Máster en Ingeniería Industrial

Trabajo fin de máster

Aplicación móvil para migrantes, que facilita el contacto con organizaciones humanitarias que les pueden prestar ayuda

Autor
Nerea Zabala Orive
Director
Rafael Palacios

Madrid
2021

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título **APLICACIÓN MOVIL PARA MIGRANTES, QUE FACILITA EL CONTACTO CON ORGANIZACIONES HUMANITARIAS QUE LES PUEDEN PRESTAR AYUDA**

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el curso académico **2020/21** es de mi autoría, original e inédito y no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.

Fdo.: **NEREA ZABALA ORIVE** Fecha: 13/ 08/ 21

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: **RAFAEL PALACIOS** Fecha: 15/ Ago/ 2021

Máster en Ingeniería Industrial

Trabajo fin de máster

Aplicación móvil para migrantes, que facilita el contacto con organizaciones humanitarias que les pueden prestar ayuda

Autor
Nerea Zabala Orive
Director
Rafael Palacios

Madrid
2021

Resumen

Aplicación móvil para migrantes, que facilita el contacto con organizaciones humanitarias que les pueden prestar ayuda

Autor: Zabala Orive, Nerea.
Director: Palacios, Rafael.

Los flujos migratorios son un fenómeno común en todo el mundo, coexisten con la humanidad desde el principio de los tiempos, allí donde existen fronteras y sociedades distintas, hay personas que quieren cambiar su residencia de unos sitios a otros por infinidad de motivos. En la actualidad existe una tendencia al alza de este fenómeno, "Según las estimaciones de las Naciones Unidas, el número de migrantes internacionales a nivel mundial aumentó durante los últimos veinte años (entre 2000 y 2020), llegando a 281 millones en 2020" [1], por tanto, es un fenómeno al que se le debe de prestar atención de cerca. Uno de los principales problemas que tienen los migrantes mas vulnerables, es el difícil acceso a información dirigida a ellos, aunque existen numerosas asociaciones y empresas que disponen de servicios dirigidos a migrantes, el flujo de información entre estas entidades y los migrantes es difuso.

El abaratamiento y la fácil accesibilidad a las nuevas tecnologías, en concreto los *Smartphone* y la conexión a internet, brinda la oportunidad de desarrollar plataformas para mejorar el flujo de información entre los migrantes y las ONG´s. Por ello, este proyecto tiene como objetivo realizar una aplicación móvil mutilplataforma donde las ONGs puedan exponer información sobre sus servicios a los migrantes que los necesitan en su país destino.

Una aplicación multiplataforma es una aplicación que funciona en varios sistemas operativos, realizando un único desarrollo de código[2]. Para el desarrollo de este tipo de aplicaciones es necesario utilizar un framework especifico, que permita con un mismo código crear una aplicación para diferentes sistemas operativos. En este proyecto el framework utilizado es Flutter, un framework de código abierto, que permite crear aplicaciones nativas en los sistemas operativos iOS y Android. En las Figuras 6 y 7 se puede ver como queda tras el desarrollo la aplicación en un dispositivo con sistema operativo iOS y Android.

Figura 1: Resultado de la aplicación en un dispositivo iOS

Figura 2: Resultado de la aplicación en un dispositivo Android

La aplicación desarrollada, expone información sobre los servicios de las ONGs, de manera intuitiva, clara y en varios idiomas. Para ello, los servicios están englobados dentro de distintas categorías, facilitando así la búsqueda de información por parte de los usuarios. Estas categorías aparecen en la página de inicio de la aplicación, en forma de lista, cada una con su nombre, icono y color; al hacer click en cualquiera de las categorías, se pasa a otra pantalla, donde se muestra una lista desplazable con la información de cada servicio dividida en recuadros. En los recuadros se muestra la información disponible del servicio en español, francés y árabe. Además de la página de inicio, existe otra página principal, a la que se accede desde la barra inferior de la aplicación, esta muestra un mapa interactivo vía Google Maps, donde se puede ver la localización de los servicios en el mapa con chinchetas rojas y la localización del usuario.

La información sobre los servicios que se van a exponer en esta aplicación, es proporcionada y verificada por asociaciones humanitarias. Esta información se vuelca en una base de datos, para poder acceder a ella desde la aplicación. Como proveedor de base de datos, se decide usar Firebase, una plataforma para el desarrollo de aplicaciones en la nube; en concreto, se utiliza *Firestore Database* una base de datos NoSQL que permite almacenar datos en documentos y colecciones. Al usar una base de datos en la nube, la aplicación es fácilmente escalable, en número de usuarios y en número de servicios. Pero Firebase es mucho más que un servicio de base de datos en la nube, esta plataforma ofrece herramientas para el desarrollo y potenciación de las aplicaciones en el mercado [3]. La herramienta de Firebase más usada en este proyecto es *Analytics*, ya que permite visualizar estadísticas sobre el uso de la base de datos.

Una de las ventajas de esta aplicación, es que se ha programado y diseñado para poder tener una trazabilidad sobre ciertas características de los usuarios, y el uso que le dan a la aplicación. Es decir, al final de este proyecto, la asociación propietaria de la aplicación, será capaz de acceder a un Dashboard donde se encuentren métricas sobre: la localización geográfica de sus usuarios, el porcentaje de usuarios que accede a la aplicación desde los diferentes sistemas operativos y la cantidad de visitas que tienen las diferentes categorías.

Esta información ayuda a las asociaciones a entender mejor los flujos migratorios y analizar las tendencias de sus usuarios. Una vez realizado este mecanismo de recogida de datos para el proyecto, se ha procedido a realizar pruebas de uso de la aplicación, con usuarios cercanos que voluntariamente acceden a instalarse la aplicación en sus dispositivos móviles y usarla durante un tiempo. Aunque las métricas conseguidas con esta pruebas no dan información relevante sobre tendencias migratorias ya que sus usuarios no son migrantes, se ha conseguido verificar que la aplicación funciona correctamente y que los datos se recaban y se visualizan en el Dashboard de Firebase Analytics.

En las Figuras 8, 9 y 10 se pueden ver las métricas que aparecen en el Dashboard tras las pruebas de uso. La métrica sobre visualizaciones de pantalla en la Figura 8 refleja que servicios son más demandados y cuales menos, lo que puede ayudar a la asociación a saber que servicios son críticos y deberían reforzar. La métrica sobre localización geográfica, en la Figura 9, puede ayudar a las asociaciones a entender rasgos geográficos sobre sus usuarios, es decir de que países proceden o por que países pasan antes de llegar a España. Por ultimo, la métrica de la Figura 10 indica que tipo de sistema operativo se usa para acceder a la aplicación.

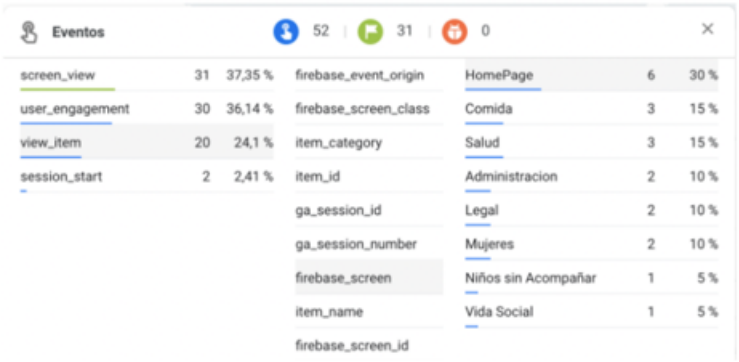


Figura 3: Resultados de las pruebas de uso de la aplicación sobre el uso de las diferentes pantallas

Figura 4: Resultados de las pruebas de uso de la aplicación sobre la localización geográfica de sus usuarios

Figura 5: Resultados de las pruebas de uso de la aplicación sobre el sistema operativo de los usuarios

Tras la realización de esta aplicación se puede concluir que los objetivos principales de esta proyecto se han completado satisfactoriamente. La aplicación se ha desarrollado correctamente, y funciona en ambos sistemas operativos iOS y Android. Además, la base de datos está conectada a la interfaz de usuario y se muestra la información de los servicios en la aplicación y su localización en el mapa. Por último, el mecanismo de recogida y visualización de datos extrae datos relevantes para las asociaciones humanitarias del uso de la aplicación. Tras la finalizar este proyecto se han preparado varios manuales; tanto para la subida de la aplicación por parte de la asociación a las plataformas oficiales de descarga, como para el uso

y administración de la aplicación. Con estos manuales la asociación debería de ser capaz de lanzar y administrar la aplicación ellos mismos.

Referencias

[1] Portal de datos mundiales sobre la migración. *Poblaciones de migrantes internacionales*. URL: https://migrationdataportal.org/es/themes/poblaciones-de-migrantes-internacionales.

[2] Kewal Shah, Harsh Sinha y Payal Mishra. «Analysis of Cross-Platform Mobile App Development Tools». En: *2019 IEEE 5th International Conference for Convergence in Technology (I2CT)*. 2019, págs. 1-7. DOI: 10.1109/I2CT45611.2019.9033872.

[3] Doug Stevenson. *What is Firebase? The complete story, abridged.* URL: https://medium.com/firebase-developers/what-is-firebase-the-complete-story-abridged-bcc730c5f2c0.

Abstract

Mobile application for migrants, that facilitates contact with humanitarian organizations that can help them.

Author: Zabala Orive, Nerea.
Director: Palacios, Rafael.

Migratory flows are a common phenomenon throughout the world, coexisting with humanity since the beginning of time, wherever borders and societies exist, there are people who want to change their residence from one place to another for countless reasons. Currently there is a growing trend in this phenomenon, "According to united nations estimatations, the number of international migrants worldwide increased during the last twenty years (between 2000 and 2020), reaching 281 million in 2020" [1], therefore, it is a phenomenon that should be closely looked at. One of the main problems that face the most vulnerable migrants today, is the difficulty of accessing information aimed at them, although there are numerous associations and companies that have services aimed at migrants, the flow of information between these entities and migrants is sometimes diffuse.

The cheapening and easy accessibility of new technologies, in particular *Smartphone* and internet connection, provides an opportunity to develop platforms that improve the flow of information between migrants and associations. Therefore, this project consists of crsting a mutilplatform mobile application where NGOs can present information about their services to migrants who need them in their destination country.

A cross-platform application is an application that works on multiple operating systems, only needing a single code development[2]. For the development of this type of applications it is necessary to use a specific framework. In this project the framework used is Flutter, an open source framework, which allows to create native applications on iOS and Android operating systems. In the Figures 6 and 7 the final application can be seen in an iOS and Android phone.

Figura 6: Results of the aplication in a iOs operating system

Figura 7: Results of the aplication in an Android operating system

The application developed, shows information about the services of NGOs, in an way that it is intuitive, clear and in several languages. To achieve this, the services are included in different categories, thus facilitating the search for information by users. These categories appear on the app's home page, in the form of a list, each with its own name, icon, and color; when you click on any of the categories, you go to another screen, where a scrollable list is displayed with the information of each service divided into boxes. The boxes display the available information of the service in Spanish, French and Arabic. In addition to the home page, there is another main page, which is accessed from the bottom bar of the application, this shows an interactive map via Google Maps, where you can see the location of the services on the map with red pins and the location of the user.

The information about the services in this application is provided and verified by humanitarian associations. This information is included into a database, so that it can be accessed from the application. As a database provider, it is decided to use Firebase, a platform for cloud application development; specifically, *Firestore Database* is used a NoSQL database that allows data to be stored in documents and collections. By using a cloud database, the application is easily scalable, in number of users and in number of services. But Firebase is much more than just a database service in the cloud, this platform offers tools for the development and enhancement of applications in the market [3]. The most commonly used Firebase tool in this project is *Analytics*, as it allows to visualize statistics about database usage.

One of the advantages of this application is that it has been programmed and designed to be able to have traceability on certain characteristics of the users, and the use they give to the application. At the end of this project, the association that owns the application will be able to access a Dashboard where metrics are found on: the geographical location of its users, the percentage of users who access the application from the different operating systems and the number of visits that the different categories have.

This information helps associations to better understand migration flows and analyse trends in their users. Once this data collection mechanism has been carried out, tests of application usage have been carried out, with nearby users that voluntarily agree to install the application on their mobile devices and use it for a while. Although the data obtained with this test do not give relevant information on migration trends since its users are not migrants, it has been possible to verify that the application works correctly and that the data is collected and displayed in the Firebase Analytics Dashboard.

In Figures 3, 4 and 5 the metrics that appear in the Dashboard after the tests of use can be seen. Figure 3 shows the screen visualizations of users, this metric can help us know which services are most demanded and which are less, thus helping the association to know which services are critical and need to be strengthened. The geographical location metric in Figure 4 can help associations understand geographical caracteristics about their users, i.e. which countries they come from or pass through before reaching thier final destination. Finally, the metric in Figure 5 indicates what type of operating system is used to access the application.

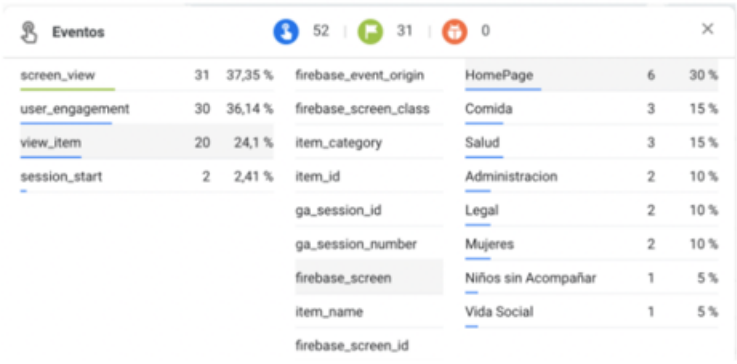


Figura 8: Results on screen viewing

Figura 9: Results on geografical location of users

Figura 10: Results on operating systems of users

After the completion of this proyect it can be concluded that the main objectives have been successfully completed. The app has been developed successfully, and works on both iOS and Andoid operating systems. In addition, the database is connected to the user interface and the information of the services and their location is displayed on the app. Finally, the data collection and visualization mechanism extracts relevant data for humanitarian associations from the use of the application. A number of manuals have been prepared following the completion of this project; both for the upload of the application by the association to the official download platforms, as well as for the use and administration of the application. With these manuals the association should be able to launch and manage the application for themselves.

Bibliography

[1] Portal de datos mundiales sobre la migración. *Poblaciones de migrantes internacionales*. URL: https://migrationdataportal.org/es/themes/poblaciones-de-migrantes-internacionales.

[2] Kewal Shah, Harsh Sinha y Payal Mishra. «Analysis of Cross-Platform Mobile App Development Tools». En: *2019 IEEE 5th International Conference for Convergence in Technology (I2CT)*. 2019, págs. 1-7. DOI: 10.1109/I2CT45611.2019.9033872.

[3] Doug Stevenson. *What is Firebase? The complete story, abridged.* URL: https://medium.com/firebase-developers/what-is-firebase-the-complete-story-abridged-bcc730c5f2c0.

Índice general

Índice de figuras

Capítulo 1

Introducción

1.1. Motivación

Los flujos migratorios son un fenómeno común en todo el mundo, coexisten con la humanidad desde el principio de los tiempos, allí donde existen fronteras y sociedades distintas, hay personas que quieren cambiar su residencia de unos sitios a otros por infinidad de motivos. En la actualidad existe una tendencia al alta de este fenómeno, "Según las estimaciones de las Naciones Unidas, el número de migrantes internacionales a nivel mundial aumentó durante los últimos veinte años (entre 2000 y 2020), llegando a 281 millones en 2020" [1], por tanto, es un fenómeno al que se le debe de prestar atención de cerca. Dentro de los diferentes perfiles de personas migrantes, existe un grupo de ellos que corre un riesgo importante al realizar su migración, suelen ser personas con pocos recursos, poca información y en situación de vulnerabilidad, que buscan cambiar de residencia para mejorar su calidad de vida.

Uno de los principales problemas que tienen los migrantes mas vulnerables hoy en día, es el difícil acceso a información dirigida a ellos, aunque existen numerosas asociaciones y empresas que disponen de servicios dirigidos a migrantes, el flujo de información entre estas entidades y los migrantes es a veces difuso.

El abaratamiento y la fácil accesibilidad de las nuevas tecnologías, en concreto los *Smartphone* y la conexión a internet, brinda la oportunidad de desarrollar plataformas para mejorar el flujo de información entre los migrantes y las asociaciones. Este proyecto pretende aprovechar estas tecnologías para mejorar el acceso a información de los migrantes, la solución concreta pasa por la realización de una aplicación móvil multiplataforma que exponga información sobre los servicios disponibles a los migrantes que lo necesiten.

1.2. Objetivos

Este proyecto tiene como objetivo la realización de una aplicación móvil multiplataforma, que exponga información sobre servicios de ONG´s disponibles dirigidos a migrantes, estos servicios estarán englobados dentro de distintas categorías para poder facilitar la búsqueda de información por parte de los usuarios, además, se expondrá un mapa interactivo donde se pueda ver la localización de los servicios y la localización del usuario.

El primer objetivo es realizar una interfaz de usuario multiplataforma, es decir que se pueda usar tanto desde un Android como iOS, además, la interfaz debe de ser responsiva, es decir que se adapte a las dimensiones de la pantalla del dispositivo; y accesible, que se ajuste la pantalla al tamaño de letra y que funcione con ayudas de accesibilidad como el sintetizador de voz. Como se ha comentado antes, esta interfaz debe tener un listado de servicios divididos en categorías y mostrar un mapa con la localización de todos ellos.

Durante este proyecto, se esta en estrecha colaboración con asociaciones de ayuda a migrantes en España, son ellos los que proporcionan la información sobre los servicios disponibles que se tendrán que volcar en la base de datos de esta aplicación y serán ellos los propietarios de la aplicación al finalizar el trabajo.

Además, tras el desarrollo de la aplicación se lleva a cabo un análisis de usabilidad de la aplicación, con ello se tratara estudiar que datos podrían ser beneficiosos para las organizaciones humanitarias a la hora de entender los flujos migratorios y tendencias entre los usuarios, para la mejora de sus servicios.

1.3. Metodología

De cara a la realización del proyecto, se dividió el trabajo en bloques, para poder llevar un seguimiento claro del progreso en cada etapa, además se establece una reunión semanal con el tutor de cara a resolver posibles problemas, exponer los avances y planear los futuros desarrollos. Los bloques de trabajo son los siguientes:

1. Estudio de Mercado
    a) Estudio de Aplicaciones parecidas
    b) Estudio de Tecnologías para aplicaciones multiplataforma
    c) Estudio de Hostings
2. Desarrollo

   a) Desarrollo de interfaz de usuario
   b) Desarrollo de base de datos
   c) Integración
   d) Subida de la aplicación en los mercados de aplicaciones

3. Análisis y Resultados

   a) Análisis de la Interfaz de usuario en distintos dispositivos
   b) Análisis de la Interfaz con integrada con la base de datos
   c) Análisis de usabilidad de la aplicación

4. Redacción

1.3.1. Cronograma

Para el seguimiento del proyecto se realiza un cronograma con las tareas principales, el cronograma se puede ver en la Figura 1.1.

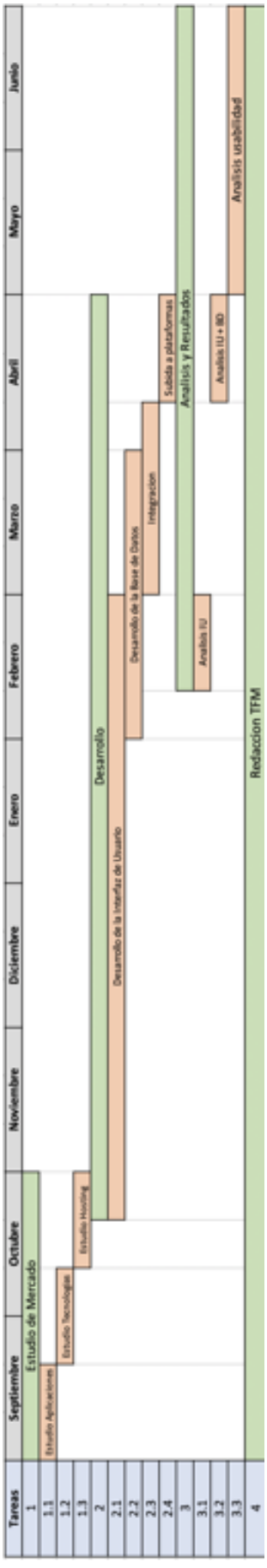


Figura 1.1: Cronograma

Capítulo 2

Estado de la cuestión

En esta sección del documento, se tratar el estado del arte de este proyecto, donde se exponen las tecnologías ya existentes para el desarrollo de aplicaciones móviles generales y multiplataforma, se estudia los tipos de hostings disponibles y se comentan otras aplicaciones que existen actualmente para la ayuda a migrantes.

2.1. Tecnología Móvil

La tecnología móvil, hace referencia a un conjunto de tecnologías que permiten la comunicación entre dos o mas dispositivos, sin estar físicamente conectados.

2.1.1. Redes de comunicación

Las redes de comunicación son una parte esencial de la tecnología móvil, es el canal que permite la comunicación entre distintos dispositivos. Esta tecnología ha ido evolucionando con los años y se pueden distinguir cinco generaciones:

- **Primera Generación 1G:** Se transmite información por señales FM, lo que hace posible comunicar voz entre dispositivos, es un sistema de baja velocidad y baja capacidad de transmisión. [2]
- **Segunda Generación 2G/GSM:** Se dio a principios de los años 90, esta generación proporciona mejor calidad en la comunicación por voz, es la primera generación con la que fue posible mandar SMS, además, se incorporó una mejora que dio capacidad de enviar imágenes y navegar por internet gracias al GPRS (General Packet Radio Service). [2]
- **Tercera Generación 3G:** La tercera generación trae consigo mejor calidad de voz y mas velocidad, lo que permite el envío y recepción de archivos

multimedia desde el dispositivo móvil, y acceso a internet con el protocolo TCP/IP [2]

- **Cuarta Generación 4G:** Con la cuarta generación, la velocidad de datos se hace significativamente mejor, lo que habilita la posibilidad de consumir contenido multimedia desde el dispositivo móvil.[2]

- **Quinta Generación 5G:** Esta generación, aun por evolucionar, proporciona comunicaciones de ultra baja latencia y alta fiabilidad, además permite incrementar drásticamente el numero de dispositivos conectados a la red al mismo tiempo.

2.1.2. Dispositivos

Los dispositivos móviles no están únicamente relacionados con la telefonía, sino que existen otro tipo de dispositivos englobados en este termino, las características generales de estos dispositivos son: que tienen capacidad de procesamiento, que son pequeños o transportables y que pueden tener conexión permanente o intermitente a la red [3].

El termino *Smartphone* hace referencia a un tipo especifico de teléfonos móviles con pantalla táctil, caracterizados porque sus usuarios a través del sistema operativo consiguen conectarse a internet y descargarse aplicaciones que dan la posibilidad de tener multitud de funcionalidades adicionales.

2.1.3. Sistemas Operativos

Los sistemas operativos (SO) son un conjunto de programas de software que gestionan el funcionamiento del hardware, y proveen servicios a otros programas que quieran acceder al hardware del dispositivo [4]. Su función es; abstraer la complejidad del hardware a otros programas y gestionar la capacidad computacional del dispositivo. Existen dos sistemas operativos principales en el mercado de los teléfonos móviles, iOS y Android. iOS está dedicado a móviles de la marca Apple inc. y no puede instalarse en otros dispositivos, mientras que el SO Android es libre y lo puede utilizar cualquier fabricante de dispositivos móviles adaptando los drivers al dispositivo en concreto.

Como se puede ver en la Figura 2.1, actualmente el mercado de móviles esta dominado por los sistemas operativos Android e iOS, siendo Android el mayor de los dos con el 86,1 % del mercado en 2019. Como se puede ver en la Figura 2.1 , el mercado ha pasado de tener fragmentación entre competidores, como BlackBerry

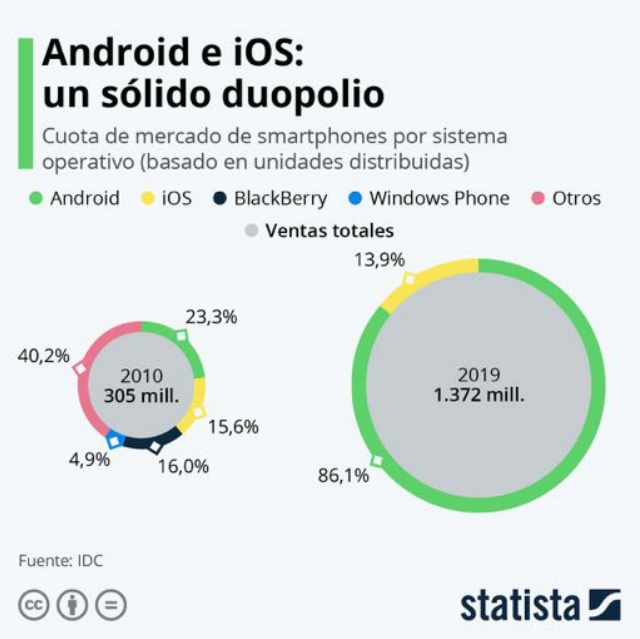


Figura 2.1: Gráfico del % del mercado que tiene iOS y Android en 2019 y en 2010 [5]

o Windows Phone a estar completamente dividido entre Android e iOS dando pie a una especie de duopolio.

- **iOS** El sistema operativo iOS salió al mercado en el 2007 a la vez que el iPhone [6], y se ha ido adaptando a las nuevas tecnologías que han ido adquiriendo los móviles de Apple inc., como el Touch Id o el reconocimiento de voz, entre otros. Es un sistema operativo dedicado a los equipos de Apple, por tanto, ningún tercero puede utilizarlos. El sistema operativo iOS esta basado en Unix y su arquitectura consta de cuatro capas que se pueden ver en la Figura 2.2

  - **Capa de Cocoa Touch:** Es la capa que habla directamente con las aplicaciones, es decir, es el framework utilizado para el desarrollo de aplicaciones en iOS, el lenguaje nativo de este entorno es Objective-C .
  - **Capa media:** Esta capa provee la tecnología de gráficos a las aplicaciones desarrolladas en este sistema operativo.
  - **Capa Core Services:** Esta capa contiene los servicios principales del sistema operativo, que luego son usados por las aplicaciones, como por ejemplo, el almacenamiento en iCloud.

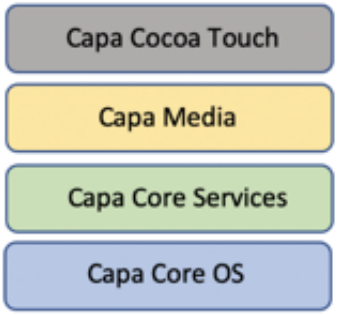


Figura 2.2: Capas de la arquitectura de iOS

- **Capa Core OS:** Esta es la capa de bajo nivel, esta situada directamente encima del hardware y utiliza el sistema operativo Darwin BSD. Esta capa Engloba al Kernel de los móviles y por tanto realiza la gestión de memoria, procesos e hilos, entre otros.[7]

En el SO iOS la única manera de instalarse las aplicaciones es a través de su propia aplicación llamada AppStore ya que IOs solos ejecuta aplicaciones firmadas digitalmente por Apple. Antes de ofertar en su plataforma cualquier aplicación Apple realiza un chequeo de seguridad, privacidad y usabilidad. Con este proceso Apple minimiza el riesgo de que haya desarrolladores creando aplicaciones con actividad maliciosa, además, se asegura de que todas las aplicaciones ofertadas en su plataforma cumplen con un estándar mínimo de calidad. [8]

- **Android**, a diferencia de iOS, es un SO de código abierto y puede ser usado en cualquier dispositivo móvil. Android inc. fue comprado por Google en 2005 y en 2008 se lanzó al mercado una primera versión de su sistema operativo. El SO Android esta basado en Unix y a diferencia de este competidor tiene una arquitectura que permite ser usado en cualquier hardware, independientemente del modelo y marca, en la Figura 2.3 se pueden ver las capas de la arquitectura Android

  - **Aplicaciones:** En esta capa residen todas las aplicaciones del sistema operativo, y las que el usuario final utiliza.
  - **Framework de Aplicaciones:** En esta capa intermedia, están las librerías de java utilizadas por los desarrolladores a la hora de generar alguna funcionalidad, estas librerías acceden a los recursos mas primitivos a través de la maquina Dalvik que se explica en la siguiente capa.

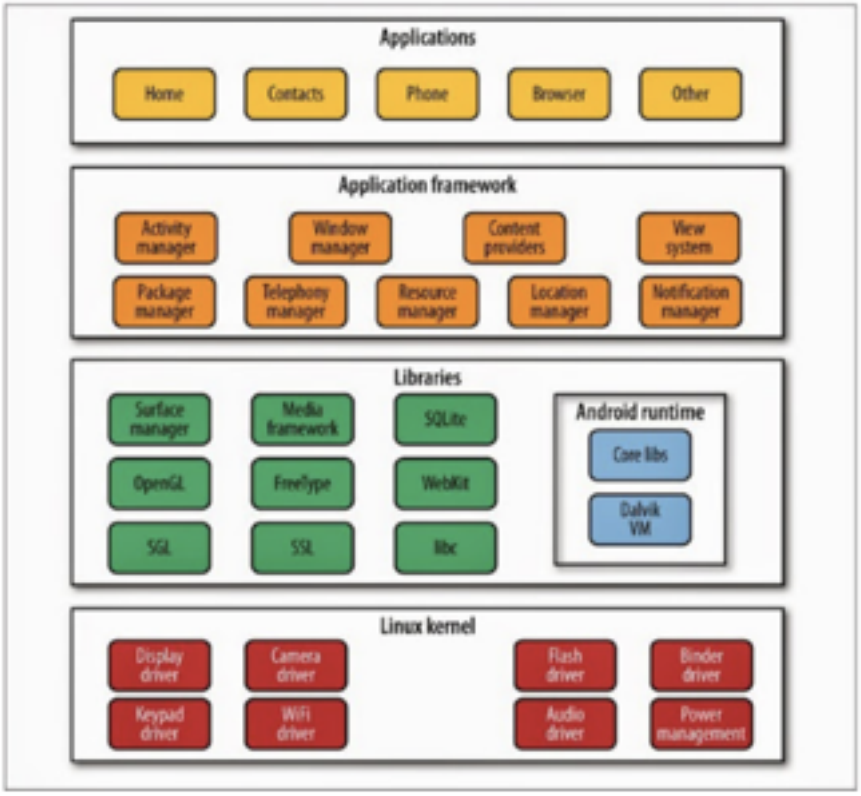


Figura 2.3: capas de la arquitectura Android [9]

- **Runtime:** Esta capa no es una capa en si, ya que utiliza librerías de la capa en la que se encuentra, en esta subcapa se instancia una maquina virtual Dalvik para cada aplicación que se este utilizando.
- **Librerías:** Estas librerías proporcionan las funcionalidades que se suelen repetir con frecuencia en las aplicaciones. Están implementadas en C/C++.
- **Kernel:** Esta capa es la de más bajo nivel, sirve para abstraer el hardware a las aplicaciones. Está basado en Linux y permite tener los servicios más básicos.

A diferencia de su competidor, en este sistema operativo las aplicaciones se pueden descargar tanto en Google play, que es el sitio oficial de aplicaciones, como desde cualquier pagina web. Esto crea más inseguridad ya que las aplicaciones descargadas por paginas web no pasan ningún tipo de validación ni registro. [8]

2.2. Aplicaciones Móviles

Las aplicaciones móviles (App´s) son aplicaciones informáticas, específicamente diseñadas para su uso en *smartphones*, tabletas u otros dispositivos inteligentes [10]. Estas aplicaciones usan liberarías expuestas por el sistema operativo para dar servicios al usuario; de ocio, de información, o de reservas entre muchos otros. En un principio, las aplicaciones del teléfono venían predefinidas por el sistema operativo que usase el teléfono móvil y los usuarios no podían acceder a otro tipo de aplicaciones. En el año 2007 con la llegada del primer iPhone Apple lanza App Store, una plataforma donde los desarrolladores pueden ofrecer aplicaciones a los usuarios finales, poco después otros competidores realizan la misma estrategia y así se "liberalizo" el mercado de las aplicaciones móviles, llegando a ser el que se conoce hoy en día.

2.2.1. Categorías de aplicaciones móviles

- **Aplicaciones Web (Web App):**

  Una Aplicación web o Web App es una pagina web diseñada de manera fluida para que sea responsiva al formato del dispositivo en el que se esta utilizando [11], es decir, se puede visualizar correctamente tanto desde un ordenador como desde una Tablet o smartphone. A diferencia de las aplicaciones móviles (de las que se hablará a continuación), se accede a ellas desde el navegador, por tanto, es imprescindible tener acceso a internet para su uso. Se basan en lenguajes como HTML, CSS o JavaScript. Debido a que estas aplicaciones sólo se pueden usar desde los navegadores web, no están tan bien diseñadas para un hardware especifico, pero tienen una ventaja importante, que se pueden actualizar a nivel servidor, y por tanto, si sucede algún fallo en el sistema, o se quiere actualizar el software, solo habría que solucionarlo en el servidor central y con ello todos los clientes tendrán acceso a la nueva versión.

- **Aplicaciones móviles (Mobile App):**

  Estas aplicaciones, a diferencia de las anteriores, tienen que ser descargadas, normalmente desde la tienda de aplicaciones del dispositivo. Las APP´s tienen la capacidad de funcionar sin acceso a internet. Suelen ser mas rápidas que las aplicaciones web y pueden tener servicios mas avanzados, ya que se programan sobre el sistema operativo del móvil y tienen acceso directo al hardware. Dentro de esta categoría existen dos grupos, las aplicaciones nativas, creadas únicamente para un tipo de sistema operativo, y las multiplataforma que consiguen que un mismo código sea válido para mas de un

sistema operativo.[12]

- **Nativa:** Estas aplicaciones se desarrollan para un sistema operativo en concreto, por tanto, se desarrollan en el lenguaje que habilita el framework de cada SO, en el caso de Apple es Objective-C y en el caso de Android es Java, para su programación se utilizan las librerías de bajo nivel de cada SO. Estas aplicaciones suelen proporcionar una mejor experiencia de usuario al estar específicamente diseñadas para su SO.
- **Multiplataforma:** Las aplicaciones móviles multiplataforma son exportables a distintos sistemas operativos con un mismo desarrollo [13], este tipo de aplicaciones abaratan el coste de desarrollo, ya que no se tienen que programar dos componentes distintos para cada sistema operativo, sino que con un mismo código se consigue su uso en varios SO.

2.2.2. Tecnologías para el desarrollo de la interfaz de usuario

Para el desarrollo de aplicaciones multiplataforma, es necesario usar tecnologías especificas para ello, estas plataformas permiten programar un solo código y que este se traduzca a códigos nativos, generalmente de iOS y Android. Son tecnologías relativamente nuevas, y están saliendo al mercado cada vez mas porque permiten abaratar el coste de desarrollo de las aplicaciones móviles.

- **Flutter:**

  Es un Kit de herramientas UI que permite programar una aplicación únicamente en Flutter y con ello generar aplicaciones compiladas nativamente en iOS, Android y web, este framework de Google ha crecido exponencialmente en popularidad desde que salió al mercando en 2017 [14].

  Dart, es el lenguaje de programación utilizado en este framework, este lenguaje desarrollado por Google, es orientado a objetos y con compilación anticipada. Su curva de aprendizaje es relativamente baja, si ya se conocen lenguajes comunes orientados a objetos como, C++ o Java. Este Framework, a diferencia de algunos de sus competidores, no necesita un bridge o proxy para comunicarse con los módulos nativos, ya que la mayoría de ellos están dentro del mismo framework [15]. En la Figura 2.4 se puede ver un esquema de la arquitectura de esta plataforma, como se puede leer no existen bridges ni proxys.

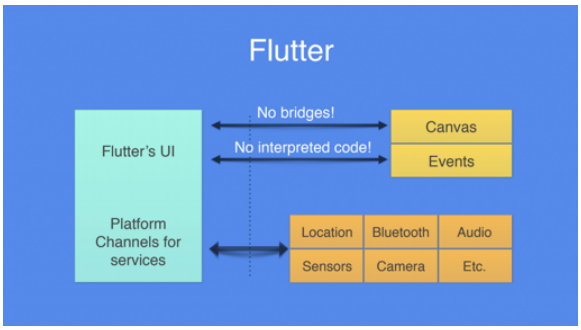


Figura 2.4: Arquitectura Flutter [15]

- **React Native:**

  React Native es un framework para el desarrollo de aplicaciones multiplataforma desarrollada por Facebook internamente desde el año 2015, permite generar aplicaciones con un único código base para plataformas iOS y Android [16].

  JavaScript es el lenguaje de programación utilizado en esta plataforma, es el lenguaje mas conocido para el desarrollo web del lado del cliente, es orientado a objetos e interpretado. La arquitectura de este framework hace necesario el uso de JavaScript Bridge para comunicarse con los módulos nativos[17]. En la Figura 2.5 se visualiza el esquema de la arquitectura de este framework, se puede ver como a diferencia de Flutter, esta si que posee proxies y Bridges para comunicarse entre módulos.

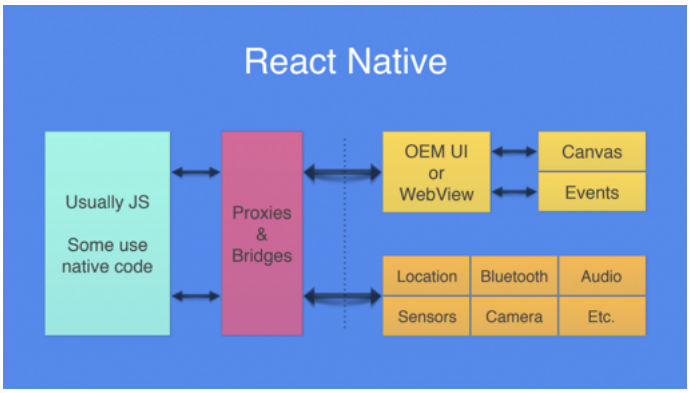


Figura 2.5: Arquitectura React Native [15]

- **Apace Cordova:**

En particular, este framework habilita el uso de tecnologías web como HTML5, CSS3 o JavaScript para diseñar aplicaciones web que podrán ser tratadas como aplicaciones nativas. Además, Apache Cordova es compatible con otras herramientas, en particular Angular e Ionic.

En la Figura 2.6, se ve la arquitectura de este framework, las aplicaciones web creadas se comunican vía APIs con el Webview, esto es lo que permite traducir las tecnologías web utilizadas, en funcionalidades para aplicaciones nativas, además también acceden a las librerías de Cordova vía ese mismo WebView.[18]

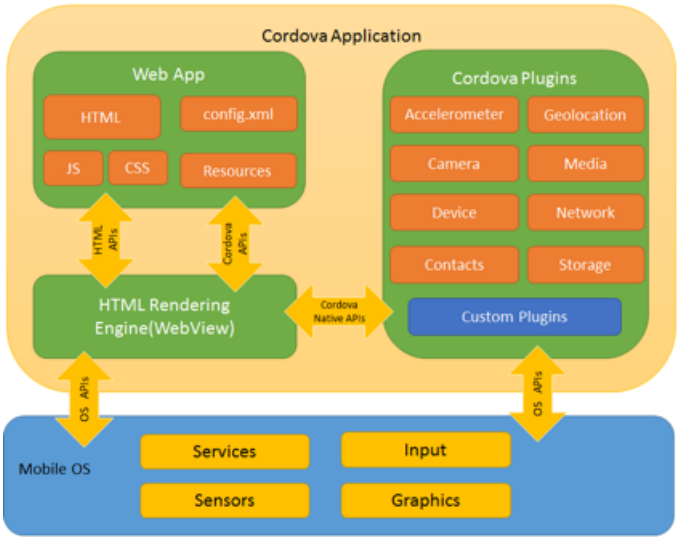


Figura 2.6: Arquitectura Apache Cordova [19]

- **Xamarin:**

  Xamarin es una plataforma para crear aplicaciones nativas múltiplataforma con un mismo código base, esta plataforma fue comprada por Microsoft en el año 2016. Se programa en C# un lenguaje de programación muy utilizado para tecnologías .NET y orientado a objetos. MVVM es lo que hace a Xamarin distinguirse de otras tecnologías parecidas, este modelo de arquitectura permite separa al máximo la Interfaz de Usuario de la lógica del programa.

- **Ionic:**

  Ionic permite crear aplicaciones multiplataforma, utiliza tecnologías muy conocidas en el desarrollo web como AngularJS para la estructura de las apli-

caciones o Cordova plugins para el acceso a múltiples funciones del sistema operativo. Este framework permite programar aplicaciones en CSS y JavaScript.

Para terminar con el análisis de las tecnologías existentes para crear aplicaciones móviles multiplataforma, cabe destacar que en la Figura 2.7, se expone la tendencia de búsqueda de las diferentes tecnologías de aplicaciones multiplataforma en el buscador de Google. Como se puede ver la más buscada a lo largo del año 2018 y 2019 ha sido React Native, pero Flutter tiene una clara tendencia ascendente durante los últimos años y sobrepasa a React Native desde principios del 2020. Esto nos hace pensar que Flutter es una tecnología nueva pero que está ganando interés entre el publico general.

Figura 2.7: Gráfico de búsquedas en Google de cada tecnología multiplataforma

2.3. Alojamiento de la aplicación

En esta sección se va a exponer las tecnologías existentes de bases de datos y plataformas para el alojamiento hosting de datos de aplicaciones.

2.3.1. Bases de datos

Las bases de datos son un conjunto de información, perteneciente al mismo contexto y ordenada de manera sistemática [**ECDB**], los tipos de bases de datos que se van a tratar en esta sección son las digitales, estas son necesarias a la hora

de almacenar información digital de una manera ordenada para su posterior uso. [20]

Para empezar, conviene distinguir entre los diferentes tipos de bases de datos que existen, la primera distinción que se puede realizar es por la flexibilidad de modificación de las bases de datos y existen dos tipos:

- **Estáticas:** En estas bases de datos la información no puede ser modificada, únicamente leída.
- **Dinámicas:** En este tipo de base de datos la información almacenada puede ser modificada en cualquier momento.

La segunda distinción que se puede realizar en torno a las bases de datos es según su forma de organización

- **Jerárquicas:** Estas son organizadas según importancia, y su estructura es de relación padre e hijo, cada hijo puede tener como máximo un padre, pero cada padre puede tener varios hijos.
- **De red:** Son bases de datos en las que los registros están relacionadas entre si creando una red.
- **Relacionales:** Estas bases de datos son de estructura relacional, almacenan los registros en forma de tablas que, a su vez, pueden estar relacionadas entre ellas. Es el tipo de base de datos mas usado en la actualidad.
- **Deductivas:** Son sistemas capaces de deducir información adicional con los datos almacenados en ellas.
- **Multidimensionales:** Son bases de datos que alojan la información en tablas, pero a diferencia de las relacionales, estas tablas tienen forma de cubo.

2.3.2. Proveedores de bases de datos en la nube

Las plataformas que alquilan recursos computacionales en la nube se han hecho un parte vital de los desarrollos tecnológicos, en ellas se tienen numerosos servicios distintos y entre ellos están las bases de datos.

- **Google Cloud Plataform:**

  Google Cloud Plataform es una plataforma que ofrece multitud de servicios en la nube, son servicios modulares que se pueden ir incorporando poco a

poco y tiene diferentes tarifas en función del uso y características del servicio escogido.

Para bases de datos en concreto, Google Cloud ofrece varias opciones donde elegir, se ofrecen servicios de bases de datos SQL como *Google Cloud SQL* o *Cloud Big Table* y bases de datos NoSQL como *Mongo DB*. [21]

- **Firebase:**

  Firebase es una plataforma de Google que proporciona servicios para el desarrollo de aplicaciones en la nube, en ella se pueden encontrar; bases de datos, sistemas de autenticación de usuarios y sistemas de gestión de anuncios entre otros. Esta plataforma ayuda a los desarrolladores a crear y mantener aplicaciones y a retener a los usuarios [22]. Este servicio esta ligado a Google Cloud Plataform al ser de la misma compañía, la diferencia reside en que Firebase está exclusivamente diseñada para el desarrollo de aplicaciones móviles, mientras que Google Cloud Plataform tiene una cartera mas amplia de servicio.

  Este servicio ofrece dos tipos de bases de datos, el primero es *Cloud Firestore* y el segundo *Realtime Database*, ambos tipos son bases de datos NoSQL. La base de datos *Realtime Database* es la más antigua de las dos, siendo la base de datos original de Firebase, los datos se almacenan en formato JSON creando un árbol con la jerarquía de los datos. La base de datos *Cloud Firestore* es la ultima versión de Firebase y es una mejora de la anterior, la diferencia reside en que los datos se almacenan en documentos y colecciones, en lugar del arbol de JSON´s, para implantar jerarquías en este modelo se crean subcolecciones dentro de las colecciones, esto genera una mejora con respecto al tipo de base de datos anterior, ya que es mas fácil e intuitiva cuando la jerarquía de los datos es compleja [23].

- **Amazon Web Services:**

  Al igual que los servicios anteriores, Amazon Web services es una plataforma de computación en la nube, que ofrece servicios modulares como bases de datos, contenedores o APIs.

  Ofrece un total de quince bases de datos distintas, con estructuras y modelos personalizables, las más utilizadas son *DynamoDB* como base de datos NoSQL y *RDS* como base de datos relacional. [24]

2.4. Mapas Interactivos para aplicaciones

Cada vez es más común que las aplicaciones móviles tengan servicios de otros proveedores, esto suele ocurrir con los mapas interactivos que se añaden a las APPs, crear y actualizar un mapa de una región es un trabajo complicado; hay muchas información a tener en cuenta y es crucial tener la información actualizada en este tipo de desarrollos. Por todo esto, lo mas común es utilizar servicios de terceros, en el mercado existen varios y a continuación se van a mencionar los mas relevantes.

- **Google Maps**

  Google Maps es un plataforma de la empresa Google donde se pueden consultar mapas de todo el mundo, servicios e información sobre rutas. Esta plataforma fue lanzada por primera vez en 2005 [25] y haciendo mejoras y extensiones se ha convertido en el servicio que es ahora. En la Figura 2.8 se puede ver como es el aspecto de Google Maps.

Figura 2.8: Pantallazo de Google maps

  Para poder incluir mapas de este servicio en una aplicación, es necesario recurrir a Google Maps API, un servicio perteneciente a Google Cloud Plataform, que permite crear claves API para la conexion de servicios externos con Google Maps.

- **OpenStreetMap**

Este servicio es un proyecto de código abierto y colaborativo, en el que se crean mapas de todo el mundo y donde cualquiera puede ayudar a subir información sobre ellos, la licencia es libre y por tanto es gratis de utilizar. En la Figura 2.9 se puede ver el aspecto de los mapas en este servicio.

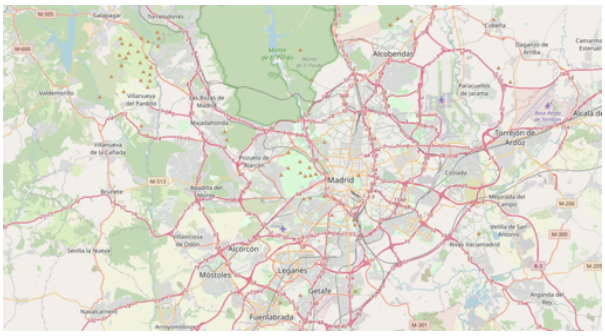


Figura 2.9: Pantallazo de Open Street View

Para poder incluir este servicio en aplicaciones móviles, solo hace falta incluir las librerías dedicadas a ello en el código, al no tener ningún tipo de facturación por su uso, no es necesario tener una clave API independiente.

- **MapBox**

MapBox es un servicio que realiza mapas por encargo, al igual que el anterior es de código abierto y su ventaja radica en que es altamente personalizable, con esta plataforma se puede realizar mapas que se adapten a tus necesidades. En la Figura 2.10 se puede ver un ejemplo de mapa creado con MapBox.

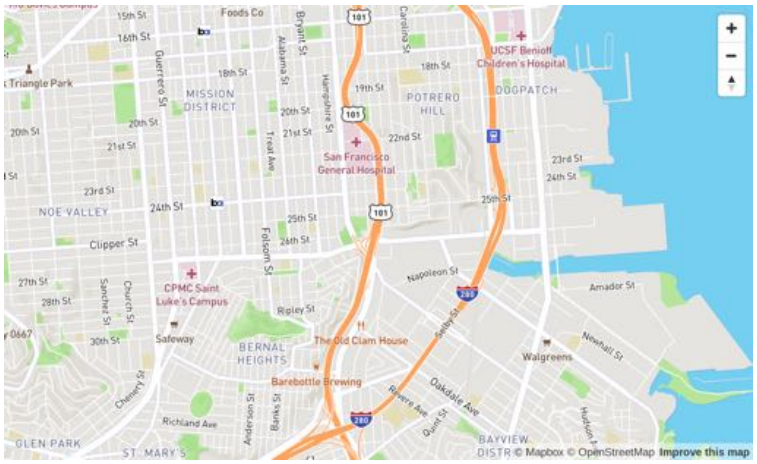


Figura 2.10: Pantallazo de MapBox

Para incluir estos mapas a la aplicación es necesario tener un token, que es en un principio gratuito, pero luego hay servicios de pago dentro de la plataforma.

2.5. Soluciones existentes. Aplicaciones de ayuda a migrantes

En la actualidad existen algunas aplicaciones móviles, con el objetivo de ayudar a los migrantes, entre ellas existe un rango amplio de servicios y funcionalidades ofrecidos, en esta sección se van a estudiar las más relevantes para este proyecto.

- **OASI on the street:**

  Esta aplicación nace de la mano de *associazione mosaico*, una asociación italiana de ayuda a migrantes, fue lanzada en el año 2018 tras los trágicos eventos que ocurrieron en los Alpes con los migrantes que trataban de cruzar desde Italia a Francia. Esta aplicación trata de ayudar a los migrantes a informarse y pedir ayuda, para poder sobrellevar momentos difíciles con apoyo e información.[26]

  Es una aplicación hecha exclusivamente por refugiados, la interfaz de usuario de esta aplicación contiene un mapa de la ciudad de Turín, donde indica los servicios que pueden ser interesantes para los migrantes, los servicios van categorizados por colores para que les sea más fácil a los usuarios distinguirlos en el mapa. Esta aplicación únicamente esta disponible para usuarios de sistema operativo Android. [26]

Figura 2.11: Pantallas que componen OASI on the street

En la Figura 2.11 se pueden ver las pantallas que componen esta aplicación, la pantalla principal consta de un mapa que incluye chinchetas de colores para diferentes tipos de servicios.

- **MigrAdvisor:**

  Esta aplicación fue desarrollada por Caritas Italia en colaboración con la embajada EEUU en Italia [27]. Esta APP no esta únicamente diseñada para migrantes sino que trata de abarcar un publico más amplio, siendo este cualquiera que esté en un situación de vulnerabilidad o que necesite ayuda para la integración.

  Utilizando la localización del usuario se identifican servicios cercanos como embajadas, centros de Caritas, centros médicos y oficinas de policía entre otros, además tiene un apartado dedicado a avisar sobre abusos, explotación o situaciones irregulares en distintos sitios. Esta aplicación está disponible para iOS y Android, pero el único idioma que soporta es Italiano[28].

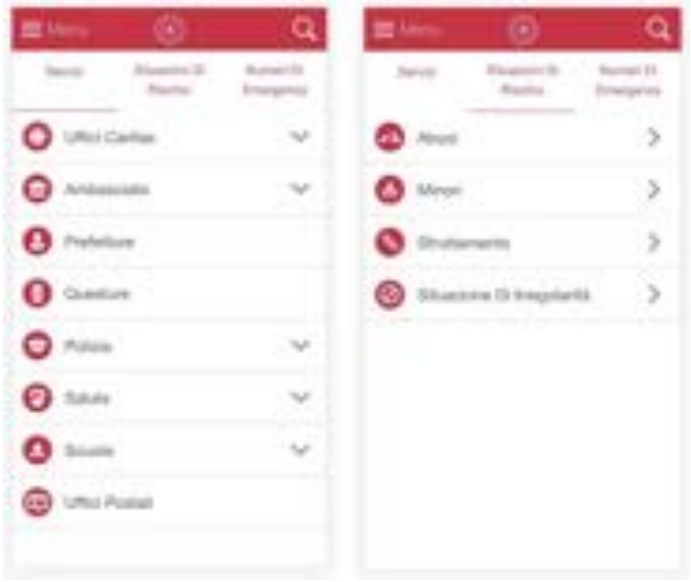


Figura 2.12: Pantallas que componen MigrAdvisor

En la Figura 2.12 se pueden ver la pantallas de la APP, la forma de navegar a través de la aplicación es con la barra superior que contiene una pestaña para servicios, otra para situaciones de riesgo y la última para números de emergencia.

- **MigAPP:**

  Esta aplicación provee información a los migrantes sobre servicios enfocados a ellos, informa sobre servicios de cambio de moneda y financiación, y alerta con notificaciones sobre eventos relevantes para su publico, además es un sitio donde leer historias de otros migrantes. [29] La aplicación viene en numerosos idiomas como Arabe, chino, frances español o portugués entre otros. Esta aplicación desarrollada por OIM (Organización Internacional para las Migraciones), esta disponible para dispositivos tanto Android como iOS.

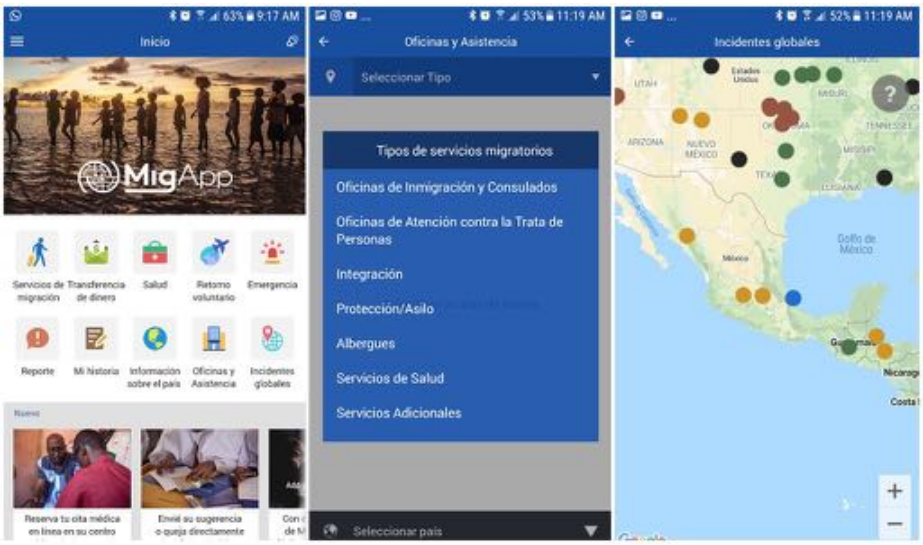


Figura 2.13: Pantallas que componen MigAPP

- **RefAid:**

  Esta aplicación quiere acercar los servicios cercanos disponibles a los migrantes, para ello muestra una lista de servicios englobados en ciertas categorías, además provee de un mapa para situar esos servicios y la ubicación del usuario. Esta aplicación esta disponible tanto para Android como para iOS. Esta plataforma quiere universalizar todos los servicios de migraciones en una misma aplicación, haciendo así posible generar un punto de encuentro entre los migrantes y las asociaciones.

Figura 2.14: Pantallas que componen REFAID

En la Figura 2.14 se puede ver el aspecto que tiene esta aplicación, parece una aplicación fácil de usar y muy guiada por colores, además se tiene tanto pantallas con diferentes servicios, como el mapa.

Capítulo 3

Desarrollo y Tecnologías

En esta sección del proyecto se exponen las tecnologías utilizadas para la realización de la aplicación y se explica el desarrollo de las diferentes partes del proyecto.

3.1. Tecnologías utilizadas

Las tecnologías utilizadas se pueden dividir en cuatro categorías, todas ellas vienen detalladas a continuación.

3.1.1. Framework multiplataforma

Para el desarrollo de un aplicación multiplataforma, es decir, una aplicación que funcione en varios sistemas operativos, es necesario utilizar un framework especifico, que permita con un mismo código, generar los archivos necesarios para los diferentes sistemas operativos. En este caso se quiere realizar una aplicación que funcione tanto en iOS, como en Android.

El framework utilizado es Flutter, este framework de código abierto permite crear aplicaciones nativas en ambos sistemas operativos con un mismo código. Una de las principales razones por las que se utiliza esta plataforma, es debido a la buena conexión de este framework con las bases de datos y servicios de Google, ya que pertenece a esta misma empresa; además, la documentación de la plataforma es verdaderamente extensa y detallada. Para la programación en Flutter, el lenguaje de desarrollo es Dart, que permite tanto desarrollar la interfaz de usuario mediante widgets, como conectarse con la base de datos y plataformas externas mediante funciones dedicadas y librerías especificas.

3.1.2. Entornos de desarrollo

Los entornos de desarrollo (*IDE*) son herramientas que permiten el desarrollo de código, haciendo posible generar código, compilarlo y probarlo. Para este proyecto se usan tres entornos de desarrollo distintos, el primero y el más utilizado es *Android Studio*, el segundo es *Xcode* y por ultimo, *Visual Basic Code* .

- **Android Studio:** Este entorno ha sido el más utilizado durante el proyecto, en él se programa todo lo relacionado con Flutter. Es un *IDE* especialmente diseñado para el desarrollo de aplicaciones móviles, cuenta con un emulador que permite simular dispositivos Android, para ver el funcionamiento de las aplicaciones, sin tener que volcarlo físicamente en un móvil. Además, al usarse un MacBook Pro como ordenador de desarrollo, el uso de Android Studio es conveniente, ya que XCode se puede utilizar para simular el funcionamiento de cualquier iPhone.

Figura 3.1: Android Studio logo

- **Xcode:** Xcode es un entorno de desarrollo exclusivamente dedicado al desarrollo de software en ordenadores MacOS. Es el entorno oficial para el desarrollo de aplicaciones para el sistema operativo iOS. Aunque el código principal de la aplicación se desarrolla en Android Studio, hay cosas que se deben hacer en el proyecto nativo de iOS y para ello se utiliza Xcode.

Figura 3.2: Xcode logo

- **Visual Basic Code:** El entorno Visual Basic Code es un *IDE* desarrollado por Microsoft, este se utiliza para una tarea muy especifica en el proyecto, en concreto, para volcar la base de datos desde Excel a Firebase, para ello se utiliza el lenguaje de programación JavaScript.

Figura 3.3: Visual Studio Code logo

3.1.3. Interfaz de usuario

La interfaz de usuario se programa en Flutter, siendo el lenguaje de programación Dart, que mediante Widgets permite realizar una interfaz de usuario responsiva, que se visualice correctamente en ambos sistemas operativos, y en dispositivos con diferentes escalas de pantalla. Estos Widgets tiene características y estados propios, que definen como se ven en la pantalla del usuario, los tipos mas utilizados son: texto, contendedores y columna o fila.

La interfaz de usuario se programa en un único código y luego Flutter se encarga de que el código se ejecute como aplicaciones nativas en los diferentes sistemas operativos. En la Figura 3.4 se puede ver la estructura de carpetas de un proyecto en Flutter, a continuacion se va a explicar los directorios mas importantes del proyecto:

- ***/lib*** : El código principal esta dentro de éste directorio y contiene todos los archivos *.dart* con los que se monta la interfaz de usuario.
- ***/pubspec.yaml***: En este archivo de configuración están tanto las dependencias del proyecto para ambos sistemas operativos, como las configuraciones generales del proyecto de Flutter.
- ***/ios***: Es el directorio donde se genera el código para aplicaciones nativas de iOS, en ella está tanto el código, como todas los archivos de dependencia y compatibilidades.
- ***/android***: Es el directorio donde se genera el código nativo para aplicaciones Android, al igual que la anterior, tiene todo el código en formato nativo para poder compilar la aplicación en este sistema operativo.

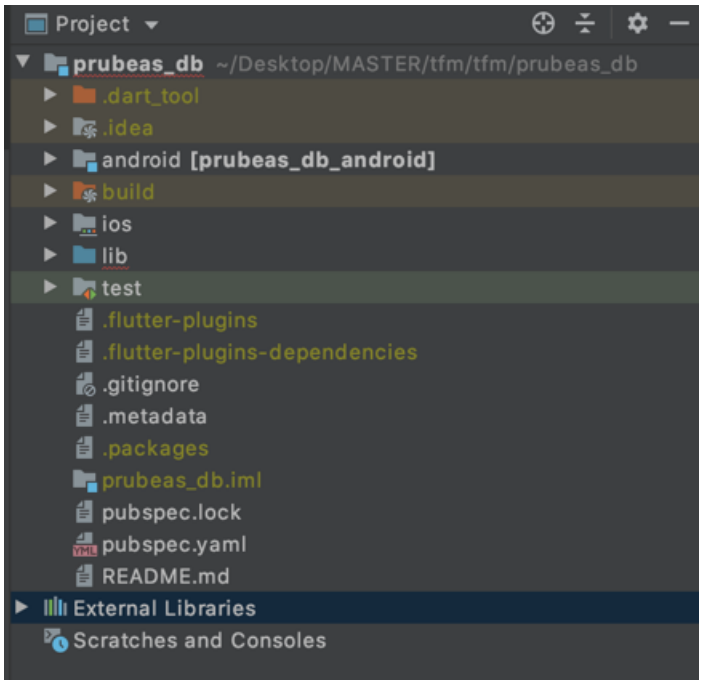


Figura 3.4: Estructura de carpetas de un proyecto Flutter

3.1.4. Base de datos

El alojamiento de la base de datos se realiza en Firebase, una plataforma para el desarrollo de aplicaciones en la nube, la plataforma tiene diferentes herramientas que facilitan el desarrollo y potenciación de las aplicaciones en el mercado. Sus productos se pueden dividir en tres categorías: desarrollo de aplicaciones, lanzamiento, monitorización y atracción de usuarios; dentro de estas categorías existen diferentes servicios prestados por Firebase.

En este proyecto se utiliza *Firestore Database* una base de datos NoSQL, que permite almacenar datos en documentos y colecciones, los documentos se encuentran dentro de las colecciones, que a su vez puede contener subcolecciones. Al ser una base de datos en la nube, se puede escalar la aplicación tanto en numero de usuarios, como en numero de servicios fácilmente; además tiene numerosos servicios para la autenticación de usuarios o manejo de anuncios que se pueden implantar en un futuro.

Para terminar, mencionar que esta plataforma tiene una tarifa gratis para empezar a desarrollar aplicaciones, permitiendo así a los desarrolladores tener una base de datos suficientemente grande como para probar la aplicación y empezar a

usarla sin tener que pagar. La tarifa se llama *spark* y tiene generosos limites en casi todas las funciones de Firebase. En la Figura 3.5 se pueden ver los servicios gratuitos de la tarifa spark en relación a Cloud Firestore que es la base de datos utilizada en este proyecto. Tal y como se ve en la figura, se pueden tener de manera gratuita hasta 1GB de datos almacenados, 10GB de salida de red por mes y numerosas operaciones de escritura y lectura por día. Por tanto, este plan es mas que suficiente para las dimensiones iniciales de esta aplicación.

Figura 3.5: Tarifa Spark Cloud Firestore

3.1.5. Google Cloud Plataform

Google Cloud Plataform, se usa para crear el mapa interactivo de la aplicación, esta plataforma brinda la oportunidad de generar claves API que permitan a la aplicación mostrar y acceder al contenido de Google Maps. Para darse de alta en este servicio, es necesario tener una dirección de correo Gmail y una tarjeta de crédito, a los nuevos desarrolladores de la plataforma, Google les concede un crédito gratuito de 300 USD durante un periodo de prueba de 90 dias, posterior a eso tiene multitud servicios sin cobro alguno, en concreto, para el uso del Google Maps API, se tienen carga ilimitadas de mapas por parte de los usuarios.

3.2. Desarrollo de la Interfaz de Usuario

En esta sección, se va a explicar como se desarrolla la interfaz gráfica de la aplicación, cada página principal: lista de categorías, lista de servicios y mapa interactivo vienen extensamente explicadas en cada una de sus secciones.

La interfaz de usuario de esta aplicación debe ser intuitiva y fácil de usar, en ella se muestra tanto la lista de servicios disponibles agrupados por categorías, como un mapa con la localización de cada servicio y la localización del usuario. Durante la realización de la interfaz de usuario, se tiene en cuenta que es crucial mostrar las informaciones en diferentes idiomas. Esto hace que la aplicación pueda ser usada por un público más extenso, que la aplicación pueda servir como medio de comunicación entre dos personas que no hablen el mismo idioma y para que los migrantes puedan ir aprendiendo conceptos clave en el idioma del país en que se encuentran.

El color principal de la interfaz de usuario es el azul y las letras e iconos vienen en color blanco, para poderlos visualizar correctamente los títulos de cada pagina principal, vienen centrados y en la parte superior de la pantalla.

Figura 3.6: Paginas principales y su navegación con la barra inferior

En la primera capa de la aplicación existen tres paginas principales: la lista de categorías, el mapa de servicios y la configuración de usuario. Para navegar por estas tres páginas se utiliza la barra inferior, en la Figura 3.6 se puede ver qué icono corresponde a qué página principal. El icono de barras de la izquierda lleva al usuario a la lista de categorías, el icono del mapa en el centro lleva al mapa interactivo y por ultimo, el icono de la derecha lleva a las configuraciones de usuario.

3.2.1. Lista de categorías. Página de inicio

La página de inicio aparece al abrir la aplicación, consta de una lista de categorías disponibles en la base de datos de Firebase. En la Figura 3.7 se puede ver su apariencia. Cada categoría viene definida por tres caracteristicas; su nombre (en español, francés y árabe), un color y un logo. Esta información tiene que venir definida por el dueño de la aplicación y se explica cómo configurarla en el Anexo III, sección 8.3.

Figura 3.7: Paginas principal

Desde la pagina principal de la aplicación, se accede a la información concreta de cada servicio haciendo click en la categoría correspondiente, una vez se accede a una categoría en concreto, se muestra la información de todos los servicios que contiene en una nueva pantalla.

En la Figura 3.8 se puede ver el diagrama de pantallas desde la lista de categorías, es decir, se muestra a qué partes de la interfaz de usuario se puede acceder desde la página principal. Al hacer click en una categoría especifica, en este caso "Legal", se accede a una pantalla secundaria donde se expone la lista de servicios que pertenecen a esa categoría. En esta pantalla el usuario se puede desplazar hacia abajo para ver todos los servicios, cada servicio viene en un recuadro diferente y, si tiene su localización en la base de datos, el icono de mapa se despliega a la izquierda del cuadro de información. Clickeando en él se accede al mapa interactivo que muestra una única chincheta indicando la ubicación de este servicio.

Figura 3.8: Diagrama de pantallas desde la lista de servicios

La programación de esta pantalla de inicio se lleva a cabo mediante un Widget personalizado llamado *MyHomePage()*, en él se genera la página de inicio; generando el titulo y barra superior, la barra inferior de navegación y la lista de categorías, cada categoría es un objeto detector de gestos que engloba de otro Widget personalizado llamado *ProductBox()*. Este Widget recibe la información concreta de una categoría; titulo, color y logo, con estos datos genera una caja personalizada y la devuelve a su objeto de llamada.

3.2.2. Servicios

Desde la lista de categorías descrita en la sección anterior, se accede a la listas de servicios, donde se despliega la información concreta de cada servicio en recuadros independientes.

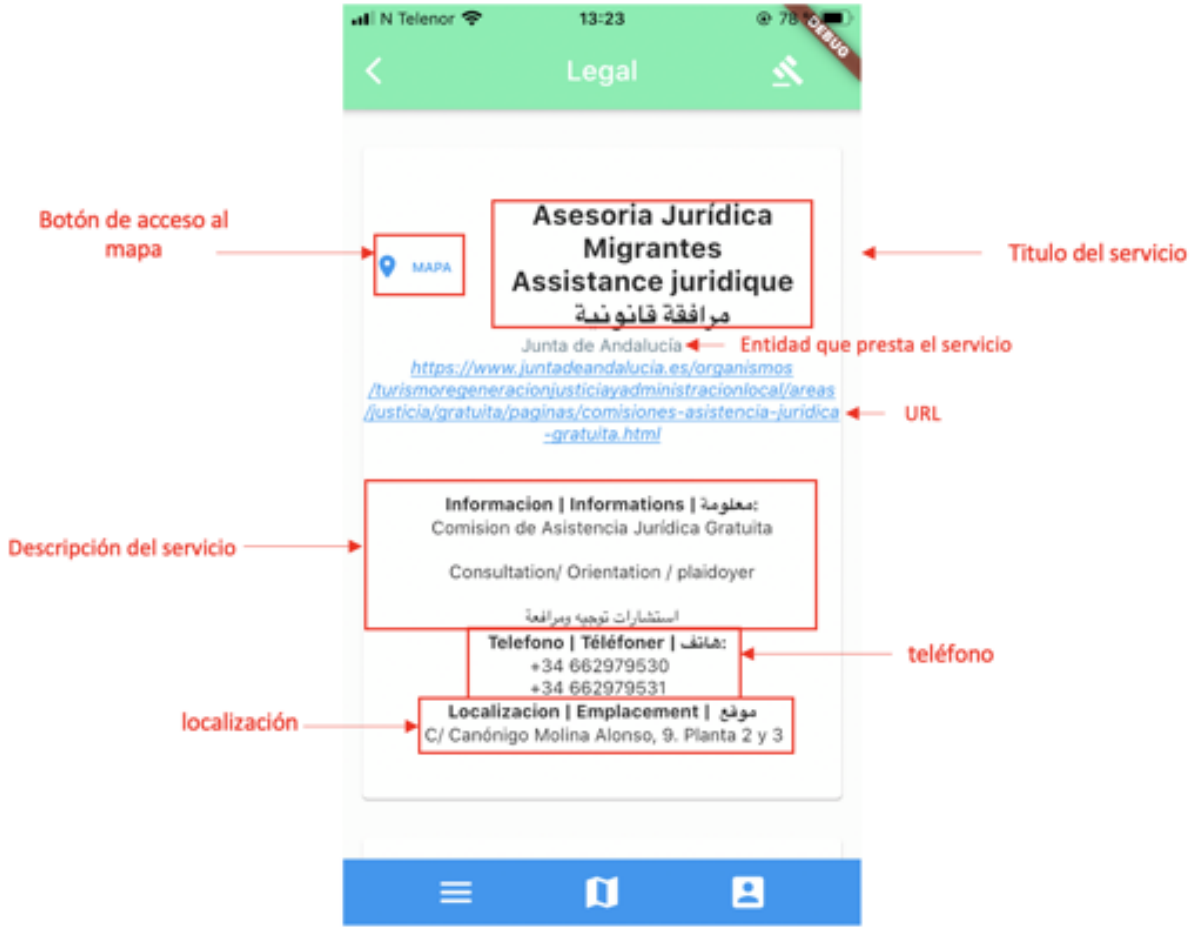


Figura 3.9: Descripción de la caja de servicios

La información disponible para cada servicio se puede ver en la Figura 3.9. En la parte superior del recuadro se encuentra el titulo en los tres idiomas disponibles; como se ha mencionado anteriormente, es importante presentar los datos en varios idiomas.

A la izquierda del titulo, se encuentra el icono del mapa, al hacer click sobre él se redirige al usuario a la pantalla del mapa, mostrando una única chincheta con la localización de este servicio.

Justo debajo del titulo, se despliega la entidad que presta el servicio en color gris. Debajo de eso, se puede presentar la url de la pagina web de la entidad o del servicio, siendo posible acceder desde ella directamente a la pagina web.

Por ultimo, se despliega la información sobre el servicio, que consiste en una breve descripción del servicio y los datos de contacto, que son el teléfono y la localización.

Cabe decir, que toda esta información que aparece en el recuadro del servicio, depende de los campos rellenos en la base de datos, es decir, si en la base de datos alguno de los campos queda vacío, esa información no sera visible en el recuadro del servicio. Además, si el servicio no tiene rellena su ubicación el icono del mapa de arriba a la izquierda no aparecerá para evitar confusiones por parte del usuario.

La programación de esta pantalla de servicios, se realiza mediante Widgets personalizados en Flutter, en este caso el nombre del Widget que genera la pantalla es *ServicePage()*, en él se genera para empezar el titulo de la pagina, que es el titulo de la categoría en español y el color de la barra superior que corresponde al color asociado a la categoría. Además, a diferencia de otras paginas se inserta una flecha en la parte superior izquierda de la pantalla para poder volver a la pagina de inicio. También, se genera como siempre la barra inferior de navegación y por ultimo se llama al Widget *SubProductBox()* pasándole los datos de un servicio en concreto, y este widget devuelve el recuadro del servicio completamente creado.

3.2.3. Mapa interactivo

La página correspondiente al mapa interactivo, despliega un mapa vía *Google Maps* en la pantalla del usuario. La primera vez que se accede al mapa con la barra inferior, todos los servicio en la base de datos, que tengan una ubicación serán mostrados con una chincheta roja en el mapa. Una vez abierto el mapa se puede filtrar qué ubicaciones se quieren ver con los botones inferiores de colores. Cada botón representa una categoría y al hacer click únicamente los servicios de esa categoría serán mostrados con chinchetas.

Figura 3.10: Diagrama de pantallas desde el mapa

En la Figura 3.10 se puede ver la pagina principal del mapa a la izquierda del diagrama, a esta pantalla se ha accedido desde la barra inferior, y por tanto, muestra todas las ubicaciones disponibles en la base de datos. Además, con el icono de abajo a la derecha del mapa se accede a la localización del usuario, en caso de que haya dado la autorización para poder acceder a ella que se puede ver en la imagen superio derecha. Por último, desde los cuadros de abajo se ve como al filtrar por categorías, únicamente las chinchetas de esa categoría aparecen en el mapa. En la imagen inferior derecha de esta figura se ha filtrado por la categoría mujeres y por tanto, aparecen la chinchetas correspondientes a los servicios de ayuda a mujeres.

3.3. Desarrollo de la base de datos

La base de datos de este proyecto viene proporcionada por asociaciones humanitarias que ya tienen recabada y verificada la información sobre servicios disponibles. Para el desarollo de la APP esta información fue proporcionada en formato Excel. Además, al ser una aplicación dedicada a migrantes la información disponible esta en tres idiomas español, francés y árabe, para poder llegar mejor al publico potencial y que la información sea mas clara y entendible para ellos.

La base de datos se decide realizar en Firebase, esta elección viene justificada en el apartado 3.1. Como la base de datos es NoSQL, se diseñan colecciones y documentos que englobarán la información necesaria para esta aplicación. En las siguientes secciones se explica cómo se diseña la base de datos y cómo se procede para volcar la información desde un documento Excel a la base de datos de Firebase. Estas acciones solo las llevan a cabo los administradores de la aplicación.

3.3.1. Diseño de la base de datos

En la Figura 3.11 se muestra el diseño conceptual de la base de datos, la colección 'pruebas-upload' es la coleccion principal que engloba todos los campos de información de esta aplicación, en ella se tiene un documento distinto por cada tipo de categoría que existe.

Cada documento de tipo categoría, engloba unas variables que la definen y son: Categoría, Categoría_ar y Categoria_fr estas tres variables contienen el nombre de la categoría en los tres idiomas disponibles. También se tiene la variable color que es un número entero que volcado en las funciones de Flutter representa un color, y la variable logo que es otro numero que representa un logo en Flutter, y la variable id que es un identificador único para la categoría. Por último, cada categoría contiene una colección llamada servicios que engloba los servicios que contiene.

La colección servicios tiene tantos documentos como servicios que engloba, cada servicio viene definido por un id único y tiene múltiples variables que recogen la información necesaria de cada servicio. Para la descripción del servicio existen las variables entity que almacena el nombre de la entidad proveedora del servicio, subject que es una breve mención al titulo del servicio y message que describe el servicio en unas cuantas lineas. Para la localización del servicio están address, city, country, latitud y longitud, y por ultimo, para el contacto existen website y phone.

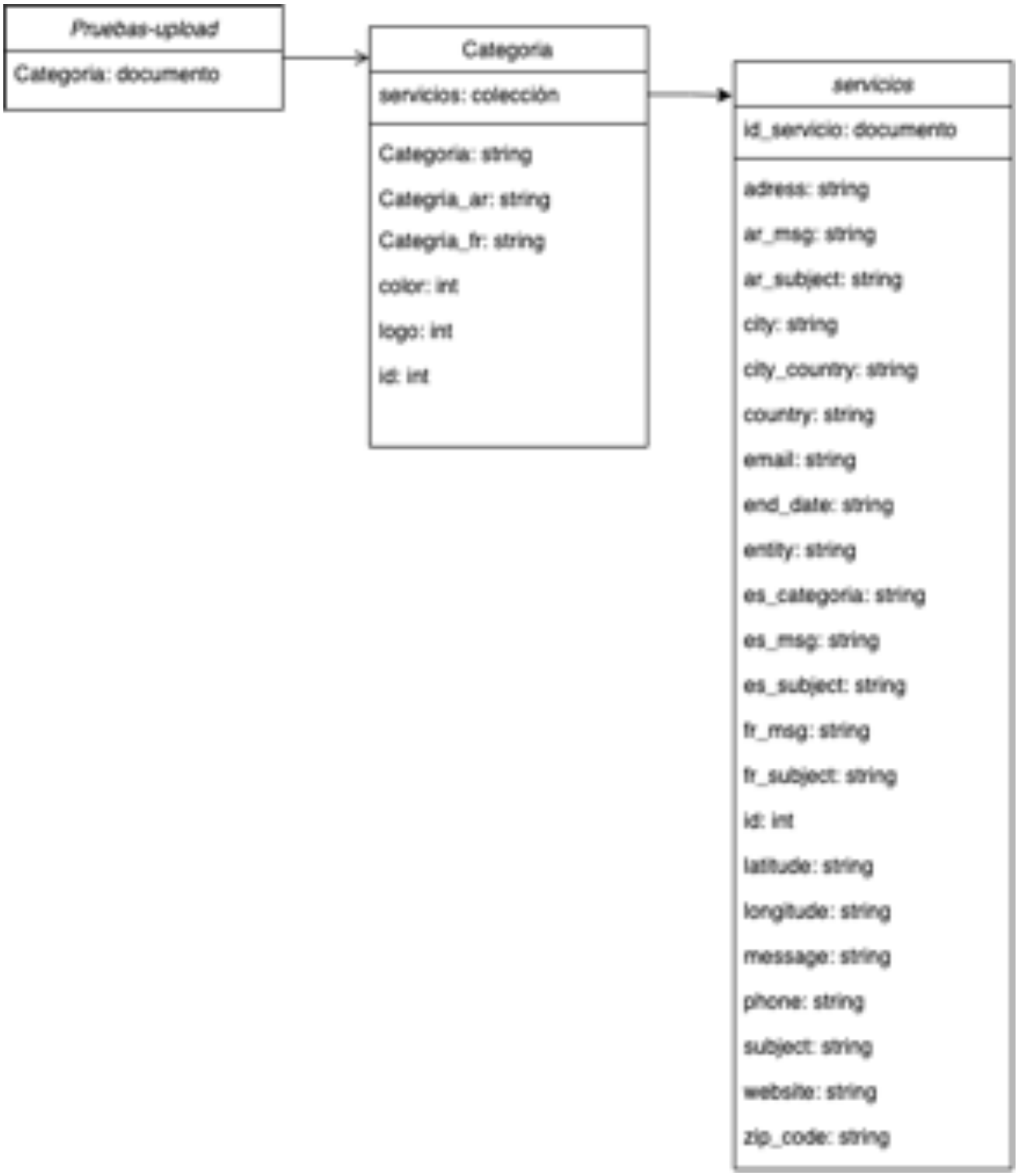


Figura 3.11: Diagrama de conceptual de la base de datos

3.3.2. Volcar la base de datos en Firebase

Para poder volcar la base de datos desde documentos Excel a Firebase se ha desarrollado un programa que lo hace automáticamente. Para ello, lo primero que hay que hacer es arreglar el formato del Excel, y que contenga una primera fila con el nombre de las variables, teniendo un servicio por cada fila del Excel, como se puede ver el la Figura 3.12. En esta figura se muestra parte del Excel con los datos, los títulos vienen en la primer fila en color negro y la información de cada servicios viene en las filas posteriores.

id	entity	es_categoria	es_subject	es_msg	location_phone
1	Médicos del Mundo	Administracion	Prostitución y explotación sexual	Servicio de información, orientación y atención a mujeres en situación de prostitución y víctimas de trata.	+34 638 560 345
2	Fegamp - Federación Gallega de Municipios y Provincias	Administracion	Migración	Información, orientación, servicios sociales. Encargada: María Viqueira Iglesias.	+34 981685001
3	Fegamp - Federación Gallega de Municipios y Provincias	Administracion	Migración	Información y asesoramiento; aula de apoyo para extranjeros escolarizados y para adultos; programas de formación y sensibilización. Encargada: Conchi Pernas Díaz	+34 982580609
4	Fegamp - Federación Gallega de Municipios y Provincias	Administracion	Migración	Información, orientación y asesoramiento sobre los recursos sociales, sanitarios, educativos; realización de itinerarios de inserción personalizados; sesiones informativas sobre rasgos culturales de la sociedad de acogida y sensibilización social. Encargada: Violeta Bernardo Vázquez	+34 982530383

Figura 3.12: Extracto del Excel de servicios

Una vez hecho esto se guarda el archivo como CSV (Comma Separated Values), este tipo de archivo es usado para guardar tablas de información con un formato universal, además proporciona un formato valido para posteriormente poder convertirlo a formato JSON, que es el formato en el que los datos se pueden subir a Firestore. Para convertir el archivo desde CSV a JSON se utiliza un convertidor online.

Para comenzar a escribir valores en la base de datos, es necesario primero crear y configurar un proyecto Node.js. Para ello se usa el comando **npm init** en el terminal del ordenador, que crea e inicializa un proyecto. A continuación, se descarga el archivo *ServiceAccountKey.js* del proyecto de Firebase, este archivo es necesario porque contiene las claves para poder tener acceso a escribir información en la base de datos desde plataformas externas a Firebase.

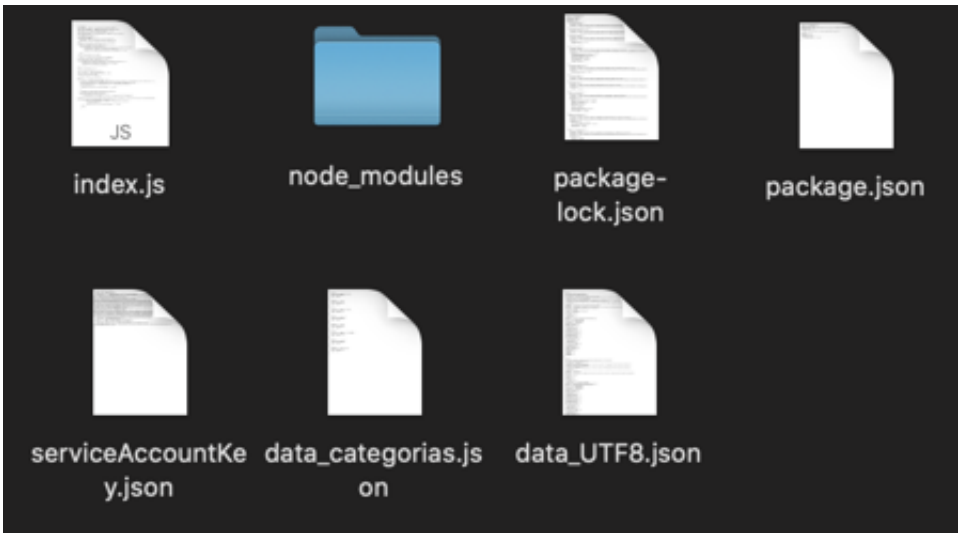


Figura 3.13: Directorio con los archivos del proyecto para subir datos a Firebase

Una vez se tienen los datos en el formato correcto y el proyecto Node.js creado, es necesario generar un código que vaya creando las diferentes colecciones y categorías e introduciendo los servicios a sus correspondientes colecciones.

El directorio de trabajo utilizado para está sección debe ser como el mostrado en la Figura 3.13, los datos están almacenados en dos documentos *data_categories.json*, que contiene la información sobre las categorías: titulo, color y logo, y *data_UTF8.json* que contiene la información sobre los servicios proveniente del Excel. El código principal que extrae los datos y los sube a Firebase está en el archivo *index.json* y por ultimo, *package.json*, *package_lock.json* y *node_modules* son archivos y directorios que se generan automáticamente al crear el proyecto *Node.js*.

La primera tarea es crear la colección principal, que en este caso se llama 'pruebas-upload', para ello se usa la interfaz de usuario de Firebase ya que es mas sencillo. A continuación, se rellena esta colección con documentos que engloban las categorías, para ello, se usa el archivo data_categories.json, extrayendo de él toda la información de las categorías y añadiéndola a Firebase, en la Figura 3.14 se puede ver cómo queda la base de datos tras esto, cada documento de categoría tiene sus propias variables, en la Figura 3.14 se pueden ver las de Administración.

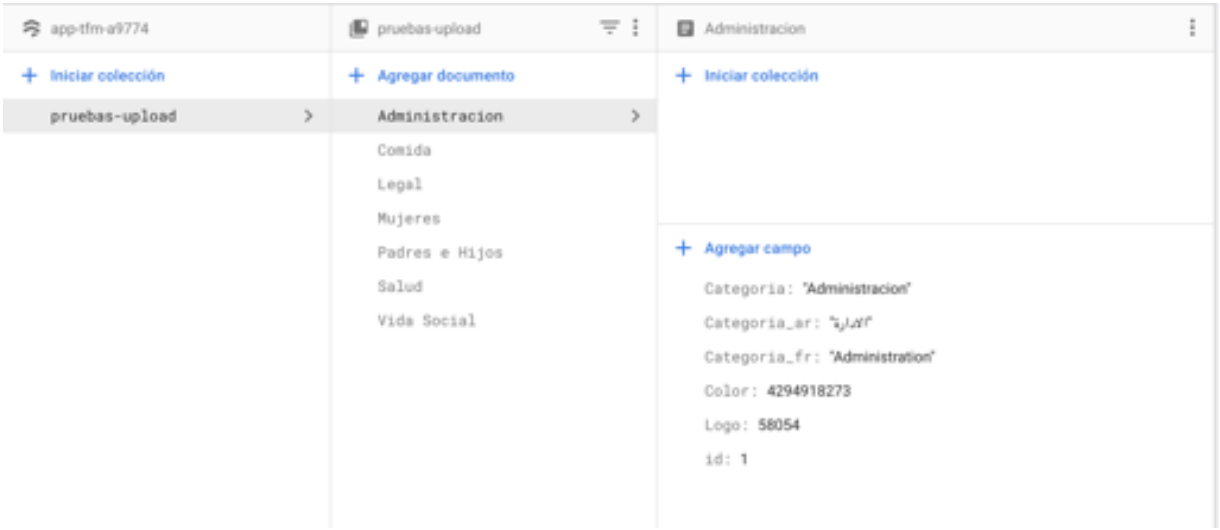


Figura 3.14: Base de datos tras subir las categorías, en concreto se ve el documento de la categoría 'Administración'.

Una vez se tienen las categoría en la base de datos, se filtran los servicios según categoría y se añade una nueva colección a cada categoría llamada 'servicios', donde se subirán todos los servicios de la categoría. En la Figura 3.15 se puede ver como queda la base de datos tras rellenar los servicios, en concreto los de la categoría Administración, una vez en este punto, la base de datos esta completa y solo se tendrá que actualizar con nuevos servicios o nuevos datos.

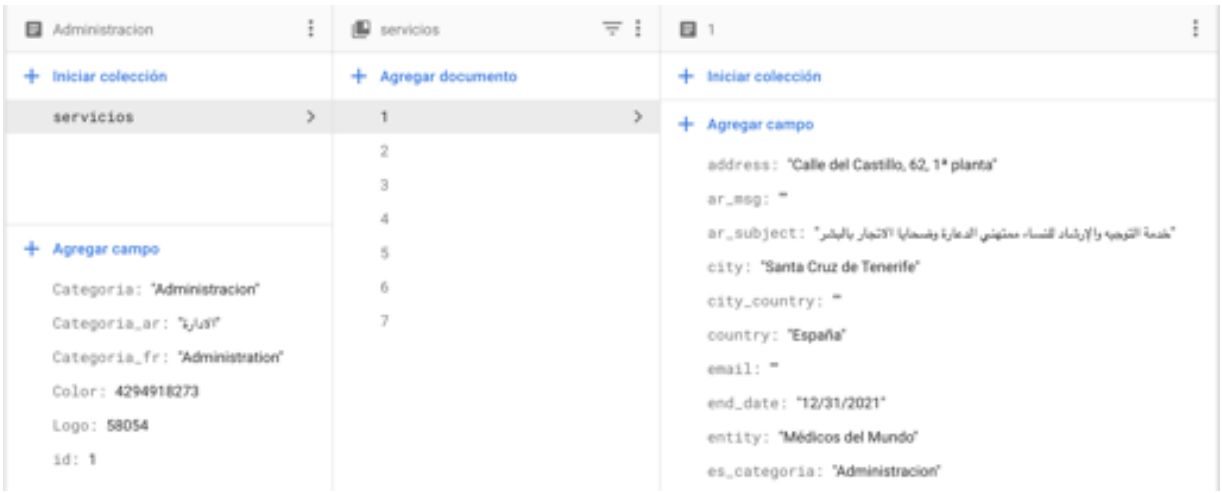


Figura 3.15: Base de datos tras subir los servicios a las categorías, en concreto se ve el documento 'Administración' y su colección servicios

3.4. Integración

En este apartado se explica con qué tecnologías se ha integrado la interfaz de usuario de la aplicación, es decir, con qué otras tecnologías se comunica la interfaz de usuario para poder generar la aplicación en su conjunto. Esta sección tiene dos partes, la integración con Google Maps mostrada en el apartado 3.4.1 y la integración con la base de datos, mostrada en el apartado 3.4.2

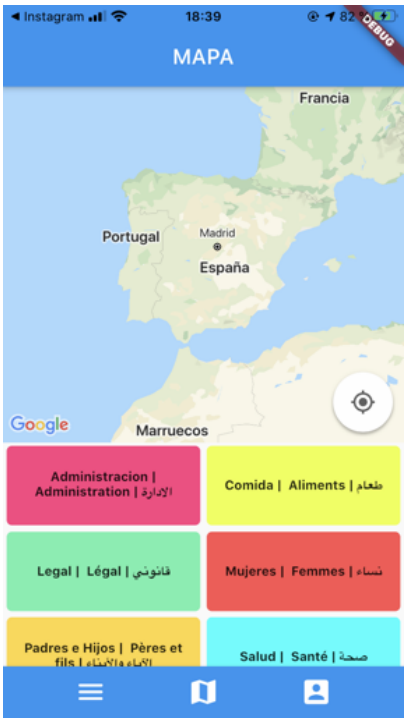
3.4.1. Integración con Google Maps

El mapa interactivo que forma parte de esta aplicación es debido a la integración de Google Maps en ella, se decide usar esta plataforma porque proporciona un servicio de mapas extenso y actualizado, además permite al usuario redirigirse a Google Maps con las localizaciones de los servicios para poder disfrutar de todas las funcionalidades de esta aplicación externa como por ejemplo la generación de rutas, Google Maps es compatible tanto con iOS como con Android.

Para llevar a cabo la integración de Google Maps con Flutter existe una librería dedicada a ello *google_maps_flutter*, por tanto, el primer paso a seguir es añadir esta dependencia al archivo *pubspec.yaml*.

Una vez hecho esto, es necesario crear y agregar un clave API a nuestro código, esta clave dará acceso a la aplicación a los servicios de *Google Maps*. Para poder tener una clave API, es necesario tener una cuenta en Google Cloud Plataform, en la que se habilita el SDK de iOS Maps y de Android Maps. Una vez hecho esto se copia la clave correspondiente a Android al fichero AndroidManifest.xml y la clave correspondiente a iOS al fichero Info.plist, así la configuración del proyecto esta lista para poder usar las funciones de Google Maps en la aplicación

Una vez accedido a la API, es necesario crear las chinchetas que muestren la localización de los servicios en el mapa, la propia librería de Flutter tiene funciones que permiten crear marcadores mediante la latitud y longitud de los servicios. Para filtrar por servicios en el mapa se realizan distintas queries a la base de datos, buscando únicamente los servicios dedicados a cada categoría. En las Figuras 3.16a y 3.16b se puede ver como se muestran las chinchetas en el mapa interactivo.



(a) Mapa integrado en la APP sin chinchetas

(b) Mapa integrado en la APP con chinchetas

Figura 3.16: Utilización de la API de Google Maps.

3.4.2. Integración con Base de Datos

Una vez desarrollada tanto la interfaz de usuario como la base de datos, es necesario integrar las dos, para que se puedan comunicar y que la información proveniente de la base de datos se pueda presentar en la aplicación.

Para integrar la base de datos con la interfaz de usuario es necesario que se puedan comunicar, para ello, lo primero que se tiene que realizar es registrar la aplicación en Firebase. Se deben registrar dos aplicaciones independientes, una Android y otra iOS en el proyecto donde se ha realizada la base de datos.

En la Figura 3.17 se muestra cómo es el primer paso al registrar la aplicación en Firebase. En este primer paso es necesario introducir el nombre del paquete Android que viene en el archivo *build.gradle* de la aplicación de Flutter y un código SHA-1 que es opcional y en este proyecto no se ha usado.

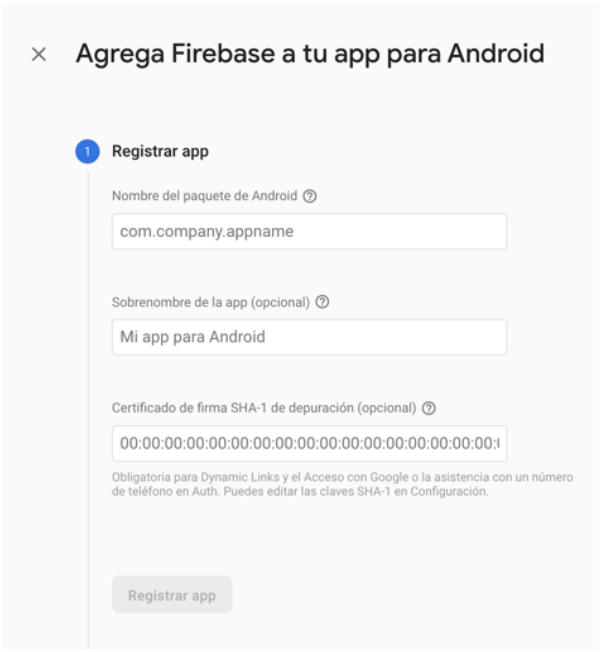


Figura 3.17: Datos para agregar una APP a firebase, primer paso

El siguiente paso es descargarse el archivo *google-services.json* que genera Firebase automáticamente. Este archivo permite corroborar que la aplicación que está intentando usar la base de datos esta dada de alta como cliente de la misma, en caso contrario la comunicación no se puede realizar. En la Figura 3.18 se puede ver la pantalla de Firebase para descargarse este archivo, cabe destacar, que es necesario descargarse dos archivos *google-services.json*; uno para el proyecto de Android y otro para el proyecto de iOS. En el proyecto de Android el archivo se introduce dentro de la carpeta $\backslash android\ \backslash app$ y el de iOS en la carpeta $\backslash Runner\ \backslash Runner$

Figura 3.18: Datos para agregar una APP a firebase, segundo paso

Por ultimo, se agrega el SDK de Firebase tanto al proyecto de Android como de iOS, para ello se añaden estas linea de código:

- **Android:** se añade el SDK en las dependencias del archivo *build.gradle*, en concreto dentro de la sección buidlscript se inserta google() en repositories y en allprojects, además en las dependencies deben estar los servicios de google.

```
buildscript {
  ...
    repositories {
        google()
    }
        ...
    dependencies {
        classpath 'com.google.gms:google-services:4.3.8'
    }
}
allprojects {
    ...
    repositories {
        google()
        ...
    }
}
```

- **iOS:** En este caso iOS usa CocoaPods para administrar las dependencias de la aplicación, por tanto lo primero que hay que hacer es inicializar el Podfile desde el terminal con la siguiente instrucción:

  ```
  $ pod init
  ```

  A continuación se agrega a *Pods \Podfile* el siguiente código:

  ```
  pod 'Firebase/Analytics'
  ```

  Por ultimo, se instalan las nuevas dependencias en el terminal con la instrucción:

  ```
  $ pod install
  ```

Tras estos pasos, Firebase y el proyecto de Flutter se comunican entre ellos, por tanto, ya es posible hacer llamadas a la base de datos desde el código mediante queries para recoger la información necesaria en cada momento.

3.5. Arquitectura de la aplicación

En esta sección se va a explicar la arquitectura de la aplicación, haciendo referencia al flujo de información entre las diferentes entidades del proyecto y la estructura y orden en la que todo funciona. Las entidades del proyecto son:

- Interfaz de Usuario de Flutter
- Firebase
- Google Maps

3.5.1. Comunicaciones

A groso modo, la comunicaciones de los distintos módulos de la aplicación se puede ver en la Figura 3.19, la aplicación reside en el dispositivo móvil del usuario y se comunica tanto con Firebase como con Google Maps de manera bidireccional, es decir, la aplicación les pide contenido a las dos entidades y ellas se lo devuelven. Ambas plataformas utilizan APIs para poder comunicarse con la aplicación. Al usar el framework Flutter durante la programación de la aplicación, estas comunicaciones se realizan mediante funciones ya creadas en las librerías correspondientes, que abstrae al programador de la aplicación de la complejidad de gestionar las comunicaciones entre los servicios de terceros y el código. Las funciones utilizadas son las siguientes:

- **Comunicación Flutter - Firebase:** las librerías utilizadas son firestore_core y firestore_cloud con ellas se tienen todas las funciones necesarias para la comunicación entre la aplicación y la base de datos.
- **Comunicación Flutter - Google Maps:** la librería utilizada es google_maps_flutter que gestiona las comunicaciones del usuario con las plataforma de Google Maps.

Figura 3.19: arquitectura de las comunicaciones entre la app y los distintas tecnologías

3.5.2. Lógica de la aplicación

La lógica de la aplicación se puede dividir dependiendo de las acciones del usuario, el usuario interactúa con la aplicación mediante la interfaz gráfica y el código responde con diferentes acciones a lo que el usuario le pide, a continuación se va a hacer una breve mención a todas las acciones que el usuario puede realizar y como reacciona la aplicación a ellas.

En la Figura 3.20 se ven las acciones que el usuario puede realizar y que se explican más abajo. El icono rojo del cursor representa la acción que el usuario debe hacer y cada sección de la tabla muestra una acción diferente que se puede realizar.

Figura 3.20: Acciones que puede realizar el usuario en las distintas pantallas

- **Inicialización de la aplicación:** Al abrir la aplicación en el dispositivo móvil, el código pide a la base de datos toda la información correspondiente a las categorías: títulos en diferentes idiomas, color de cada categoría y logo, todo esto se realiza con la función que se ve a continuación:

```
FirebaseFirestore.instance.collection('pruebas-upload')
.snapshots()
```

Una vez se tiene esa información se procesa el color que en la base de datos es un numero entero pero con la siguiente función sé define como un color en Flutter:

```
Color(snapshot['Color'])
```

Algo muy parecido a lo anterior se hace con el logo, Flutter tiene su propia librería de iconos, y los iconos se guardan en la base de datos como un numero entero, luego se usa la siguiente función para definirlo como un logo en Flutter:

```
Icon( IconData(snapshot['Logo'],
fontFamily:'MaterialIcons'), size: 50)
```

Con todos estos datos de cada categoría, se monta la pantalla de inicio.

- **Selección de una categoría:** Los cuadrados de la lista de categorías son detectores de gestos, cuando el cliente selecciona una categoría en concreto, toda la información que se tiene previamente sobre esa categoría, es pasada a las funciones para construir las listas de servicios. Esto se hace con las lineas de código que se ven un poco más abajo, en ellas *snapshot.data.docs[index]* contiene la información sobre la categoría seleccionada y la función *ServicePage()* construye la pantalla de servicios con esta información.

```
onTap: () {
    Navigator.push(
        context,
        MaterialPageRoute(
            builder: (context) =>
                ServicePage(item: snapshot.data.docs[index]),
            ),
    );
 }
```

Una vez en la función que crea la pantalla de servicios *ServicePage()*, es necesario volver a comunicarse con la base de datos, para pedir la información

sobre los servicios disponibles en la categoría. Esto se hace con las siguientes lineas de código:

```
FirebaseFirestore.instance.collection('pruebas-upload').
doc(item['Categoria']).collection('servicios').snapshots(),
```

Con esta query se selecciona la subcategoría 'servicios' dentro de la categoría seleccionada, donde están todos los documentos que corresponden a los servicios disponibles y con esto se procede a montar la pantalla.

- **Acceso a la localización de un servicio:** Desde la lista de servicios el usuario puede consultar la información sobre los servicios y además, puede acceder a la localización de los servicios concretos, para ello, existe el boton de mapa que se puede ver en la Figura 3.21

Figura 3.21: Botón para acceder a la localización de un servicio desde la pagina de servicios

Cuando el usuario pide acceder a la localización del servicio, el código redirige al usuario a la pagina del mapa con la función *MapPage()* y como argumentos pasa el titulo de la categoría y el id del servicio dentro de esta categoría, esto se hace con el siguiente código:

```
onPressed: () {
    Navigator.push(context,
        MaterialPageRoute(builder: (context) =>
            MapPage(categoria: snapshot['es_categoria'],
                    product: snapshot['id'])
                    )
        );
}
```

Una vez en la pagina del mapa, el código crea una única chincheta, buscando en la base de datos al servicio que corresponde al id y la categoría proporcionadas. Una vez creados los marcadores se llama a la función que se comunica con Google Maps *GoogleMap()* con el siguiente código:

```
return GoogleMap(
        myLocationEnabled: true,
        myLocationButtonEnabled: true,
        compassEnabled: true,
        markers: Set<Marker>.of(markers),
        initialCameraPosition: CameraPosition(target:
        LatLng(40.38,-3.37), zoom: 7),
         onMapCreated: (GoogleMapController controller) {
                    mapController = controller;
                },
        );
```

Finalmente, la pantalla del mapa es construida en su totalidad y el mapa interactivo con la chincheta se visualiza en la pantalla del usuario.

- **Acceso al mapa desde la barra inferior:** Cuando el usuario accede a la pantalla del mapa vía la barra inferior, todos los servicios son desplegados en el mapa, por tanto se llama a la función *MapPage()* con los argumentos por defecto, categoría: 'default' y id: -1, con esto el código reconoce que debe mostrar todos los servicios, las instrucciones utilizadas para esto están en la función *CreateButtonAppBar()* que se dedica a construir la barra inferior de navegación y es el siguiente:

```
onPressed: () {
          Navigator.push(context,
          MaterialPageRoute(builder: (context) =>
              MapPage(categoria: "default", product: -1)
              )
        );
    }
```

Al igual que en la sección anterior, la aplicación genera los marcadores y se los pasa a la función *GoogleMap()* que despliega el mapa en la pantalla. Además, debajo del mapa se crea una lista de categorías de la misma manera que se genera la lista de categorías en la pagina principal, pero se distribuyen en la pantalla de manera distinta.

- **Filtrar por categoría en el mapa:** Cuando el usuario está en la pantalla del mapa y decide seleccionar uno de los cuadrados de categorías que se muestran debajo del mapa, el código filtra los marcadores que pertenecen a esa categoría y solo enseña esos. Para ello se vuelve a llamar a la función *MapPage()* con estos argumentos categoría: el nombre de la categoría seleccionada y producto: -1 que es el valor por defecto y con el que el código sabe

que no se esta filtrando por ningún producto en concreto. El código utilizado es el siguiente:

```
onTap: () {
    Navigator.push(context, MaterialPageRoute(
        builder: (context)=> MapPage(categoria:snapshot.data.
        docs[index]['Categoria'],
        product: -1)
        )
    );
}
```

3.6. Mecanismo de recogida de datos útiles para las asociaciones humanitarias

Parte del beneficio de tener una aplicación propietaria, es la habilidad de recoger datos a través de ella, que luego pueden ser usados para ampliar y mejorar el servicio humanitario qué se le ofrece al migrante. En concreto, esta APP permite recoger datos valiosos para las organizaciones humanitarias sobre; qué servicios son los más demandados y qué perfil de usuarios tienen.

Firebase, proporciona un Dashboard donde el propietario de la base de datos puede acceder a estadísticas y datos de uso generales. Ese Dashboard forma parte de su sección de *Analytics* suele incluir métricas generalistas, como las regiones desde donde se está usando la aplicación, el sistema operativo desde el que se accede, el numero de usuarios activos y otros muchos mas. Además del Dashboard, Firebase permite llevar un registro de ciertos eventos que pasan dentro de la aplicación, como por ejemplo; cambios de pantalla, tiempo en cada pantalla o acciones que realiza el usuario. Para ello, es necesario ajustar el código de Flutter y registrar los eventos personalizados. En este proyecto, es interesante poder saber qué categorías son las mas buscadas por los usuarios, y por ello, se quiere llevar a cabo un seguimiento de las vistas de pantalla.

Como se ha mencionado anteriormente, es necesario incluir ciertas funciones en el código de Flutter para poder registrar los eventos del seguimiento de pantallas, para ello se utiliza la librería *Firebase_Analytics*. La manera en la que se ha diseñado este sistema es; para empezar, se le da un nombre a cada pantalla de la aplicación que se quiere incluir en el seguimiento: la pantalla de inicio se registra como "HomePage", cada pantalla que expone la lista de servicios de una categoría tiene como nombre el titulo de la categoría y por ultimo, el mapa interactivo lleva

el nombre de "MAP page". La instrucción de código utilizado para dar nombre a cada pantalla es la que se muestra a continuación, en este ejemplo se registra la pantalla de inicio pero para registrar cualquier otra pantalla solo hace falta cambiar el *screenName*.

```
FirebaseAnalytics().setCurrentScreen(screenName: 'HomePage');
```

Cada vez que el usuario entra a alguna de esas pantallas un evento se envía a Firebase para ser almacenado, cada evento contiene un nombre de pantalla al que esta asociado y un id, la instrucción de código para ello es la siguiente:

```
FirebaseAnalytics().logViewItem(itemId: "1",
itemName: 'HomePage', itemCategory: 'screens');
```

A continuación se hace un breve resumen de qué métricas son interesantes y se pueden recoger desde Firebase:

- **Localización geográfica de la audiencia:** Estos datos vienen incorporados al Dashboard de Firebase, con ello se puede analizar desde dónde se está accediendo a la información de la aplicación. Por ejemplo, sería útil para las asociaciones saber si los usuarios acceden a la información desde sus países de origen o desde el destino. Además, puede proporcionar una información valiosa a la hora de entender que idiomas es recomendable que entiendan los empleados que están de cara al usuario en este tipo de servicios.

- **Sistemas operativos de los usuarios:** Esta categoría de datos, puede proporcionar una valiosa información sobre, el poder adquisitivo de los usuarios, con ellos se puede evaluar si la aplicación esta siendo utilizada en su mayoría por gente de perfil adquisitivo mas bajo que usarían un dispositivo Android, o mas alto donde usarían un dispositivo iOS.

- **Seguimiento de pantallas:** Con este conjunto de datos las asociaciones pueden ver que categorías son las más visitadas, y consecuentemente con los resultados intentar reforzar ese área de servicios. Además, este conjunto de datos también sirve para la mejora de la aplicación, con ellos se puede entender qué se usa más si el mapa interactivo o la lista de servicios, para poder enfocar una nueva versión de la aplicación hacia ello.

Capítulo 4

Análisis de Resultados

En esta sección del documento, se exponen los resultados del proyecto; en la primera sección 4.1 se muestran los resultados del desarrollo de la propia aplicación, tanto de la interfaz de usuario, como la base de datos y su integración; en la segunda sección 4.2, se muestran los resultados extraídos tras el uso de la aplicación por parte de voluntarios.

4.1. Resultados del desarrollo

Los resultados del desarrollo se muestran en esta sección, en ella se pueden ver tanto los resultados de la interfaz de usuario en móviles iOS y Android, como la base de datos y su integración con la interfaz de usuario.

4.1.1. Interfaz de Usuario. Pruebas de compatibilidad.

La aplicación realizada es multiplataforma, por tanto la interfaz de usuario debe de funcionar tanto en el sistema operativo Android como iOS, por ellos se realizan estas pruebas, en la Figura 4.1 se puede ver la interfaz de usuario en un sistema operativo iOS y en concreto en un iPhone 7, y en la Figura 4.2 se ve la interfaz en un sistema operativo Android, en concreto un Samsung Galalxy A40.

Como se pueden ver en estas dos figuras, la interfaz de usuario está bien desarrollada para ambos sistemas operativos, con esto se muestra la capacidad que tiene la herramienta de Flutter para crear aplicaciones multiplataforma. Tanto las listas de servicios y categorías como el mapa se pueden visualizar correctamente en ambos dispositivos, dejando los margenes correctos entre las secciones y las funcionalidades de la aplicación son las mismas en ambos también, todas los seleccionables y el mapa funcionan de manera correcta en ambos.

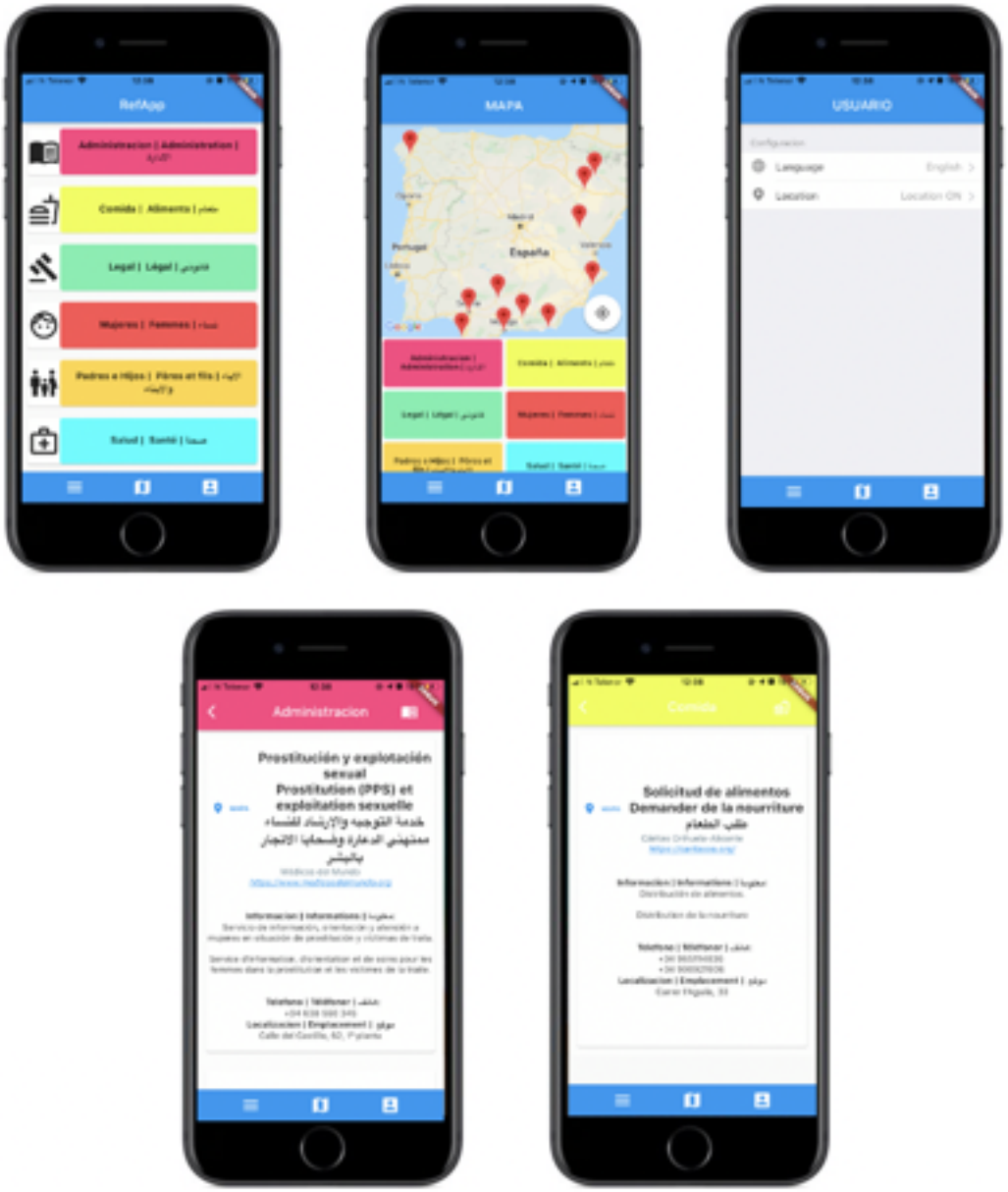


Figura 4.1: Resultados de la IU en un iPhone 7

En la Figura 4.1 se pueden ver todas las pantallas de la interfaz de usuario en un dispositivo iOS, se resalta que todos los recuadros están bien posicionados en la pantalla, los títulos aparecen en el centro de la barra superior y el mapa se muestra correctamente

Figura 4.2: Resultados de la IU en un Samsung A40

En la Figura 4.2 se ven las pantallas de la aplicación en un dispositivo con sistema operativo Android, al igual que en dispositivos iOS los recuadros están bien posicionados, con los margenes esperados y el mapa posicionado correctamente, los títulos aparecen en el centro de la pantalla y las funcionalidades de la aplicación son correctas.

Con los resultados de esta sección, se destaca que la aplicación funciona, tanto en sistemas operativos Android como iOS, pudiendo así llegar a un publico mas amplio.

4.1.2. Base de Datos

La base de datos es una entidad crucial para el desarrollo y funcionamiento de la aplicación, en ella se guardan los datos de todos los servicios disponibles y de las características de la aplicación. En estos resultados se escogen dos servicios ejemplo, y se muestra donde quedan esos servicios en la base de datos, tras volcar los datos desde Excel a Firebase.

Ejemplo 1: Se escoge aleatoriamente un servicio del Excel de datos proporcionado por las asociaciones. El servicio se puede ver en la Figura 4.3 donde se ven los campos de id, entidad, categoría, titulo, mensaje en español y por ultimo teléfonos. Se destaca que no son los únicos datos en el Excel sobre este servicio, pero valen para poder evaluar la base de datos y reconocer el servicio.

id	entity	es_categoria	es_subject	es_msg	location_phone
13	Junta de Andalucía	Legal	Asesoria Jurídica Migrantes	Comision de Asistencia Jurídica Gratuita	+34 662979530 +34 662979531

Figura 4.3: Parte de los datos del Excel sobre el servicios ejemplo1

Se busca este servicio en la base de datos de Firebase, para verificar que esta posicionado en los documentos y colecciones correctos.

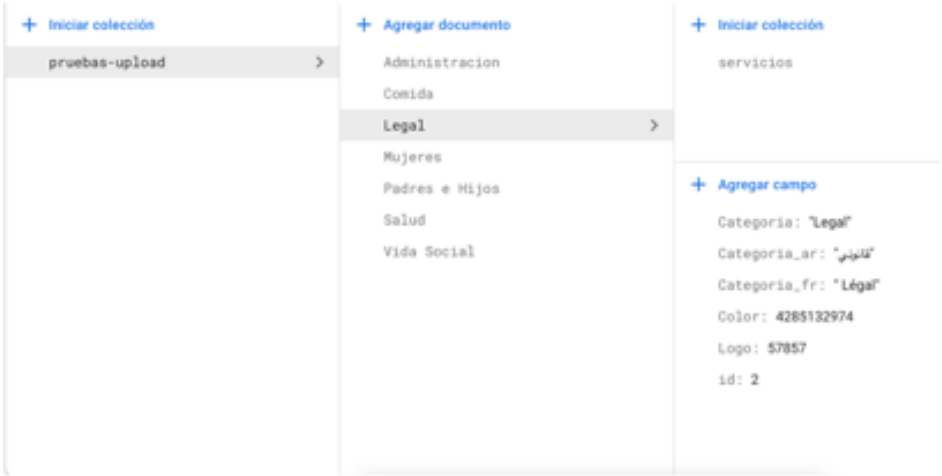


Figura 4.4: Base de datos de Firebase primera colección donde se encuentra el servicio del ejemplo 1

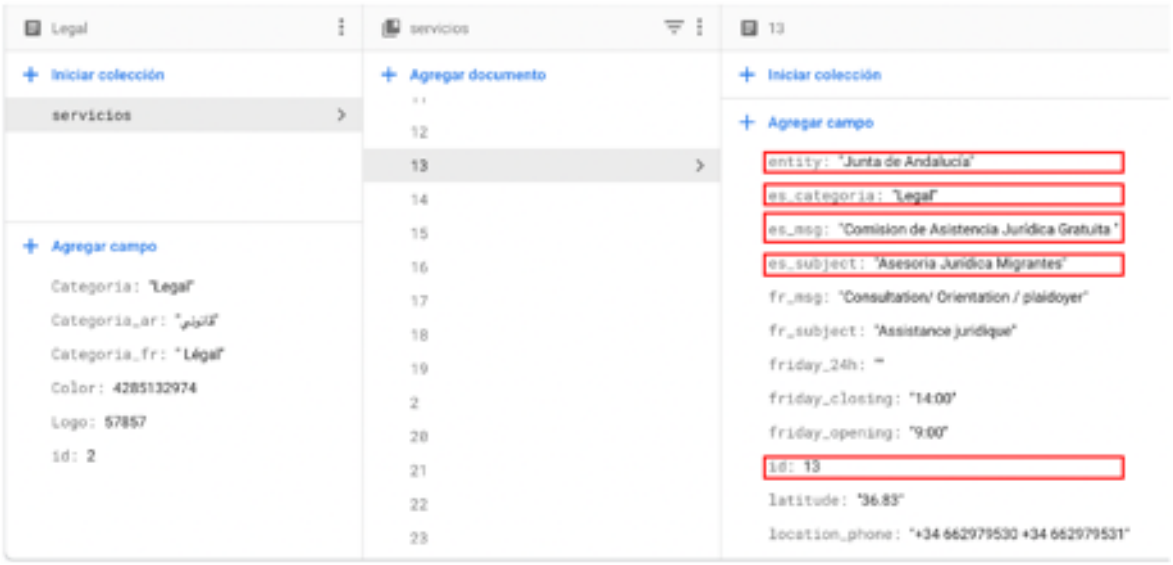


Figura 4.5: Base de datos de Firebase segunda colección donde se encuentra el servicio del ejemplo 1

Como se ve en las Figuras 4.4 y 4.5, el servicio ejemplo se encuentra dentro de la categoría 'Legal' y dentro de la sub-colección servicios bajo el id 13. Esta entidad tiene todos los campos detallados en la Figura 4.3 insertados en la base de datos de Firebase correctamente, la Figura 4.5 contiene en rojo los campos que coinciden.

Para obtener la información de este servicio desde Flutter y no directamente desde la interfaz de usuario de Firebase se puede utilizar el siguiente querie:

```
FirebaseFirestore.instance.collection('pruebas-upload').
doc('Legal').collection('servicios').
doc('13').get()
```

Ejemplo 2: En este ejemplo se elige otro servicio del Excel, en la Figura 4.6 se pueden ver algunos campos del servicio escogido. Al igual que en el ejemplo anterior se pueden ver el id, la entidad, la categoría, el tema en español, mensaje en español y el teléfono.

id	entity	es_categoria	es_subject	es_msg	location_phone
10	Fegamp - Federación Gallega de Municipios y Provincias	Vida Social	Migración	Información y asesoramiento; aula de apoyo para extranjeros escolarizados y para adultos; programas de formación y sensibilización. Encargada: Conchi Pernas Díaz	+34 982580609

Figura 4.6: Parte de los datos del Excel sobre el servicios ejemplo2

A continuación, se busca el servicio en la base de datos de Firebase para verificar que el servicio está en la categoría correcta y que los datos son correctos también.

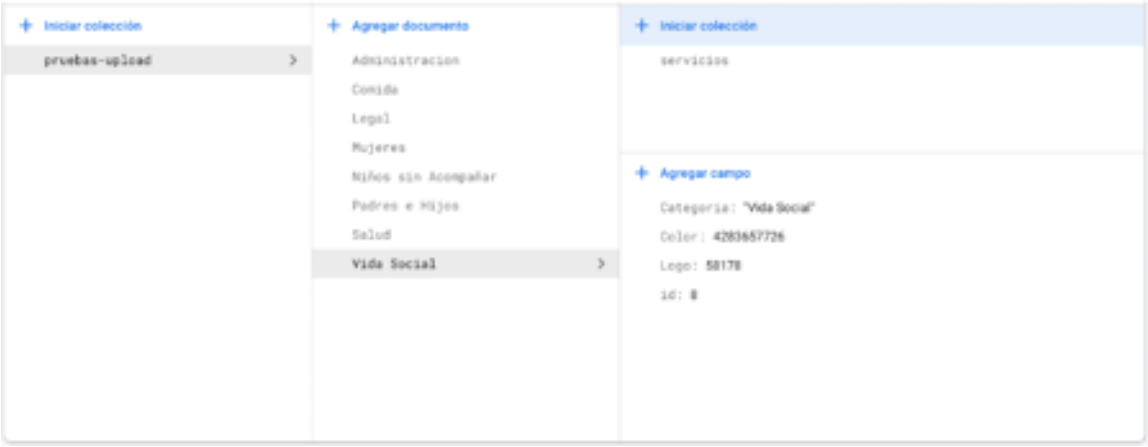


Figura 4.7: Base de datos de Firebase segunda colección donde se encuentra el servicio del ejemplo 2

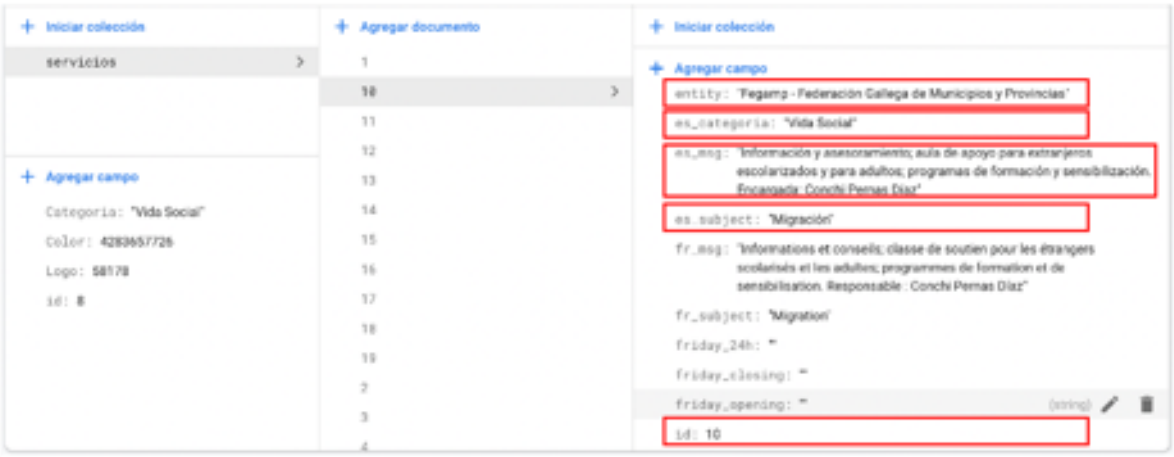


Figura 4.8: Base de datos de Firebase segunda colección donde se encuentra el servicio del ejemplo 2

En las Figuras 4.7 y 4.8 se puede ver las colecciones y documentos a los que pertenece el servicio escogido para el ejemplo 2, en la Figura 4.7 se ve como la categoría a la que pertenece es 'Vida Social' y en la Figura 4.8 se puede comprobar como los campos del Excel están correctamente rellenados en la base de datos.

Para hacer una consulta a la base de datos y obtener la información necesaria de ese servicio, la consulta sería la siguiente:

```
FirebaseFirestore.instance.collection('pruebas-upload').
doc('Vida Social').collection('servicios').
doc('10').get()
```

4.1.3. Integración

La integración en este proyecto, hace referencia a la fusión de la interfaz de usuario con la base de datos y con Google Maps API, para que la aplicación tenga todas las funcionalidades diseñadas. En esta sección de resultados, se quiere mostrar como funciona la aplicación una vez llevada a cabo la integración, para ello se muestran resultados de dos secciones diferentes; la primera es la integración Flutter - Firebase y la segundo la integración entre Flutter - Google Maps.

- **Integración Flutter - Firebase**

  Para esta sección de resultados se analiza como llegan las peticiones de datos desde Flutter a Firebase, y qué tipo de acción sobre la interfaz de usuario crea estas peticiones.

  Para recoger estos resultados, se accede a la aplicación y se selecciona una de las categorías, simultáneamente se analizan las peticiones de lectura a la base de datos para corroborar que coinciden con las acciones que el usuario a realizado.

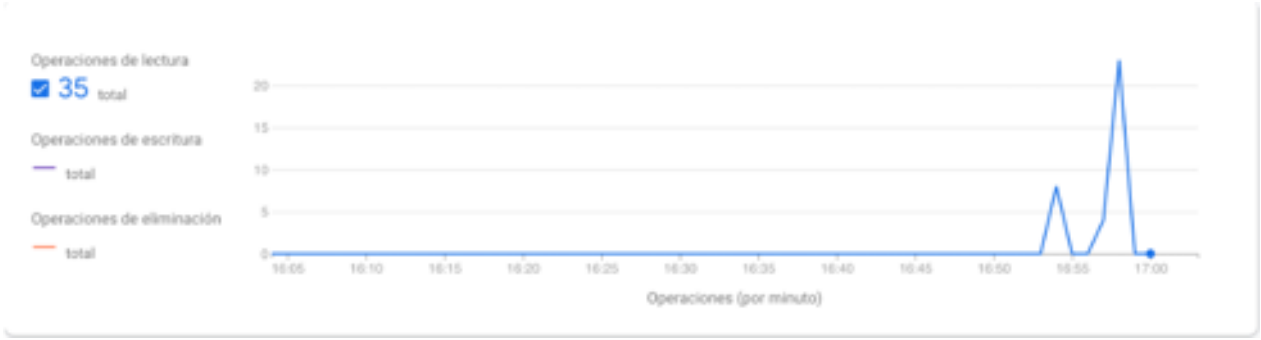


Figura 4.9: Operaciones de lectura en la base de datos de Firebase

La Figura 4.9 contiene las operaciones de lectura en la base de datos de la aplicación, en ella se pueden ver dos picos de actividad que corresponden al

uso de la aplicación. El primer pico corresponde a la solicitud de las categorías generado cuando el usuario abre la aplicación y el segundo pico corresponde a la solicitud de los servicios de la categoría 'Legal'.

En las Figura 4.13a y 4.13b se pueden ver las pantalla que genera estas dos operaciones de lectura, se observa como la hora del dispositivo coincide con los picos lecturas en la base de datos. En la Figura 4.13a son las 16:53 que coincide con el primer pico de actividad de la Figura 4.9. El segundo pico coincide con la hora del la Figura 4.13b que es las 16:57. Además, se observa que el numero de solicitudes de lectura del segundo pico es mayor que el primero, esto es debido a que hay mas cantidad de datos requeridos para desplegar todos los servicios de la categoría 'Legal' que los datos requeridos para desplegar la lista de tipos de servicios.

(a) Captura de pantalla de la acción que provoca el trafico de la API a las 16:53

(b) Captura de pantalla de la acción que provoca el tráfico de la API a las 16:57

- **Integración Flutter - Google Maps**

Para esta sección de resultados se analiza el tráfico por la API de Google Maps ligada a la aplicación, cuando los usuarios de la aplicación usan el

mapa interactivo debe de haber tráfico en esta API. En este caso, las APIs de Android e iOS son distintas y por tanto tienen distintos tráficos, en esta sección se analiza únicamente la de iOS pero los resultados pueden ser extrapolados a la de Android.

Para evaluar el tráfico por la API, se usa la interfaz de usuario de la aplicación y en concreto el mapa interactivo, una vez hecho esto se guarda la hora, para poder analizar si hay trafico en la API a esas hora concreta.

Figura 4.11: Tráfico por la API de Google Maps para dispositivos iOs

En la Figura 4.11 se muestra el trafico por la API de Google Maps para dispositivos iOS, en ella se puede ver como existe tráfico alrededor de las 16:17, ese tráfico es generado por el uso del mapa interactivo por parte de un usuario de la aplicación. En la Figura 4.12 se puede ver una captura de la acción que estaba generando ese tráfico y la hora de la captura coincide con el tráfico por la API 16:17. Por tanto, se concluye como resultados que la integración Google Maps - Flutter esta realizada de manera correcta y que el uso del mapa desde la aplicación se traduce en trafico por la API de Google Maps.

Figura 4.12: Captura de pantalla de la acción que genera el trafico por la Google Maps API

4.1.4. Accesibilidad

En esta sección de los resultados se detalla como responde la aplicación a las funciones de accesibilidad de los dispositivos móviles. Las funciones de accesibilidad están diseñadas para ajustar el sistema operativo a diferentes necesidades de oído, visión o motrices. Tanto en sistemas operativos Android como iOS existen este tipo de funcionalidades.

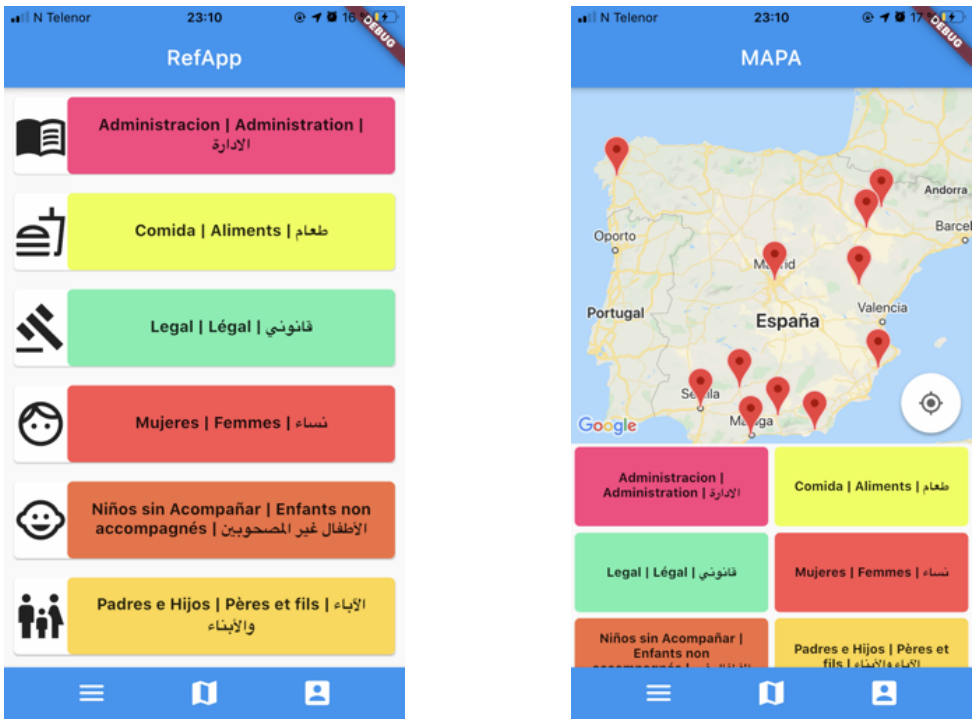
En concreto se va a evaluar como responde la aplicación a la funcionalidad llamada *Voiceover* del sistema operativo iOS, esta funcionalidad permite a personas con dificultad de visión, oír todo lo que ocurre en la pantalla del dispositivo para que pueden usar de manera cómoda el móvil y sus aplicaciones.

Tras llevar a cabo varias pruebas se obtienen como resultados que la funcionalidad *Voiceover* funciona en esta aplicación, al navegar por los diferentes componentes, el sistema va dictando lo que ve en la pantalla haciendo posible usar la aplicación con niveles bajos de visión e incluso sin ninguna visión .Además se concluye que el diseño de la aplicación de cara a esta funcionalidad podría ser

mejorado. Aunque es posible utilizar la aplicación con problemas de visión, ya que al cambiar de pantalla el *Voiceover* dicta el titulo y esto proporciona al usuario una manera de orientarse dentro de la aplicación, sería conveniente incluir texto junto a los iconos de la barra inferior de navegación, así el usuario con problemas de visión sería capaz de cambiar de sección más fácilmente. El *Voiceover* le podría dictar que significan esos iconos, haciendo así más fácil el uso de la aplicación.

4.1.5. Uso de la aplicación sin conexión a Internet

El público objetivo de esta aplicación, son personas que realizan una migración de un país a otro, esto hace bastante habitual que los usuarios no tengan conexión a Internet en el momento en el que se quiere acceder a la información sobre determinado servicio. Por ello, es crucial que la aplicación siga siendo útil cuando se pierde la conexión a una red de internet.



(a) Pantalla principal accediendo sin conexión a internet

(b) Pantalla del mapa accediendo sin conexión a internet

Figura 4.13: Uso de la aplicación sin acceso a internet.

En estos resultado se detalla como se comporta la aplicación tras una perdida de conexión, es decir, tras instalarse la aplicación y usarla, se va a desconectar tanto

la conexión wifi como la red 4G. En las Figuras 4.13a y 4.13b se puede comprobar como se ha desactivado cualquier conexiona internet, ya que en la barra superior no están ni el icono de wifi ni el icono de la conexión de red al lado del nombre del operador de red ("N Telnor").

Además, se corrobora en estas mismas figuras que la aplicación sigue funcionando aún habiendo perdido la conexión, funciona tanto las lista de categorías como el mapa interactivo permitiendo así al usuario de la aplicación acceder a la información sin conexión.

4.2. Pruebas de uso. Recogida de datos.

En esta sección de los resultados, se detallan la pruebas de uso de la aplicación, en ellas se detallan resultados sobre lo desarrollado en la sección 3.6 donde se decide qué datos son útiles para las organizaciones humanitarias, en temas de entender el flujo de migrantes y sus necesidades.

4.2.1. Usuarios cercanos

La primera prueba de uso, se realiza con usuarios cercanos que voluntariamente acceden a instalar la aplicación vía cable en sus dispositivos móviles. Con ello se puede ver si los datos se muestran en el Dashboard de Firebase de manera clara, para que las asociaciones en un futuro sepan entenderlos y analizarlos.

- **Localización geográfica de la audiencia**

  En estos resultados, las asociaciones pueden visualizar donde esta localizada geográficamente la audiencia de la aplicación. En la Figura 4.14 se ve como Firebase desde su Dashboard personalizable despliega un mapa donde se ve la distribución de usuarios al rededor del mundo, en este caso hay audiencia de la aplicación tanto en noruega como España e Italia, pero al ser usuarios meramente voluntarios estos datos en concreto no reflejan ninguna situacion real. Los datos geográficos vienen plasmados tanto en un mapa del mundo donde se resalta en azul la localizacion de los usuarios, como en un diagrama de barras donde se muestra la cantidad total de sesiones abiertas en cada país, y el porcentaje que representa cada una.

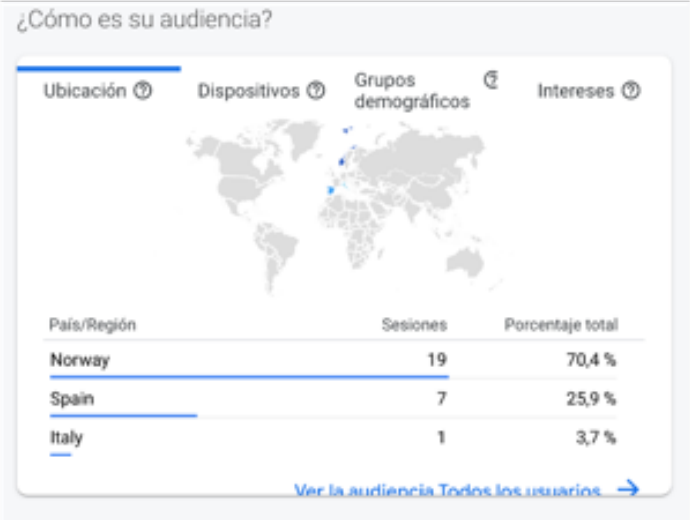


Figura 4.14: Localización geográfica de los voluntarios que usan la aplicación

- **Sistemas operativos**

  Con estos resultados es posible saber, qué porcentaje de la audiencia usa el sistema operativo Android e iOS, estos resultados también aparecen en el Dashboard personalizable de Firebase, en la Figura 4.15 se pueden ver estos datos para el grupo de usuarios voluntarios. En este caso la mayoría de usuarios son de Android en concreto el 83 % y el resto de iOS.

Figura 4.15: % de los usuarios que usan los distintos sistemas operativos para acceder a la base de datos

- **Visualizaciones de pantalla**

  En esta sección de resultados, las organizaciones son capaces de ver qué categorías son las más demandas por los usuarios, esto les puede ayudar a reforzar sus servicios en ese área concreta.

  En la Figura 4.16 se pueden ver estos datos con los usuarios voluntarios, cada pantalla es un *view_item* tal y como se ha definido previamente en el código de la aplicación y esto hace posible recoger los datos sobre visualización de pantallas. Con estas métricas se puede ver cuantas veces se a accedido a la *HomePage* que es la pantalla de inicio y a las distintas categorías, además de un porcentaje de visualizaciones de la pantalla en concreto en relación a las visualizaciones totales. Como es de esperar, el porcentaje más alto de visualizaciones lo tiene *HomePage* ya que es la pantalla de inicio y es por la que se pasa más veces. Dentro de las distintas categorías, la mas visitada es Comida, pero estos datos no son relevantes ya que las pruebas se han llevado a cabo con voluntarios ajenos a la situación de migración.

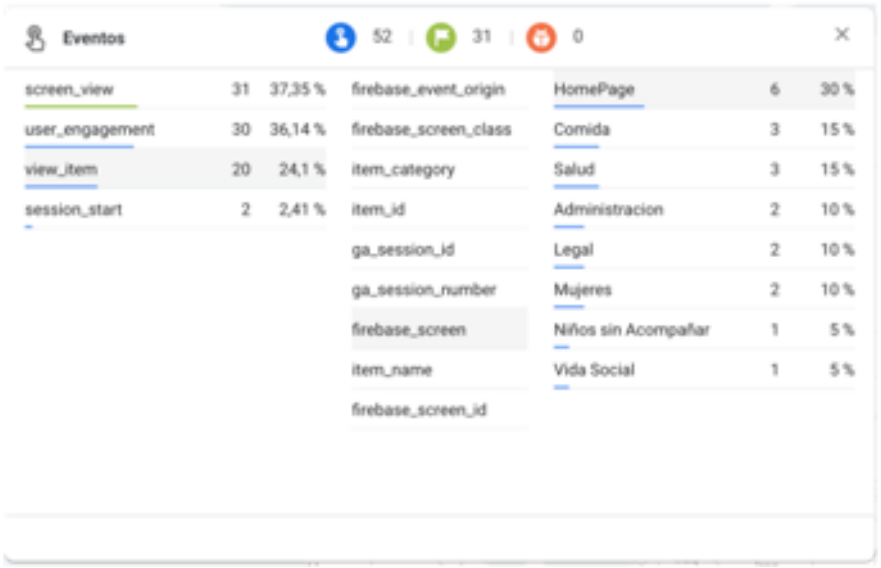


Figura 4.16: Número de veces que los usuarios se accede a las distintas categorías

Capítulo 5

Conclusiones

Este proyecto se ha terminado de manera satisfactoria y todos los objetivos iniciales se han cumplido. Está todo preparado para la subida de la aplicación a las plataformas oficiales de descarga y se ha preparado un manual para que la propia asociación realice el proceso de alta.

Se concluye que la aplicación multiplataforma realizada funciona correctamente en los sistemas operativos Android e iOS, en la sección 4.1.1 se demuestra que la interfaz de usuario se visualiza correctamente en ambos. Este desarrollo multiplataforma ahorra costes a la hora de implantar las aplicación en los mercados, ya que sólo se tiene que desarrollar un único código para móviles de cualquier sistema operativos.

La base de datos realizada en Firebase se rellena correctamente con los datos proporcionados por asociaciones humanitarias dedicadas a prestar este tipo de ayudas. En la sección 4.1.2 del documento se corrobora como la base de datos está creada y rellenada de manera satisfactoria, además en la sección 4.1.3 se comprueba que la base de datos está bien integrada en la aplicación. Firebase es una plataforma áltamente efectiva para la realización de aplicaciones móviles, su incorporación al desarrollo es áltamente eficiente y permite abstraer la complejidad del 'backend' durante el desarrollo de una aplicación.

Cabe mencionar, que tanto Firebase, como Flutter y Google Cloud Plataform pertenecen a la empresa; Google. Tras utilizar estos servicios en el proyecto, se concluye que usar plataformas de una misma empresa al desarrollar cualquier tipo de software es ventajoso, ya que la interacción entre las distintas plataformas esta muy bien diseñada y contiene abundante documentación.

Por último, se concluye que es posible extraer datos relevantes para las aso-

ciaciones humanitarias desde la aplicación realizada. Tal y como se detalla en la sección 3.6, no solo se hace uso de las funcionalidades básicas que proporciona Firebase, sino que se han desarrollado funcionalidades especificas, insertadas en el código de la aplicación, para poder obtener información relativa a la manera en la que se utiliza la aplicación. Con todo ello, se obtiene información muy valiosa sobre el uso de la aplicación por parte de migrantes y esta información se organizan en tres grupos de datos: la localización geográfica de la audiencia, el sistema operativo de los usuarios y el número de visualizaciones en cada tipo de pantalla. Estos datos pueden ser utilizados para mejorar los servicios disponibles a los migrantes, ya que se conoce un poco mejor las categorías más buscadas o de donde se necesitan esos servicios.

Capítulo 6

Trabajos Futuros

En esta sección se va a detallar varios puntos de mejora y extensión de funcionalidades de la aplicación.

Al empezar este proyecto, una de las metas que se pusieron fue subir la app a las plataformas de descarga y analizar datos de usuarios reales de esta aplicación. Finalmente, este objetivo no se ha podido cumplir porque la ONG que coordina los servicios a migrantes ha decidido posponer el lanzamiento, y por ello queda como desarrollo futuro llevarlo a cabo.

A parte de lo anterior, también existen puntos a mejorar en la programación de la aplicación. El primer punto de mejora, es la gestión de los idiomas en la aplicación, aunque tras la realización de este proyecto, la información crucial está disponible en los tres idiomas más importantes; español, francés y árabe, es recomendable que se realice una página de inicio donde el usuario pueda elegir cuál sería su idioma principal. Con esta funcionalidad los títulos se podrán mostrar en el idioma adecuado, ahora mismo están por defecto en español, además se puede implantar un lógica que muestre primero el idioma seleccionado por el usuario y debajo de ello el resto de idiomas. Esto permitiría expandir la aplicación a otras regiones del mundo.

El segundo punto a mejorar, es la pantalla de preferencias de usuario. Ahora mismo no tiene funcionalidades, pero sería conveniente poder gestionar el acceso a la localización y la configuración del idioma desde ahí. La idea sería, que el usuario pueda cambiar si quiere dar su localización o no lo que lanzaría una llamada a la aplicación de preferencias del usuario, y que hubiese un seleccionable donde elegir el idioma principal de la aplicación.

El siguiente punto a extender es el uso de la información sobre horarios y fechas

de apertura de los servicios, que se tiene en la base de datos. Las fechas de apertura, se pueden filtrar y crear una lógica que sólo exponga el servicio al usuario si se está en fecha de apertura. Además, con el horario del servicio se puede crear un icono que le informe al usuario si el servicio esta abierto o cerrado.

El ultimo punto de trabajo futuro está relacionado con el mapa interactivo. En este momento al seleccionar una de las chinchetas aparece un desplegable con el titulo de ese servicio, seria conveniente que ese desplegable tuviese un botón que redirija a la lista de servicios, y en concreto al cuadro que contiene la información de ese servicio en concreto. Además, sería bueno incorporar colores a las chinchetas de los servicios, y que cada chincheta fuese del color de su categoría, esto ayudaría a los usuarios a reconocer de manera visual que tipo de servicios existen cerca de ellos.

Capítulo 7

Alineación del proyecto con ODS

En esta sección del documento se va a comentar la alineación del proyecto con los Objetivos de Desarrollo Sostenible (ODS). Los ODS son un conjunto de objetivos globales que se acordaron en 2015 por la ONU para erradicar la pobreza, proteger nuestro planeta y garantizar la paz y prosperidad. Los ODS constan de 17 objetivos que pertenecen a la agenda de desarrollo del 2030, estos objetivos se desglosan en 169 metas concretas, que deben ser tratadas de cumplir tanto por los estados, como organizaciones y ciudadanos.

A continuación se van a mencionar que objetivos se alinean con este proyecto y porque:

- **Objetivo 1: Poner fin a la pobreza en todas sus formas en todo el mundo**

  Uno de los mayores motivos de la inmigración hacia España, es el objetivo de huir de la pobreza, muchos de los inmigrantes viene de países subdesarrollados o menos desarrollados que España, a buscar una forma de vida mejor. Además, los acontecimientos actuales de la pandemia del covid genera aún mas pobreza y en concreto desigualdad, los países desarrollados consiguen vencer la pandemia de manera mas rápida e eficiente que el resto, por tanto, la migración tenderá a aumentar en los años posteriores.

  Esta aplicación puede brindar ayuda a los migrantes que por intentar huir de una situación de pobreza, a veces terminan en una situación de mayor riesgo para su vida. Esta aplicación no fomenta las migraciones ni el tráfico ilegal de personas, pero permite ayudar a las personas desplazadas a recibir los servicios esenciales que necesitan y favorece su inclusión y protección en el país destino. Esta APP aproxima a los migrantes información crucial para ellos que hasta ahora era difícil de encontrar.

- **Objetivo 5: Lograr la igualdad entre los géneros y el empoderamiento de todas las mujeres y niñas**

  Este objetivo marca la necesidad de erradicar la discriminación de las mujeres y niñas, que además son uno de los grupos mas vulnerables al realizar las migraciones. Las mujeres que provienen de regiones mas pobres sufren más la desigualdad a lo largo de su vida, muchas de ellas no tienen acceso a la educación y por tanto, se encuentran en clara inferioridad a la hora de entender sus derechos y de manejarse en un país distinto.

  Esta aplicación, al estar en varios idiomas puede ayudar a las mujeres y niñas a comunicarse con alguien que les pueda ayudar, pensando en este publico en concreto, la aplicación siempre tiene disponible la información en los tres idiomas cruciales, francés, árabe y español, para que puedan fácilmente pedir ayuda aunque no hablen el mismo idioma, simplemente enseñando la información que proporciona la APP se puede comunicar que servicio necesitan, donde esta y que características tiene.

  Además, dentro de la aplicación existe una categoría dedicada a mujeres, donde se encuentran servicios de ayuda contra la violencia de genero entre otros muchos servicios dedicados exclusivamente ayuda a las mujer.

- **Objetivo 16: Promover sociedades pacíficas e inclusivas para el desarrollo sostenible, facilitar el acceso a la justicia para todos y crear instituciones eficaces, responsables e inclusivas a todos los niveles**

  Según fuentes de las Naciones Unidas, en el año 2018 las personas que huyen de conflictos, guerras o persecuciones superó los 70 millones [30], una de las consecuencias de estos conflictos suele ser la migración huida de los ciudadanos a otros países, en este caso los migrantes son llamados refugiados. Muchos de ello, no conocen sus derechos en el país al que se mueven, y es importante que tengan a mano fuentes de información, o servicios que les ayuden a entender y saber que derechos y deberes tienen en su país destino. Esta aplicación acerca esos servicios a los inmigrantes ayudando así a facilitar el acceso a la justicia.

Capítulo 8

Anexos

8.1. Anexo I: Requisitos y programas para gestionar la aplicación

En este Anexo se detallan que programas y cuentas es necesario tener para poder gestionar la aplicación posterior a este proyecto.

Android Studio

El IDE utilizado para el desarrollo del código en Flutter es Android Studio, si se quiere hacer alguna modificación posterior al código es necesario tener instalado este programa. Para su instalación se accede a este link:
`https://developer.android.com/studio?hl=es`

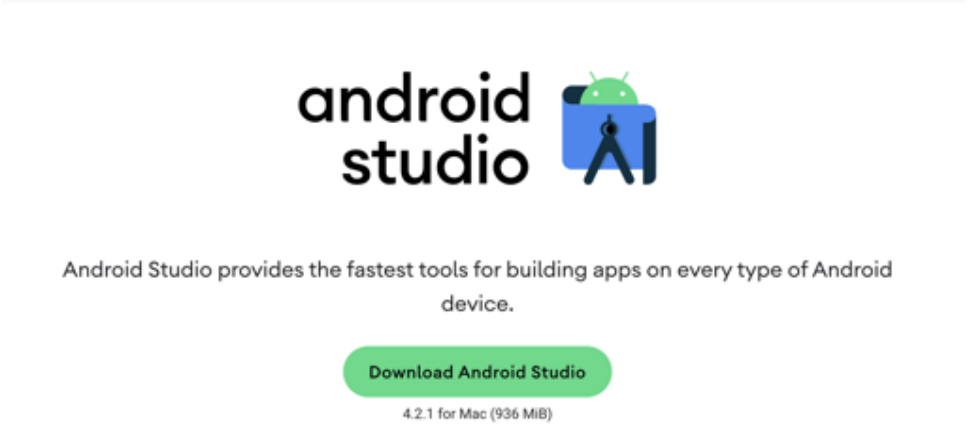


Figura 8.1: Pantalla para la descarga de Android Studio

Una vez en la pagina se muestra algo parecido a la Figura 8.1 se hace clikc en 'Download Android Studio' y se procedera a descargar los archivos de descarga del IDE.

Visual Studio Code

Para cargar datos desde los archivos de Excel a la base de datos de Firebase, es necesario un IDE que permita modificar y compilar código de Javascript. El IDE necesario es Visual Studio Code, y para su descarga se accede desde el siguiente link:

`https://code.visualstudio.com/download`

Figura 8.2: Pantalla para la descarga de Visual Studio Code

Una vez en la pantalla de descarga se muestra algo parecido a la Figura 8.2, donde habrá que seleccionar la descarga en función del tipo de sistema operativo que se use.

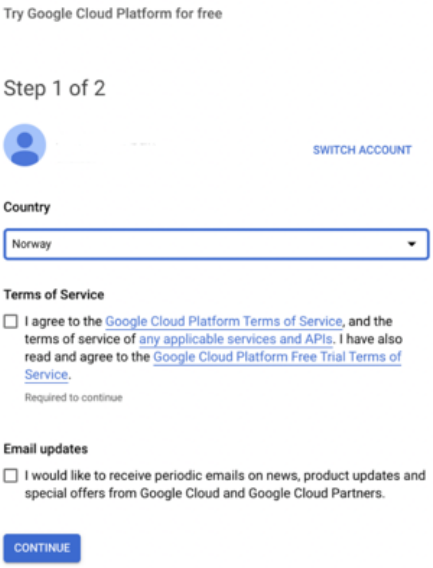
Firebase

La base de datos esta creada en la plataforma de Firebase, para poder gestionarla es necesario tener una cuenta de gmail. El link del servicio es el siguiente:

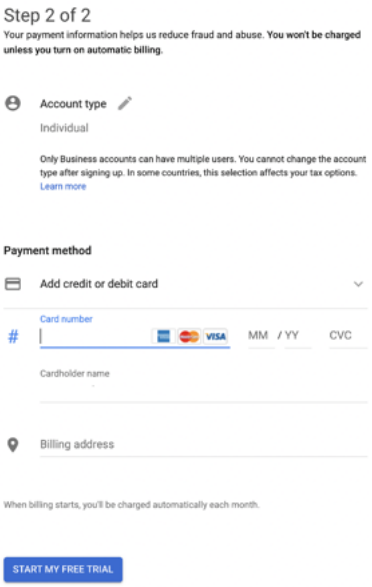
`https://firebase.google.com/`

Google Cloud plataform

Al igual que la plataforma anterior, para acceder a Google Cloud plataform, es necesario tener una cuenta de gmail. Además, para iniciar el periodo de prueba gratuito es necesario tener una tarjeta de crédito vinculada en la cuenta.



(a) primera pantalla para darse de alta GCP



(b) segunda pantalla para darse de alta GCP

En la Figura 8.3a y 8.3b se pueden ver los pasos a seguir, en el primero es necesario elegir un país desde donde se trabaja y aceptar los términos y condiciones y en el segundo se tienen que rellenar las datos de la tarjeta de crédito asociada.

8.2. Anexo II: Manual para la creación de cuentas de desarrollador en Google Play y Apple Store

Una vez creada la aplicación el siguiente paso es la subida de la misma a las plataformas de descarga, para que los usuarios se la puedan descargar en sus dispositivos. Al ser una aplicación multiplataforma (Android/iOS), es necesario subirla tanto a Google Play como a App Store y para ello hay que tener una cuenta de desarrollador, a continuación se explica como llevar esto a cabo.

Google Play

Para empezar con las explicaciones, se detalla una lista de cosas que es necesario tener para poder realizar el proceso en Google Play:

- Cuenta de Gmail
- Tarjeta de crédito
- Nombre publico de desarrollador (vale cualquier nombre a elegir)
- Dirección secundaria de correo electrónico (no tiene porque ser gmail)
- Numero de teléfono

Para crear esta cuenta se accede al siguiente link: `https://play.google.com/console/u/0/signup` Una vez dentro del link se tienen que realizar dos pasos:

1. Facilitar datos del usuario:

   En la Figura 8.4 se puede ver la primera pantalla para crearse una cuenta de desarrollador. En ella es necesario rellenar los campos resaltados en amarillo y seleccionar que se aceptan ambos términos y condiciones del servicio. Una vez completado se selecciona la caja azul donde dice Crear cuenta y pagar.

Figura 8.4: Imagen que detalla el primer paso para realizar la cuenta de desarrollador de Google Play

2. Facilitar datos de compra:

   Una vez facilitados los datos del usuario, es necesario proceder con el pago, la tarifa son 25 USD en total, es decir se pagan una vez y con ello se pueden

subir todas las aplicaciones que se quiera. En la Figura 8.5 se ven los campos necesarios a rellenar para realizar el pago, como en todas las compras por internet se necesita el numero de tarjeta de crédito, el nombre del titular y los datos del domicilio.

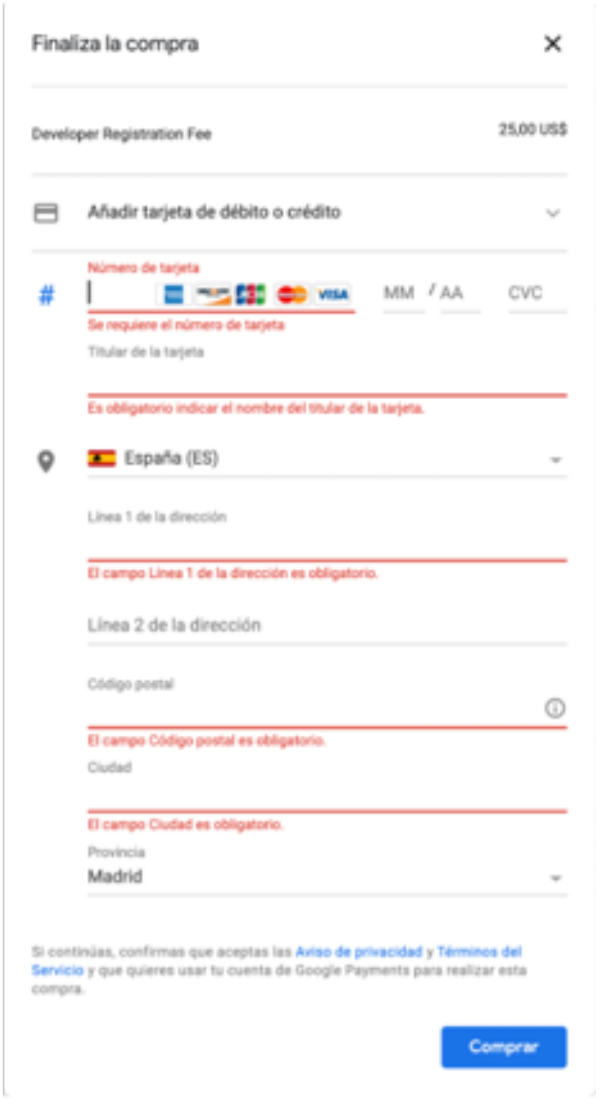


Figura 8.5: Imagen que detalla el segundo paso para realizar la cuenta de desarrollador de Google Play

App Store

Para empezar con las explicaciones, se detalla que es necesario tener para poder realizar el proceso de carga en App Store, los primero es crearse una cuenta de

Apple si no se tiene y lo segundo es asociar esa cuenta a una cuenta de desarrollador.

1. Crear Cuenta de Apple

   Para empezar, es necesario crear una cuenta de Apple, para ello es necesario tener la siguiente información:

   - Nombre y Apellidos del usuario
   - Fecha de nacimiento del usuario
   - Correo Electrónico
   - Contraseña
   - Numero de teléfono

   La cuenta se realiza accediendo al siguiente link:

   `https://appleid.apple.com/account?appId=632&returnUrl=https%3A%2F%2Fdeveloper.apple.com%2Fenroll%2F`

   Una vez se accede al link la pagina pide la información del usuario: nombre, email, contraseña, teléfono y país. Abajo del todo, vienen las opciones sobre que tipo de notificaciones quiere el usuario recibir, en la Figura 8.6 se pueden ver estas opciones, lo mas cómodo es no seleccionar ninguna si no se quiere estar recibiendo emails con publicidad.

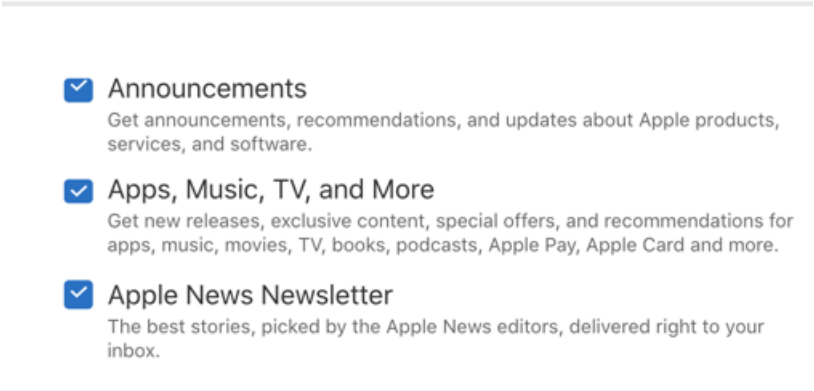


Figura 8.6: Imagen que detalla las opciones de notificaciones al realizar la cuenta de Apple

2. Crear cuenta de desarrollador

Tras terminar la cuenta de Apple, es hora de crear una cuenta de desarrollador, con la que poder subir aplicaciones al Apple Store. Para darse de alta como desarrollador en Apple hay dos opciones; darse de alta como persona física o como empresa, si se elige la opción de empresa durante la verificación de datos es posible que se pidan datos como el numero D-U-N-S, una autoridad vinculante o una pagina web de la empresa.

El link para acceder a darse de alta como desarrollador es el siguiente:

`https://developer.apple.com/programs/`

Figura 8.7: Imagen que detalla la pantalla de inicio para crearse una cuenta de desarrollador de Apple

Una vez en este link, la pantalla mostrada es la de la Figura 8.7 en la que se tiene que hacer click en la caja azul donde pone 'Enroll' para empezar el proceso. A continuación es necesario iniciar sesión con la cuenta de Apple creada anteriormente, y seguir los pasos detallados por el sistema. El costo de esta cuenta de desarrollador son 99 USD anualmente, es decir es necesario ir actualizando la cuenta de manera anual.

8.3. Anexo III: Manual para la configuración de categorías en la aplicación

En este anexo se va a explicar como llevar a cabo la configuración de las categorías en la aplicación, cada categoría va asociada a un color y un logo y esta información puede ser personalizada.

Para empezar, es importante mencionar que toda la información que se vuelca en la aplicación sobre categorías viene en el Excel con nombre 'data_categorias'. En la Figura 8.8 se puede ver el aspecto de este Excel, tiene cuatro columnas: id, Categoría, Color, Logo y cads fila es una categoria distinta.

id	Categoria	Color	Logo
1	Administracion	4294918273	58054
2	Legal	4285132974	57857
3	Padres e Hijos	4294956864	57756
4	Mujeres	4294922834	57753
5	Salud	4279828479	58051
6	Niños sin Acompañar	68718784064	57572
7	Comida	4293852993	57759
8	Vida Social	4283657726	58178

Figura 8.8: Extracto del Excel de data_categoria

El id es simplemente un número entero único que identifica la categoría, la columna categoría contiene el nombre de la categoría y Color y Logo son cadenas de números que representan tanto un color como un logo en Flutter, y se explicará como elegirlos mas abajo en este Anexo .

Elección de un color

Flutter tiene una clase Color, que permite usar distintos colores en las aplicaciones de la plataforma, cada color se distingue por un nombre y por un número hexadecimal, cualquiera de esas dos características vale para identificar un color.

La mejor manera de almacenar los colores de las categorías en Firebase es a modo de números, ya que si se quiere usar el nombre es necesario acceder a las propiedades de otra clase a la hora de volcarlo en flutter y se hace más difícil. Por tanto, se decide almacenar los colores como numero naturales.

En la Figura 8.9 se pueden ver algunos ejemplos de colores disponibles y la colección completa se puede consultar en el siguiente link:

`https://api.flutter.dev/flutter/material/Colors-class.html`

Figura 8.9: Ejemplo de colores en Flutter

Tras escoger un color se tiene que convertir el número hexadecimal que se ve a la derecha de la pantalla en un número decimal, existen conversores online que lo realizan, por ejemplo el numero 0xFFFF8A80 en decimal es 4294937216, el numero decimal es el que se inserta en la columna de Color en el documento Excel 'data_categoria'.

Elección de un logo

Al igual que con los colores, Flutter tiene una librería dedicada a los logos, cada logo se distingue o por un nombre o por un número, y en la base de datos de esta aplicación se almacenan los números de los logos.

El listado de logos de Flutter se pueden consultar en el siguiente link:

`https://api.flutter.dev/flutter/material/Icons-class.html`

En la Figura 8.10 se puede ver un extracto de esa pagina, debajo de cada logo viene una función como la siguiente:

```
IconData( 45674, fontFamily: 'MaterialIcons')
```

La información interesante de esta sección es el número entero que tiene IconData como primer argumento, este número identifica al logo y es el que se incluirá en el Excel mencionado anteriormente, bajo la columna de Logo.

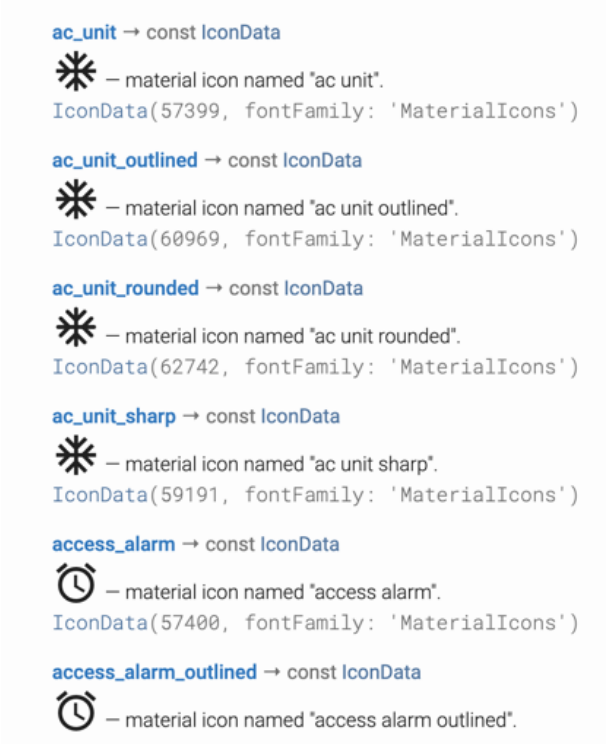


Figura 8.10: Ejemplo de logos en Flutter

Para terminar este manual, cabe destacar que una vez modificado el documento Excel, es necesario guardarlo con csv (Comma Separated Value) y convertirlo a JSON con un convertidor online sin modificarle el nombre. Para que estos nuevos datos se reflejen en la aplicación se tiene que actualizar la base de datos, para ello se guarda el archivo *data_categorias.js* en el directorio *Node.js*, sustituyendo al documento anterior con el mismo nombre. Una vez hecho eso, es necesario correr el código de index.js para que la base de datos se actualice. En la sección 3.3.2 se tiene mas información sobre este proceso.

8.4. Anexo IV: Manual para añadir nuevos servicios a la base de datos Firebase

Para añadir o modificar nuevos servicios en la aplicación existen dos opciones, la primera es hacerlo desde Firebase y la segunda es hacerlo en el documento Excel con nombre *data.xlsx* y luego volcar los nuevos datos a Firebase tal y como se hizo la primera vez. En este manual se va a explicar como llevar a cabo ambas maneras.

1. **Añadir o modificar servicios desde Firebase** En este caso únicamente se usa la interfaz de usuario de Firebase, en la pantalla de inicio de la plataforma se accede a la pestaña de Cloud Firestore, en la Figura 8.11 se puede ver en rojo donde esta esa pestaña.

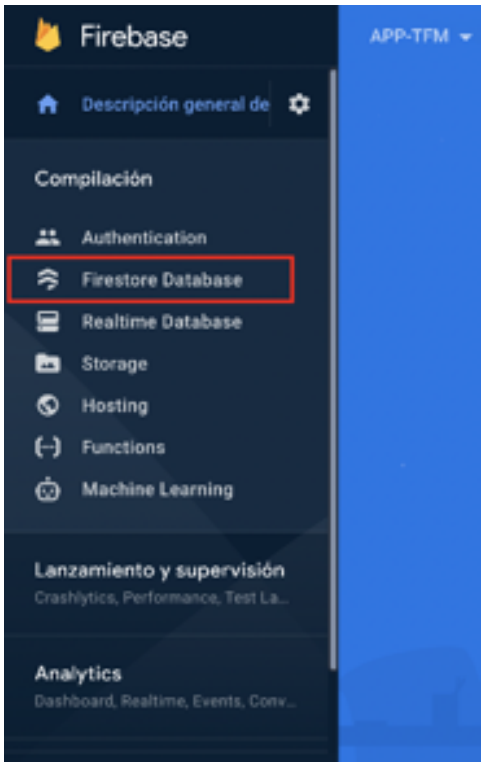


Figura 8.11: Seleccionable de la pantalla de inicio de Firebase, en rojo la opción a seleccionar para acceder a la base de datos

Al entrar en la base de datos, se navega hasta el servicio que se quiere modificar o la categoría en la que se quiere incluir un nuevo servicio. Una vez actualizada la base de datos, la aplicación de los usuarios se actualizará de manera automática.

Si se quiere añadir un nuevo servicio, se navega hasta la categoría en la que está englobado, se abre la colección 'servicios' de esta categoría y se crea un nuevo documento. Este documento debe tener un identificador único, la manera en la que se está configurando este 'id' es asignarle el siguiente numero entero libre, entre los id´s del resto de servicios pertenecientes a esta categoría. Es decir si el id mas grande que existe dentro de una categoría es diez, el id del nuevo servicio será once. Además, se tiene que rellenar los campos del servicio con los nombre de las variables exactamente igual a los del Excel.

Si se quiere modificar un servicio, se navega hasta el mismo, y se modifican las variables que tiene, es importante no modificar el nombre de las variables.

2. **Añadir o modificar servicios desde Excel**

   Para añadir un nuevo servicio o modificar uno existente desde el documento Excel es necesario, modificarlo, guardarlo en formato JSON y luego volcar otra vez todos los datos a la base de datos con los ficheros del proyecto Node.js.

5	4	Fegamp - Federación Gallega de Municipios y Provincias	Administracion	Migración	Información, orientación y asesoramiento sobre los recursos sociales, sanitarios, educativos; realización de itinerarios de inserción personalizados; sesiones informativas sobre rasgos culturales de la sociedad de acogida y sensibilización social. Encargada: Violeta Bernardo Vázquez
6	5	Fegamp - Federación Gallega de Municipios y Provincias	Administracion	Migración	Acogida: información y asesoramiento, clases de castellano y cultura gallega. Cobertura de necesidades básicas, integración social y cultural. Encargada: Inmaculada Diego Covelo
7	6	Fegamp - Federación Gallega de Municipios y Provincias	Administracion	Migración	Información, orientación, asesoramiento. Integración social. Emergencia social. Formación. Encargada: Ángeles Fernández Lago
8	7	Fegamp - Federación Gallega de Municipios y Provincias	Administracion	Migración	Integración inmigrantes: alfabetización, información, asesoramiento, jornadas, campamentos, intercambio cultural. Encargada: Genma Vilas Martínez
9					
10	1	Cáritas Orihuela-Alicante	Comida	Solicitud de alimentos	Distribución de alimentos.
11	1	Gobierno de Aragón	Legal	Asistencia jurídica	Servicio de asistencia y orientación jurídica a personas inmigrantes. Con cita previa.

Figura 8.12: Ejemplo de como añadir un servicio al Excel de datos

Si se quiere añadir un nuevo servicio, se inserta una fila a continuación del ultimo servicio de esa categoría, en la Figura 8.12 se puede ver como si se quiere insertar un nuevo servicio en la categoría 'Administración', se inserta una fila como la fila verde de la figura, esta fila se encuentra después del ultimo servicio disponible de la categoría 'Administración'. Tras esto, queda rellenar la fila con los valores correspondientes, es importante rellenar el cuadro de id con el numero entero siguiente al del servicio de arriba, es decir en el ejemplo de la figura el id debe ser ocho.

Una vez modificado el Excel, se guarda el archivo con el mismo nombre en formato csv y se convierte con un convertidor online a formato JSON. A continuación, se sustituye el fichero *data_UTF8* por este nuevo fichero con el mismo nombre, y se corre el fichero Index.js para actualizar la base de datos de Firebase. Una vez actualizada la base de datos los usuarios podrán acceder a esta nueva información.

8.5. Anexo V: Acceso a los datos de uso de la aplicacion en Firebase

Como se ha explicado en la sección 3.6 de este documento, existen numerosos grupos de datos, útiles a la hora de entender tanto el uso de la aplicación como el comportamiento de los migrantes. Para acceder a esos datos, se utiliza directamente la plataforma de Firebase. A continuación se va a detallar uno por uno que datos útiles se pueden acceder y cómo.

- **Localización Geográfica de la audiencia**

  En la Figura 8.13 se puede ver la pantalla donde se visualizan estos datos del proyecto, en concreto, se encuentran en la sección de *Analytics* en el subapartado de Dashboard. En la figura se puede observar el desplegable de secciones a la izquierda y el seleccionable de Dasboard esta de color azul porque esta seleccionado.

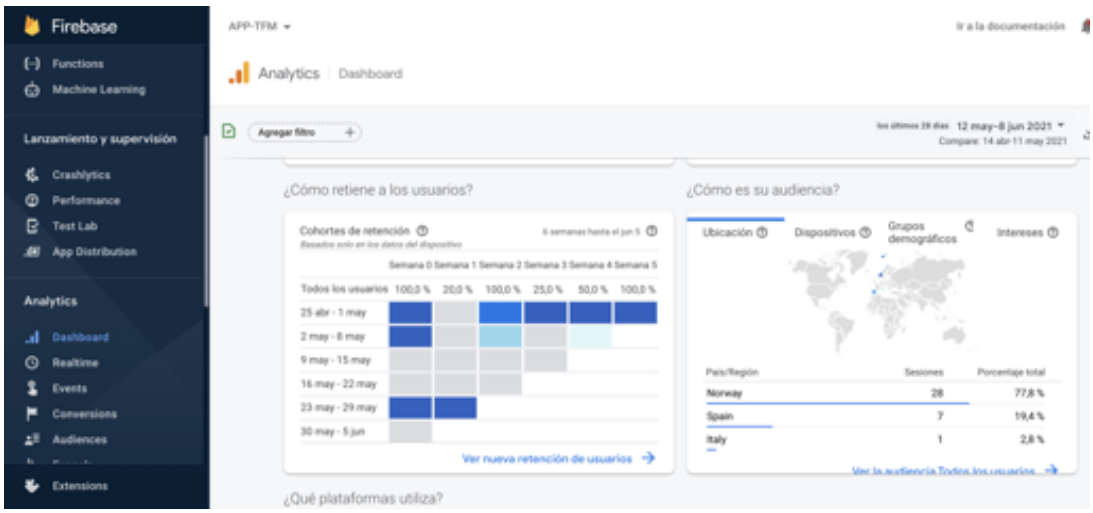


Figura 8.13: Sección de Firebase donde se encuentran los datos geográficos

- **Sistemas operativos de los usuarios**

  Al igual que en el punto anterior, estos datos se encuentran en el Dashboard de la sección de *Analytics*. En la Figura 8.14 se puede ver la pantalla que contiene estos datos, que al igual que antes tiene resaltado en azul la sección de Dashboard.

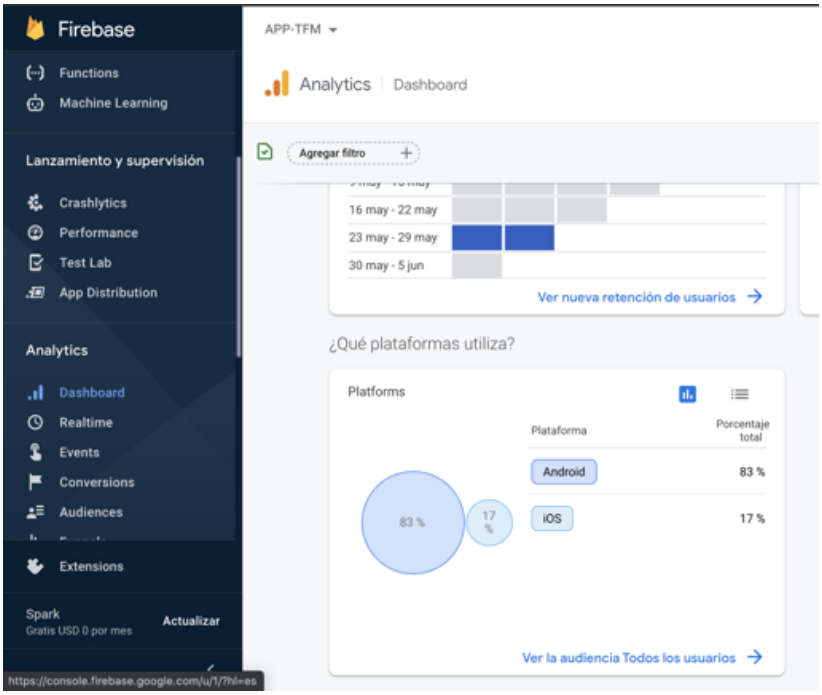


Figura 8.14: Sección de Firebase donde se encuentran los datos sobre sistemas operativos

- **Visualizaciones de pantalla**

Estos datos se han personalizado en el código de la aplicación, en concreto pertenecen al grupo de datos de tipo evento. En la Figura 8.15 se detalla donde están estos datos, en concreto están en la sección de *events* dentro de *Analytics*, en la figura se puede ver a la izquierda esta categoría resaltada de color azul.

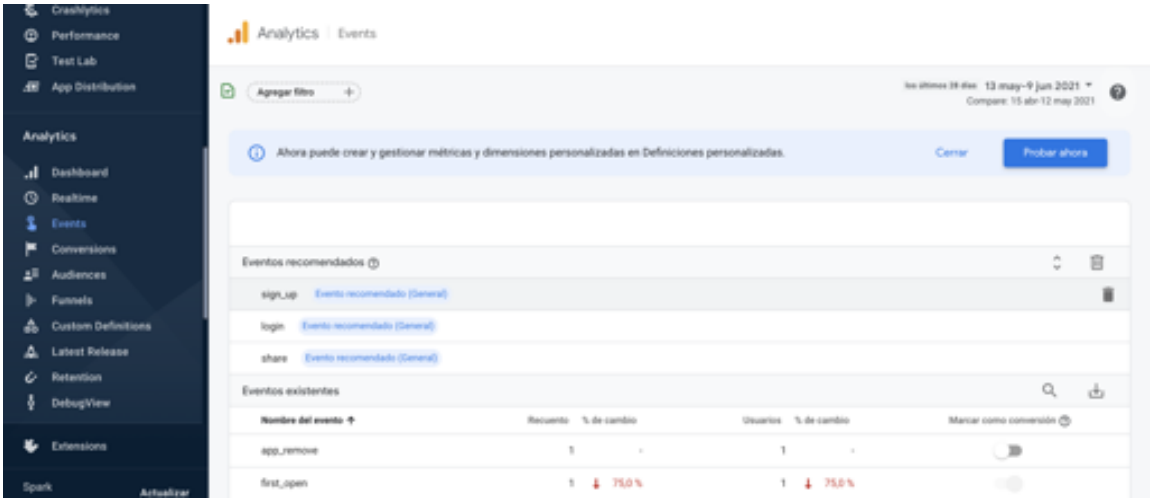


Figura 8.15: Sección de Firebase events

Una vez se ha accedido a la pantalla *events*, se busca dentro de la sección 'Eventos Existentes' el evento con nombre *ScreenView*, y se hace doble click sobre él. Una vez realizado esto, debería aparecer una pantalla como la de la Figura 8.16, es importante seleccionar en el desplegable a la derecha del titulo 'Interacción de los usuarios' el tema 'Nombre de pantalla'. En estos datos se puede ver que % de visualizaciones tiene cada pantalla, si se necesita mas información sobre cual es cada pantalla, viene explicado en la sección 3.6

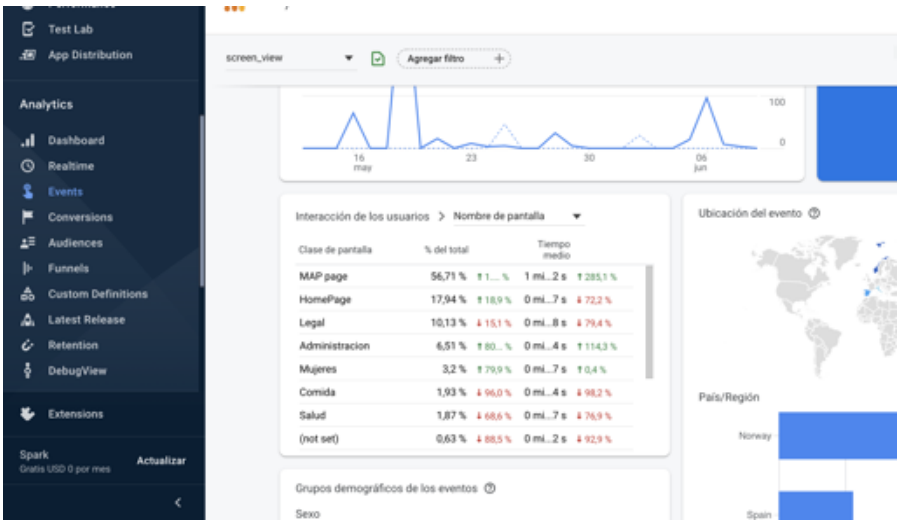


Figura 8.16: Pantalla de ScreenView en Firebase

Bibliografía

[1] Portal de datos mundiales sobre la migración. *Poblaciones de migrantes internacionales*. URL: https://migrationdataportal.org/es/themes/poblaciones-de-migrantes-internacionales.

[2] A. Perez Laureano. *Evolucion de la Tecnologia Movil*. URL: https://es.slideshare.net/pellokoto/evolucin-de-la-tecnologia-movil.

[3] Catalina Cruz Cuéllar. *¿Que es un dispositivo movil?* URL: https://arielrodrigoreyes.wordpress.com/que-es-un-dispositivo-movil/.

[4] Wikipedia. *Sistema operativo*. URL: https://es.wikipedia.org/wiki/Sistema_operativo.

[5] Mónica Mena Roa. *Android e iOS dominan el mercado de los smartphones*. URL: https://es.statista.com/grafico/18920/cuota-de-mercado-mundial-de-smartphones-por-sistema-operativo/.

[6] Brian Voo. *A look Into The History of iOS (And Its Features)*. URL: https://www.hongkiat.com/blog/ios-history/.

[7] Eve Porras. *Sistemas operativos móviles: iOS*. URL: http://eve-ingsistemas-u.blogspot.com/2012/04/sistemas-operativos-moviles-ios.html.

[8] Ibtisam Mohamed. «Android vs iOS Security: A Comparative Study». En: (2015). DOI: 10.1109/ITNG.2015.123. URL: https://ieeexplore.ieee.org/document/7113562/authors#authors.

[9] Nency Barron. *Blog de Programación de Aplicacines Móviles CBTIs73*. URL: http://nencybarrondam.blogspot.com/2016/02/arquitectura-del-sistema-operativo.html.

[10] *Aplicación Móvil*. URL: https://es.wikipedia.org/wiki/Aplicaci%5C%C3%5C%B3n_m%5C%C3%5C%B3vil.

[11] IONOS. *¿Diseño web móvil, responsivo o app?* URL: https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/pagina-web-movil-responsiva-o-app/.

[12] APPandWEB. *Principales tipos de apps: ventajas e inconvenientes*. URL: https://www.appandweb.es/blog/tipos-de-apps/.

[13] ABAMobile. *Apps multiplataforma. Qué son y sus características*. URL: https://abamobile.com/web/apps-multiplataforma-que-son-y-caracteristicas/.

[14] Eduardo Colmenares Shady Boukhary. «A Clean Approach to Flutter Development through the Flutter Clean Architecture Package». En: (2020). DOI: 10.1109/CSCI49370.2019.00211. URL: https://ieeexplore.ieee.org/document/9071367.

[15] David Molina. *¿Es Flutter el framework del futuro?* URL: https://www.bbvanexttechnologies.com/es-flutter-el-framework-del-futuro/.

[16] Guo-Ping Liu Xingwei Zhou Wenshan Hu. «React-Native Based Mobile App for Online Experimentation». En: (2020). DOI: 10.23919/CCC50068.2020.9189636. URL: https://ieeexplore.ieee.org/document/9189636.

[17] Shashikant Jagtap. *Flutter vs React Native: A developer's perspective*. URL: https://blog.codemagic.io/flutter-vs-react-native-a-developers-perspective/.

[18] Sebastian Novak Stefan Bosnic Ištvan Papp. «The development of hybrid mobile applications with Apache Cordova». En: (2020). DOI: 10.1109/TELFOR.2016.7818919. URL: https://ieeexplore.ieee.org/document/7818919.

[19] Apache Cordova. *Apache Cordova Documentation*. URL: https://cordova.apache.org/docs/en/10.x/guide/overview/.

[20] María Estela Raffino. *Qué es una base de datos?* URL: https://concepto.de/base-de-datos/.

[21] Google Cloud. *Bases de datos de Google Cloud*. URL: https://cloud.google.com/products/databases/?hl=es-NI.

[22] Doug Stevenson. *What is Firebase? The complete story, abridged*. URL: https://medium.com/firebase-developers/what-is-firebase-the-complete-story-abridged-bcc730c5f2c0.

[23] Firebase. *Elige una base de datos: Cloud Firestore o Realtime Databasee*. URL: https://firebase.google.com/docs/firestore/rtdb-vs-firestore.

[24] amazon Web Services. *Bases de datos en AWS*. URL: https://aws.amazon.com/es/products/databases/.

[25] RYTEWIKI. *Google Maps*. URL: https://es.ryte.com/wiki/Google_Maps.

[26] Marion MacGregor. *Italy: A refugee app, made by refugees*. URL: https://www.infomigrants.net/en/post/21397/italy-a-refugee-app-made-by-refugees.

[27] ROME REPORTS. *MigrAdvisor: la app que ayuda a los migrantes a encontrar centros de acogida*. URL: https://www.romereports.com/2018/05/19/migradvisor-la-app-que-ayuda-a-los-migrantes-a-encontrar-centros-de-acogida/.

[28] CaritasinMigration. *MigrAdvisor*. URL: https://inmigration.caritas.it/iniziative-eventi/migradvisor.

[29] IOM. *MigAPP*. URL: https://www.iom.int/migapp.

[30] PNUD. *Objetivos de Desarrollo Sostenible / Objetivo 16: Promover sociedades justas, pacíficas e inclusivas*. URL: https://www.un.org/sustainabledevelopment/es/peace-justice/.