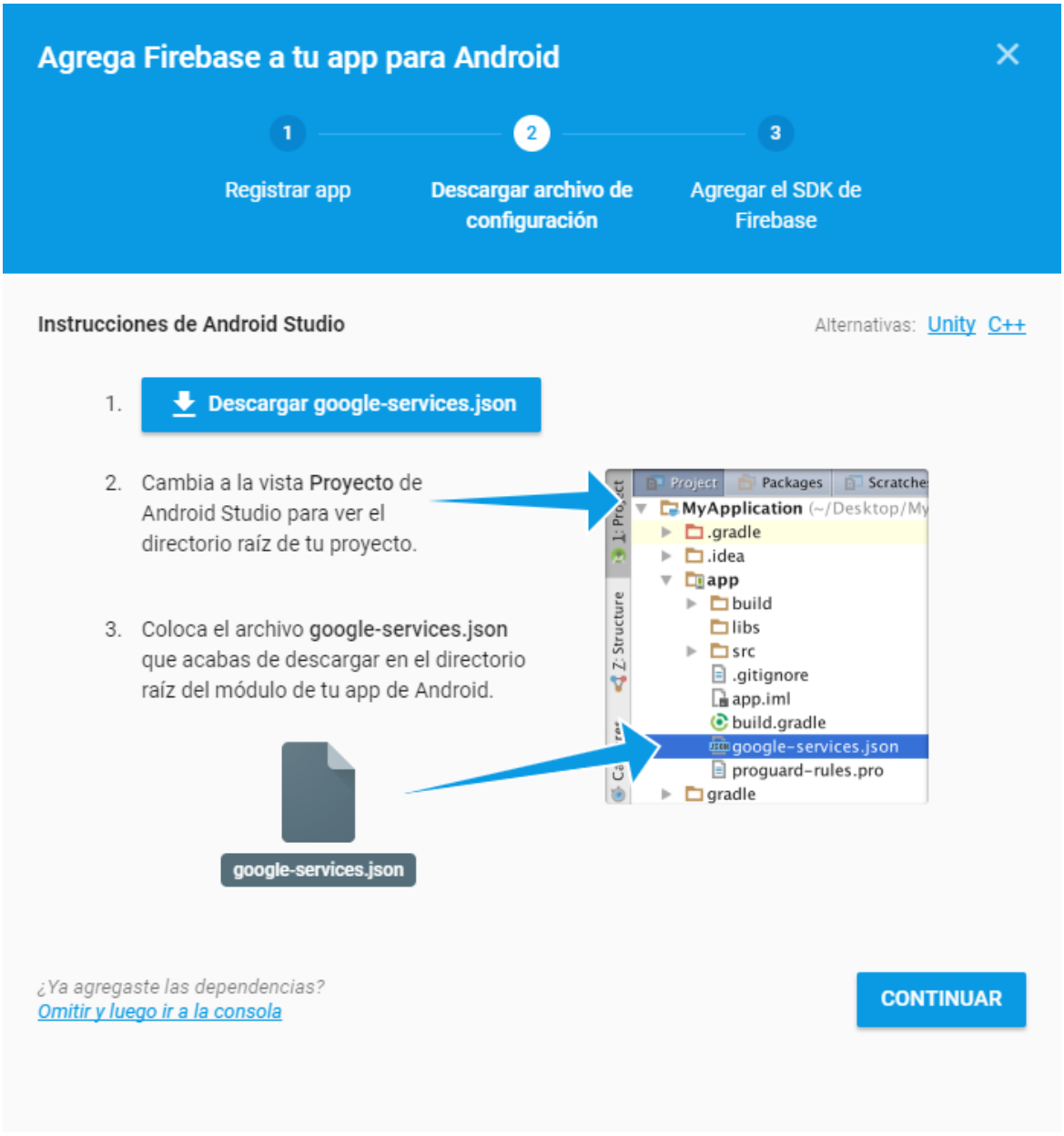

Figura 14. Agregar Firebase a Android, Paso 2.

- Paso 3, añadir el código indicado en los siguientes archivos del proyecto de Android Studio:

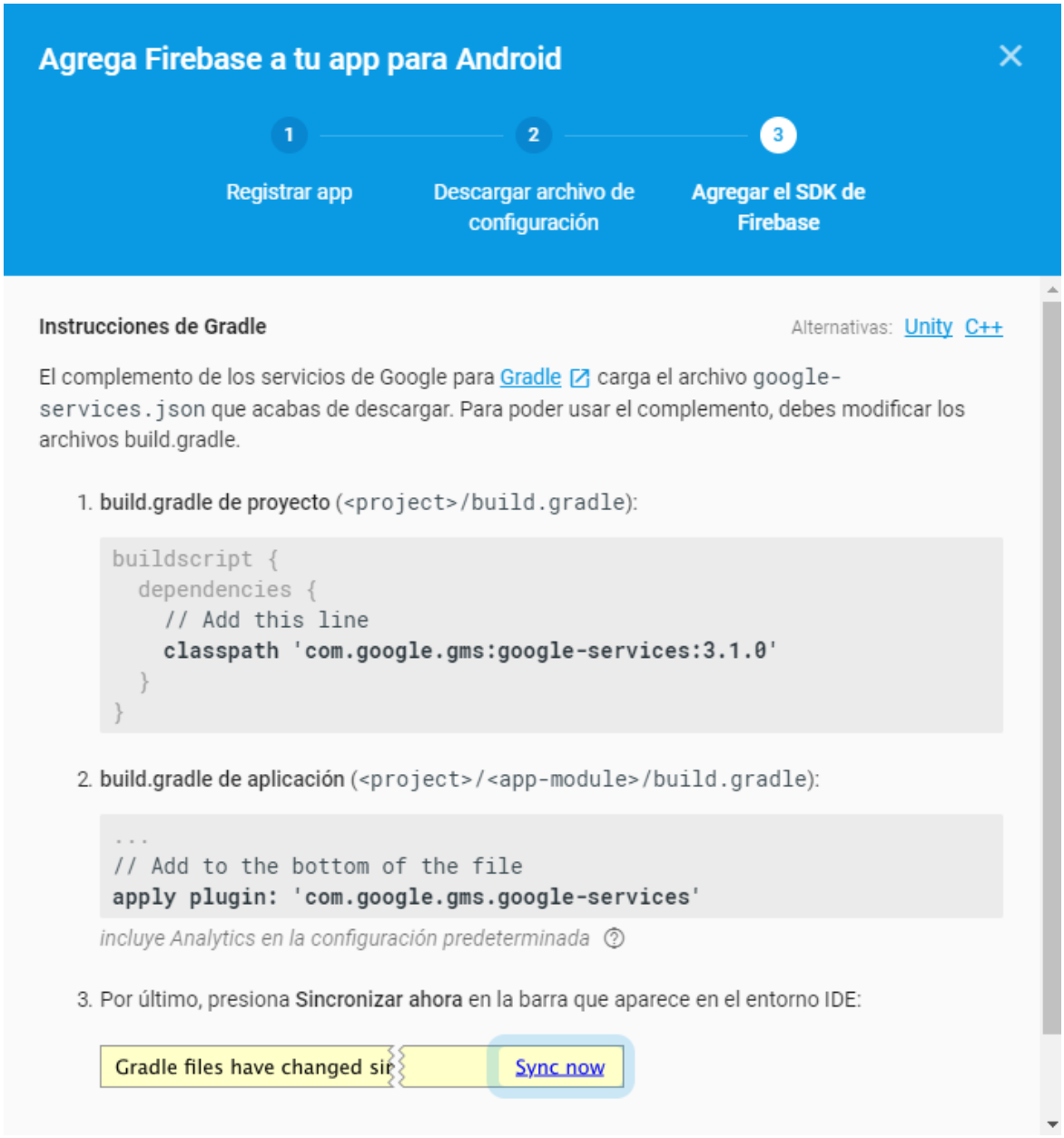

Figura 15. Agregar Firebase a Android, Paso 3.

Finalizados los pasos anteriores nuestra aplicación quedaría conectada con nuestro proyecto en Firebase. Ahora sólo queda agregar los servicios de Firebase que vallamos a utilizar, para ello debemos desde Android Studio:

- Entrar en el asistente de Firebase:

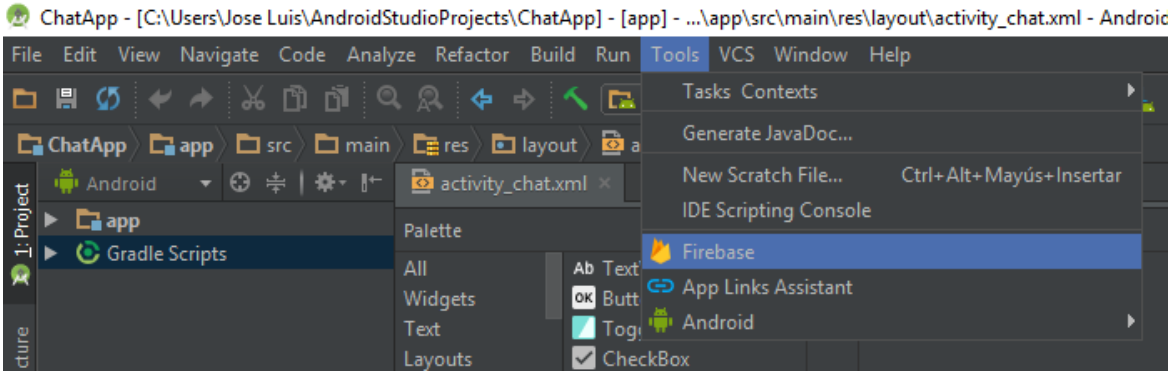

Figura 16. Acceso al asistente de Firebase.

- En la parte derecha se abrirá un apartado donde encontraremos cada uno de los servicios de Firebase:

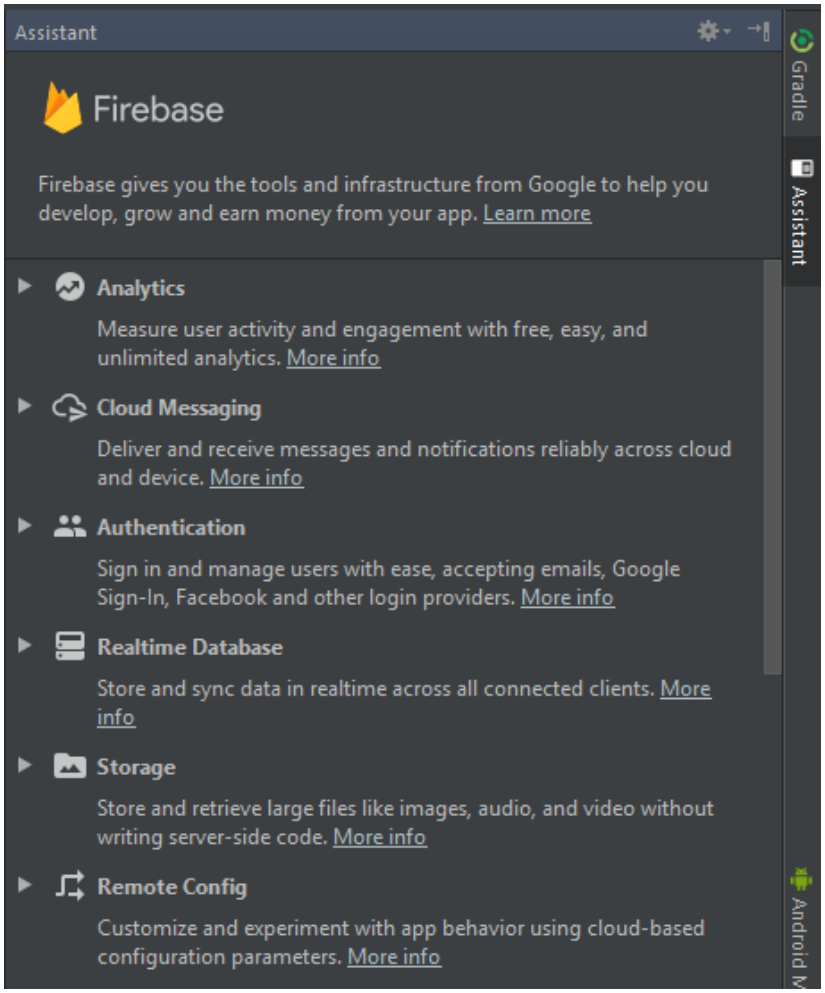

Figura 17. Asistente de Firebase.

- Si pulsamos sobre cada uno de estos servicios, se abrirá un apartado donde podremos añadir el código necesario para su utilización de manera automática, así como un medio alternativo para conectar la aplicación a Firebase, también encontraremos tutoriales para la utilización de los servicios:

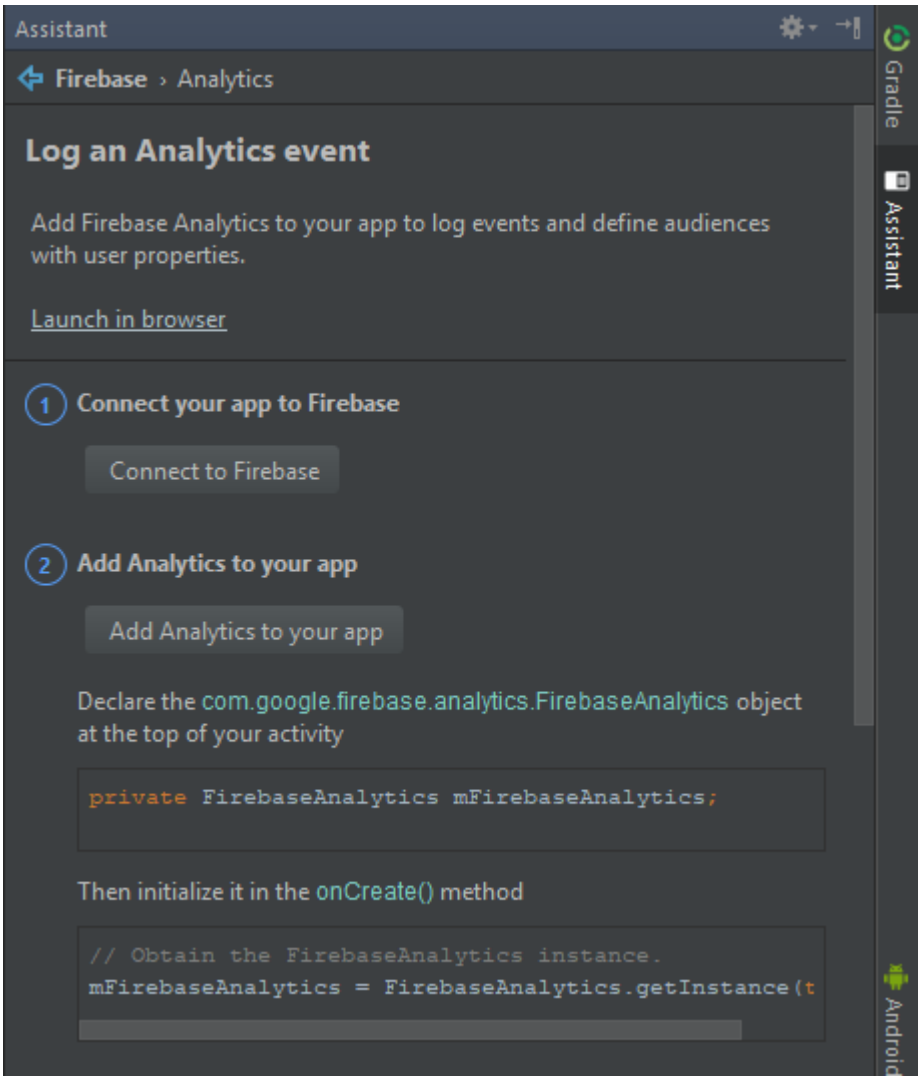

Figura 18. Asistente de Firebase, ejemplo.

Para la aplicación del proyecto se han añadido al proyecto los servicios de Realtime Database, Cloud Storage y Authentication.

5.2.2. Pantalla de bienvenida

En esta pantalla encontramos la portada con un mensaje de bienvenida para el usuario, donde al pulsar sobre la pantalla nos lleva al inicio de sesión. Dicha portada ha sido realizada mediante el software de edición de imagen gratuito GIMP, y se mostrará en el caso de que no haya ninguna sesión de usuario activa, en tal caso se iniciará la pantalla principal.

Figura 19. Pantalla de bienvenida.

Obtenemos al usuario activo con “FirebaseAuth.getInstance().getCurrentUser()”, esto será nulo en caso de no haber una sesión activa.

5.2.3. Pantalla de inicio de sesión

Aquí se encuentran los campos de texto para que el usuario introduzca su email y su contraseña, así como el botón para que inicie sesión una vez los haya introducido. También hay un botón que lleva a la pantalla de registro, en caso de que el usuario no esté registrado. Cuando el usuario inicia sesión, en caso de ser correctos los datos introducidos, se redirige a la pantalla principal.

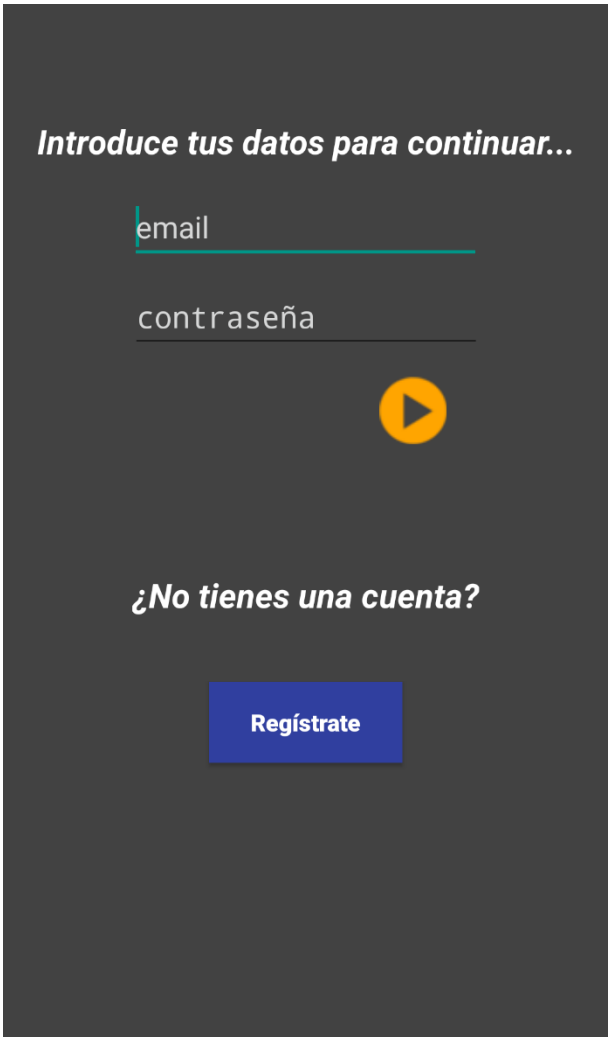

Figura 20. Pantalla de inicio de sesión.

Para el inicio de sesión se utiliza la función de Firebase "signInWithEmailAndPassword", mediante la cual podemos iniciar sesión pasándole como parámetros el email y la contraseña.

Mediante la función "onAuthStateChanged" comprobamos cuándo cambia el estado de autenticación, que cambiará cuando el usuario envíe sus datos para iniciar sesión, si los datos son erróneos, "user" será nulo. En el caso correcto obtenemos el UID del usuario e iniciamos la pantalla principal mediante un "intent", pasándole el UID del usuario.

5.2.4. Pantalla de registro

En esta pantalla encontramos los campos de texto para que el usuario introduzca su nombre, su email y su contraseña. Más abajo se encuentra el botón para enviar los datos que, en caso de creación correcta de la cuenta, se creará al usuario en la base de datos y se le asignará un usuario de Asterisk.

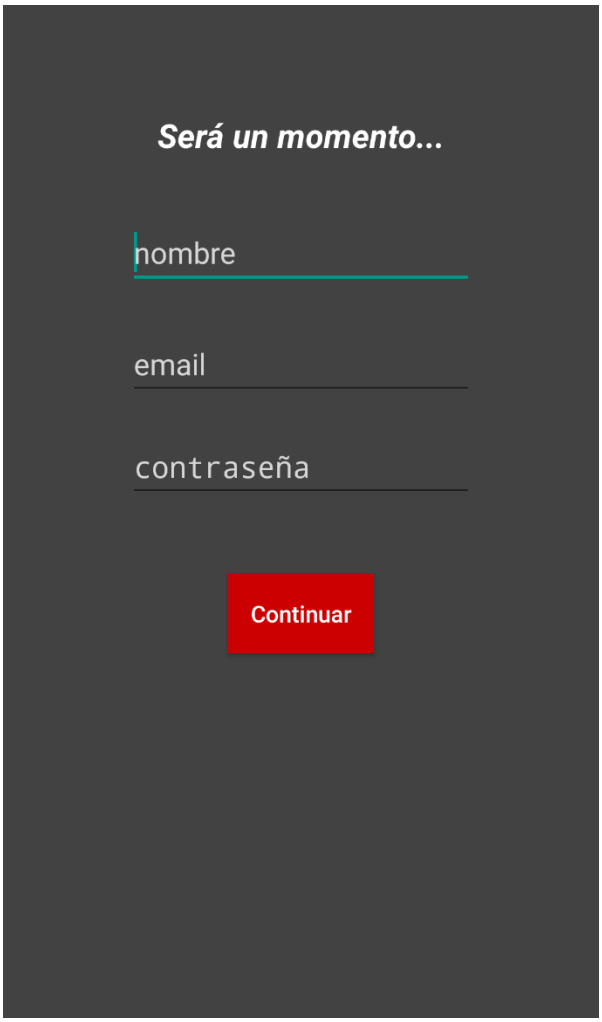

Figura 21. Pantalla de registro.

La creación de la cuenta de usuario se realiza mediante la función de Firebase “createUserWithEmailAndPassword”. Después se crea al usuario en la base de datos junto con sus campos de datos, chats y contactos, para añadir datos a la base de datos debemos obtener la instancia de la base de datos, y, con esta, la referencia a la raíz de la base de datos, acto seguido se añaden los datos de usuario en esa referencia mediante la función “.updateChildren(nuevoUsuario)”. Este nuevo usuario debe ir dentro de un HashMap. Si quisiéramos añadir datos en otra zona inferior de la base de datos, lo haríamos mediante la adición de “.child(hijoInferior)” a la referencia que tengamos, de esta manera podemos bajar niveles en la base de datos. De la misma manera se le asigna un usuario Asterisk libre a sus datos y este se pone como “ocupado” en la lista de usuarios Asterisk.

También se añade al almacenamiento cloud una imagen de perfil para el usuario por defecto, la cual podrá cambiar más adelante. Para ello el procedimiento es parecido al de añadir datos a la base de datos, teniendo que obtener la referencia del almacenamiento cloud y en esa referencia mediante “.putFile(rutaArchivo)” añadimos la imagen al almacenamiento.

5.2.5. Pantalla principal

Aquí nos encontramos, en la parte superior, cuatro botones:

- **Botones centrales:** dos botones que nos permiten cambiar entre la lista de chats y la lista de contactos, que están constituidas por dos fragmentos que se alternan en un contenedor en el activity. Al pulsar un elemento de la lista, nos lleva a la pantalla de chat de ese contacto.

- **Botón derecho:** dicho botón nos lleva a la ventana de buscar y agregar usuarios.

- **Botón izquierdo:** al pulsarlo se despliegan otros dos botones, los cuales nos permiten, uno cerrar la sesión y el otro ir a la pantalla de edición del perfil.

- A continuación se muestran imágenes de muestra de la pantalla:

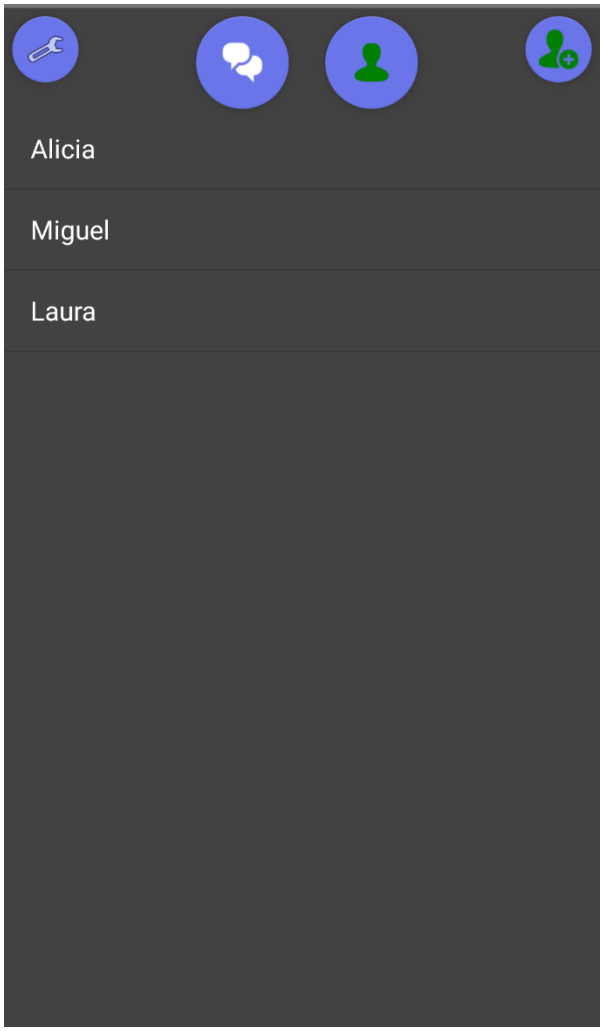

Figura 22. Lista de chats.

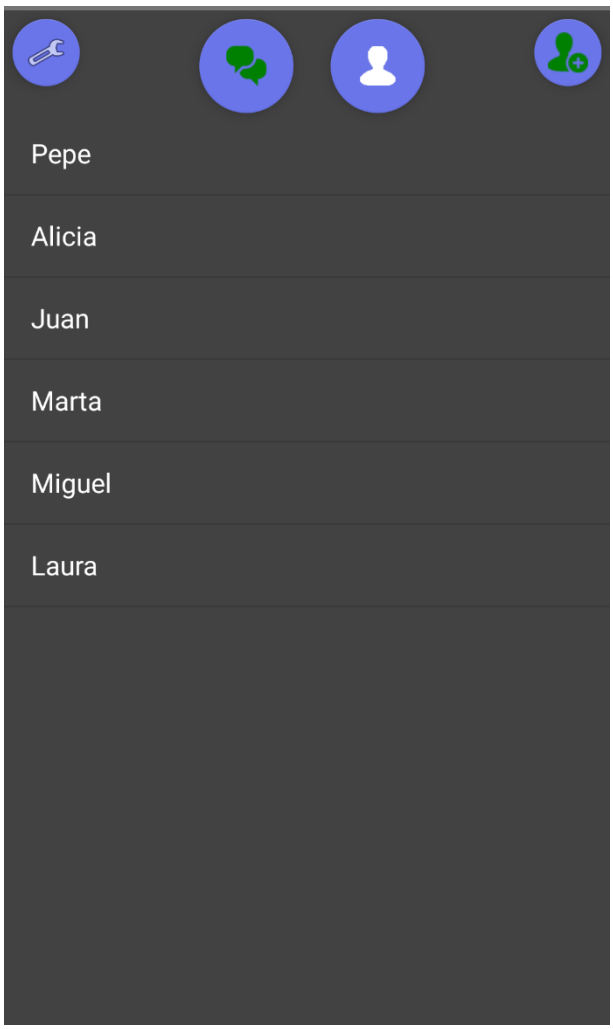

Figura 23. Lista de contactos.

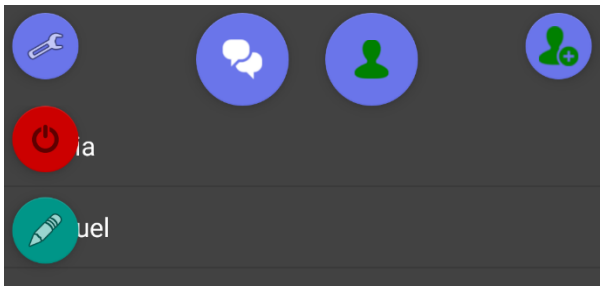

Figura24. Menú desplegable.

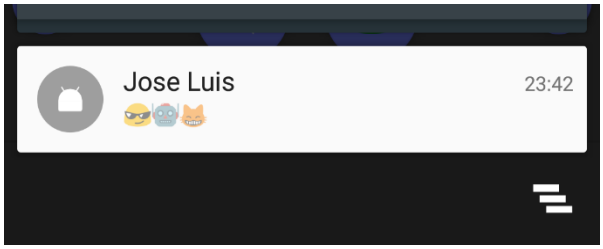

Figura 25. Notificación con mensaje.

En esta pantalla también se obtiene en background de la base de datos el usuario Asterisk correspondiente al usuario de la aplicación y se hace el registro en la centralita Asterisk con el usuario obtenido. Con respecto a la parte de VoIP, también se registra un "callReceiver" con la clase "LlamadaEntrante", la cual hereda de "BroadcastReceiver", permitiéndonos así recibir eventos de llamadas entrantes, las cuales son derivadas a la pantalla de teléfono.

Por último se inicia el servicio de notificaciones, contenido en "Notificaciones.java". Este servicio utiliza la función ".onDataChange()" la cual nos permite tener una sincronización en tiempo real con la base de datos, pues se ejecutará cada vez que haya un cambio en la parte de la base de datos donde asignemos la referencia. En esta función se buscarán en los chats mensajes que no hayan sido entregados (entregado = false), y genera una notificación con el texto del mensaje y el nombre de quien lo envía. Dicha notificación nos llevará al chat pertinente al pulsarla.

5.2.6. Pantalla de edición de perfil

En esta vista encontramos una visualización de la foto de perfil del usuario y de su nombre, estos datos pueden ser cambiados:

- **Foto de perfil:** Al pulsar sobre ella, nos aparecerán nuestros archivos para que seleccionemos una imagen nueva, al hacerlo se actualizará el nuevo archivo en la base de datos.

- **Nombre:** a su lado encontramos un botón mediante el cual se habilita un campo para escribir un nuevo nombre de usuario y guardarlo en la base de datos.

  A continuación se muestra el perfil de un usuario de ejemplo:

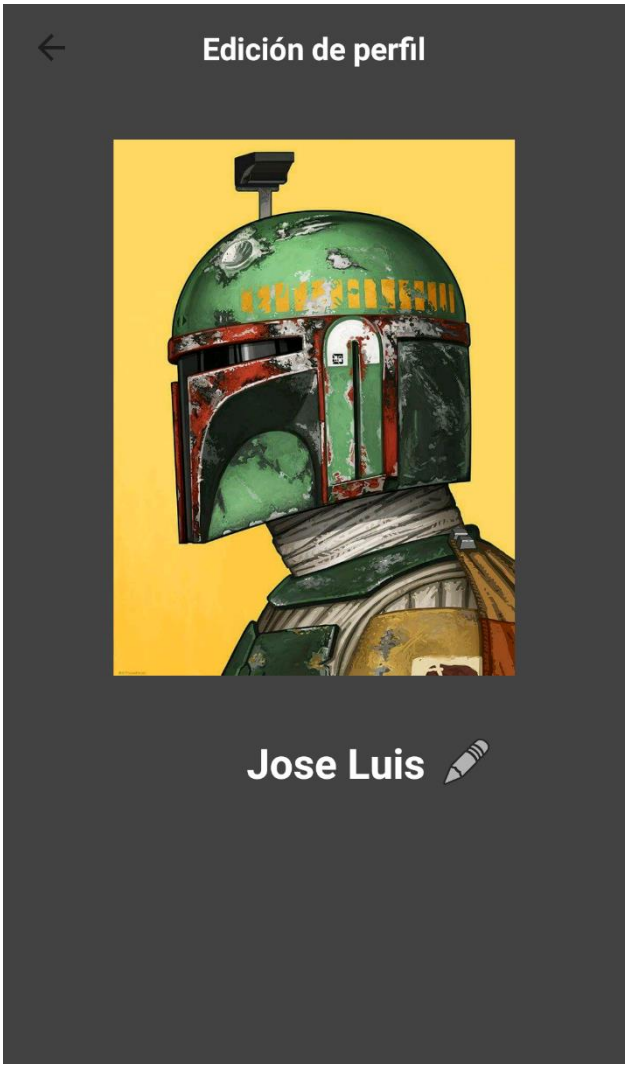

Figura 26. Pantalla de edición de perfil.

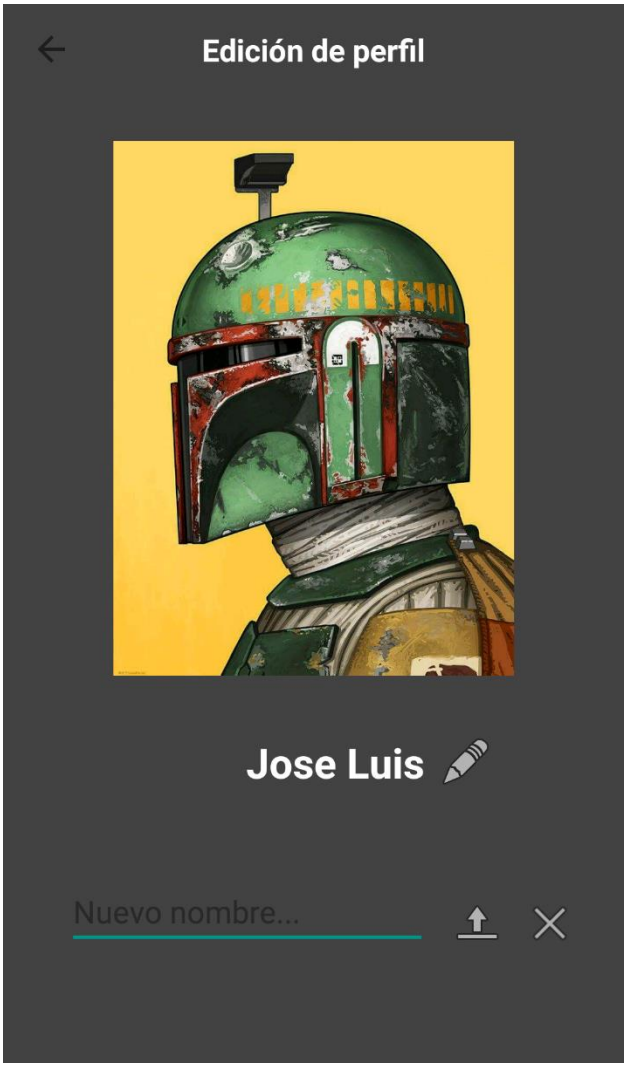

Figura 27. Pantalla de edición de perfil. (Edición de nombre activa)

5.2.7. Pantalla de búsqueda

Esta pantalla consta de un campo de edición de texto y una lista, la cual se va actualizando en tiempo real con nombres de usuarios que contienen el texto escrito en el campo de texto. Para hacer esto posible se ha añadido al campo de texto un "TextChangedListener", que nos permite controlar el evento de que el texto del campo de edición cambie. Dentro del listener ejecutamos en tiempo real la consulta a la base de datos, concretamente en la función "afterTextChanged()", la cual se ejecuta después de que cambie el texto.

Cuando pulsamos sobre uno de los nombres, se abre una ventana de alerta en la que se da la opción de agregar como contacto a ese usuario. Si agregamos a un usuario,

automáticamente este queda registrado en nuestra lista de contactos, y nosotros quedaremos registrados en la suya.

A continuación se muestran imágenes de muestra de la pantalla:

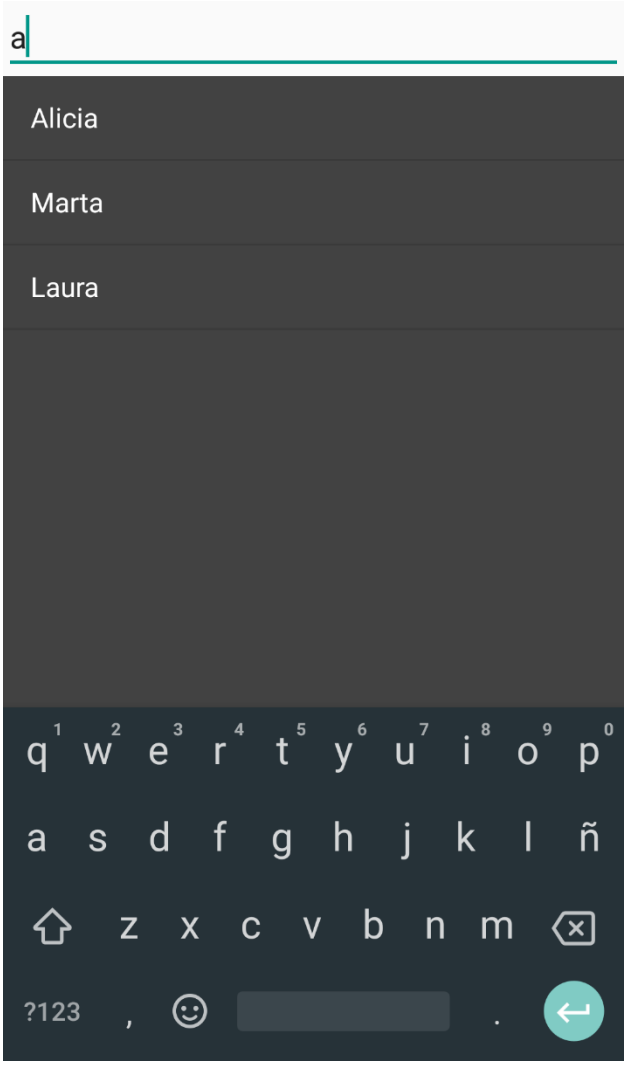

Figura 28. Pantalla de búsqueda.

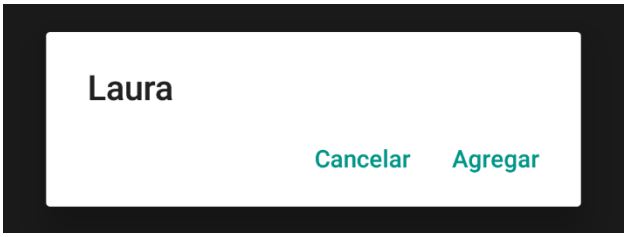

Figura 29. Ventana de alerta pantalla de búsqueda.

5.2.8. Pantalla de chat

En la parte superior encontramos un campo con:

- La foto de perfil del usuario, que al pulsarla se ve en vista ampliada haciendo uso de la pantalla de zoom, la cual consta de una vista de imagen a pantalla completa.

- El botón de llamar, el cual nos permitirá llamar al contacto. Esto se realizará a través de la pantalla de teléfono.

- En la parte derecha encontramos dos botones, uno con una papelera y otro con una equis, sirven para eliminar el chat y eliminar al contacto respectivamente. Esta eliminación se lleva a cabo mediante la función ".updateChildren()" comentada anteriormente, pero en este caso se le pasará como parámetro "null", eliminando así ese eslabón de la base de datos y todos los inferiores que dependan de ese.

En la parte inferior encontramos el campo para escribir los mensajes y el botón para enviarlos, el cual hace una actualización en la base de datos. Los mensajes se guardan en la base de datos como "entregado=false" hasta que el otro usuario los reciba en su dispositivo.

En la parte central encontramos el campo donde se irán reflejando los mensajes en tiempo real gracias a la función "onDataChange()". Estos mensajes son textViews que se añaden a la pantalla central, están dentro de burbujas de chat creadas poniendo a estos campos imágenes ninepatch, estas imágenes tienen unas marcas en los bordes que indican por donde se va a deformar la imagen en caso de ampliación, pudiendo con esto generar imágenes adaptables al tamaño del texto.

A demás cada mensaje lleva su hora de emisión, y los grupos de mensajes de un chat están separados por la fecha de estos.

A continuación se muestra un chat de ejemplo:

Figura 30. Pantalla de chat.

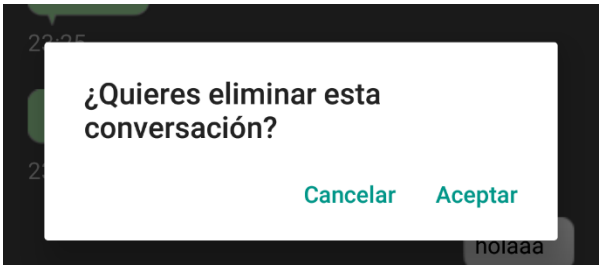

Figura 31. Ventana eliminar chat.

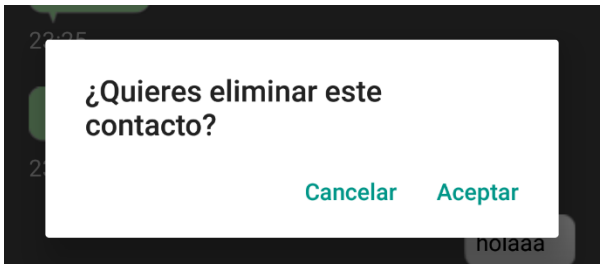

Figura 32. Ventana eliminar contacto.

La pantalla de llamada se mostrará en el siguiente apartado.

5.2.9. Pantalla de teléfono

Esta pantalla es la encargada de las llamadas entrantes y salientes, en ella encontramos un campo de texto donde aparece el nombre del usuario con el que tenemos dicha llamada, otro donde aparece el estado de la llamada y una visualización de la foto de perfil del contacto. En la parte inferior encontramos tres botones, dos de ellos representan los botones de responder y colgar llamada cuando nos están llamando, y el restante sirve para colgar cuando la llamada está en curso.

Siempre que colguemos seremos redirigidos a la pantalla principal, pero cuando:

- **Recibimos una llamada:** en este caso se obtiene de la llamada el usuarioAsterisk del llamante, con este dato se busca en la base de datos a quién corresponde dicho usuario y a partir de ahí se carga su nombre y foto de perfil. En ese momento están activos los botones de responder y colgar, cuando pulsemos responder se ocultarán dejando paso al botón segundario de colgar. La llamada se responde mediante las funciones SIP ".answerCall() y .startAudio()". Para colgar se usa la función ".endCall()".

- **Realizamos una llamada:** en este caso se inicia la pantalla de teléfono pasándole el UID del contacto al que llamamos, así diferencia que esto no es una llamada entrante. Se busca el usuario de Asterisk de ese contacto y se hace una llamada SIP mediante la función "makeAudioCall()".

A continuación se muestran unas llamadas de ejemplo, una llamada entrante del usuario "Jose Luis" y una llamada realizada a la usuaria "Laura" desde el chat:

Figura 33. Llamada entrante.

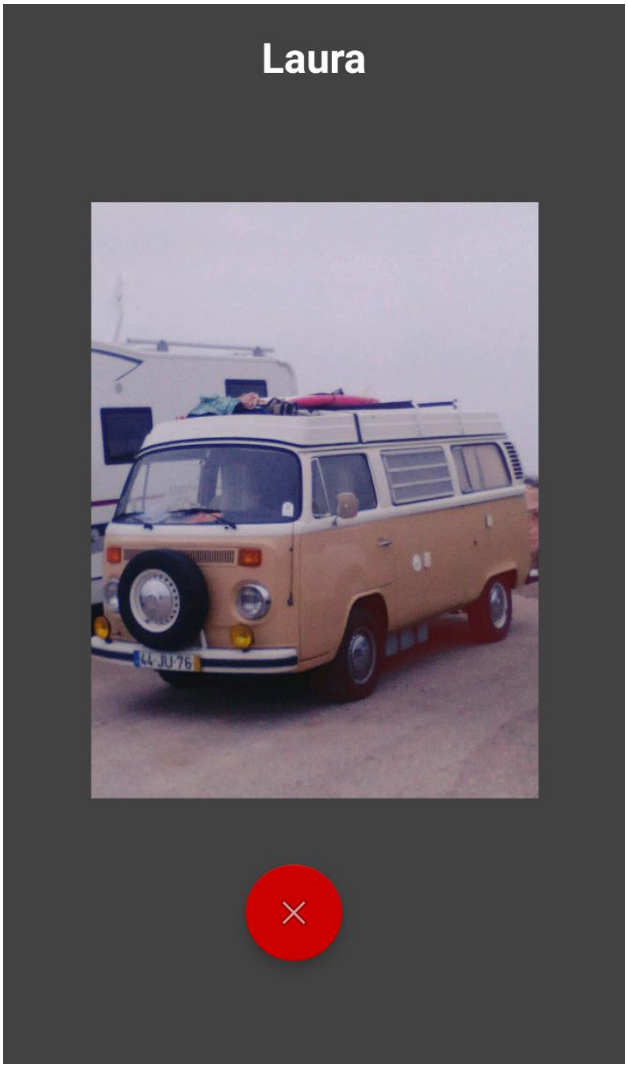

Figura 34. Llamada saliente.

5.2.10. Manifiesto

Para el correcto funcionamiento de la aplicación se necesitan lo siguientes permisos y usos en AndroidManifest.xml:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.apptfg.chatapp">

    <receiver android:name=".IncomingCallReceiver" android:label="Call Receiver" />

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.USE_SIP" />
    <uses-permission android:name="android.permission.RECORD_AUDIO" />
    <uses-permission android:name="android.permission.CAPTURE_AUDIO_OUTPUT" />
    <uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />

    <uses-feature android:name="android.hardware.sip.voip" android:required="true" />
    <uses-feature android:name="android.hardware.wifi" android:required="true" />
    <uses-feature android:name="android.hardware.microphone" android:required="true" />
```

Figura 35. Android Manifest.

6. Resultados

La aplicación ha sido probada en dos dispositivos reales, ya que la carencia de la potencia suficiente en uno de los ordenadores disponibles, y la incompatibilidad del emulador Android en otro de los ordenadores, han hecho imposible probar la aplicación en el emulador de Android Studio.

Los dispositivos utilizados han sido dos “bq Aquaris M5” con las siguientes características:

- Pantalla de 5 pulgadas.
- Procesador de ocho núcleos 1,5 GHz.
- Ram: 2 GB.
- Memoria interna: 16 GB.
- Sistema operativo: Android 6.0.1.

Los resultados han sido satisfactorios, pudiendo realizar todas las funciones como crear usuarios, iniciar sesión, cambiar el nombre y la foto de perfil, enviar mensajes, eliminar chats y, agregar y eliminar contactos contactos.

También se han podido registrar correctamente en la centralita Asterisk, la cual ha estado funcionando en una máquina virtual de Ubuntu 16 en un PC con Windows 10. Se han realizado llamadas y se podía hablar bien entre dos personas, aunque cabe destacar que al estar los dispositivos cercanos y recibir los dos el mismo sonido se creaba algo de ruido en bucle, esto quedo apaciguado al alejar los dos dispositivos.

Las comunicaciones con la plataforma Firebase están cifradas con TLS, pero las comunicaciones con la centralita Asterisk van sin cifrado, ya que no ha sido posible la conexión con la centralita a través de Firebase al usar la versión gratuita del servicio, en esta versión están limitados el número de mensajes, algo no viable para un streaming multimedia.

7. Conclusión

En este proyecto se ha creado una aplicación para dispositivos Android que hace posible las comunicaciones entre usuarios tanto por voz como mediante mensajes de texto. Se ha buscado que la aplicación se sencilla de utilizar para el usuario, que no caiga en el exceso de funciones innecesarias que acumulan muchos de los servicios similares que encontramos en la actualidad, y que a la vez le permita cierta personalización por parte del usuario para su perfil.

La parte de servidor se ha realizado con Firebase y Asterisk, ambas herramientas gratuitas. Cabe destacar que en el caso de que esta aplicación fuera a tener una cuantía considerable de usuarios, sería necesario contratar alguno de los planes que dispone Firebase, pero para el proyecto ha sido suficiente con la versión gratuita.

La principal dificultad del proyecto ha sido el estudio y comprensión de la utilización de Firebase y del protocolo SIP en el desarrollo Android. Dicha dificultad ha sido mermada gracias a la gran cantidad de documentación que podemos encontrar tanto en la web de Google Developers como en la de Firebase. Aquí queda reflejado el gran trabajo que hace Google por desarrollar software de manera abierta y buscando la divulgación de conocimiento.

En conjunto obtenemos como resultado un servicio de gestión de comunicaciones entre usuarios. Se obtiene, en la parte del usuario, una aplicación Android de fácil manejo y, en la parte de servidor, un sistema donde el administrador del servicio dispondrá por un lado de la configuración en línea de comandos de la centralita Asterisk, y por otro de Firebase, donde podrá de una manera sencilla supervisar y configurar casi todos los aspectos del servicio. Cabe destacar que en la plataforma Firebase encontramos una pestaña de analíticas donde podemos obtener un feedback de la aplicación haciendo por ejemplo recuentos del número de usuarios activos por mes o el país de estos.

8. Anexo 1, Manual de la aplicación

La aplicación se encuentra en formato “apk”, para instalar dicha aplicación debemos activar la casilla “orígenes desconocidos” en los ajustes de seguridad del dispositivo Android. Dicho dispositivo deberá contar con conexión a Internet para la utilización del servicio.

Utilización de las distintas pantallas:

- **Pantalla de bienvenida:** tan sólo debemos tocar la pantalla para continuar.
- **Pantalla de inicio de sesión:**

Figura 36. Manual, inicio de sesión.

- **Pantalla de registro:** *(para eliminar tu cuenta contacta con soporte@chatapp.com)*

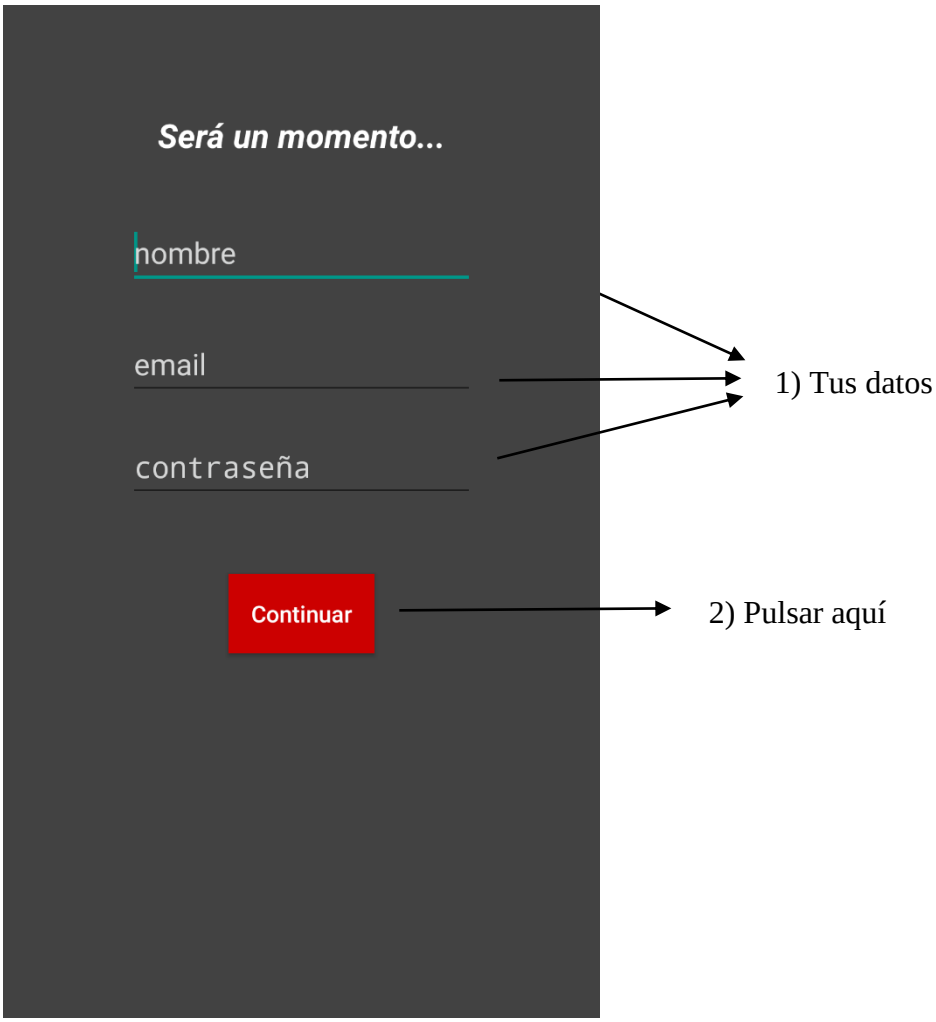

Figura 37. Manual, registro.

- **Pantalla principal:**

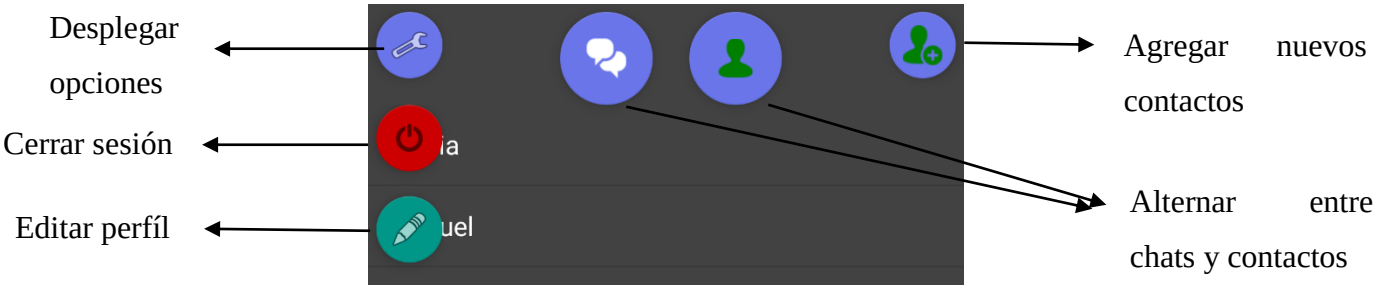

Figura 38. Manual, pantalla principal.

Pulsar sobre un contacto o chat para abrir la ventana de chat.

- **Pantalla de edición de perfil:**

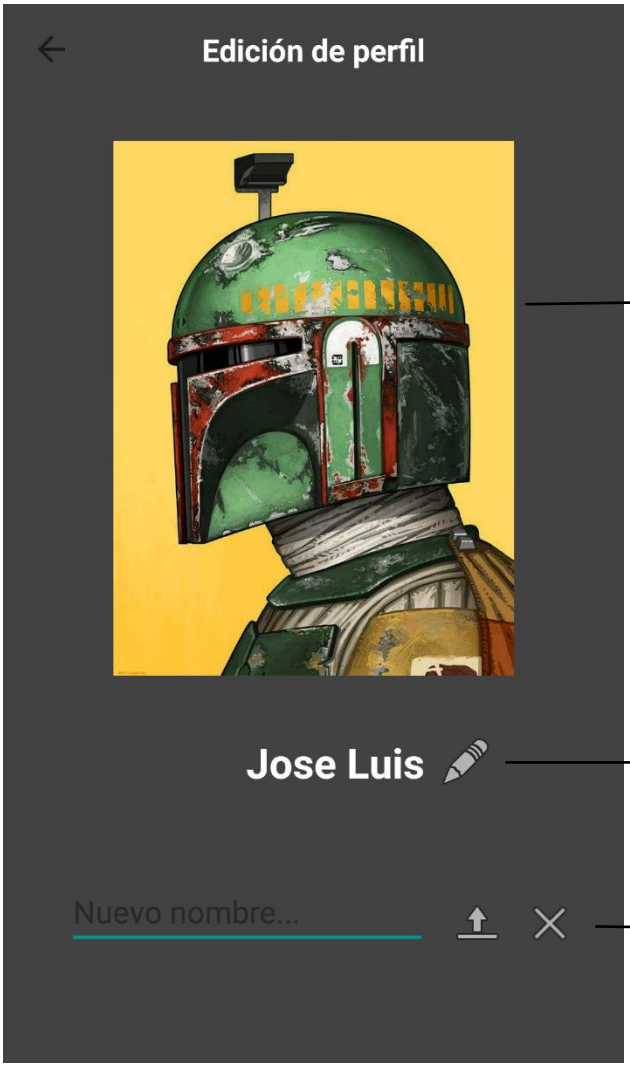

Figura 39. Manual, edición de perfil.

- **Pantalla de búsqueda:**

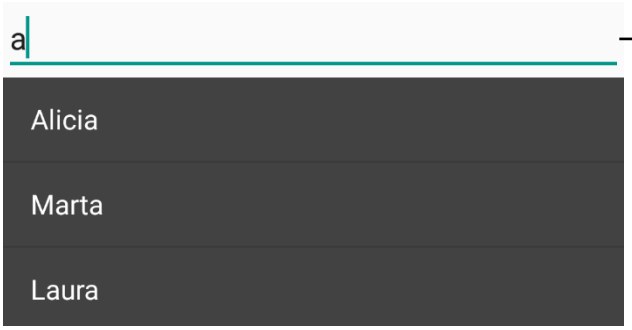

Figura 40. Manual, agregar contactos.

- **Pantalla de chat:**

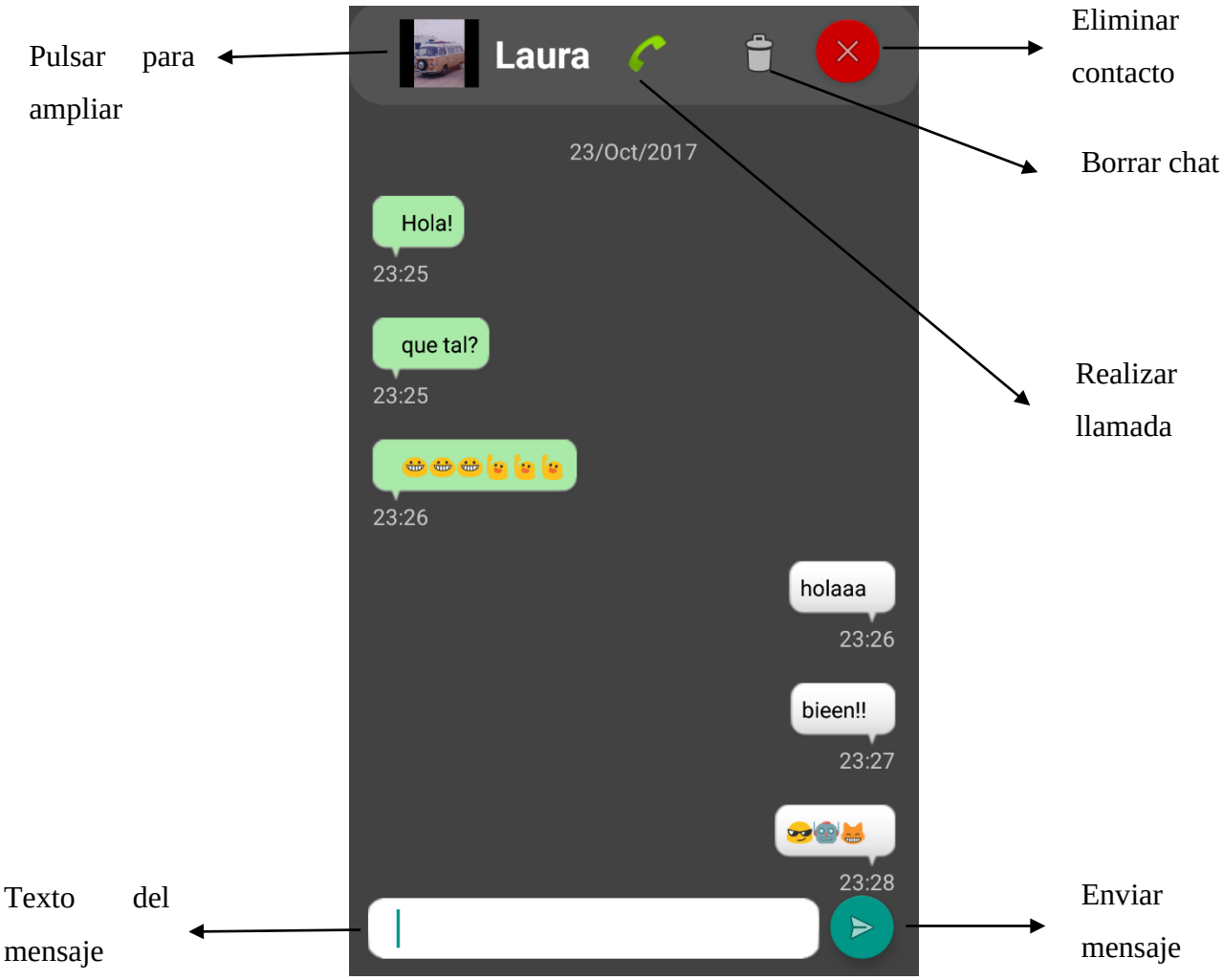

Figura 41. Manual, pantalla de chat.

- **Pantalla de teléfono:**

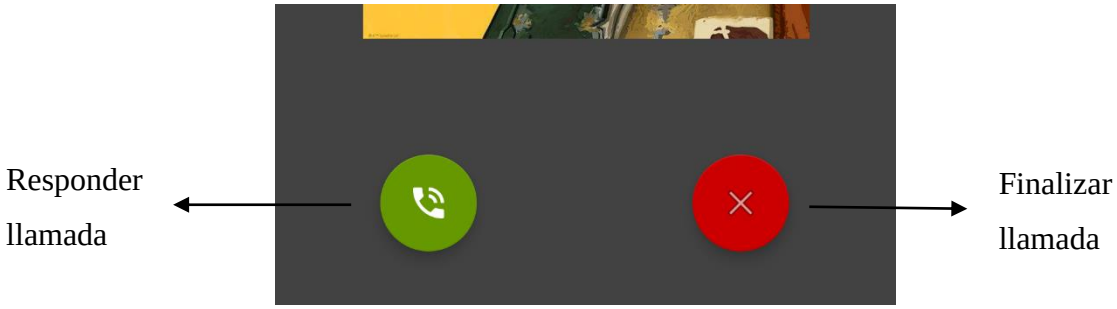

Figura 42. Manual, llamada.

9. Anexo 2, Planificación y presupuesto

A continuación se hará un recuento de las horas empleadas en el proyecto y un resumen de los costes del mismo.

- Desglose de horas empleadas:

Tabla 1. Desglose de horas.

Actividad	Horas
Estudio previo y documentación	50
Configuración de backend	30
Desarrollo de la aplicación	120
Pruebas de campo	20
Elaboración de la memoria	80
TOTAL	**300**

- Costes de materiales:

Tabla 2. Costes de materiales.

Producto	Unidades	Coste unitario	Total (€)
Ordenador	2	316	632
Dispositivo Android	2	142,2	284,4
Costes de software	4	0	0
TOTAL			**916,4**

- Recursos Humanos:

Tabla 3. Recursos Humanos.

Ocupación	Horas	Precio/Hora (€)	Importe (€)
Ingeniero	300	48	14400
TOTAL			14400

- Costes totales:

Tabla 4. Costes Totales.

Concepto	Precio (€)
Costes de materiales	916,4
Recursos Humanos	14400
Costes indirectos (20%)	3063,28
Subtotal	18379,68
I.V.A. (21%)	3859,73
TOTAL	22239,41

El coste total del proyecto es de *veintidós mil doscientos treinta y nueve euros con cuarenta y un céntimos*.

Bibliografía

[1] www.xatakamovil.com

[2] www.tel-2011-2012.wikispaces.com

[3] www.elastixtech.com

[4] www.wikipedia.org

[5] www.androidsis.com

[6] https://developer.android.com/

[7] https://firebase.google.com/