Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo de Fin de Grado

Google Actions en entornos eHealth

Google Actions in eHealth enviroments

Mireia Marta Scholz Díaz

La Laguna, 10 de junio de 2019

D. **Pino Teresa Caballero Gil**, con N.I.F. 45.534.310-Z Catedrática de Universidad adscrita al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutora.

D. **Alexandra Rivero García**, con N.I.F. 78.646.309-V investigadora adscrita al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como cotutora.

C E R T I F I C A (N)

Que la presente memoria titulada:

"*Google Actions en entornos eHealth*"

ha sido realizada bajo su dirección por D.ª **Mireia Marta Scholz Díaz**, con N.I.F. 51.150.138-R.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 10 de junio de 2019

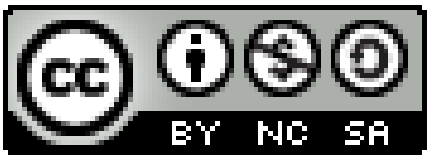
Agradecimientos

A Pino Caballero y a Alexandra Rivero por haberme permitido participar en este proyecto; así como por su tiempo, ayuda y sinceridad.

Al grupo de investigación CryptULL, especialmente a Iván Santos y a José Ángel Concepción por estar siempre dispuestos a resolver cualquier dudas y a compartir su conocimiento.

Por último, a mi familia, pareja y amigos cercanos por el apoyo incondicional durante estos meses de trabajo.

Licencia



Resumen

Los altavoces inteligentes con asistentes de voz integrados cada vez tienen más presencia en nuestros hogares y están abriendo paso a una nueva forma de interactuar con las tecnologías que utilizamos a diario: la voz. A día de hoy, la gran mayoría de aplicaciones disponibles para estos dispositivos pertenecen al sector de la domótica o del entretenimiento.

El objetivo de este proyecto es sacar el máximo provecho de los asistentes de voz, introduciendo esta nueva tecnología en el ámbito del cuidado de pacientes en hospitalización domiciliaria. Para ello se ha desarrollado un sistema que facilite el seguimiento del estado de esos pacientes a las personas encargadas de su cuidado. Este sistema está compuesto por una Action para Google Assistant y una aplicación móvil para dispositivos Android que se comunican entre sí a través de Firebase.

Las tecnologías que han sido utilizadas para el desarrollo de este sistema son Actions on Google, Google Assistant, DialogFlow, Firebase Firestore, Android Studio, GitHub en remoto y Git en local.

Palabras clave: eHealth, hospitalización domiciliaria, Actions on Google

Abstract

Smart speakers with integrated voice assistants are becoming more and more present in our homes and are introducing a new way for us to interact with the technologies we use everyday: our voice. Most of the apps that are currently available for these devices are Home Automation or entertainment apps.

The goal of this project is to unleash the potential of the smart speaker devices introducing this recent technology to the field of home hospitalized patients care. To achieve this, a system that helps the caregivers of these patients to keep track of their health status and real time updates has been developed. The system is made up of an Action for the Google Assistant and a mobile phone application for Android devices that communicate with each other through Firebase.

The technologies that have been used to develop this projets are Actions on Google, Google Assistant, DialogFlow, Firebase Firesotre, Android studio and GitHub among others.

Keywords: eHealth, home hospitalization, Actions on Google

Índice general

Índice de Figuras

Índice de Tablas

Capítulo 1

Introducción

Este trabajo se ha redactado intentando utilizar lenguaje inclusivo. Por ese motivo se han utilizado siempre que se ha podido términos colectivos. Sin embargo, términos como usuarios/as no tienen sinónimo colectivo y por eso a partir de este punto utilizaremos la palabra usuarios para referirnos a todas las personas que usen el sistema.

1.1. Motivación

Pese a que los primeros asistentes virtuales surgieron en la década de los 70 como contestadores telefónicos y como software médico de dictáfonos digitales, No fue hasta 2011 cuando por fin pudimos hacer uso de los asistentes de voz tal y como los conocemos en la actualidad con la llegada de Siri, el asistente de voz de Apple que fue presentado como la mayor novedad de iOS 5 para el iPhone 4S [1]. Los usuarios de dispositivos Android tuvieron que ser más pacientes, pues Google lanzó Google Assistant 5 años más tarde.

Mientras tanto, en 2014 Amazon ya lanzaba su altavoz inteligente con asistente de voz integrado, Echo, y a finales de 2017 salían a la venta los equivalentes de Google con Google Assistant integrado, que hasta entonces solamente podía ser utilizado en smartphones. A principios de 2018 se vendieron 31 millones de dispositivos Amazon Echo y 14 millones de dispositivos Google Home [2].

Además de las aplicaciones que ya vienen instaladas por defecto en estos asistentes, es posible instalar nuevas funcionalidades: 'Alexa Skills' en el caso de Amazon Echo y 'Actions' en el caso de Google Home. En líneas generales, a pesar de contar con una gran cantidad de aplicaciones disponibles, no son muchas las disponibles en la categoría de salud. La temática común se pueden resumir en apps de nutrición y de ejercicio. Cabe destacar, según un estudio publicado por VoiceBot sobre el uso de los altavoces inteligentes en Estados Unidos, las principales tareas que realizan los usuarios a través de estos dispositivos son hacer preguntas, escuchar música y consultar el tiempo [3].

El campo de la eHealth es un terreno prácticamente inexplorado en lo que se refiere a estos nuevos dispositivos. Líneas futuras de investigación en este ámbito podrían ayudar, no sólo a los pacientes a consultar información médica sobre tratamientos o medicamentos sin necesidad de acudir a una consulta. Sino también a poder identificar pacientes con comportamientos depresivos o ansiedad basándose en su voz. Estos son tan solo algunos ejemplos de posibles soluciones que los altavoces inteligentes, con asistentes de voz integrados, podrían aportar a este campo sobre las que ya se está investigando [4].

Debido a todos los motivos anteriormente expuestos, con este proyecto se pretende aprovechar el potencial de estos nuevos dispositivos tan populares en el campo de la eHealth, concretamente en el ámbito del cuidado de pacientes en hospitalización domiciliaria.

1.2. Objetivos

La finalidad de este proyecto es aprovechar el potencial de los altavoces inteligentes en entornos eHealth y adaptar esta nueva tecnología al ámbito del cuidado de pacientes en hospitalización domiciliaria. Para ello se ha desarrollado un sistema que permite realizar un seguimiento del estado de los pacientes por las personas encargadas de su cuidado. En este punto se debe aclarar que haremos referencia, a partir de este momento, a dos roles básicos: paciente y cuidador. El paciente será la persona que tiene sus datos médicos en el sistema y necesitará la ayuda de algún profesional. Este rol hace referencia a una persona bajo hospitalización domiciliaria a la que hay que monitorizar y/o atender. Por otro lado, el rol del cuidador está enfocado a ser cubierto por cualquier profesional sanitario (o no) que pueda llevar a cabo un seguimiento del paciente, y puede ser desde un médico dedicado a esto o un asistente de emergencias hasta un familiar del paciente.

Este sistema está compuesto por dos elementos principales: una Action para Google Assistant y una aplicación móvil.

1.2.1. Action para Google Assistant

Uno de los objetivos principales del proyecto es que los pacientes puedan informar a sus cuidadores de su estado de salud de manera sencilla utilizando únicamente su voz. Esto se consigue mediante la creación de una Action, es decir, una aplicación para Google Assistant, que puede utilizarse tanto en smartphones Android y Apple como en los altavoces inteligentes de Google. Este sistema debe de contar con diferentes comandos que permitan a los pacientes notificar a su cuidador las necesidades que tienen en cada momento. Podría ser una alerta de emergencia o si el paciente necesita ayuda para algún problema concreto.

1.2.2. Aplicación móvil

La información facilitada por los pacientes puede ser consultada a través de una aplicación móvil para dispositivos Android. Además, a través de la misma la persona con rol de cuidador puede programar diferentes notificaciones, como la toma de medicamentos para su paciente, y saber si los ha tomado o si se ha olvidado.

1.3. Estructura del documento

El contenido de este documento está estructurado en los siguientes capítulos:

- Capítulo 1: Introducción
  En este capítulo se han desarrollado la motivación del proyecto y los objetivos del mismo.
- Capítulo 2: Estado del arte
  Se explica la situación actual de los altavoces inteligentes en el mercado y sus características, las diferencias entre los asistentes de voz asistentes de voz y se exponen algunos proyectos similares.
- Capítulo 3: Preliminares
  Se definen el sistema, las partes por las que está compuesto y cómo estas se comunican entre sí.
- Capítulo 4: Tecnologías utilizadas
  Se exponen las librerías, herramientas y otras tecnologías utilizadas para desarrollar cada parte del proyecto.
- Capítulo 5: Aplicación móvil
  En este capitulo se tratan las interfaces que componen la aplicación móvil y detalles de su implementación.
- Capítulo 6: Action para Google Assistant
  Se repasan las tareas o ”intents” que se pueden realizar con la Action desarrollada, y se comenta el código que ejecutan al ser invocadas, en el caso de que lo tengan.
- Capítulo 7: Seguridad
  Se explican los mecanismos de seguridad utilizados para proteger el sistema, en concreto, en el proceso de vinculación de cuentas y en la base de datos.
- Capítulo 8: Presupuesto
  Se incluye una tabla con la estimación de costes del proyecto y las horas invertidas en cada actividad.
- Capítulo 9: Conclusiones y líneas futuros
- Capítulo 10: Conclusions and future lines

Capítulo 2

Estado del arte

En la actualidad se puede encontrar una amplia variedad de altavoces inteligentes para el hogar de la mano de diferentes marcas con precios muy variados entre sí. A continuación se exponen los más relevantes, además de otras aplicaciones del sector eHealth para monitorizar el estado de salud de otras personas.

2.1. Altavoces inteligentes

No hay duda de que a día de hoy los líderes de mercado en lo que a altavoces inteligentes se refiere son Amazon y Google, ya que en 2018 contaban con un 52 % y 32 % de la cuota de mercado respectivamente. El 16 % restante pertenece a Apple y a otras compañías (ver Figura 2.1). Hay que puntualizar que se espera que Google se ponga a la cabeza de cara a 2022.

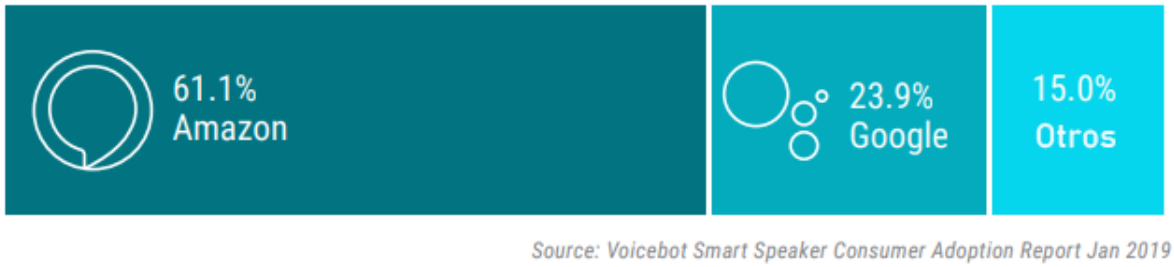


Figura 2.1: Altavoces Inteligentes vendidos en Estados Unidos en Enero de 2019 [5]

2.1.1. Amazon

Echo o Amazon Echo es la gama de altavoces inteligentes propuestos por Amazon cuyo asistente virtual integrado respone al nombre de Alexa. La primera generación de Echo fue limitada para algunos usuarios de Amazon Prime en 2014, y en 2015 se pusieron a la venta para todo el público estadounidense.

Desde entonces, la línea de altavoces inteligentes de Amazon ha aumentado considerablemente. Los principales productos que la componen son los siguientes [6]:

Figura 2.2: Echo dot (3ª generación)

Echo Dot (ver Figura 2.2)

Es la versión más pequeña de Echo, cuya primera generación fue anunciada en 2016. Cuenta con un altavoz de 41mm y cuatro micrófonos de largo alcance. Su precio es de 59,99€.

Figura 2.3: Echo (2ª Generación)

Echo (ver Figura 2.3)

La primera versión de Echo estuvo a la venta desde 2015 hasta que fue retirado en 2017 y Amazon lanzó su sucesor con mejores acabados y un precio más reducido. Viene con un Woofer de 63mm y un tweeter de 16mm, 7 micrófonos y cancelación de ruido, a la venta por 99,99€.

Figura 2.4: Echo Plus (2ª generación)

Echo Plus (ver Figura 2.4)

De todos los altavoces de Amazon, es el que cuenta con mejor calidad de sonido. Permite controlar el resto de aparatos inteligentes sin dispositivos adicionales y su acabado es de tela. Cuenta con un Woofer de 76 mm y tweeter de 20 mm, 7 micrófonos y cancelación de ruido, a la venta por 149,99€.

2.1.2. Google

A principios de 2016 se comenzó a sospechar que Google podría estar planeando lanzar su propio altavoz inteligente para competir con Amazon. En mayo de ese mismo año en la Google I/O, el congreso para desarrolladores celebrado anualmente por la compañía, Google anunció Google Home, y que el asistente virtual que tendría integrado sería Google Assistant, como era de esperar. [7, 8]

Figura 2.5: Google Home

Google Home (ver Figura 2.5)

Salió a la venta en Estados Unidos en Noviembre de 2016, pero no llegó a España hasta 2018. Su base inferior es la malla del altavoz, intercambiable por piezas de metal o de tela de diferentes colores. Cuenta con 3 altavoces de 5cm de diámetro, radiadores pasivos duales de 5cm y 2 micrófonos duales con reconocimiento de voz de campo lejano. Está a la venta por 99,00€.

Figura 2.6: Google Home mini

Google Home mini (ver Figura 2.6)

Salió a la venta en 2017 como principal competidor de Echo Dot debido a su similitud en cuanto a su precio y dimensiones. Al igual que Google Home, también está disponible en diferentes colores, cuenta con 2 micrófonos para el reconocimiento de voz de campo lejano y sonido 360, esta vez con un altavoz de 40mm de diámetro. Está a la venta por 59,00€.

También existe Google Home Max, de mayor tamaño. Es mucho más potente y ofrece una mejor calidad de sonido, aunque aún no ha salido a la venta en España.

2.1.3. Otros

A pesar de que el mercado de los altavoces inteligentes está liderado por Amazon y Google, existen otras alternativas que también están creciendo en popularidad, concretamente el Apple HomePod y SonosOne.

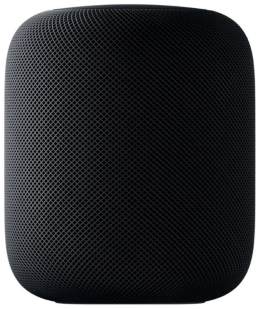


Figura 2.7: Apple HomePod

Apple HomePod (ver Figura 2.7)

Es el altavoz inteligente con mejor sonido del mercado, aunque sólo es posible escuchar música con una suscripción de Apple Music, pues si no no podrá integrarse con Siri, el asistente de voz que tiene integrado. Su app de configuración solamente está disponible para dispositivos iOS. Está a la venta desde Febrero de 2018 por 300,00€ [9]

Figura 2.8: Sonos One

Sonos One (ver Figura 2.8)

Al contrario que Apple HomePad, este altavoz es compatible con más de 60 servicios de audio. El asistente de voz que trae integrado es Alexa, al igual que los dispositivos Echo de Amazon. Su precio de venta es de unos 200,00€ [10]

2.2. Asistentes de voz

A la hora de comparar los diferentes dispositivos, además de tener en cuenta características como su diseño o la calidad de sonido, conocer el asistente de voz que traen integrados es esencial. Los asistentes de voz integrados en los altavoces inteligentes más populares actualmente, tal y como se presentó en el apartado anterior, son Alexa, Google Assistant y Siri.

Cada año Loup Ventures, una empresa americana dedicada a la investigación tecnológica, pone a prueba los diferentes asistentes con un examen de inteligencia que consiste en 800 preguntas de diferentes categorías. Aunque todos son capaces de realizar las mismas tareas (responder preguntas generales, gestionar alarmas, reproducir música, ...), la precisión de las respuestas varía considerablemente.

Según los resultados publicados en el informe de 2018 (ver Tabla 2.1), Google Assistant, el asistente de voz que se utilizará durante el desarrollo de este proyecto, fue capaz de responder correctamente el 88 % de las preguntas que comprenden el test, mientras que Siri y Alexa acertaron un 75 % y un 73 % respectivamente [11].

Cabe mencionar que el asistente de voz cuyo resultado mejoró más en comparación con el año anterior fue Siri, que en 2017 obtuvo un 52 % de precisión.

	Respuestas correctas	Preguntas entendidas
Google Assistant	87,9 %	100 %
Siri	74,6 %	99,6 %
Alexa	72,5 %	99,0 %

Tabla 2.1: Resultados del test de IQ de Loup Ventures (2018)

2.3. Proyectos similares

En el ámbito de la salud y la medicina ya se está popularizando el uso de estas nuevas tecnologías.

Los ECD o dispositivos de accesibilidad para el control del entorno (ver Figura 2.9) permiten a usuarios con discapacidad física y/o funcional controlar su entorno y les aportan cierto grado de independencia. El coste de algunos ECD puede llegar a ascender a los 1200€ [12] y deben ser utilizados a través de una pantalla táctil integrada. Soluciones como Google Home suponen alternativas considerablemente más asequibles –disponible a partir de 60€– que también permiten controlar los diferentes elementos de la casa mediante comandos de voz con micrófonos de largo alcance [13] .

Figura 2.9: Primo+ ECD

A pesar de la comodidad que supone consultar información médica a través de un altavoz inteligente sin la necesidad de desplazarse a una consulta médica, tomar medicamentos o iniciar nuevos tratamientos sin la supervisión de un médico podría suponer graves riesgos para la salud de los pacientes ([14]). En la actualidad, algunos hospitales están introduciendo asistentes de voz para facilitar la estancia de los pacientes. En centros médicos como el hospital infantil de Boston o el Centro médico Beth Israel Deaconess se está experimentando con el uso de estos dispositivos de tal manera que respondan preguntas de los pacientes como “¿Puede venir un/a enfermero/a?”, “¿Cuándo vendrá el doctor?” o “¿Qué hay para almorzar?”. También son utilizados por el personal médico de estos centros para obtener información administrativa [15]. Sin embargo, con esta solución no existe una interacción entre médicos y pacientes.

Por otra parte, también existen múltiples aplicaciones para gestionar toma de medicamentos y similares. Por ejemplo ”Medisafe: Recordatorios y alarmas para medicamentos”, se trata de una aplicación móvil que además de cumplir con lo que su propio nombre indica, entre otras funcionalidades, permite añadir amigos o ”Medfriends” para que puedan consultar si has tomado tu medicación o si por el contrario es posible que hayas olvidado tomarla. Cabe destacar que esta aplicación está disponible tanto para Android como para iOS en sus respectivos catálogos de aplicaciones [17]. A pesar de que esta aplicación se trata de la solución más aproximada al proyecto que se pretende realizar, carece de integración con altavoces inteligentes o comandos de voz.

En conclusión, al tratarse de una tecnología tan reciente, actualmente no existen soluciones como la planteada que hagan uso de estas nuevas herramientas.

Capítulo 3

Preliminares

3.1. Definición

Tal y como se ha especificado en el apartado en el que se definen los objetivos del proyecto, se ha desarrollado un sistema que facilite a los cuidadores de personas en hospitalización domiciliaria la atención y seguimiento del estado de salud de esas personas. Por lo tanto, podríamos definir dos roles o tipos de usuario que podrán interactuar con el sistema:

- Paciente: Persona que está hospitalizada en su casa.
- Cuidador: Es la persona responsable de los cuidados del paciente. No tiene que ser necesariamente personal médico, podría tratarse también de un familiar o persona de confianza.

Este sistema está formado por una aplicación móvil para dispositivos Android, una Action para Google Assistant y un proyecto de Firebase que actúa como servidor, back-end y base de datos.

Para que un cuidador pueda obtener la información del estado de salud de su paciente, ambos deben registrarse en la aplicación móvil y haber completado la información de su perfil. Podrán registrarse con una cuenta de Google o proporcionando e-mail y contraseña.

Se debe de puntualizar que la dirección de e-mail con la que los pacientes se registren deberá ser la misma cuenta que tengan asociada a su altavoz inteligente.

La información que ambos tipos de usuario deben proporcionar es su nombre completo, qué tipo de cuenta van a crear (paciente o cuidador) y su género. Además de esto, los pacientes deberán proporcionar una información adicional, la dirección de su domicilio.

Un cuidador podrá tener un único paciente, pero un paciente podrá tener múltiples cuidadores. Para la creación de esta relación es necesario que los usuarios vinculen sus cuentas mediante un sistema seguro implementado desde la aplicación móvil. Estos vínculos sólo podrán estar formados por parejas cuidador-paciente.

Las tareas que pueden realizar los pacientes a través del sistema desarrollado son:

- Por comandos de voz desde Google Assistant
    - Pedir ayuda a su cuidador
    - Preguntar qué medicamentos tiene pendientes de tomar desde el inicio de ese día

        - Indicar que ya ha tomado los medicamentos pendientes hasta ese momento
        - Preguntar qué medicamentos tiene que tomar de cara al resto del día
        - Registrar sus constantes vitales (frecuencia cardiaca y temperatura)
    - A través de la aplicación móvil
        - Iniciar sesión
        - Cerrar sesión
        - Consultar su perfil de usuario
        - Vincular su cuenta con la de un cuidador
        - Consultar las veces que haya pedido ayuda a su cuidador a través de Google Assistant
        - Consultar qué medicamentos se debe tomar qué días a qué horas
        - Consultar todos los medicamentos que tiene pendientes de tomar
        - Marcar individualmente los medicamentos que se ha tomado
        - Consultar los registros de constantes vitales que haya registrado anteriormente
        - Modificar la dirección de su domicilio en caso de mudanza

Por otra parte, las tareas que pueden realizar los cuidadores a través del sistema desarrollado son:

- A través de la aplicación móvil
    - Iniciar sesión
    - Cerrar sesión
    - Consultar su perfil de usuario
    - Vincular su cuenta con la de un paciente
    - Cuando un paciente pida ayuda, podrán abrir en Google Maps la ruta más rápida desde su ubicación actual hasta el domicilio de su paciente.
    - Añadir nuevos medicamentos que debe tomar el paciente
    - Eliminar medicamentos que el paciente ya no necesita tomar
    - Consultar qué medicamentos se debe tomar su paciente qué días a qué horas
    - Consultar los medicamentos que su paciente tiene pendientes de tomar
    - Consultar los medicamentos que su paciente ya ha tomado
    - Consultar las constantes vitales que ha registrado su paciente a través de Google Assistant

3.2. Conceptualización

En la Figura 3.1 se pueden diferenciar las tres partes que componen el sistema, tal y como se introdujo en el apartado anterior. Se puede identificar: Aplicación móvil para Android, que será utilizada tanto por pacientes como por cuidadores; la Action de Google

Assistant, que permitirá a los usuarios interactuar con el sistema mediante comandos de voz; y un proyecto de Firebase, que actuará como servidor y base de datos en la nube.

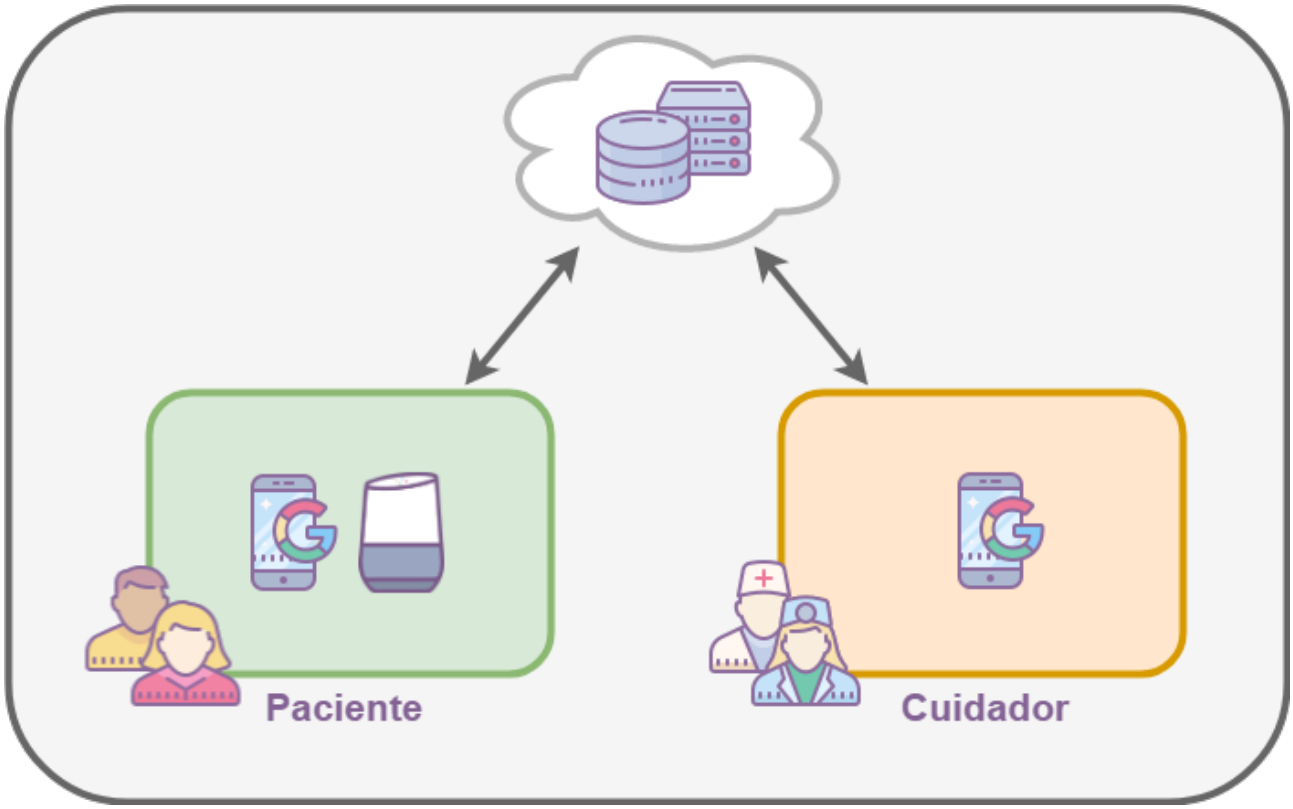


Figura 3.1: Estructura del sistema

La aplicación móvil y la Action se comunican a través de Firebase, de tal forma que cuando un usuario utiliza un comando de voz, se ejecuta el fragmento de código correspondiente de esa tarea. Este código, por norma general, implica leer y escribir en la base de datos, que en este sistema es en la Cloud Firestore de Firebase. Por otro lado, la aplicación móvil también está conectada a la base de datos, de esta forma es posible escribir en la misma y escuchar constantemente toda la información de interés que proviene desde ella. Esto es posible porque el proyecto de Firebase también está conectado a la aplicación móvil como dependencia del proyecto.

Por otro lado, mediante la Action es posible ejecutar código utilizando lenguaje natural. Esto es posible gracias a una serie de tecnologías intermedias que están interconectadas entre sí. Estas tecnologías son Google Assistant, DialogFlow y Firebase (ver Figura 3.2). Cuando el usuario dice el comando adecuado para invocar una acción (”Ok Google, hablar con -Nombre de la Action-") a Google Home ocurren varias cosas:

1. Google Assistant, a través del altavoz inteligente, se encarga de transcribir la voz a texto. La voz, inicialmente como onda analógica, se transforma en una señal digital. Se analiza la señal digital para identificar fonemas que después se combinan para formar palabras dependiendo del contexto [1]. Ahora ya se sabe que el usuario quería lanzar la Action.

2. Google Assitant invoca la Action solicitada por el usuario.

3. DialogFlow lanza el mensaje de bienvenida, que se devuelve a Google Assistant y se reproduce en el altavoz inteligente.

Ahora la Action se está ejecutando y el altavoz inteligente está esperando una tarea que realizar. En el caso de la Action desarrollada, si el usuario pronunciara ”Ok Google, hay una emergencia”, el flujo de información circularía de la siguiente manera:

1. De nuevo, Google Assistant transcribe la voz a texto
2. Esta vez, ya es DialogFlow quién identifica cuál de las tareas registradas es la que el usuario desea ejecutar con Inteligencia Artificial.
3. Esa tarea tiene un código asociado mediante un webhook, y a su vez dicho código está desplegado como una Cloud Function de Firebase. Este código registrará en la base de datos que a esa hora determinada, ese usuario ha solicitado ayuda.
4. Cuando el código se haya ejecutado correctamente, devuelve el control a Firebase, que devuelve una respuesta (”Ya he notificado a tu cuidador”) a Google Assistant y que se reproduce en Google Home.

Las tareas que quiere realizar el usuario se llaman ”intents” y el código que tienen asociado se trata de su ”fulfillment”. Ambos conceptos serán explicados más ampliamente en la sección de DialogFlow del Capítulo 4: Tecnologías utilizadas.

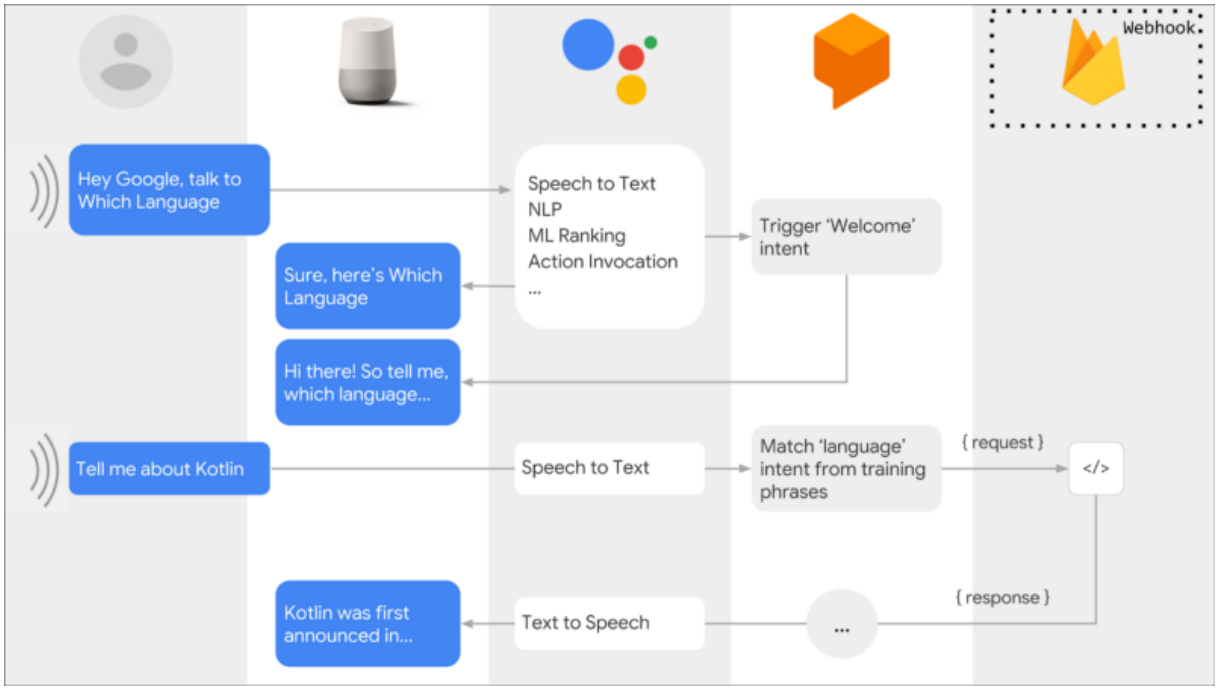


Figura 3.2: Ejemplo de flujo de la información en una Action para Google Assistant [16]

Capítulo 4

Tecnologías utilizadas

En este capítulo se expondrán y explicarán las diferentes tecnologías utilizadas para desarrollar este proyecto. Han sido agrupadas en los siguientes bloques:

1. Aplicación móvil.
2. Action de Google Assistant.
3. Backend y Base de datos.
4. Otros.

4.1. Aplicación móvil

- **Android**

  Es el sistema operativo para móviles desarrollado por Google para dispositivos con pantalla táctil como smartphones y tablets, entre otros. Está basado en el kernel de Linux y, a diferencia de iOS, es de código libre.

  Es el sistema operativo para móviles más popular, de hecho en 2018 Android ya estaba presente en casi el 90 % de los smartphones del mundo (ver Figura 4.1).

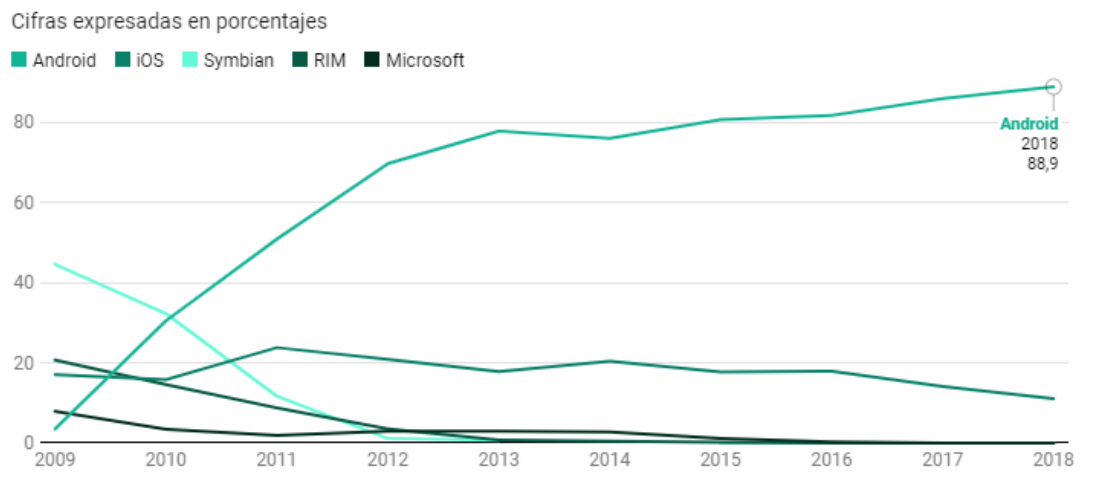


Figura 4.1: Sistemas operativos más presentes en smartphones (2009-2018) [18]

- **Android Studio**

  Es el IDE (entorno de desarrollo integrado) oficial para desarrollar de aplicaciones Android disponible para Windows, macOS y Linux. Está basado en IntelliJ IDEA de JetBrains así que también cuenta con sus características herramientas de refactorización de código para Java.

  Algunas de las funcionalidades que ofrece son la posibilidad de diseñar las interfaces de usuario arrastrando componentes de la UI a una previsualización de la misma (ver Figura 4.2). También permite el uso de emuladores de diferentes versiones del SO para probar las aplicaciones o la posibilidad de integrar sistemas de control de versiones para utilizarlos a través de la propia interfaz gráfica del IDE.

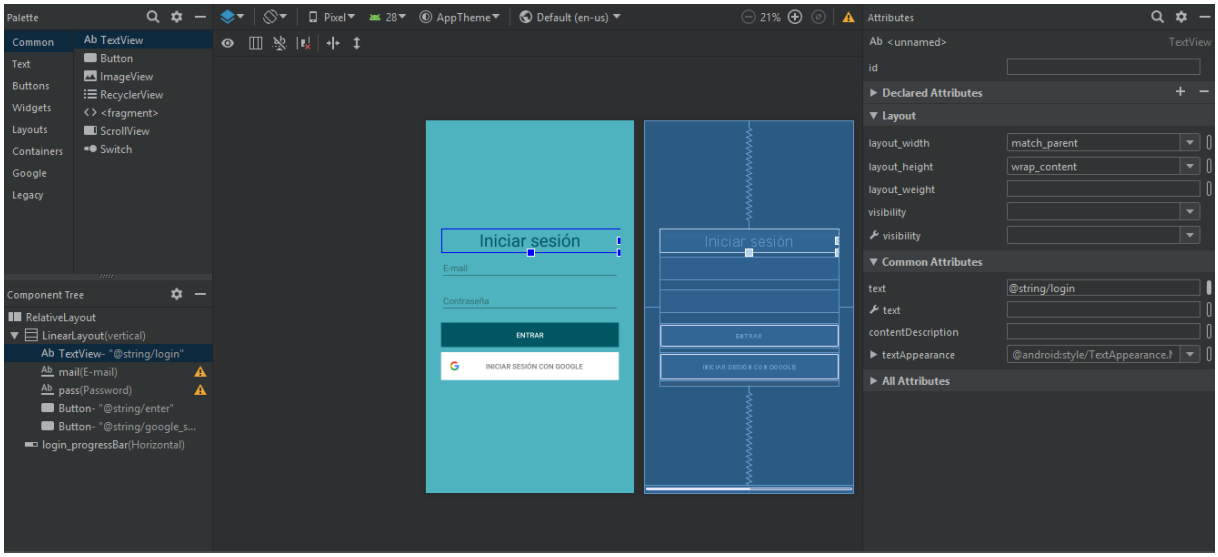


Figura 4.2: Interfaz ”drag and drop” para diseñar UI en Android Studio

- **Java**

  Java es un lenguaje de programación de propósito general orientado a objetos que fue comercializado por primera vez en 1995. Es posible encontrar Java en portátiles y centros de datos, en consolas de videojuegos y en súper computadoras, en teléfonos móviles y hasta en Internet.

  Es el lenguaje de programación utilizado para desarrollar apps en Android Studio, aunque también es posible utilizar Kotlin o C++.

- **XML**

  Es un lenguaje de marcado de propósito general. A diferencia de otros lenguajes de marcado, XML no está predefinido y se deben utilizar etiquetas propias.

  Se utiliza para diseñar las UI en Android con unas etiquetas propias. La funcionalidad ”drag and drop” previamente nombrada de Android Studio, genera código XML, aunque también es posible escribirlo directamente (ver Figura 4.3), que es más rápido una vez se cuenta con la soltura suficiente.

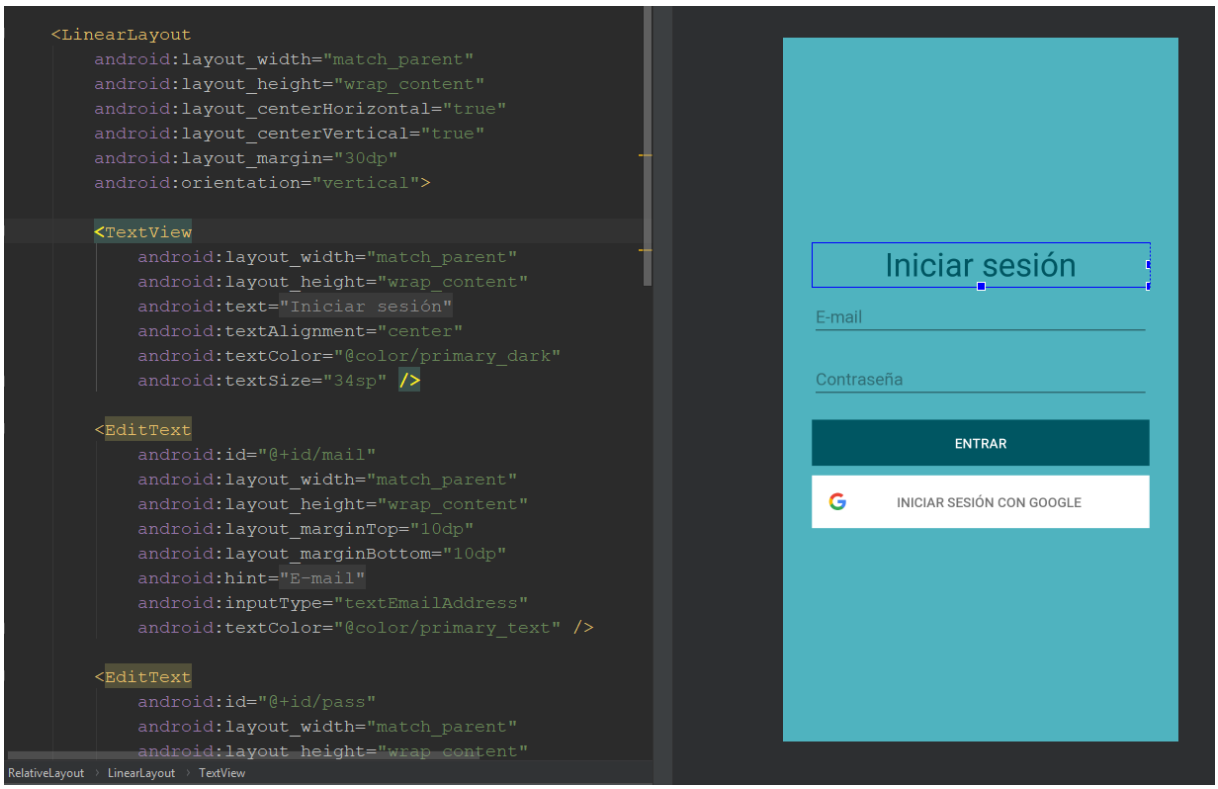


Figura 4.3: Código XML y previsualización del mismo

- **Color Tool de Material Design**

  Esta herramienta genera paletas de colores basados en las guías de estilos definidas y permite utilizar este tema en la UI en base a unos colores seleccionados (ver Figura 5.1).

  Material Design es el conjunto de guías de diseño y estilo previamente mencionado, propuestas por Google enfocadas en la apariencia de las aplicaciones móviles y web. Esta filosofía se aplica a todos los productos de Google, lo que aporta consistencia al diseño de estos, y los hace más intuitivos y familiares para los usuarios. Además, también simplifica el proceso de diseño para los desarrolladores de aplicaciones.

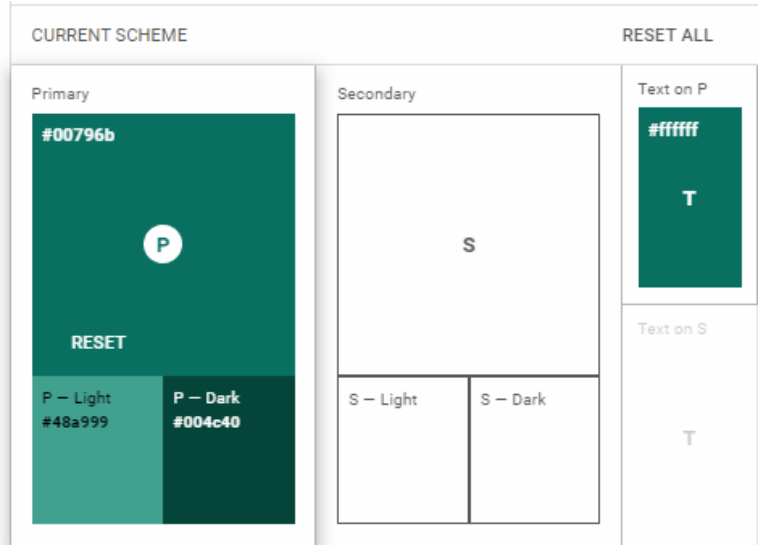


Figura 4.4: Paleta de colores generada por la herramienta de Material Design

- **SpongyCastle** [19]

  Es un paquete de Java que contiene diferentes implementaciones de algoritmos criptográficos. Se trata de una actualización de la librería de criptografía BouncyCastle con una serie de cambios menores para que sea posible utilizarla en Android, ya que al utilizar la librería original se producían conflictos con nombres de clases con los propios del SO.

- **ZXing barcode scanner** [20]

  Zxing –abreviatura de ”Zebra crossing”– barcode scanner, es una librería de código abierto multiplataforma cuya funcionalidad principal es el procesamiento de imágenes de código de barras tanto 1D como 2D.

- **QR Generator** [21]

  Librería para Android que permite generar códigos QR y almacenarlos como imagen.

- **Google Maps**

  Es la aplicación de mapas de Google que permite visualizar mapas, buscar ubicaciones concretas y definir rutas entre dos puntos.

  Es posible lanzar Google Maps con una ruta entre dos puntos A y B desde una aplicación de Android. Se lanza como una actividad convencional con ambas direcciones formateadas de manera adecuada.

4.2. Action de Google Assistant

- **Google Home mini y Google Assistant**

  Durante todo el proceso de desarrollo del proyecto se han utilizado estas dos tecnologías. Se han presentado en profundidad en el Capítulo 2: Preliminares.

- **Actions on Google**

  Es una plataforma para desarrolladores que permite crear aplicaciones que extienden las funcionalidades de Google Assistant. Estas aplicaciones, llamadas Actions, se encargan de procesar las peticiones de los usuarios y de realizar la acción solicitada. Esa acción puede consistir en buscar contenido en la web, abrir una aplicación u otras funcionalidades del asistente. Las Actions se ejecutan en la nube, independientemente del dispositivo desde el que se hayan invocado.

  Es posible crear Actions para diferentes entornos: embebidas en contenido web, como extensión de aplicaciones móviles, o para altavoces inteligentes con código propio o a partir de plantillas prediseñadas.

  Las Actions están compuestas por un conjunto de conversaciones o ”intents” de Dialogflow.

- **DialogFlow**

  DialogFlow es la plataforma de Google para crear agentes conversaciones de voz y texto con inteligencia artificial.

  Estas conversaciones se llaman ”intents”. Un intent representa la tarea que el usuario quiere realizar. Para completar esta tarea, el usuario obtendrá una respuesta

por parte del asistente de voz. Esta respuesta puede requerir acceder a un servicio externo para recuperar información, lo que conoce como ”fulfillment”, o por el contrario se puede establecer una respuesta predefinida.

En la Figura 4.5 se muestra un intent muy sencillo que forma parte de la Action desarrollada para este proyecto:

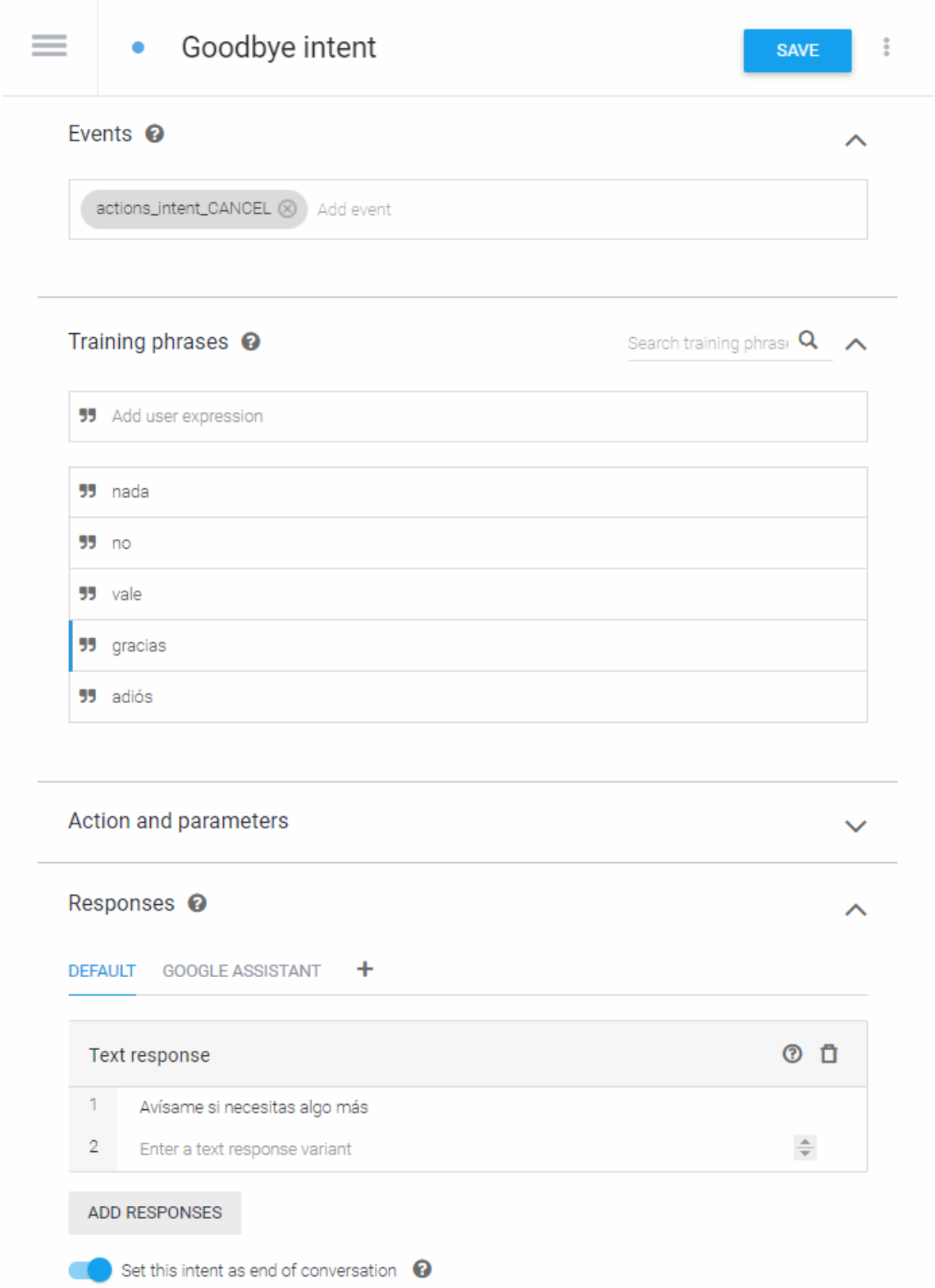


Figura 4.5: Intent de despedida de la Action desarrollada

Concretamente, el intent mostrado en la Figura 4.5, es la despedida para abandonar la Action. Cuando el usuario pronuncie alguna de las frases de entrenamiento o ”training phrases” el asistente de voz responderá, en este caso con una respuesta será predefinida. En la Figura 4.6 se muestra el intent en funcionamiento:

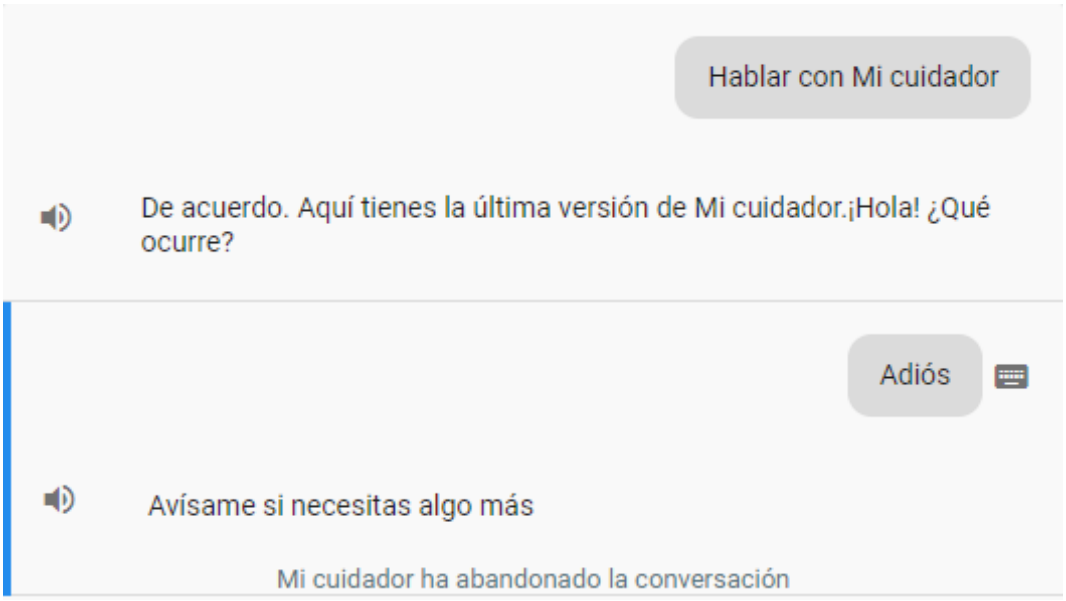


Figura 4.6: Intent de despedida en Google Assistant

- **JavaScript y NodeJs**

  Para los casos en los que devolver una respuesta predefinida no es suficiente, existe la posibilidad de ejecutar un fulfillment, que como se ha comentado anteriormente, consiste en una llamada a un servicio externo como por ejemplo puede ser una base de datos.

  Los fulfillments se escriben en JavaScript, se despliegan como Cloud Functions para Firebase y son ejecutados en un entorno NodeJS en la nube.

  JavaScript es un lenguaje de programación interpretado orientado a objetos asíncrono y débilmente tipado que se ejecuta en el lado del cliente (client-side). NodeJS es un entorno de ejecución de código JavaScript para la capa del servidor.

- **Firebase CLI**

  La Firebase CLI (Command Line Interface) permite desplegar una Action en Cloud Functions, que se tratarán a continuación en el siguiente apartado. Se debe instalar localmente con el gestor de paquetes de NodeJS (npm).

  Sólo será necesario para los proyectos con al menos un intent cuya respuesta requiera ejecutar un fulfillment.

4.3. Backend y Base de datos

Para desarrollar este proyecto también se han utilizado diferentes funcionalidades de las ofrecidas por Firebase, la plataforma de desarrollo en la nube de Google. Proporciona diferentes herramientas para gestionar la autenticación de usuarios, el almacenamiento

de datos en la nube y diferentes estadísticas de uso, entre otras funcionalidades, sin necesidad de gestionar servidores. La API de Firebase está disponible para proyectos Android, iOS, Web, C++ y Unity, además de disponer una SDK de administrador para servidores. Las herramientas de Firebase que se han utilizado son:

- **Authentication**

  Facilita la gestión de usuarios de forma simple y segura. Ofrece diferentes métodos de autenticación para los usuarios mediante e-mail y contraseña o inclusive gracias a la utilización de otros proveedores como Google, Facebook o Twitter.

- **Cloud Firestore**

  Es una base de datos noSQL alojada en la nube. Permite mantener los datos sincronizados en tiempo real mediante listeners y ofrece soporte offline de tal manera que todas las aplicaciones que estén consumiendo datos de esta base de datos pueden seguir en funcionamiento independientemente de la latencia de la conexión a internet.

  La principal diferencia entre Cloud Firestore y Realtime Database (las dos bases de datos que ofrece Firebase) es que con Cloud Firestore es posible realizar consultas más complejas, como pueden ser operaciones de filtrado u ordenación. Por otro lado, mediante el uso de Cloud Firestore se pueden leer datos de una sola vez, sin la necesidad de mantener un listener escuchando todo el tiempo.

  Se integra fácilmente con otros productos como Cloud Functions sin necesidad de disponer servidores que actúen como intermediarios.

- **Cloud Funcions**

  También es posible ejecutar código backend gracias a Cloud Functions. Estas funciones estarán escritas en JavaScript y se ejecutarán en un entorno seguro de Node JS. Tan sólo se ejecutan cuando ocurre un evento específico, que pueden ser emitidos por otros productos de Firebase, servicios de Google Cloud o de terceros utilizando webhooks.

  Estas funciones se despliegan utilizando únicamente un comando y los recursos asignados escalan automáticamente según sea necesario.

  Además, es posible consultar el informe de errores, depuración y mensajes de información sobre el uso de la Cloud Function desplegada (ver Figura 4.7).

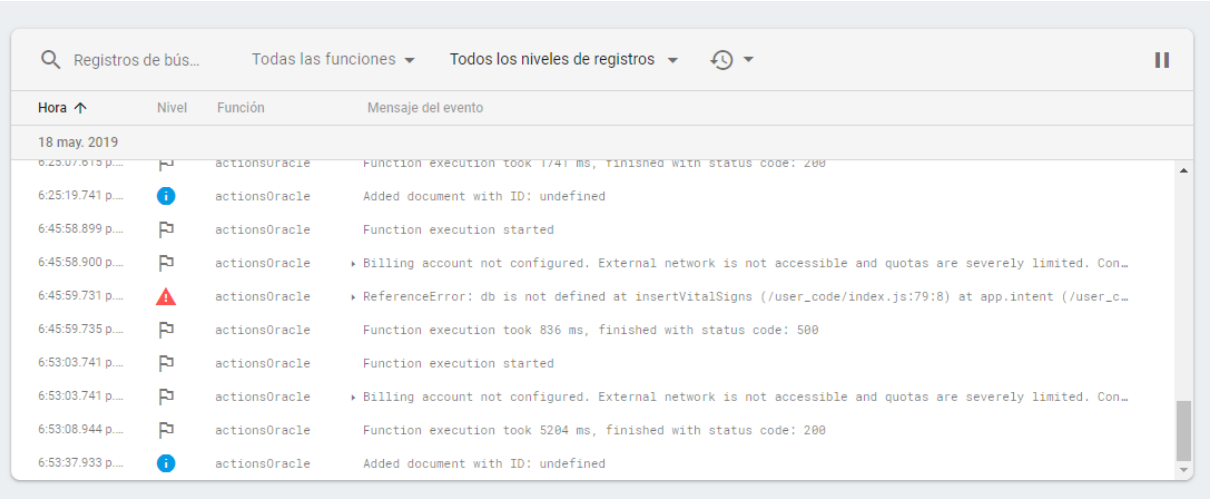


Figura 4.7: Informe de errores y de depuración sobre una Cloud Function

4.4. Otros

El sistema de control de versiones elegido para registrar los cambios tanto en la aplicación móvil como en la Action desarrollada (en repositorios independientes) es Git. Se ha utilizado:

- **GitHub**

  La popular plataforma de alojamiento del código de proyectos en remoto que utiliza Git.

- **GitHub Desktop**

  Es la aplicación de escritorio que permite utilizar GitHub desde el escritorio, tal y como su propio nombre sugiere. Es una forma sencilla y rápida de gestionar los repositorios en local a través de una interfaz gráfica. Además, es posible integrarlo con IntelliJ IDEA y por lo tanto, con Android Studio.

Capítulo 5

Aplicación móvil

En este apartado nos adentraremos en todos los aspectos relacionados con el diseño y desarrollo de la aplicación móvil. Se debe de puntualizar que la aplicación desarrollada ha sido especialmente implementada para que sea compatible con todos los dispositivos Android cuya versión del sistema operativo sea igual o superior a la versión Android 5.0 (Lollipop).

La aplicación móvil se ha desarrollado con diferentes versiones de Android Studio, ya que durante el transcurso del trabajo en este proyecto se han publicado dos nuevas versiones. Por lo tanto, se han utilizado las versiones 3.1, 3.3.1, y 3.3.2, habiendo sido esta última publicada en marzo de 2019.

5.1. Diseño

Durante las primeras etapas del proyecto se estableció la paleta de colores a utilizar (ver Figura 5.1) para expresar consistencia y uniformidad en todas las interfaces de la aplicación, cuyos valores están definidos en el fichero $style.xml$.

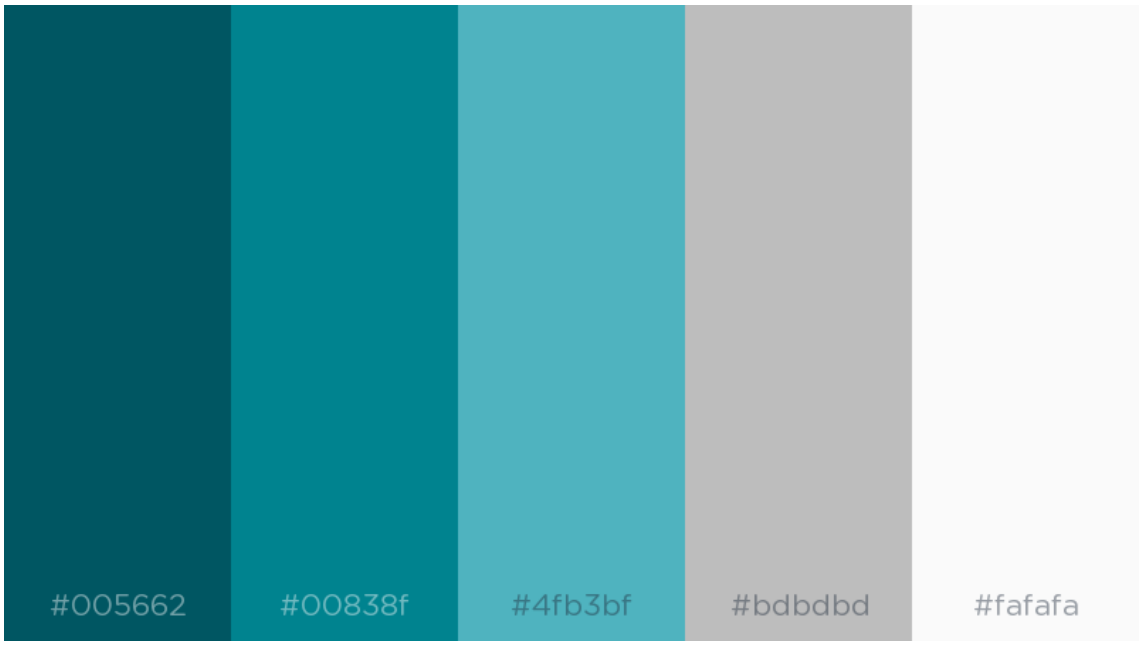


Figura 5.1: Paleta de colores utilizada

Con ella se han definido los estilos y ”themes” a aplicar en cada elemento de la interfaz de usuario. El uso de diversos colores permite diferenciar los diferentes componentes e identificar las acciones que es posible realidad (botones, pestañas, ...) de manera intuitiva.

Se han seguido las normas de diseño establecidas en la normativa Material Design propuestas por Google. Concretamente, los elementos en los que más atención se ha puesto son las pestañas o ”tabs” y en los botones (ver Figura 5.2).

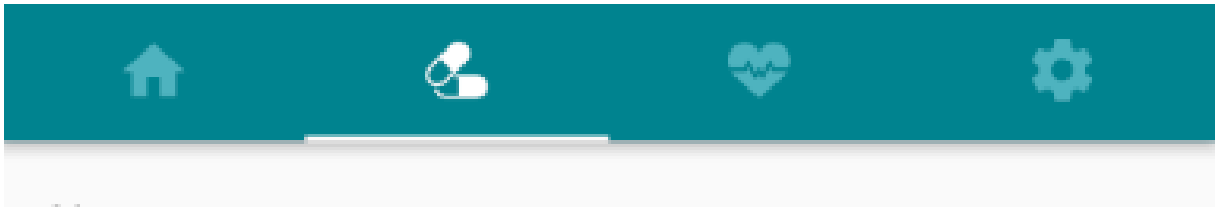


Figura 5.2: Tabs de la actividad principal de la aplicación

En la actividad principal de la aplicación el elemento principal de navegación es una TabBar (ver Figura 5.2). Es posible navegar entre sus diferentes pestañas deslizando el dedo desde cualquier parte de la pantalla o pulsando sobre las propias pestañas. A pesar de que las pestañas con texto son más claras, se han utilizado iconos descriptivos para indicar el tipo de contenido que se puede encontrar en cada pestaña debido a que las palabras ”medicamentos” o ”constantes vitales” tienen muchos caracteres y no se visualizaban correctamente en ciertas pantallas. También debe ser posible identificar la pestaña activa del resto de pestañas, para lo cual se resalta el color del icono y el texto (si lo hubiera) cambiarían de color y la propia pestaña debería aparecer subrayada con el accent color, en este caso el blanco.

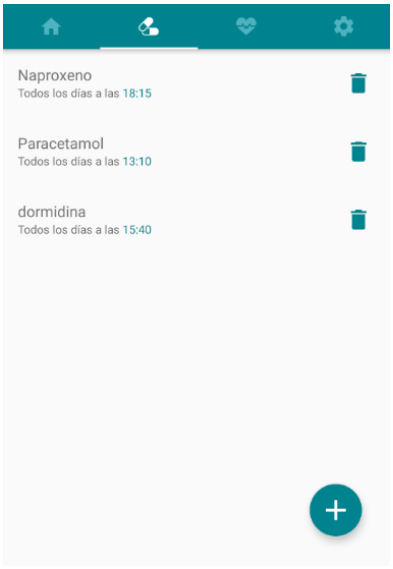


Figura 5.3: FAB en la esquina inferior

Se ha utilizado un FAB (Floating Action Button) en la interfaz de medicamentos para que el cuidador pueda añadir nuevos medicamentos (ver Figura 5.3). Se trata de uno de los elementos con ciertas características específicas. Estos botones aparecen en primero plano (delante de todos los demás elementos de la interfaz) y representan la acción principal de una interfaz. Tienen forma redondeada con un icono en el centro, ubicados normalmente en la parte inferior derecha de la pantalla.

También existen guías de diseño cuyo objetivo es incluir botones para iniciar sesión a través de diferentes servicios externos en una aplicación, como por ejemplo con una cuenta de Google, que es una de las formas de iniciar sesión en la aplicación desarrollada (ver Figura 5.4).

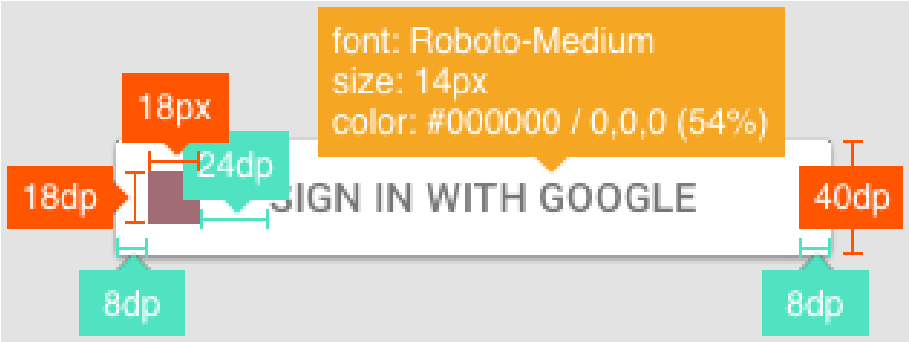


Figura 5.4: Guía de diseño para los botones de inicio de sesión con Google [22]

Cabe mencionar que todos los iconos (tanto el logotipo de la aplicación como las imágenes que aparecen en las pestañas) y avatares de los usuarios que se han incluido han sido utilizados según las indicaciones de los repositorios de imágenes en línea de acuerdo a su licencia correspondiente.

5.2. Funcionalidades

A continuación se muestran las diferentes funcionalidades implementadas en la aplicación ordenadas por interfaces.

5.3. Acceso a la aplicación

Al abrir la aplicación lo primero que se observa es la pantalla de carga que aparece en la Figura 5.5 con una barra de carga en la parte inferior que indica al usuario que se están cargando los datos necesarios. El algoritmo que se utiliza para determinar qué actividad lanzar a continuación es el siguiente:

1. Comprobar si ya hay alguna sesión iniciada.

   Si no hay una sesión iniciada, redirige al usuario a la interfaz de crear una cuenta (ver Figura 5.6), desde la que podrá acceder a la interfaz de iniciar sesión (ver Figura 5.7). Los métodos de autenticación disponibles en la misma son por dirección de correo electrónico y contraseña o con una cuenta de Google.

2. Si por el contrario ya hay una sesión iniciada, se comprueba si el usuario al que esa sesión corresponde ha completado su perfil o no.

   Si no ha completado su perfil, se redirige al usuario a la interfaz en la que debe completar la información de su perfil para acceder a la aplicación (ver Figura 5.8).

3. En caso de que el usuario sí haya completado su perfil, será necesario realizar una serie de acciones antes de que pueda acceder a la interfaz principal de la aplicación

   - Recuperar de la base de datos la información de su cuenta para guardarla en el almacenamiento local de la aplicación (SharedPreferences) durante el tiempo que dure la sesión.
   - Generar el par de claves del usuario. Ambas claves se codifican y se transforman en cadenas de caracteres para poder almacenarlas. La clave privada se

almacenará en local, mientras que la clave pública se registrará en la base de datos. Esto es necesario para la vinculación de cuentas, que se explicará en detalle en el Capítulo 7: Seguridad.

- Si el usuario es un cuidador y su cuenta está vinculada con la de un paciente, se deberá recuperar cierta información del perfil del paciente adicionalmente (dirección de correo electrónico, nombre completo y domicilio)

Antes de enviar los formularios que aparecen en las actividades de crear cuenta, iniciar sesión o de completar perfil se comprueba que los formularios se hayan rellenado correctamente para evitar inconsistencias en la base de datos.

Figura 5.5: Pantalla de carga

Figura 5.6: Crear una cuenta

Figura 5.7: Iniciar sesión

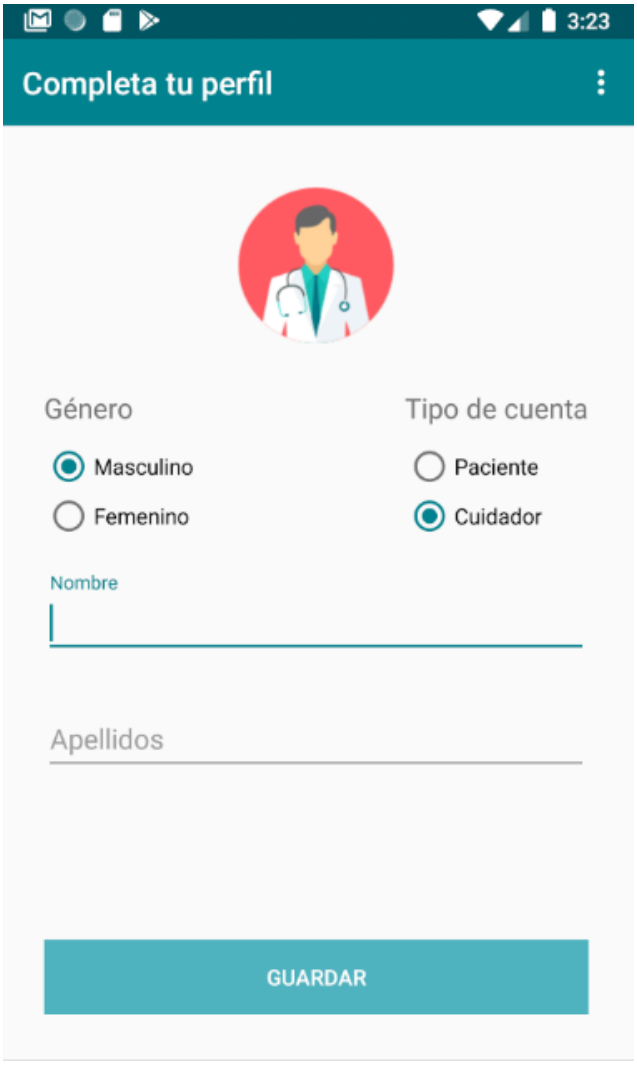


Figura 5.8: Completar el perfil

5.4. Actividad principal

Una vez se haya iniciado sesión y se haya completado el perfil de usuario se accederá a la actividad principal. Está compuesta por un TabLayout que cuenta con cuatro pestañas: la bandeja de eventos principal, medicamentos, registros de constantes vitales y ajustes (ver Figura 5.2).

Cada pestaña está asociada a un Fragment que se cargará en el espacio reservado para ello, el ViewPager. El elemento que se encarga de cargar el fragmento adecuado en cada momento es el FragmentPagerAdapter.

Cuando se crea la actividad principal, se asocia este Adapter al ViewPager de la interfaz (el espacio donde se van a cargar los fragmentos). A su vez, este ViewPager vincula al TabLayout de la interfaz para que los cambios que se realicen en alguno de los dos, es decir, cuando se cambie de pestaña, el otro se actualice automáticamente.

A continuación se va a exponer el contenido y funcionalidad de cada tab.

5.4.1. Bandeja de eventos principal

Al entrar en la actividad principal, se carga por defecto la primera pestaña, el bandeja de eventos principal (ver Figuras 5.9, 5.10 y 5.11). En esta pestaña se mostrará una lista de elementos ordenados cronológicamente en la que los nuevos elementos se añadirán por la parte superior sin necesidad de recargar la aplicación.

Estos elementos pueden ser de dos tipos:

- Notificaciones de emergencias

  Se crean cuando un usuario pide ayuda desde la Action para Google Assistant (explicaremos con más detalle esta funcionalidad en el capítulo siguiente). Estos items son tarjetas de color naranja y cada una de ellas hace referencia a una petición de ayuda del paciente, dentro de la misma se pueden ver detalles como la hora de la generación del evento. Si un cuidador toca una notificación de emergencia, se abre Google Maps con una ruta desde su ubicación actual hasta el domicilio de su paciente.

- Recordatorios de medicamentos

  Estos elementos representan los medicamentos que el paciente tiene que tomar o que ya ha tomado. Si estas tarjetas son de color blanco y poseen un icono verde, significa que el paciente ya ha tomado el medicamento. En el caso en el que la tarjeta sea de color amarillo claro con una imagen de alerta amarilla, el significado de la tarjeta es totalmente diferente. Al tener una tarjeta de estas, el evento al que hace referencia es a la falta de confirmación por parte del paciente de que se haya tomado un medicamento, de esta forma probablemente se pudo haber olvidado de tomarlo.

  Los pacientes pueden indicar que ya han tomado un medicamento concreto, tocando la tarjeta de color amarillo correspondiente. Por otro lado, también podrán indicar la ingesta del mismo mediante comandos de voz desde la Action para Google Assistant, en cuyo caso el medicamento correspondiente se marcaría como tomado sin necesidad de recargar la página gracias a los listeners configurados.

Las cuentas de los dos usuarios que aparecen en las figuras están vinculadas (ver Figuras 5.10 y 5.11). En la bandeja de eventos de ambas cuentas los elementos representan la misma información, pero el texto aparecerá escrito en primera o tercera persona dependiendo de el rol del usuario.

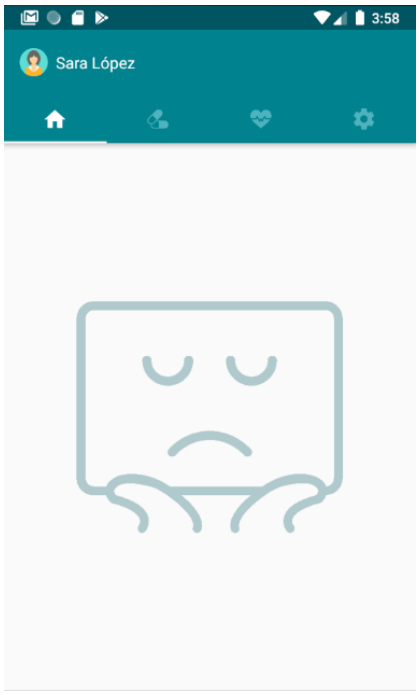


Figura 5.9: Bandeja vacía

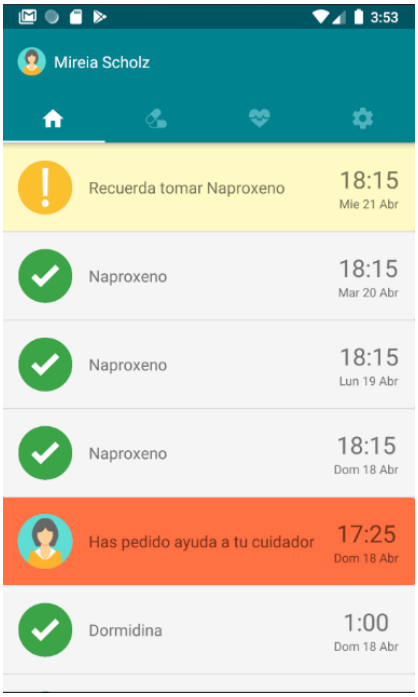


Figura 5.10: Bandeja

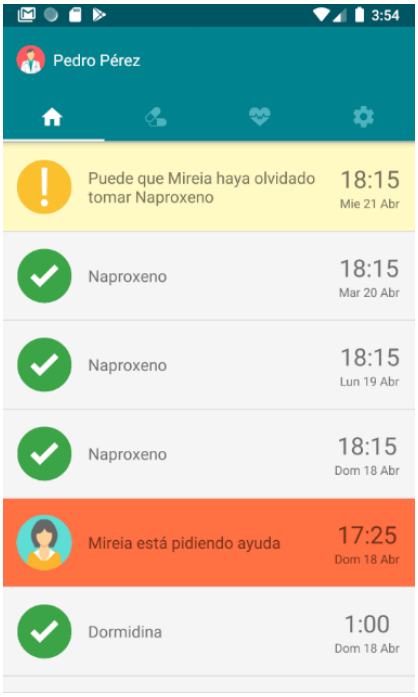


Figura 5.11: Bandeja

Ese listado de elementos se trata de un RecyclerView. Para poder inyectar los datos que queremos mostrar en ese RecyclerView de la interfaz de usuario también es necesario un Adapter.

La aplicación está constantemente escuchando de la base de datos por si se registra un nueva emergencia o hay un nuevo recordatorio de medicamento. Cuando esto ocurre se crea un DTO (Data Transfer Object) que se transfiere al Adapter. Por último, el Adapter lee la información del DTO y crea un nuevo elemento en la interfaz de usuario con la información correspondiente a partir de una plantilla (ver Figura 5.12).

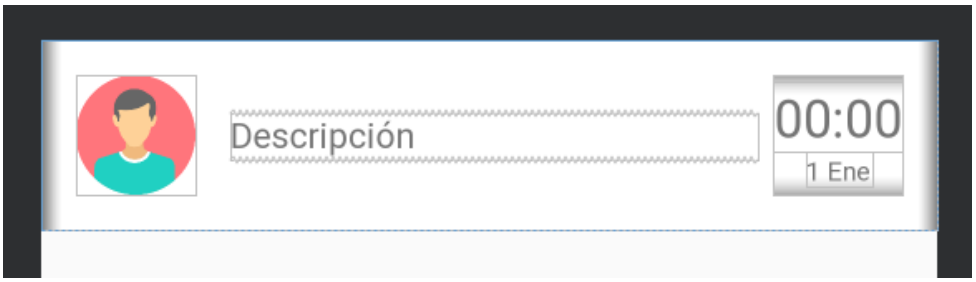


Figura 5.12: Plantilla de un elemento de la bandeja de eventos

El DTO debe poder representar tanto una notificación de emergencia como un recordatorio de medicamento. Sin embargo, como ambos tienen características diferentes (realizan diferentes acciones al tocarlos, sus colores son diferentes y el texto que aparecerá también) representarlos como una única clase no es la solución más acertada.

La solución propuesta es una jerarquía de clases que represente los elementos de la bandeja de eventos. Se desarrolló una clase FeedItem abstracta que representa la bandeja de eventos, es decir, no se pueden instanciar objetos de ella, con dos clases hijas: EmergencyItem y MedItem. Los atributos que ambas clases comparten son los siguientes:

- Identificador del elemento en la base de datos
- Nombre del usuario que lanzó el evento (el paciente).
- Color del elemento.
- Texto que se mostrará en la interfaz.

En el constructor de cada una se gestionan sus características específicas.

5.4.2. Medicamentos

La segunda pestaña es donde aparecen todos los medicamentos que el paciente debe tomar y la hora a la que deben tomarlo. Por lo tanto, cuando sea la hora de tomar alguno de esos medicamentos, se registrará en la base de datos y aparecerá en la bandeja de eventos para que el paciente recuerde tomarlo. Los pacientes no podrán realizar ninguna tarea en esta interfaz (ver Figura 5.13) más que consultarla. Sin embargo, los cuidadores pueden eliminar (ver Figura 5.14) y añadir (ver Figura 5.15) medicamentos. Este listado también se trata de un RecyclerView con su Adapter correspondiente.

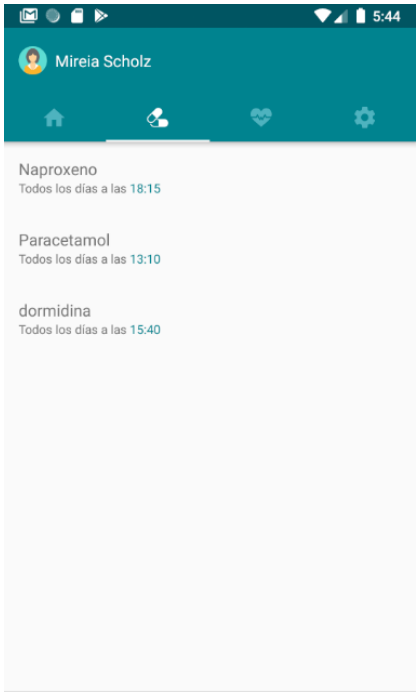


Figura 5.13: Medicamentos

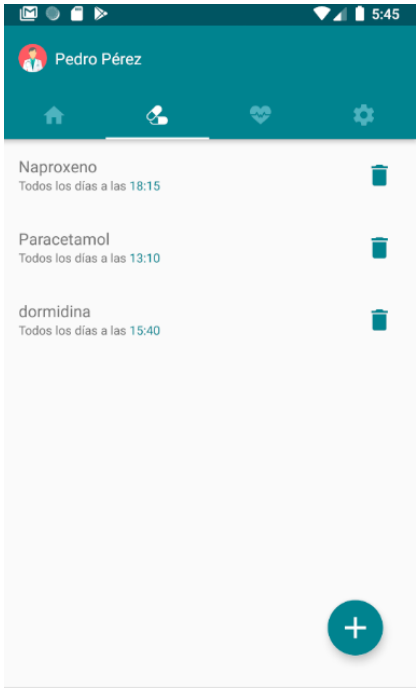


Figura 5.14: Medicamentos

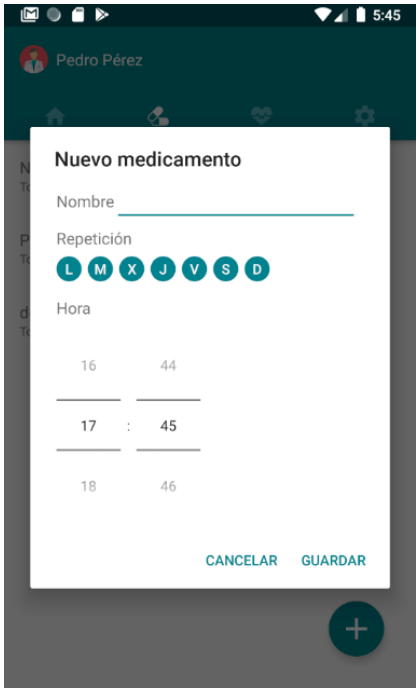


Figura 5.15: Añadir

Para determinar cuándo crear una nueva notificación de medicamento en la base de datos que se muestre en la bandeja de eventos, se desarrolló una tarea asíncrona que

compara el día y la hora a las que se deben tomar los medicamentos con el día y la hora actual. Todo esto mediante la utilización de diferentes listeners configurados. En caso de que las dos fechas coincidan, se registra ese medicamento en la base de datos.

5.4.3. Constantes Vitales

Desde esta tercera pestaña tanto los pacientes como los cuidadores pueden consultar los registros de constantes vitales que los pacientes hayan añadido utilizando los comandos de voz de la Action de Google Assistant (ver Figura 5.16), que trataremos en el siguiente capítulo.

Como en las pestañas anteriores, la lista de registros de constantes vitales también se trata de un RecyclerView gestionado por un Adapter desde el fragmento correspondiente.

5.4.4. Ajustes

En la última pestaña se encuentran los ajustes desde donde los usuarios podrán gestionar diferentes funcionalidades asociadas a su cuenta como puede ser cerrar su sesión (ver Figura 5.17). Desde esta vista de configuración los usuarios pueden acceder a su perfil y consultar la información de su cuenta (ver Figura 5.18). También podrán vincular su cuenta con otra cuenta del rol opuesto (funcionalidad que se explica más en detalle en el Capítulo 7). Al final de esta pestaña, en créditos, podemos encontrar toda la información relacionada a los iconos y logos utilizados (ver Figura 5.19).

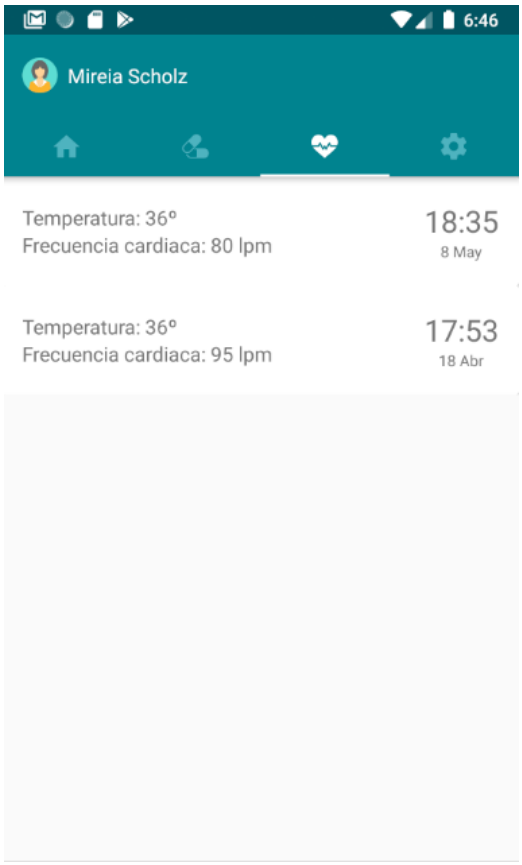


Figura 5.16: Registros de constantes vitales

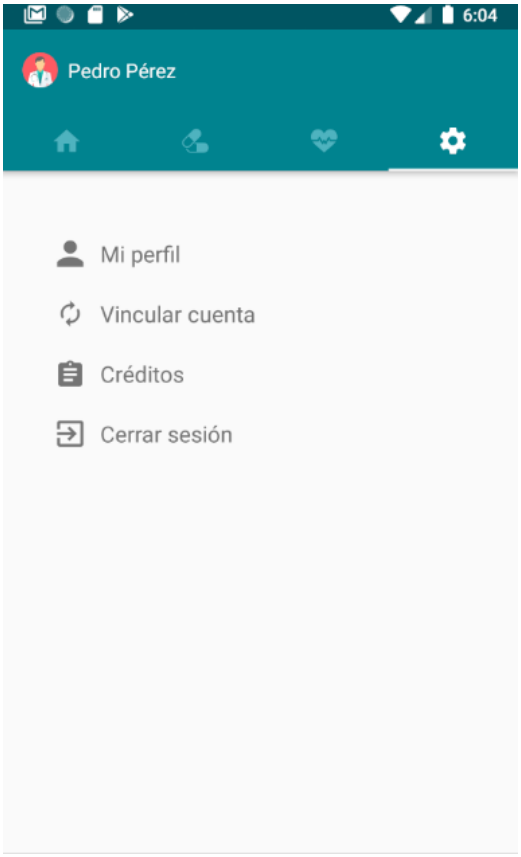


Figura 5.17: Configuración

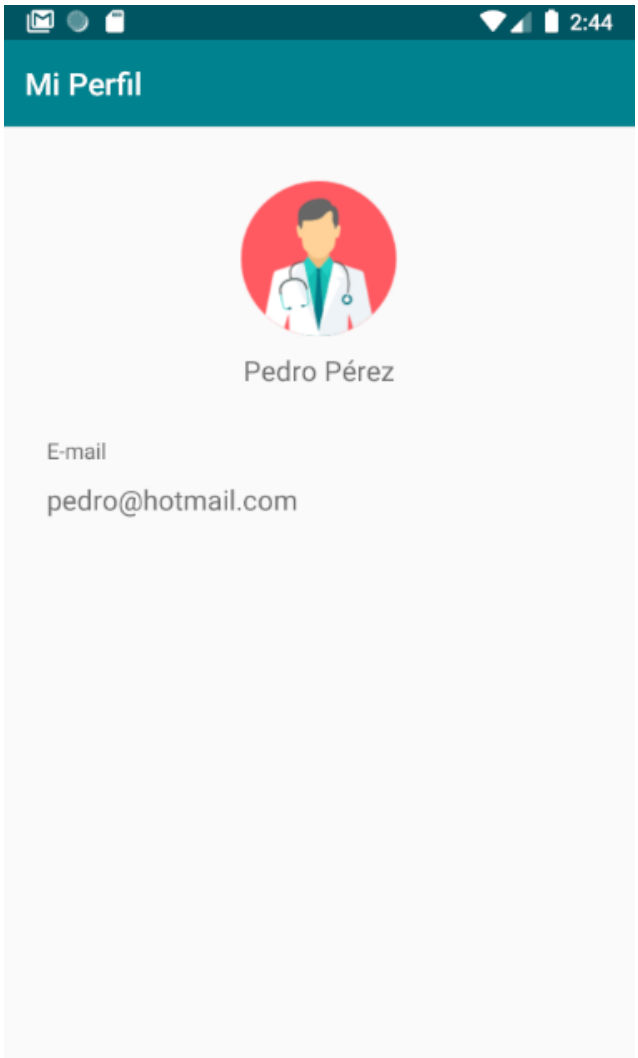


Figura 5.18: Perfil de usuario

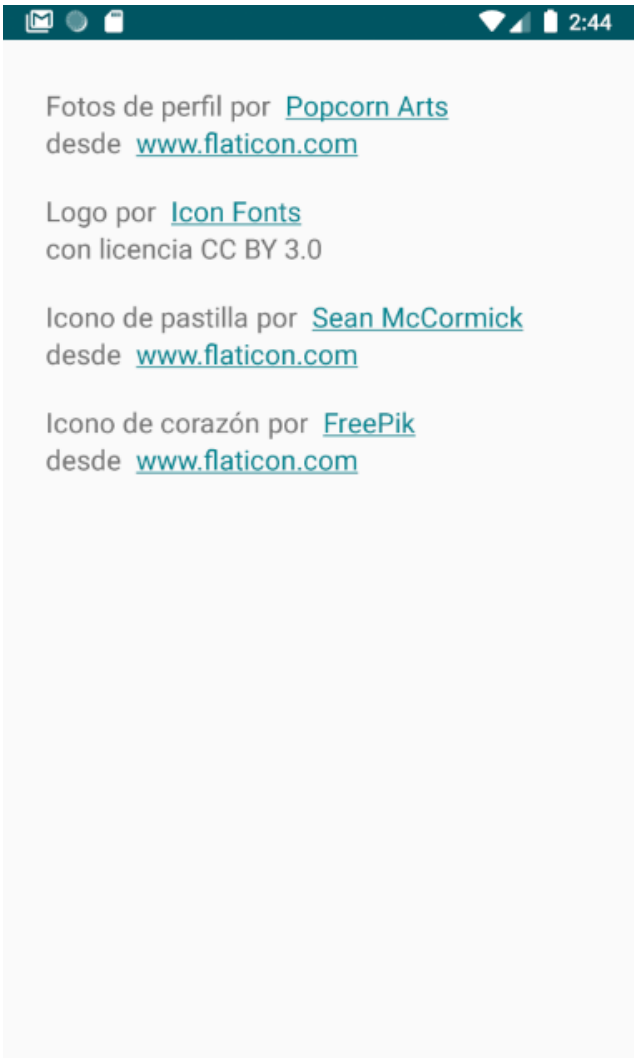


Figura 5.19: Créditos

Capítulo 6

Action para Google Assistant

En este capítulo se describirá la implementación de la Action para Google Assistant, desarrollada con el fin de que los pacientes puedan comunicarse con el sistema a través de comandos de voz.

Para poder utilizar esta Action basta con invocarla desde cualquier dispositivo con Google Assistant, como un teléfono móvil o un dispositivo Google Home, que esté asociado a la cuenta de Google con la que se creó la Action. Esto es debido a que aún no está publicada en la tienda de Google.

Para que sea posible leer y escribir en la base de datos desde la Action también es necesario estar autenticado, al igual que en la aplicación móvil. Sin embargo, en este caso no será necesario realizar ninguna acción adicional ya que el acceso se realizará con la cuenta de Google asociada al dispositivo desde el que se esté utilizando Google Assistant. Por ello, es necesario que previamente los usuarios se hayan registrado en la aplicación móvil.

Se ha decidido llamar a la Action ”Mi cuidador” porque de esta manera la Action se invoca diciendo ”Ok Google, hablar con Mi Cuidador” y el flujo de conversación es más natural.

La Action para Google Assistant se ha desarrollado principalmente en la nube, tanto a través de la consola de Actions on Google, que tiene integrado un simulador para probar la Action rápidamente, como a través de la consola de DialogFlow. Antes de comenzar, fue necesario habilitar una serie de permisos en el control de actividad de cuenta de la cuenta de Google con la que iba a desarrollar la Action: permisos de actividad en la web y aplicaciones, de información de dispositivos y de actividad de voz y audio.

El siguiente paso fue crear un un nuevo proyecto de bot conversacional personalizado, y automáticamente se crearía también un agente de DialogFlow, en cuya consola se programó la Action.

Mientras se desarrollaba la Action ha sido posible probar los diferentes intents desde el simulador de Google Assistant integrado en la consola de Actions on Google, de donde se han obtenido las figuras con las conversaciones adjuntas a este capítulo.

A continuación se van a tratar todos los intents desarrollados y su fulfillment respectivo en los casos en los que proceda.

6.1. Intent de bienvenida

Cuando se crea el agente, el Default Welcome Intent está ya creado y configurado. Este intent tiene un evento WELCOME asociado que se lanza cada vez que el usuario comienza una nueva conversación con el agente, y devuelve una respuesta para indicar al usuario que está escuchando (ver Figura 6.1).

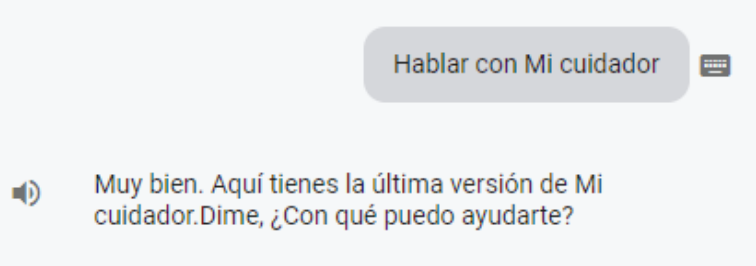


Figura 6.1: Welcome intent

Este intent de bienvenida también puede ser invocado a través de sus frases de entrenamiento o ”training phrases”, que por defecto son saludos comunes (Hola, hey, buenos días, etc.). Es posible modificar tanto las training phrases como sus respuestas. En este caso no se han sustituido las respuestas por defecto ya que al tratarse de un intent sencillo, no ha sido necesario escribir ningún código adicional ni se han utilizado webhooks.

6.2. Intent de despedida

La Action debe permitir a los usuarios abandonar rápidamente en cualquier punto de la conversación y que la conversación no termine de forma inesperada. Por defecto, la Action abandona la conversación (es decir, se cierra) cuando el usuario dice ”cancelar”, ”parar” o ”adiós”. Este comportamiento se puede modificar creando un intent con un evento ACTIONS_INENT_CANCEL y en este intent definir las respuestas personalizadas desde la propia interfaz de la consola de DialogFlow, sin necesidad de escribir código de fulfillment. Las training phrases que activan este intent son: adiós, nada, vale, gracias, no y cancelar.

Como se puede observar en la Figura 6.2, tras devolver la respuesta el asistente abandona la conversación y sale de la Action.

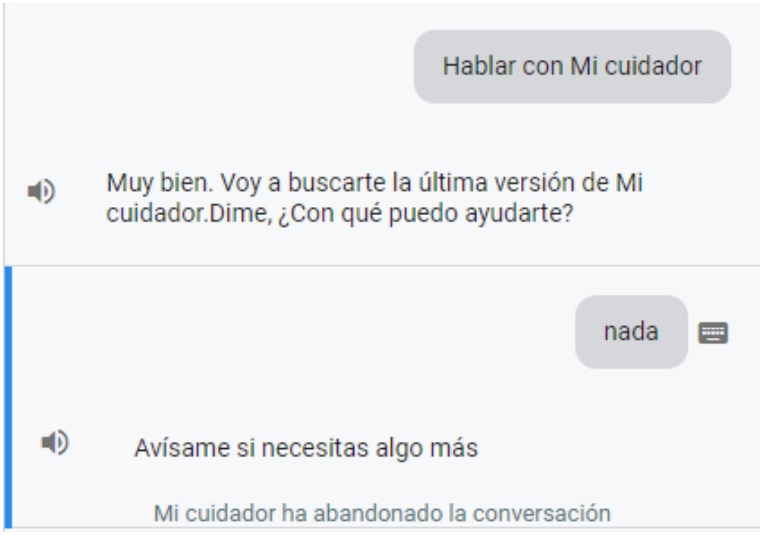


Figura 6.2: Goodbye intent

6.3. Start SignIn y Get SignIn

La primera vez que se utiliza la Action será necesario solicitar al paciente su consentimiento para poder acceder a su información de perfil de Google, incluyendo su dirección de correo electrónico, su nombre y foto de perfil. Esto es necesario para poder asociar la sesión de Google Assistant con la información asociada a esa cuenta en Firebase y establecer así un canal de comunicación entre ambas.

Para conseguir esto hubo que seguir una serie de pasos:

- Desde el panel del proyecto de Actions on Google, es necesario habilitar la opción que permite a los usuarios iniciar sesión mediante comandos de voz y seleccionar Google SignIn como método de vinculación.
- Crear dos intents: Start SignIn y GetSignIn
  - Start SignIn comienza el flujo de la vinculación de cuenta. Se añadieron a este intent las training phrases ”iniciar sesión” y ”quiero iniciar sesión”.
  - Get SignIn debe tener asociado un evento actions_intent_SIGN_IN. Si el usuario ha permitido acceder a sus datos, recuperará la información necesaria y la almacenará en una constante payload que podrá ser utilizada por el resto de los intents para acceder, en este caso, al correo electrónico del paciente.

El código del fulfillment de ambos intents estaba disponible en la documentación de Actions on Google y solamente fue necesario traducir las respuestas, ya que estaban inicialmente en inglés.

En la Figura 6.3 se puede observar cómo ocurriría la conversación. Eso solamente ocurrirá la primera vez que el paciente utilice la Action, aunque podrá revocar el acceso desde la configuración de su cuenta de Google.

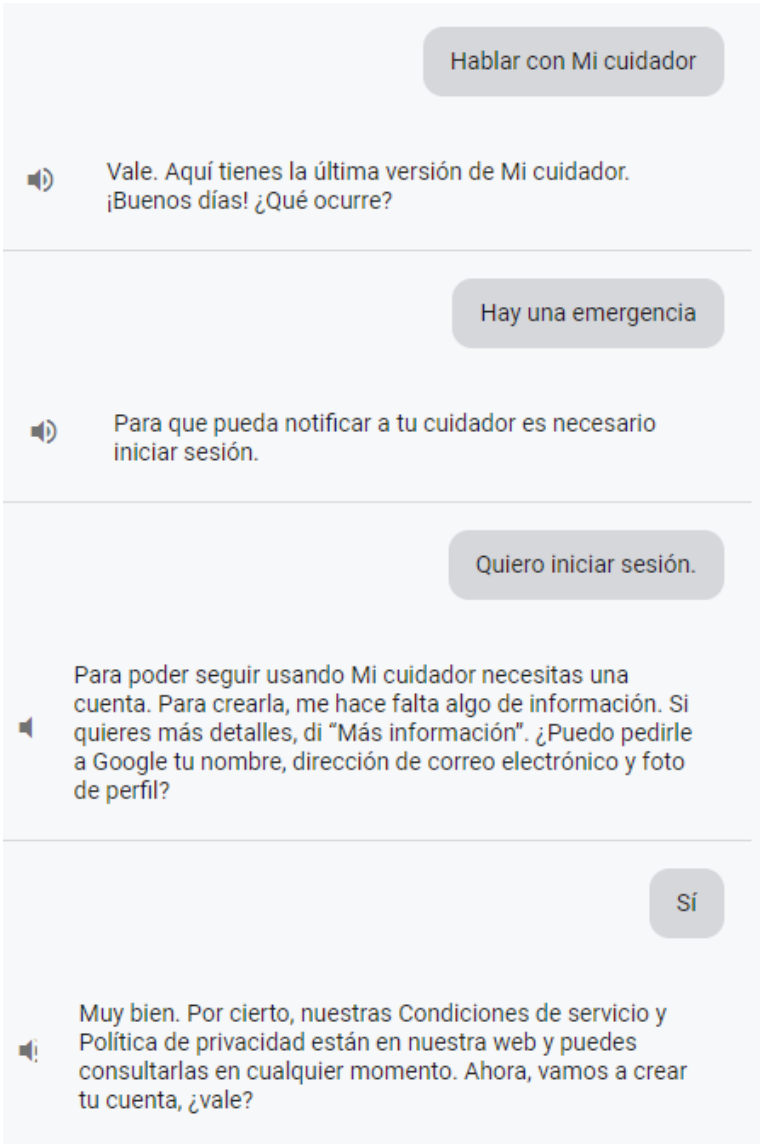


Figura 6.3: Solicitud de acceso a la información del perfil del usuario

6.4. Notificar emergencias

Este intent es el que se ejecuta cuando el paciente pide ayuda. Este intent es más complejo que los dos anteriores, debido a que en este caso es necesario escribir código adicional, y buscar palabras claves como ”urgencia” o ”emergencia”, que serán los parámetros que recibirá dicho código.

Las training phrases que activan este intent son: hay una emergencia, avisa de que hay una emergencia, ha ocurrido una emergencia, etc.

Como se acaba de comentar, en este caso será necesario ejecutar código para completar la tarea en un servicio externo, es decir, un fulfillment (ver Figura 6.6). Este código, que se escribe en JavaScript, registra el evento en la base de datos y devuelve una respuesta al usuario (ver Figura 6.7) . Para que todo funcione correctamente, será necesario desplegar el código con Firebase CLI desde la consola de NodeJS como Cloud Function. Cada vez que se invoque el intent, se ejecutará el código desplegado en la nube en un entorno NodeJS. El comando para desplegar los fulfillments del proyecto es $firebasedeploy$.

En el caso de que ocurriera una emergencia, sería un poco tedioso tener que pronunciar ”Ok Google, hablar con mi cuidador” antes de pedir ayuda (ver Figura 6.4). Por ello, este intent se ha habilitado para ser invocado con una invocación implícita. Es decir, se puede ejecutar este intent directamente sin escuchar el intent de bienvenida (ver Figura 6.5).

La conversación sigue sin sonar del todo natural, así para agilizar aun más la invocación de este intent se puede encapsular esa frase en una rutina de Google Home. Las rutinas de Google Home permiten encadenar comandos, así que se podría crear una rutina ”Hay una emergencia” que ejecutara ”Hablar con mi cuidador sobre que hay una emergencia”. De esta forma, si el paciente dijera ”Ok google, hay una emergencia” se ejecutaría esa rutina y se notificaría al cuidador de la misma manera que se haría en las figuras 6.5 y 6.6, pero de forma mucho más rápida y natural.

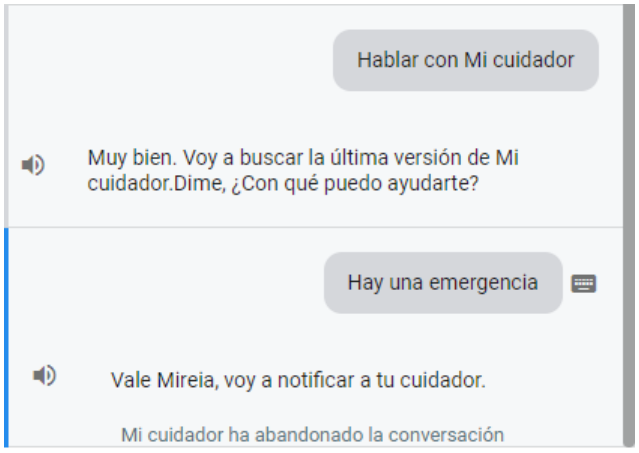


Figura 6.4: Invocación explícita del intent

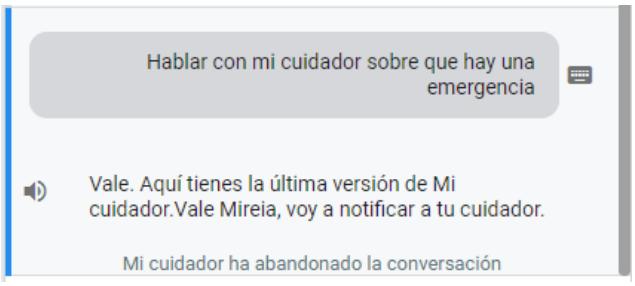


Figura 6.5: Invocación implícita del intent

```
app.intent('Send notification', (conv) => {
    const {payload} = conv.user.profile;
    if (payload) {
        const givenName = payload.given_name;
        const db = admin.firestore();
        const email = payload.email;
        var userRef = db.collection('data').doc(email);
        insertNotification(userRef, "emergency");

        conv.close(`Vale ${givenName}, voy a notificar a tu cuidador.`);

    } else {
        conv.ask("Para que pueda notificar a tu cuidador es necesario iniciar sesión.");
        conv.ask(new Suggestions('Quiero iniciar sesión.'));
    }

});
```

Figura 6.6: Ejemplo del código del fulfillment de una action

6.5. Recordatorios de medicamentos

Utilizando este intent, el paciente puede preguntar si tiene medicamentos pendientes de tomar y el agente enunciará el listado de pastillas programadas que falten por ingerir en el caso de que los tuviera (ver Figura 6.7).

Para este intent también ha sido necesario escribir un fulfillment. El listado de medicamentos que el paciente tiene pendientes de tomar se obtiene de la siguiente manera:

1. Primero se realiza una consulta a la base de datos para recuperar el listado de notificaciones de medicamentos del usuario. Para ello, la consulta debe filtrar los elementos de tipo emergencia y quedarse con los de tipo medicamento.

2. Todos los recordatorios de medicamentos que coincidan con el día y la fecha actual, y que además tengan el flag de ”tomado” marcado como falso (se añaden a un array).

3. Por último, este array se recorre y se formatean los nombres de los medicamentos como cadena de caracteres. De esta manera, el asistente de voz enumera los medicamentos que el paciente tiene pendientes de tomar para ese día.

A la hora de formatear la cadena de salida que leerá el asistente de voz, cuando hay varios medicamentos pendientes hay que tener en cuenta que todos, salvo el último par, deberán estar separados por comas, y este ultimo par por la letra ’y’.

Las training phrases que activan este intent son: Qué medicamentos tengo que tomar, qué pastilla tengo que tomar, medicamentos pendientes, etc.

6.6. Recordatorios de medicamentos para después

Con este intent el paciente puede preguntar al asistente de voz qué medicamentos tendrá que tomar el resto del día, sin contar los medicamentos que ya tiene pendientes.

En este caso, no se consultan a la base las notificaciones de medicamentos, pues estas aun no existen. Se consulta en la colección de medicamentos las pastillas que ese paciente debe tomar en general, es decir, los medicamentos que su cuidador ha añadido desde la interfaz móvil. Aquellos cuyo día de la semana coincida con el actual y además, que la hora a la que estén programados no haya pasado aún, se añadirán a un array. Este array se formateará de la misma manera que en el intent anterior, pero con la diferencia de que también se añadirá la hora a la que deberá tomarlos, y Google Assistant lo leerá como respuesta.

Las training phrases que activan este intent son: Qué medicamentos tengo que tomar después, qué pastilla tengo que tomar más tarde, medicamentos pendientes, y después, y más tarde, etc. Las últimas dos training phrases permiten una conversación con el asistente más fluida. Como se puede observar en la Figura 6.7, el usuario puede preguntar por los medicamentos que tiene pendientes de tomar y cuáles tendrá que tomar más tarde de forma natural.

6.7. Registrar constantes vitales

Con este intent los pacientes pueden registrar sus constantes vitales, de tal manera que esta información se almacena en la base de datos y su cuidador las puede consultar a través de la aplicación móvil.

La diferencia entre este intent es que va a estar esperando por dos valores: la temperatura corporal y la frecuencia cardiaca en latidos por minuto. Si el paciente a la hora de invocar este intent no proporciona esos dos datos, el asistente se los pedirá explícitamente, y no ejecutará el fulfillment (registrar esos dos valores en la base de datos) hasta que el usuario los proporcione (ver Figura 6.8). Para conseguir esto, en la consola de Dialogflow se indica que se están esperando dos parámetros, y que si no han sido proporcionados debe pedirlos (ver Figura6.9).

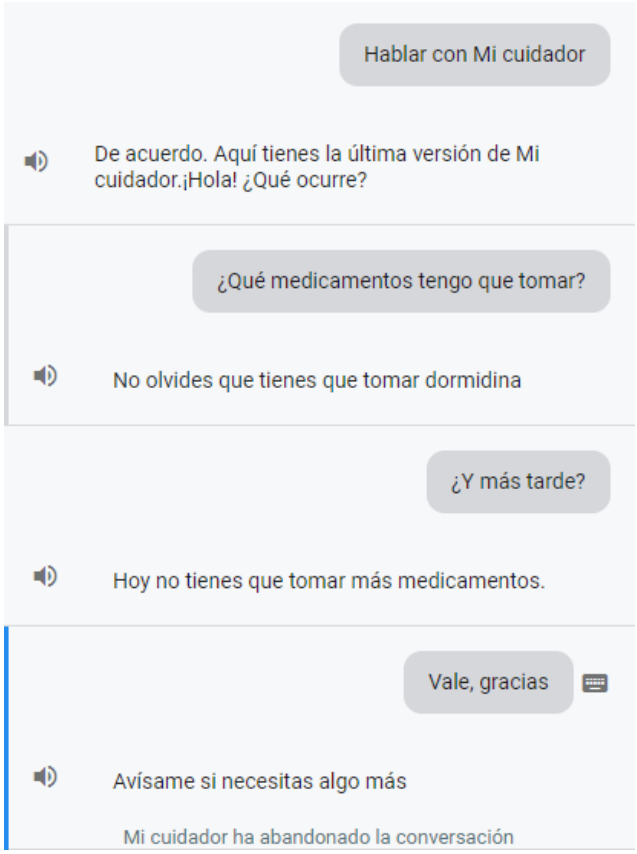


Figura 6.7: Recordatorios de medicamentos

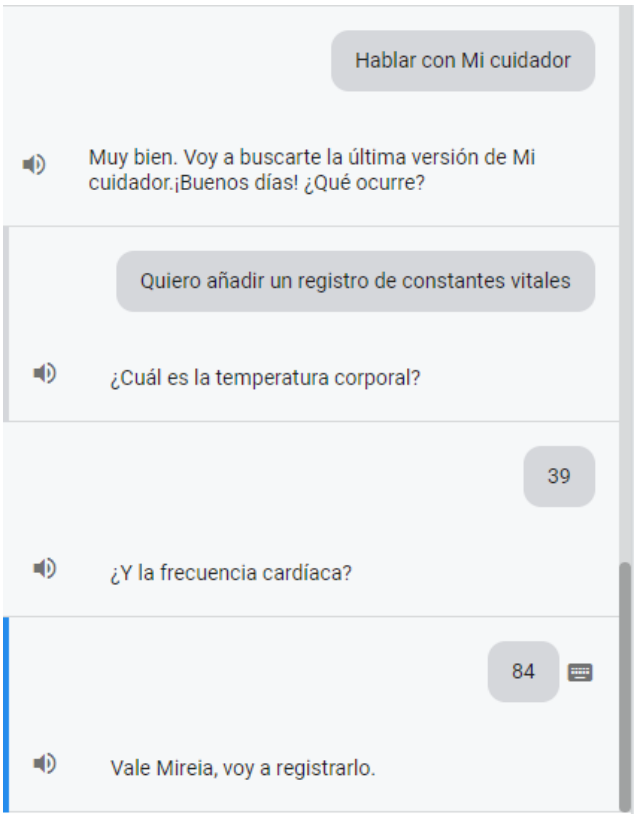


Figura 6.8: Registro de constantes vitales

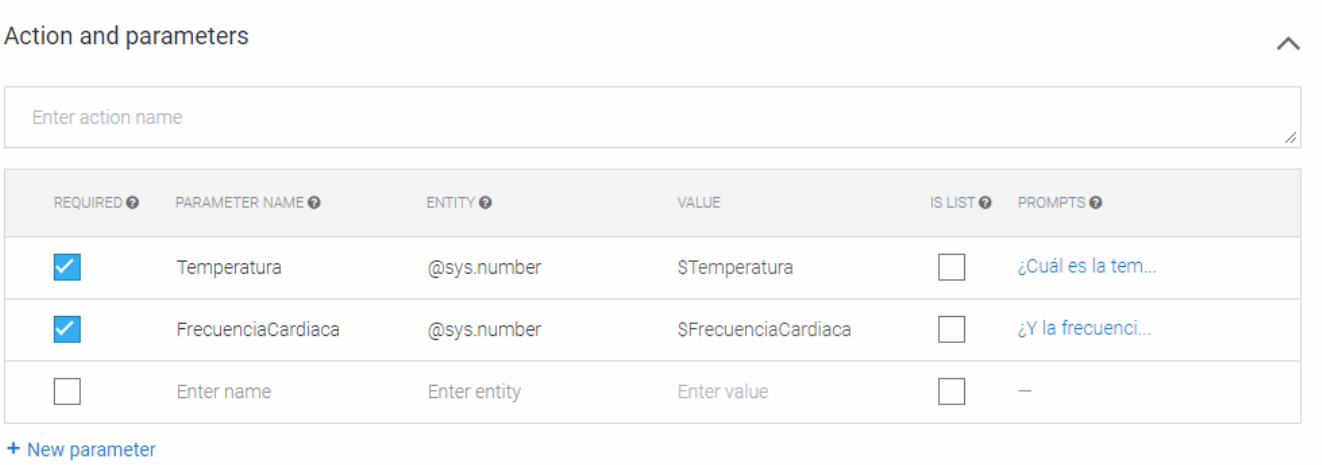


Figura 6.9: Parámetros que espera el intent Vital Signs

6.8. Medicamentos tomados

Si después de haber tomado los medicamentos que tenía pendientes, el paciente se los toma y quiere notificar a su cuidador utilizando comandos de voz, puede indicar que ya se los ha tomado (ver Figura 6.14). Como ya se ha comentado, la aplicación móvil está constantemente escuchando la base de datos, así que los cambios se verán reflejados en ella a tiempo real.

El fulfillment de este intent es muy similar al de Pill Reminder.

- Se consulta en la base de datos los medicamentos pendientes del paciente para ese día y que tengan el flag ”tomado” con valor falso, estos se devuelven como un conjunto.
- El el conjunto que se obtiene está vacío, es que no había ningún medicamento pendiente y el asistente de voz informará al paciente de ello.
- En caso de que el conjunto posea elementos, es decir que hay medicamentos pendientes, se recorre el conjunto y para cada notificación de medicamento pendiente, se actualiza el valor de ”tomado” a verdadero. Google Assistant indicará al paciente que la tarea se ha llevado a cabo con éxito.

Solamente se registrarán como tomados los medicamentos pendientes de ese día. Si hubo medicamentos pendientes de días anteriores no se registrarán como tomados.

Las training phrases que activan este intent son: Ya he tomado los medicamentos, ya tomé las pastillas, etc.

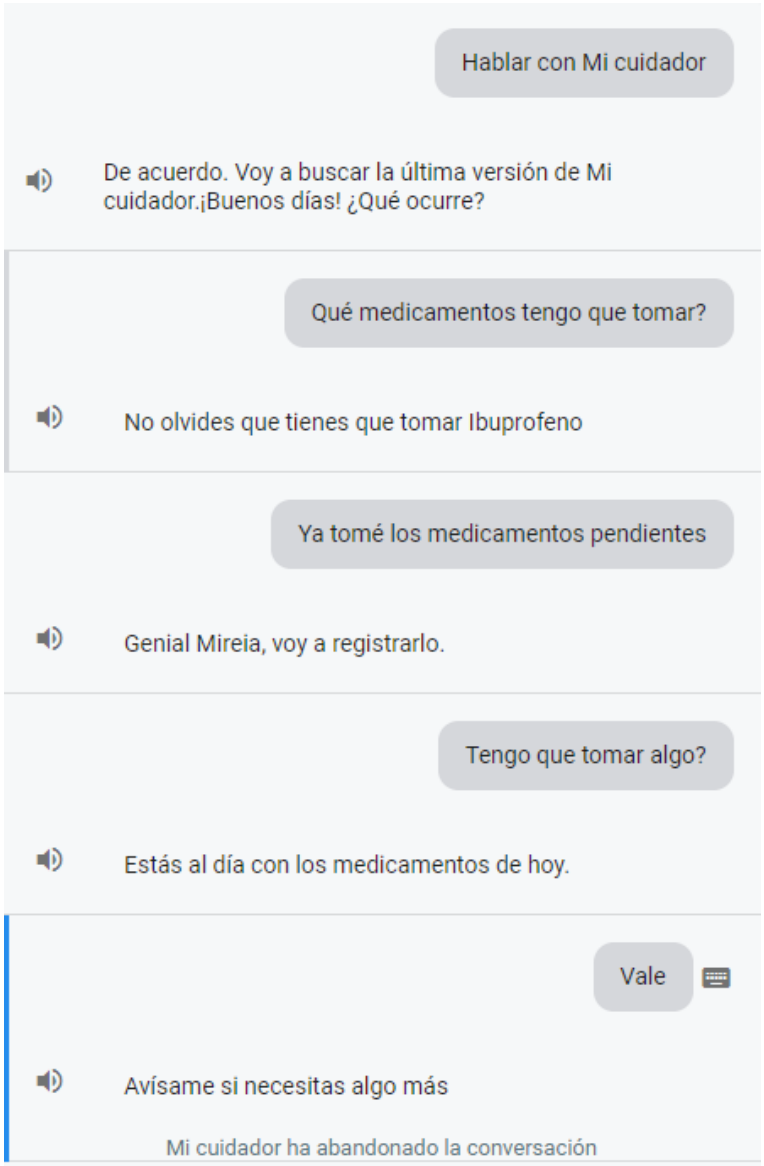


Figura 6.10: Intent de medicamentos tomados

Capítulo 7

Seguridad

7.1. Aplicación móvil

En las primeras etapas del proyecto, la decisión de vincular una pareja de cuentas cuidador-paciente era unilateral tomada por el cuidador. Éste escribía la dirección de e-mail del paciente que quería monitorizar y, si esa cuenta existía, ambas cuentas quedaban vinculadas. El problema de esta forma de vincular las cuentas es que los pacientes no participan ni deciden si desean vincular sus cuentas o no, y además no saben quién es la persona que está accediendo a sus información (los registros de constantes vitales, gestión de medicamentos, etc).

Se decidió implementar un mecanismo de vinculación de cuentas más sofisticado en dos pasos con el algoritmo criptográfico ECDH (Elliptic-Curve Diffie-Hellman) con un Out of Band Authentication (OOBA) para que esta vinculación deba realizarse de forma presencial.

7.1.1. Diffie-Hellman

El protocolo Diffie-Hellman, nombrado así en honor a Whitfield Diffie y Martin Hellman, que lo publicaron en 1976 [24], es un protocolo cuyo objetivo principal es generar una clave secreta compartida a través de un canal inseguro entre dos partes que no han tenido comunicaciones previas.

Para calcular la clave compartida, será necesario un intercambio previo de claves públicas por ambas partes, todo esto a través de este canal. Con este protocolo, aunque un intruso estuviera escuchando la comunicación y conociera las claves intercambiadas, no tendría la información suficiente para calcular la clave secreta. El secreto compartido se obtiene con el algoritmo mostrado en la figura 7.1.

Este protocolo se fundamenta en el problema del logaritmo discreto (PLD), que consiste en que dados los enteros a, b y p, se debe hallar un x tal que

$$a^x = b(mod\ p)$$

Si p es un número primo lo suficientemente grande, el PLD tiene una complejidad equivalente al problema de la factorización de números primos, aunque hay métodos para resolverlo en algunas circunstancias.

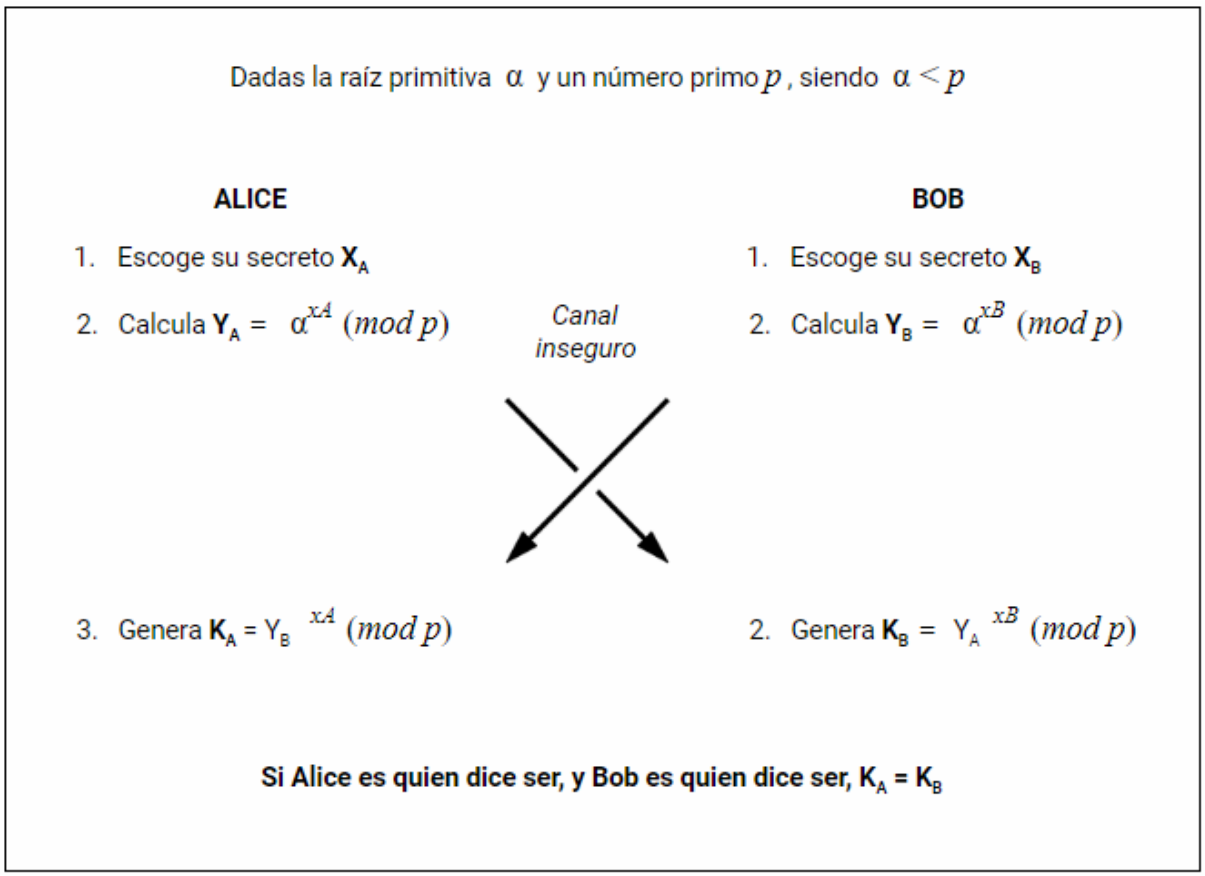


Figura 7.1: Establecer un secreto compartido con Diffie-Hellman

7.1.2. Elliptic Curve Diffie-Hellman

El Elliptic Curve Diffie-Hellman es una variante del algoritmo de Diffie-Hellman convencional. En esta versión, los parámetros necesarios para calcular la clave compartida son un número primo grande p, una curva elíptica E y un punto generador G perteneciente a esa curva (ver Figura 7.2)

La criptografía de Curva Elíptica (ECC, Elliptic Curve Cryptography) es un tipo de criptografía asimétrica que se basa en curvas elípticas. Las operaciones a realizar son más lentas que las utilizadas en los sistemas basados en factorización como el RSA, pero se puede obtener la misma seguridad con longitudes de clave mucho más cortas utilizando curvas elípticas, y en este caso podría resultar más rápido. La seguridad que proporcionan es equivalente a la de otros cifrados asimétricos. Los criptosistemas elípticos se fundamentan en la versión elíptica del PLD, conocido como el PLD elíptico.

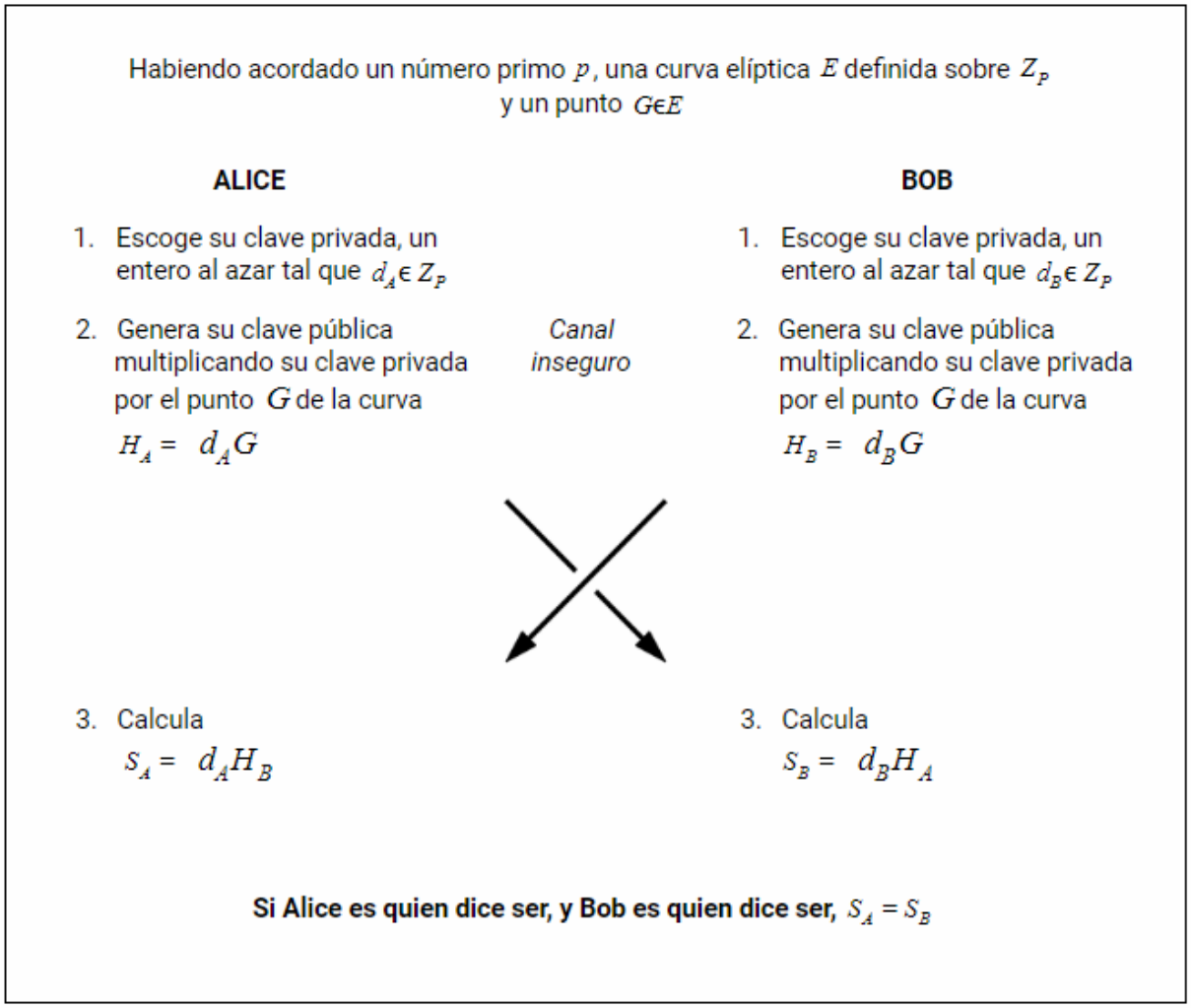


Figura 7.2: Establecer un secreto compartido con ECDH

7.1.3. Implementación

Cada vez que un usuario inicia sesión en la aplicación móvil, se genera un nuevo par de claves que será válido tanto tiempo como dure esa sesión. Se ha evitado utilizar claves fijas dado que cuanto mayor sea el tiempo que se esté utilizando una misma clave, mayor es el riesgo de que se comprometa.

Cuando una pareja paciente-cuidador desea vincular sus cuentas, ambos deben acceder a la tab de settings y pulsar ”Vincular cuenta”.

1. El primer paso de la vinculación de cuentas (ver Figura 7.1) consiste en que ambos escriban la dirección de correo electrónico de su pareja y pulsen siguiente.

2. Si las direcciones introducidas son válidas y pertenecen a una cuenta de su rol opuesto (no es posible vincular cuentas paciente-paciente o cuidador-cuidador), se muestra el paso 2. (7.2)

3. Uno de los dos miembros de la pareja deberá generar el código QR y el otro miembro deberá escanearlo con su teléfono móvil.

4. En caso de que todo haya sido realizado de forma correcta, las cuentas se vincularán.

Cuando el usuario pulsa el botón ”Siguiente” del primer paso, se recupera su clave privada, que está almacenada localmente en su teléfono, y la clave pública de la dirección de correo electrónico introducida de la base de datos. Antes de proceder a calcular el secreto compartido, primero será necesario codificar ambas claves (que se almacenan como texto) como objetos PublicKey y PrivateKEy, y entonces será posible calcular el secreto compartido con los métodos proporcionados por la librería de criptografía para Android SponyCastle (ver Figura 7.5).

El código QR se utiliza para la realización de una autenticacíon OOBA. El código que tiene que generar uno de los miembros de la pareja contiene la clave compartida que acaba de calcular codificada (ver Figura 7.4). Cuando su pareja lo escanea, lee ese secreto compartido y lo compara con la información que ha calculado anteriormente de forma independiente. Si ambos han calculado el mismo secreto, es que ambos son realmente quien dicen ser y se vincularán ambas cuentas.

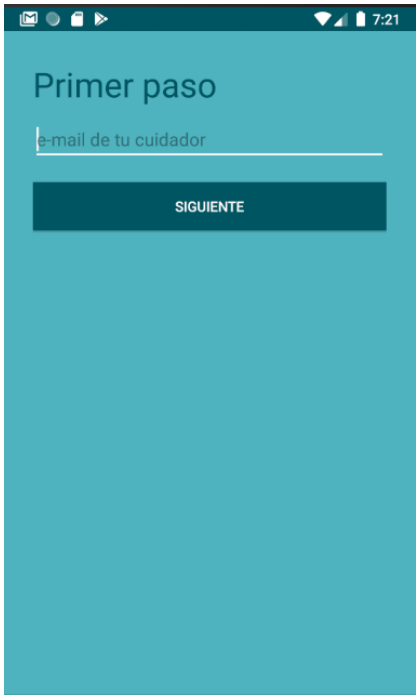


Figura 7.3: Paso 1

Figura 7.4: Paso 2

Figura 7.5: Código QR

```
    String privateKeyStringA = sharedPref.getString( key: "privateKey",  defValue: "no Key");
    String publicKeyStringB = document.getString( field: "publicKey");

    try {

        KeyFactory keyFact = KeyFactory.getInstance( algorithm: "ECDH",  provider: "SC");

        byte[] privateKeyABytes = Base64.decode(privateKeyStringA,Base64.DEFAULT);
        PKCS8EncodedKeySpec privateKeySpec = new PKCS8EncodedKeySpec(privateKeyABytes);
        PrivateKey privateKeyA = keyFact.generatePrivate(privateKeySpec);

        byte[] keyBytes = Base64.decode(publicKeyStringB,Base64.DEFAULT);
        X509EncodedKeySpec spec = new X509EncodedKeySpec(keyBytes);
        PublicKey publicKeyB = keyFact.generatePublic(spec);

        KeyAgreement aKeyAgree = KeyAgreement.getInstance( algorithm: "ECDH",  provider: "SC");
        aKeyAgree.init(privateKeyA);
        aKeyAgree.doPhase(publicKeyB,  lastPhase: true);

        MessageDigest hash = MessageDigest.getInstance( algorithm: "SHA1",  provider: "SC");
        key = new String(hash.digest(aKeyAgree.generateSecret()));

        findViewById(R.id.nextStepButton).setVisibility(View.GONE);
        findViewById(R.id.secondStepTitle).setVisibility(View.VISIBLE);
        findViewById(R.id.step2).setVisibility(View.VISIBLE);

    } catch (NoSuchAlgorithmException e) {
        e.printStackTrace();
    } catch (NoSuchProviderException e) {
        e.printStackTrace();
    } catch (InvalidKeyException e) {
        e.printStackTrace();
    } catch (InvalidKeySpecException e){
        e.printStackTrace();
    }
}
```

Figura 7.6: Cálculo del secreto compartido

7.2. Cloud Firestore de Firebase

Es posible controlar el acceso a los datos almacenados en Cloud Firestore, una de las dos bases de datos de Firebase, mediante reglas de seguridad.

En la documentación de Firebase se incluyen algunos casos de uso que podrían ser útiles dependiendo del tipo de aplicación a desarrollar, y de la fase del desarrollo en la que se esté. Durante la fase de desarrollo del proyecto se podría mantener la base de datos desprotegida, aunque no es recomendable. Por el contrario, durante esta fase se ha optado por restringir el acceso a personas no autenticadas y permitir el acceso a los datos a todos los usuarios que ya hayan iniciado sesión. Esta opción tampoco es recomendable para las releases que van a ser llevadas a producción, pero se puede mantener de esta forma durante la fase de desarrollo.

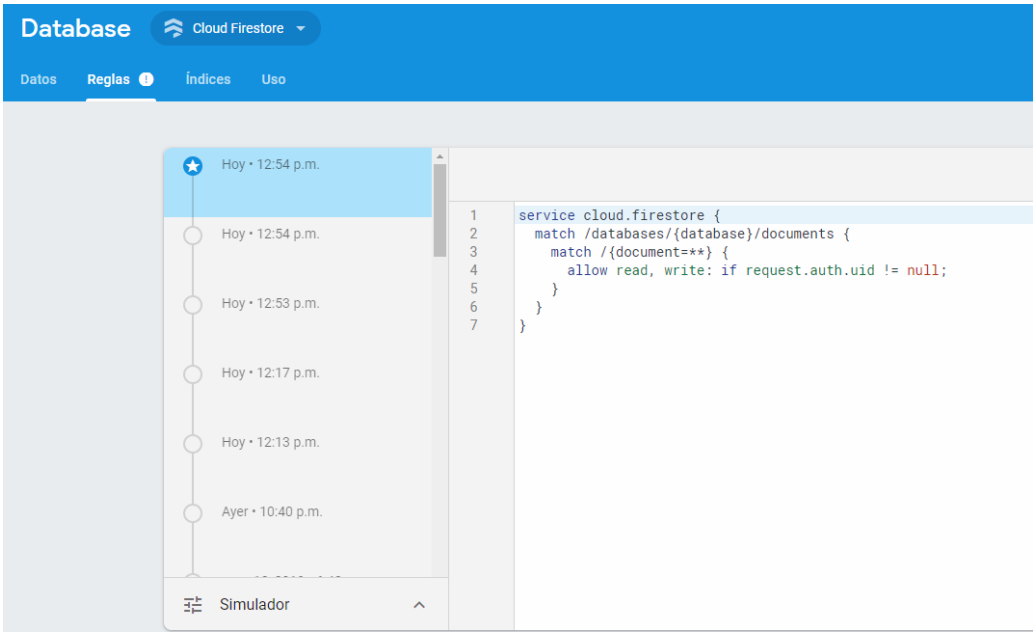


Figura 7.7: Reglas de seguridad de Cloud Firestore

Capítulo 8

Presupuesto

En este capítulo se desglosa brevemente un posible presupuesto del proyecto.

Descripción	Horas	Precio/Hora (€/h)	Total (€)
Planificación del proyecto	10	6	60
Lectura de bibliografía y documentación	45	6	270
Maquetado de la aplicación	45	15	650
Desarrollo	160	15	2400
Preparación de la memoria	40	10	400
Ordenador de sobremesa	-	-	1100
Smartphone Moto G5	-	-	170
Google Home mini	-	-	60
Total	300	-	5110

Tabla 8.1: Presupuesto del proyecto

Capítulo 9

Conclusiones y líneas futuras

El sistema desarrollado, compuesto por una aplicación móvil y una Action para Google Assistant cumple con los objetivos propuestos inicialmente para este Trabajo de Fin de Grado. Este sistema, desarrollado con la intención de introducir los altavoces inteligentes en el campo de la eHealth y así aprovechar su potencial en el mismo, tiene como objetivo facilitar la labor de las personas responsables de pacientes en hospitalización domiciliaria, y proporcionar una nueva herramienta de comunicación para dichos pacientes. Debido a que la forma principal de comunicación de los pacientes con el sistema es la voz, este proyecto puede ser especialmente trascendental para pacientes con problemas de movilidad como enfermos de Parkinson, esclerosis múltiple, artritis, o lesiones en la médula espinal entre otros, aunque podría ser utilizado por cualquier cuidador y paciente que lo necesite.

En cuanto a mi experiencia personal, considero que el desarrollo de este proyecto ha sido muy enriquecedor y que he aprendido una gran variedad de nuevas habilidades, ya que ha sido el primero de esta magnitud que realizado de forma individual –con la ayuda de mi tutora y cotutora– haciendo uso de muchas tecnologías con las que nunca había trabajado, como Actions on Google y DialogFlow, y profundizando en otras que ya había utilizado, como el desarrollo de aplicaciones móviles con Android Studio.

Este Trabajo de Fin de Grado será expuesto en la TLP Innova a celebrar el próximo mes de julio de 2019. Como el proyecto desarrollado es un prototipo, es decir, una primera versión, aun es posible añadir nuevas funcionalidades que aporten valor al mismo y seguir mejorándolo, tales como las siguientes:

- Crear un nuevo agente de conversación para el cuidador a través del cual pueda indicar al paciente que va a ir a visitarlo, que va a llegar tarde, programar los medicamentos utilizando su voz o consultar la actividad reciente de su paciente, todo mediante comandos de voz. De esta manera la aplicación móvil quedaría en un segundo plano y simplemente sería una herramienta de apoyo, y dejaría de ser el principal elemento de interacción con el sistema del cuidador.
- Añadir notificaciones push para los recordatorios de medicamentos.
- Añadir notificaciones push y sonido de alarma para las notificaciones de ayuda.
- Desarrollar un servicio web a través del cual poder monitorizar la actividad de todos los pacientes registrados en la app mediante una interfaz gráfica.

Capítulo 10

Conclusions and future lines

The developed system, which is made up of an Android Mobile Application and an Action for the Google Assistant has met the objectives defined during the first stage of the project. One of the main goals of the system is to introduce smart speaker devices into eHealth and fulfill their potential in this field. It will also provide a new tool for any person who is in charge of people hospitalized at home in order to make this task easier for them, while being an aditional way for the patients to communicate with their caregivers. This would be an especially usefull resource for patients who suffer mobility problems such as multiple sclerosis, arthritis, or spinal cord injuries amongst others, although it could be used by any caregiver and patient that could make a good use of it.

With regard to my personal experience, developing this project has been very enriching and I have I have learnt and grown many personal and technical skills, because this is the very first project of this magnitude by myself –along with my teachers– . I have been able to use some technologies that I wasn't familiar with, such as Actions on Google and DialogFlow, and to continue working on others that I am more confident with, such as Java and Android Studio.

This Final Degree project will be presented at the TLP Innova in June 2019. This project is a prototype or early release of a larger system. It can of course be improved by adding valuable features such as:

- Developing a new conversational agent that would be used by the caregiver so she/he can communicate with her/his patient using voice commands as well. One of those commands could be for telling the patient that she/he is going to be late, or asking the assistant about the recent events (emergencies or pills forgotten to be taken by the patient). In this way, the mobile app would be only a support tool and no longer be the main element for the caregivers to communicate with their patients.
- Set push notifications for pill reminders.
- Set push notifications and alarm sounds for the emergency notifications.
- Develop a web service with a graphic interface that shows the activity of the patients registered in the system, especially in the emergency events.

Bibliografía

[1] Speech Recognition,
https://www.britannica.com/technology/speech-recognition

[2] Analyst firm: Google Home gains ground on Amazon Echo, now 44M total devices sold,
https://searchengineland.com/analyst-firm-google-home-gains-ground-amazon-echo-44-million-total-units-sold-290544

[3] Smart Speaker Owners Agree That Questions, Music, and Weather are Killer Apps,
https://voicebot.ai/2019/03/12/smart-speaker-owners-agree-that-questions-music-and-weather-are-killer-apps-what-comes-next/

[4] How Alexa, Google and smart speakers are helping the disabled, elderly and depressed,
https://www.gearbrain.com/smart-speakers-and-mental-health-2609300358.html

[5] U.S. Smart Speaker Consumer Adoption Report 2019
https://voicebot.ai/smart-speaker-consumer-adoption-report-2019/

[6] Amazon Echo
https://www.amazon.es/Amazon-Echo-Dot-3-generacion-altoparlante-inteligente-Alexa/dp/B0792HCFTG

[7] Android N, Daydream VR, Google Home and more: Everything announced at Google I/O 2016
https://www.cnet.com/news/android-n-daydream-vr-google-home-and-more-everything-announced-at-google-io-2016/

[8] Shop Google Home
https://support.google.com/googlehome/answer/7072284?hl=en

[9] Shop Apple HomePod
https://www.apple.com/es/homepod/

[10] Shop Sonos One
https://www.sonos.com/es-es/shop/one.html

[11] Annual Smart Speaker IQ Test
https://loupventures.com/annual-smart-speaker-iq-test/

[12] *Environmental Control Systems For Disability Access*. (Inglés) [*Sistemas de control ambiental para la accesibilidad de personas son discapacidades*]. Tecsol.com.au, 2016

[13] Noda, Kenichiro. *Google Home: Smart Speaker as Environmental Control Unit*. (Inglés) [*Google Home: Altavoces inteligentes como unidades de control ambiental*]. Disability and Rehabilitation: Assistive Technology, vol. 13, no. 7, 2017, pp. 674–675., doi:10.1080/17483107.2017.1369589.

[14] Bickmore, T. W., Trinh, H., Olafsson, S., O'leary, T. K., Asadi, R., Rickles, N. M. y Cruz, R. *Patient and Consumer Safety Risks When Using Conversational Assistants for Medical Information: An Observational Study of Siri, Alexa, and Google Assistant*. (Inglés) [*Riesgos para pacientes y consumidores del uso de información médica proporcionada por asistentes conversacionales: Un estudio observacional de Siri, Alexa y Google Assistant*]. (Bickmore et al., 2018)

[15] The Future Of Voice AI In Patient Care
`https://www.forbes.com/sites/insights-intelai/2019/02/11/the-future-of-voice-ai-in-patient-care/`

[16] How to implement local fulfillment for Google Assistant actions using Dialogflow
`https://www.freecodecamp.org/news/how-to-implement-local-fulfillment-for-google-assistant-actions-using-dialogflow-1b3b3a13075f/`

[17] Medisafe App
`https://www.medisafeapp.com/?lang=es`

[18] Así es como Android se ha comido el mercado en diez años
`https://www.xatakamovil.com/sistemas-operativos/asi-como-android-se-ha-comido-mercado-diez-anos`

[19] Spongy Castle
`https://rtyley.github.io/spongycastle/`

[20] Zxing
`https://github.com/zxing/zxing`

[21] QRGenerator
`https://github.com/androidmads/QRGenerator`

[22] Guías del desarrollo de la marca para el acceso
`https://developers.google.com/identity/branding-guidelines?hl=es-419`

[23] Guías del desarrollo de la marca para el acceso
`New Directions in Cryptography`

[24] Whitfield Diffie y Martin Hellman. *New Directions in Cryptography* . (Inglés) [*Nuevas direcciones en criptografía*]. IEEE Transactions on Information Theory, vol. IT-22, NO. 6, Noviembre 1976