Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo de Fin de Grado

Grado en Ingeniería Informática

BLOOXUM: Aplicación móvil de donación de sangre

Blooxum: A blood donation mobile app

Daniel Rodríguez Martín

La Laguna, 6 de septiembre de 2021

D. **Juan José Salazar González**, con N.I.F. 43356435D profesor Asociado de Universidad adscrito al Departamento de Matemáticas y Estadística e Investigación Operativa de la Universidad de La Laguna, como tutor

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

Que la presente memoria titulada:

"*BLOOXUM: Aplicación móvil de donación de sangre***"**

ha sido desarrollada bajo su dirección por D. **Daniel Rodríguez Martín**, con N.I.F 79062837-S

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 6 de septiembre de 2021

Agradecimientos

A D. Juan José Salazar González por su gran apoyo y ayuda en la toma de muchas decisiones a lo largo del desarrollo de este proyecto. A mis padres, por confiar en mí y apoyarme a lo largo de todo este recorrido, así como a mis abuelos que también se han preocupado constantemente por mi avance, sin todos ellos no estaría aquí.

También agradecer a mi hermano y mis amigos más íntimos, a Álvaro, Eduardo, David, Daniel, Marcos y Santiago, que también han sido pilares fundamentales en mi trayectoria universitaria.

Licencia

Resumen

A lo largo de este documento se presenta la información necesaria para la implementación de una aplicación móvil destinada a los servicios de hemoterapia, utilizando para ello un único lenguaje de programación denominado Dart, a través de un Kit de Desarrollo de Software denominado (SDK) Flutter, destinado a funcionar en cualquiera de las plataformas más utilizadas del mercado actual. Esta aplicación cuenta con una serie de funcionalidades básicas esenciales para facilitar el proceso de donar sangre en las Islas, como la localización de un puesto de donación, la comunicación con laboratorio para la obtención de los resultados de la analítica de sangre y la posibilidad de rellenar el formulario correspondiente que habilita al usuario a poder realizar la donación pertinente. Este servicio puede ser extensible a ser utilizado en cualquier parte del mundo.

En la fase de desarrollo, ha sido imprescindible realizar un profundo análisis sobre la existencia de aplicaciones de la misma índole que pudieran estar en el mercado actualmente.

Para poder implementar alguno de los servicios de la aplicación ha sido necesario obtener la geolocalización actual del usuario, así como la implementación de un registro de usuarios a través de algunas plataformas como google, facebook o apple id.

***Glosario de términos**: flutter, dart, google, facebook, sangre.*

Abstract

Throughout this document, you will find the information necessary for the implementation of a mobile application for hemotherapy services, using a single programming language called Dart, throughout a software development kit called Flutter, designed to work on any of the most widely used platforms on the current market. This application has a series of essential basic functionalities to facilitate the process of blood donate in the Islands, such as the location of a donation station, communication with the laboratory to obtain the results of the blood analysis and the possibility of filling in the corresponding form that allows the user to make the relevant donation. This service can be extended to be used anywhere in the world.

In the development phase, it has been essential to carry out an in-depth analysis of the existence of applications of the same nature that may be on the market today.

In order to implement any of the application services, it has been necessary to obtain the current geolocation of the user, as well as the implementation of a user registry through some platforms such as google, facebook or apple id.

Glossary of terms*: flutter, google, facebook, blood.*

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Antecedentes

Desde hace ya varios años hasta el día de hoy, se han ido desarrollando distintas aplicaciones que guardan relación con el ámbito o sector hospitalario. Por medio de las distintas tiendas digitales que existen actualmente en el mercado, es posible incluso hallar aplicaciones de donación de sangre. Las características son bastante similares entre ellas. No obstante, en las Islas Canarias no contamos actualmente con un servicio de este tipo.

Si bien es cierto que con las nuevas tecnologías los sectores responsables de la donación de sangre han tenido que ser actualizados y han mejorado considerablemente las formas en las que ofertan sus servicios, también es importante destacar que podrían unificarse todas y cada una de las funcionalidades que estos medios proporcionan, en una única aplicación, al alcance de todo el mundo.

1.2. Objetivos

Lo que se pretende lograr con esta aplicación móvil es poner a disposición de cualquier persona un servicio capaz de mostrar al usuario el lugar más cercano donde pueda donar sangre. Asimismo, proporcionar una sección de chat donde el usuario puede mantenerse en contacto con los laboratorios responsables de analizar la muestra de sangre donada. También el usuario puede ser capaz de mostrar su carnet de donante en base a los datos personales de dicha persona que previamente, ha debido registrarse en la aplicación, ya sea por medio de alguna red social o a través de un registro normal. Además, una vez el usuario está autenticado en la aplicación, podrá escanear un código QR localizado en el puesto de donación donde se encuentre y poder así rellenar el formulario de consentimiento que le habilita para proceder con la donación.

Para desempeñar todas estas funciones se ha trabajado utilizando el editor de código fuente Visual Studio Code, que nos permite instalar distintos plugins que permiten trabajar con mayor fluidez y eficacia, y corresponden al kit de desarrollo de software Flutter y el lenguaje de programación en el que está basado dicho SDK, denominado Dart. La aplicación cuenta además con un servicio de base de datos Firebase que permiten alojar distinta información como los usuarios registrados en la aplicación, ya sea de forma manual o a través de una red social, y por otro lado también nos permite almacenar las encuestas que realizan dichos usuarios al donar sangre. Por último, también es importante señalar que en la base de datos se

almacenan las ubicaciones reales de los puntos de donación, así como las conversaciones del chat.

Cabe destacar que, dadas las magnitudes del proyecto, basta con utilizar los recursos gratuitos que ofrece la plataforma de Firebase, ya que para el propósito inicial al que está destinado es más que suficiente para lograr un correcto funcionamiento de la app. En un futuro si sería interesante buscar otras aproximaciones de pago que permitan ofrecer un servicio mucho mayor.

1.3. Tecnologías utilizadas

Hoy por hoy, existen distintas alternativas con las que poder abordar determinados servicios online. En el mundo de la telefonía móvil, desde la aparición de los smartphones, han existido de la misma manera diversos lenguajes de programación para implementar aplicaciones, sin embargo, cada uno de estos lenguajes son vinculantes y de uso exclusivo para la plataforma móvil para la que han sido creados, de modo que es necesario utilizar varios de estos lenguajes para poder desarrollar una aplicación en los sistemas operativos más utilizados del mercado como son Android y iOS. Es por ello que desde hace apenas muy poco tiempo, surgió una alternativa a este problema, un kit de desarrollo de software denominado **Flutter** [1], que se caracteriza por ser de código abierto, y utiliza para ello el lenguaje de programación **Dart**. Además, otras de las principales características de este SDK estriban en su funcionamiento, que se basa en el uso de **Widgets**.

***Figura 1**. Logotipo de Flutter*

1.3.1. ¿Qué es un SDK?

Principalmente un SDK consiste en un kit de desarrollo de software que se traduce en un conjunto de herramientas de desarrollo de software que permite a un desarrollador crear una aplicación informática para un sistema concreto, por ejemplo, ciertos paquetes de software, entornos de trabajo, plataformas de hardware, computadoras, videoconsolas, sistemas operativos, etcétera.

1.3.2. ¿Qué es Dart?

Dart [2] es un lenguaje de programación de código abierto desarrollado por Google destinado a ofrecer mejores resultados mediante alternativas a algunos problemas de *JavaScript*, así como también ser una herramienta sencilla para proyectos más grandes y ofrecer una mejor seguridad.

Figura 2. *Logotipo del lenguaje de programación Dart*

1.3. Objetivos generales

Dentro de las distintas funcionalidades del proyecto, es importante distinguir cuales constituyen características principales y cuales representan finalidades específicas.
Los principales objetivos del proyecto son los siguientes:

- **Localización de puestos móviles de donación** → Implementación de un servicio mediante el uso de mapas que muestra al usuario los puestos móviles de donación de sangre que están disponibles en el momento de la consulta, permitiendo que éste pueda acceder al que más cerca se encuentra respecto a su posición, logrando ahorrar así tiempo para el usuario.

- **Notificar al usuario si hay carencias de sangre de su tipo** → Dado que en muchas ocasiones no hay stock de sangres de determinados tipos en los bancos de sangre de los hospitales, se dispone a crear un apartado en la aplicación que consiste en mostrar una alerta al usuario en caso de que la sangre de su tipo coincida con la que urgentemente se esté demandando en ese momento.

1.4. Objetivos específicos

A continuación, se definen aquellos objetivos cuya funcionalidad está centrada en resolver algunos problemas más concretos:

- **Notificar al usuario que puede volver a donar** → La idea de poner a disposición del usuario un apartado que le notifique que ha finalizado el período de espera para volver a donar sangre, le permitirá saber que ya puede volver a contribuir a la ayuda de muchas otras personas, habiendo pasado para ello 4 o 3 meses, dependiendo de si se trata de un hombre o una mujer, respectivamente.

- **Chat entre personal del sector y paciente** → Desarrollar un apartado en el que el paciente y el personal encargado de dar los resultados del análisis de sangre puedan comunicarse entre sí, con la finalidad de informar a dicho paciente sobre los resultados del análisis de sangre de su donación, alertando a este sobre el hallazgo de una enfermedad si se diera el caso.

- **Rellenar formulario de donación** → Permitir mediante la lectura de un código QR disponible en el lugar donde se realiza la donación, leer y rellenar el formulario de consentimiento que es obligatorio facilitar al personal sanitario que extrae la sangre a través de un apartado disponible en la aplicación, con el fin de facilitar los procesos administrativos y centralizar correcta y eficientemente la información privada del paciente. De esta forma además dados los tiempos que corren ahora con la pandemia del COVID-19, esto permite el mínimo contacto y contagio entre el personal sanitario que se encuentra constantemente en riesgo y las personas que van a donar, ya sea en un puesto móvil o en un hospital donde las posibilidades de contagio son más altas.

- **Carnet de donante** → Habilitar una sección donde se muestran los principales datos del donante, y que a través de la lectura de un código QR, el personal sanitario pueda consultar algunos datos relevantes para la donación, en caso de que así sea requerido por dicho personal.

Capítulo 2

Desarrollo de la aplicación

A lo largo de este capítulo se muestran las distintas herramientas, bocetos y funcionalidades que se han llevado a cabo en nuestra aplicación, así como los resultados visuales finales de la misma.

2.1. Herramientas

2.1.1. Visual Studio Code

Visual Studio Code [3] es un editor de código fuente desarrollado por Microsoft gratuito y de código abierto para Windows, Linux y macOS y se caracteriza por incluir soporte para la depuración, control integrado de Git, y resaltado de sintaxis, entre otras funcionalidades. Es también personalizable, por lo que es posible cambiar el tema del editor, los atajos de teclado y las preferencias, lo que la convierte en una gran herramienta para desarrollar nuestra aplicación.

***Figura 3**. Logotipo del programa Visual Studio Code*

El motivo de su elección estriba en la gran facilidad que proporciona a la hora de desarrollar aplicaciones en Flutter, ya que como se comentaba anteriormente, nos permite instalar diversas extensiones o plugins que facilitan la incorporación de código nuevo a nuestra aplicación, resaltando errores y aportando documentación en cada sección del código fuente de nuestra aplicación.

En concreto, para este proyecto ha sido necesario instalar las extensiones de **Flutter**, **Dart**, **Flutter Widget Snippets** y **Awesome Flutter Snippets**, siendo estas dos últimas las encargadas de proporcionar cierto atajos de teclado en la implementación de funcionalidades a nuestro código.

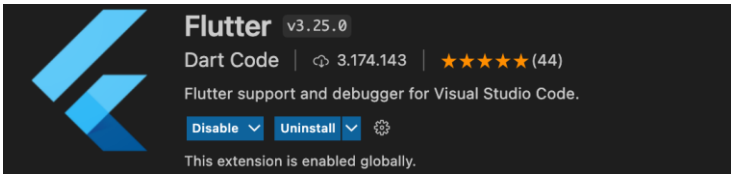

***Figura 4**. Extensión de Flutter para VSCode*

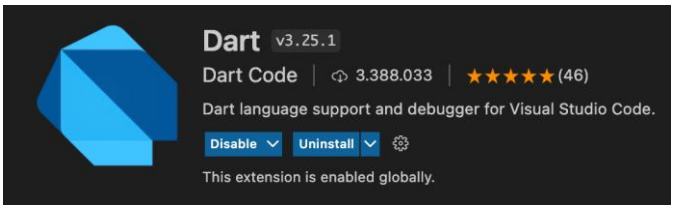

***Figura 5**. Extensión de lenguaje Dart para VSCode*

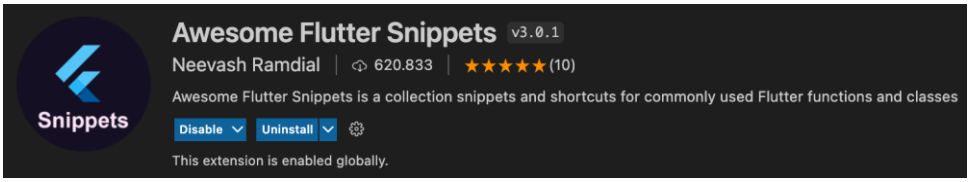

***Figura 6**. Extensión que nos permite crear atajos de teclado para implementar funcionalidades*

2.1.2. Android Studio (Simulador)

Android Studio [4] es el entorno de desarrollo integrado oficial para la plataforma Android que está basado en el software IntelliJ IDEA de JetBrains. Está disponible para las plataformas GNU/Linux, macOS, Microsoft Windows y Google Chrome OS y ha sido diseñado específicamente para el desarrollo de Android.

***Figura 7**. Logotipo Android Studio*

No obstante, en este caso esta herramienta se utilizará exclusivamente para poder visualizar nuestra aplicación en un simulador de un dispositivo con sistema operativo Android, con el fin de garantizar el correcto funcionamiento y visualización de los servicios implementados en dicha aplicación. En dicho emulador o simulador, son accesibles casi todas las funciones de un dispositivo Android real. Se pueden simular llamadas y mensajes de texto entrantes,

especificar la ubicación del dispositivo, utilizar diferentes velocidades de red, probar sensores de rotación y otros sensores de hardware, acceder a Google Play Store, entre otros.

2.1.3. XCode (Simulador)

Otro de los programas necesarios para desarrollar nuestra aplicación móvil es **Xcode** [5]**,** que consiste en un entorno de desarrollo integrado para el sistema operativo macOS que contiene un conjunto de herramientas creadas por Apple destinadas al desarrollo de software para macOS, iOS, entre otros.

***Figura 8**. Logotipo del programa XCode*

En este caso, este programa nos permitirá realizar diversas configuraciones en determinados apartados de nuestra aplicación destinada a funcionar en dispositivos con iOS, que no son posibles realizar en otros entornos de desarrollo como Visual Studio Code y, por supuesto, Android Studio.

Además, también se hará uso de una herramienta desarrollada paralelamente a XCode que nos permitirá visualizar de igual forma que con el emulador de Android, la versión de nuestra aplicación en dispositivos Apple.

***Figura 9**. Logotipo simulador XCode*

2.2. Diseño

Cuando se habla de diseño, es muy importante tener en mente siempre que la aplicación está destinada a cualquier tipo de usuario que desee, de una forma sencilla y rápida, donar sangre en cualquier lugar de las islas. Es por ello que el diseño de nuestra aplicación ha de ser preciso y sencillo para que pueda ser entendida sin ningún tipo de inconvenientes.

En primer lugar, creamos las pantallas o páginas de inicio de sesión y registro de usuarios. Para ello, se opta por definir un esqueleto que sigue algunos patrones básicos de diseño de aplicaciones móviles.

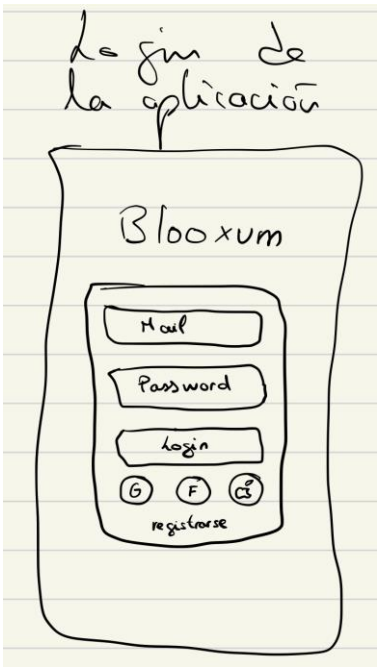

Figura 10. *Boceto del diseño de login de usuarios*

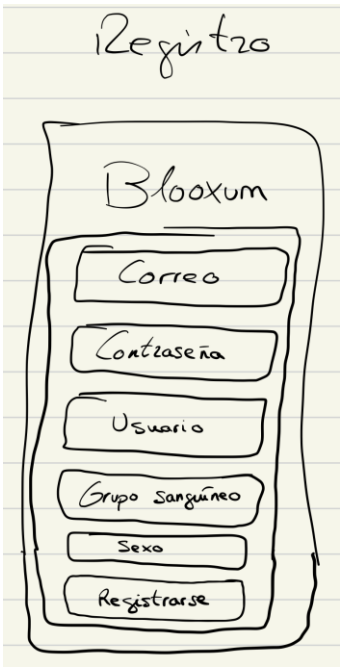

Figura 11. *Boceto del diseño del registro de usuarios*

Como se muestra en la figura 10, la validación de los datos de los usuarios de nuestra aplicación se basa en verificar simplemente su correo y contraseña. No obstante, el usuario tendrá la opción de registrarse o acceder en ocasiones posteriores a los servicios correspondientes a través de la validación de usuarios por medio de algunas redes sociales, tales como una cuenta de Google, una cuenta de Facebook o bien por el contrario una cuenta de Apple ID, tanto para usuarios de dispositivos iOS como para aquellos con dispositivos Android.

Igualmente, aquellos usuarios que deseen registrarse de una forma clásica, deben adjuntar los datos que se muestran en la figura 11.

A continuación, pasamos a hablar sobre el diseño de la página o pantalla que corresponde al **localizador**, visible en la figura 12, que se encargará de mostrar tanto los puestos móviles de donación como los puestos fijos. Para ello se sigue un esquema compuesto por una barra superior o también denominada *App Bar*, que además de mostrar el título de la página, pone a disposición del usuario un botón que le permite cerrar sesión.

Por otro lado, en el pie de la página se localizan las distintas secciones de nuestra aplicación, que de igual forma que el App Bar se mantienen de forma fija a lo largo de todas los apartados de la aplicación.

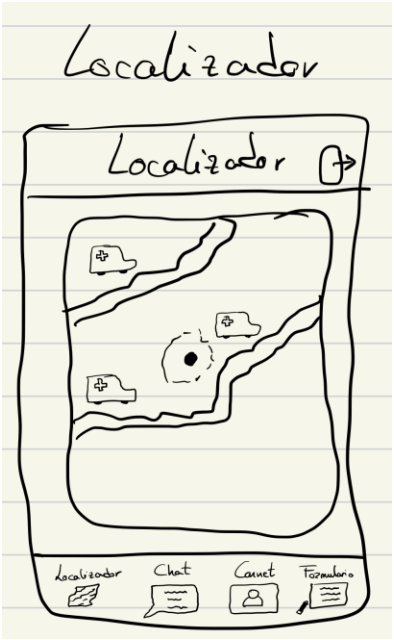

***Figura 12**. Boceto de la pantalla encargada de localizar los puestos de donación*

Pasamos ahora a explicar el esquema propuesto para la sección del **chat** que se muestra en la figura 13. En este caso se opta por un diseño que ya es comúnmente utilizado en otras muchas aplicaciones del mercado, con una estructura básica donde simplemente se muestran los mensajes que se intercambian entre el donante y el responsable de laboratorio encargado de suministrar los resultados de la donación al usuario, así como por supuesto una pequeña barra donde escribir el mensaje previo al envío.

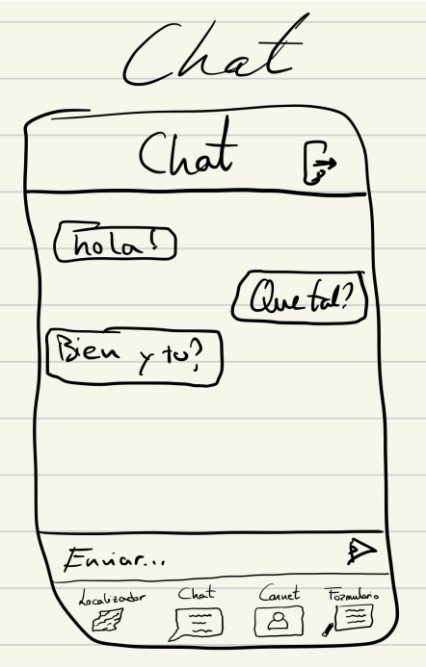

***Figura 13**. Boceto del chat*

En el boceto que se muestra a continuación en la figura 14, nos encontramos con el aspecto visual provisional para el apartado de carnet de donante del usuario. Realmente este apartado está pensando con el objetivo de cubrir dos servicios, donde una parte corresponde a la información relevante del donante que es sumamente útil e indispensable para el/la responsable del puesto de donación que se muestra en el carnet, y por otro lado un pequeño apartado anexo en la parte inferior de la pantalla que avisa al donante el tiempo restante para volver a donar. El resto de la página sigue el mismo patrón de diseño del resto de páginas.

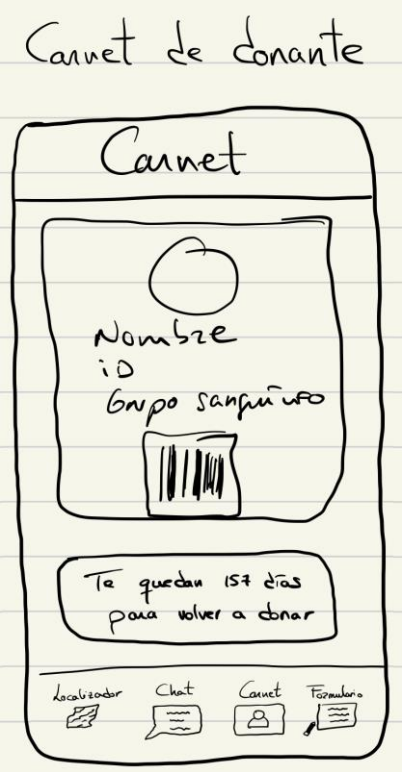

***Figura 14**. Boceto de la sección del carnet de donante*

Para acabar, tenemos como última sección de nuestra aplicación el apartado del formulario que es indispensable rellenar por parte del donante para habilitar su proceso de donación. En este caso continuamos con el mismo diseño para el AppBar y en la barra de navegación en la parte inferior, y en el cuerpo de la página se habilita un botón que permite al usuario acceder a un escaneo de un código QR disponible en el puesto de donación para rellenar las preguntas pertinentes que sirven para eludir cualquier riesgo que imposibilite donar al usuario.

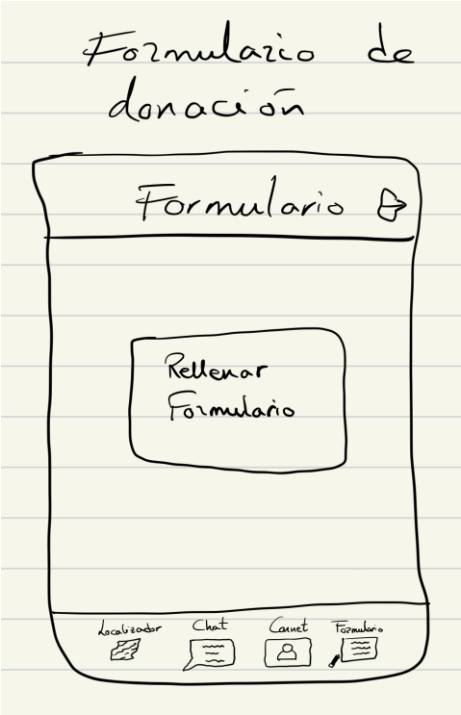

Figura 15. *Boceto para la sección del formulario de donación*

2.3. Funcionalidades e interfaz gráfica

2.3.1. Inicio de sesión

Es imprescindible tener en mente que las características del proyecto exigen establecer un mecanismo de validación para cada uno de los usuarios que deseen hacer uso de los servicios de la aplicación Blooxum. De modo que como se ha comentado anteriormente en el apartado de bocetos, la primera funcionalidad de nuestra aplicación móvil constituye al **inicio de sesión** o validación de las credenciales de aquellos usuarios ya registrados previamente en el sistema. Es por ello que, como se está mostrando en la figura 16, disponemos de varios campos tal y como se habían propuesto en los bocetos iniciales de la aplicación, donde cada uno de ellos corresponde al email utilizado para registrarse y la contraseña elegida por el

usuario, por supuesto, siempre y cuando el usuario se haya dado de alta en el sistema de forma convencional.

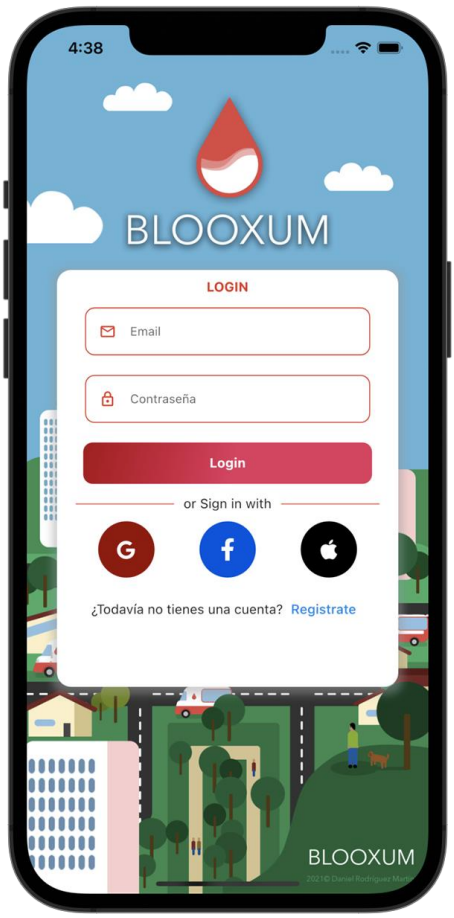

***Figura 16**. Login de Blooxum con Splashcreen personalizada*

Si por el contrario, el usuario ha elegido alguno de los métodos de validación alternativos, tendrá la posibilidad acceder con las credenciales pertinentes de su cuenta de **gmail** [6] , o su cuenta de **Facebook** [7] , así como también puede optar por validarse a través de su cuenta de **Apple ID** [8], estando esta funcionalidad disponible tanto para usuarios iOS como para aquellos usuarios con dispositivos bajo otros sistemas operativos, como puede ser **Android**.

El motivo por el que se ha decidido agregar este último método de autenticación está justificado en base a la nueva política de restricciones de nuevas aplicaciones que se suben a la tienda de aplicaciones móviles App Store, que se puso en funcionamiento a partir del año 2020.

La forma de validación es ligeramente distinta para ambos casos. En primer lugar, la validación de las credenciales a través del método convencional se basa en comprobar que el correo introducido por el usuario se encuentra disponible en la base de datos de la aplicación.

Por otro lado, la contraseña es validada a través de un token que se genera al registrar dicho usuario previamente en la aplicación. A diferencia de este método, para la validación de un usuario a través del uso de sus redes sociales, simplemente se comprueba el token que previamente es generado y asignado a cada cuenta individual, ya sea de Google, Facebook o Apple. El procedimiento de comprobación de la existencia de los correos electrónicos utilizados en cualquiera de las cuentas de las redes sociales se realiza automáticamente por medio de la API ya desarrollada de forma nativa en el SDK de Flutter.

En cuanto a la interfaz gráfica utilizada en este apartado, se ha optado por un fondo que transmite a primera vista tranquilidad y armonía al usuario, diseñado por **Danielson Design** [9]. Esto es realmente importante para satisfacer una buena *User Experience (UX)*, ya que en algunas ocasiones muchas personas tienen algunas preocupaciones a la hora de donar sangre, el procedimiento se puede hacer algo estresante en determinadas situaciones. Es por ello que, no solo se pretende facilitar en tiempo y dificultad el proceso de donación, sino también desarrollar cierto interés en las personas para motivarlos a donar sangre en futuras ocasiones.

Por otro lado, los campos de *correo electrónico* y *contraseña* siguen el estándar de bordes redondeados, delimitados por colores asociados al color rojo característico de la sangre, aunque en un tono mucho más claro, siguiendo siempre las premisas comentadas anteriormente. Ocurre lo mismo con el botón de *Login*.

2.3.2. Registro

Para el apartado de registro, seguimos las mismas bases de diseño que mostramos en la página de inicio de sesión, salvaguardando detalles que no corresponden a este apartado, como son el uso de las redes sociales, ya que en esta parte de la aplicación está destinada exclusivamente a registrar de forma clásica a los usuarios en el sistema.

Sin embargo, el funcionamiento es ligeramente distinto, ya que en este caso el campo del correo electrónico cerciora que ese email sigue unos patrones específicos, y lo hace en base a una expresión gráfica predeterminada.

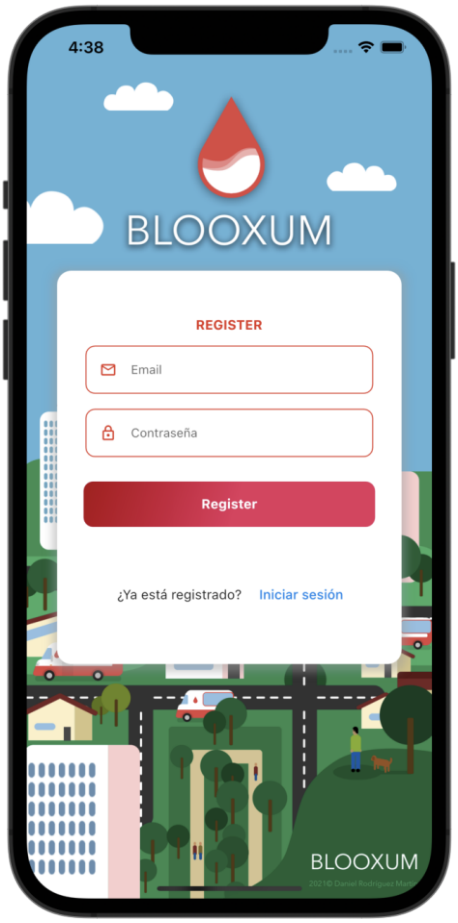

***Figura 17**. Registro de Blooxum con Splashcreen personalizada*

2.3.3. Localizador de puesto de donación

Pasamos al apartado de *tracker page* o **Localizador**. Tal y como adelantamos en el apartado de diseños, las barras de navegación superior e inferior se mantienen como en el resto de secciones. En este caso, lo que cambia por supuesto es el cuerpo de la página, que en este caso muestra un recuadro con bordes redondeados, una vez más siguiendo dicho patrón escogido por defecto para nuestra aplicación.

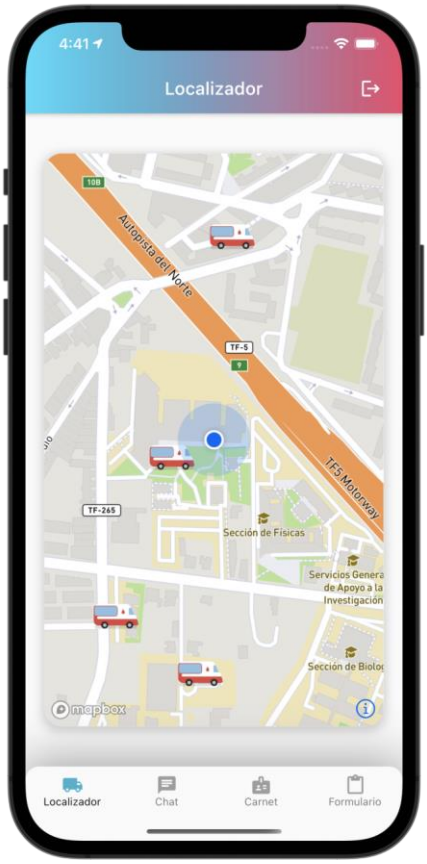

Figura 18. *Localizador de puestos móviles*

En dicho cuerpo encontramos un servicio de mapas que muestra a su vez la localización actual del usuario, cuyo consentimiento se pide al usuario cuando inicia sesión por primera vez en la aplicación, como se muestra en la figura 19. De esta forma, se observan alrededor los distintos puestos móviles o fijos de donación de sangre que puedan estar cerca de la posición geográfica del usuario.

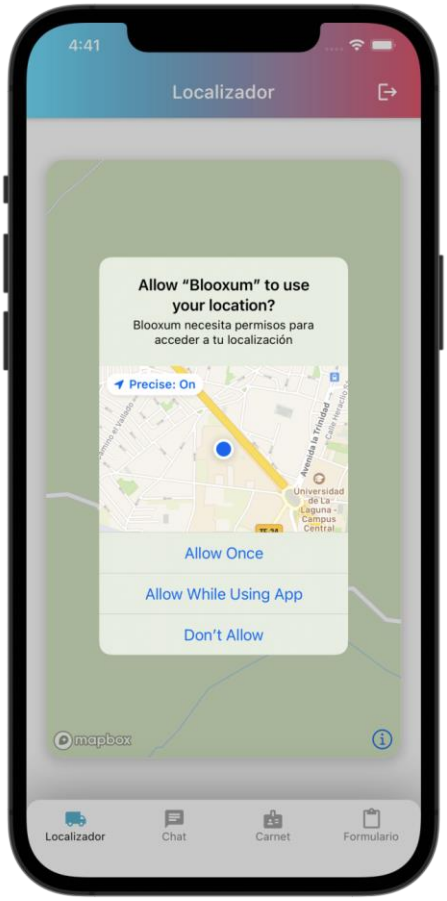

Figura 19. *Localizador de puestos móviles. Permiso de ubicación del usuario*

Al pulsar en cualquiera de los iconos de puesto de donación, se indica al usuario por medio de un recuadro blanco que sale de la parte inferior de la pantalla, un pequeño texto informativo sobre la ubicación exacta del puesto.

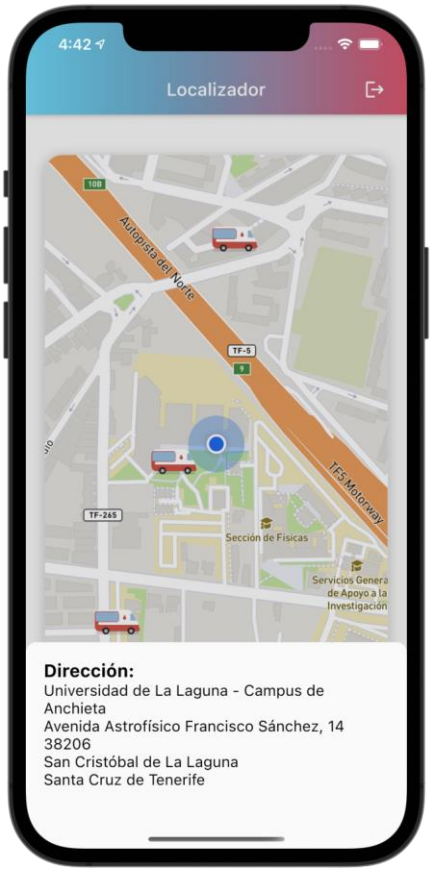

***Figura 20**. Localizador de puestos móviles: información de la ubicación del puesto*

El mapa utilizado para la aplicación trabaja bajo los servicios de **MapBox** [10], un proveedor de mapas en línea que cuenta con ciertas mejoras significativas respecto a otros servicios del mismo sector, como pueden ser los mapas de Google, motivo por el cual se ha seleccionado para el proyecto.

2.3.4. Chat para comunicación entre los laboratorios y donante

Para la función del **chat**, se ha decidido utilizar todo el campo disponible del cuerpo de esta sección para mostrar todos los mensajes. En este caso, el funcionamiento que hay detrás de este campo funciona gracias a nuestro backend cuyo servicio está basado en Firebase, aspecto que se comentará en detalle en secciones posteriores.

Su funcionamiento principal consiste en seleccionar el nombre del usuario que está con una sesión iniciada en ese momento y situar dicho nombre anexo al mensaje que envía el usuario.

En cuanto a diseño, se ha optado por destacar el recuadro de los mensajes enviados con un color azul celeste claro, similar al color mostrado en el gradiente que se puede observar en la barra de tareas superior de la aplicación.

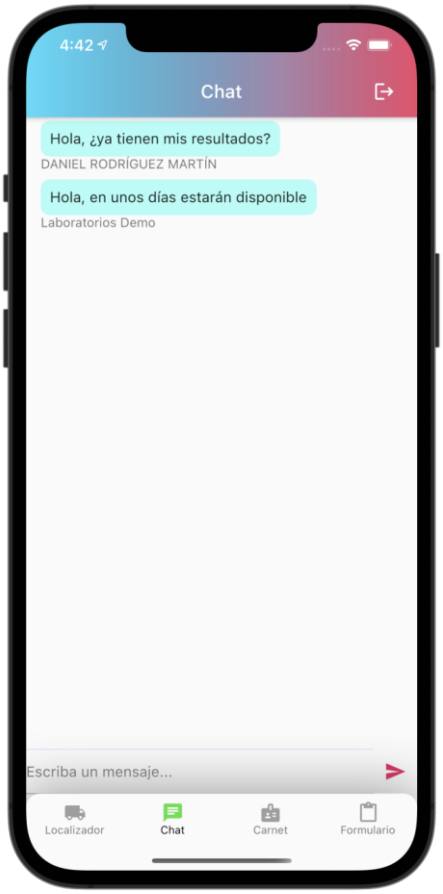

Figura 21. *Chat de Blooxum*

2.3.5. Carnet de donante

Es en esta parte de la aplicación donde el usuario será capaz de ver su información personal, información que se encuentra registrada y validada en la base de datos del sistema. La página de **carnet de donante** como se había comentado anteriormente, consta de un fragmento del cuerpo de la sección con los datos necesarios para el personal del puesto de donación, pues será aquí donde, a través del código QR disponible en la tarjeta, se pueden obtener toda la información que se muestra visible en el apartado.

Para esta página de la aplicación se ha pensado en utilizar un fondo en con distintos gradientes en tonos rojos con el fin de ofrecer una tarjeta estilizada y actual.

En cuanto al funcionamiento de esta tarjeta, cabe destacar que la información mostrada es recabada tanto de las cuentas ordinarias como de cuentas provenientes de redes sociales, siendo en este último caso gracias a una serie de tokens de los cuales es posible obtener todo tipo de información vinculada a esa cuenta (foto de perfil, nombre completo, correo, etc).

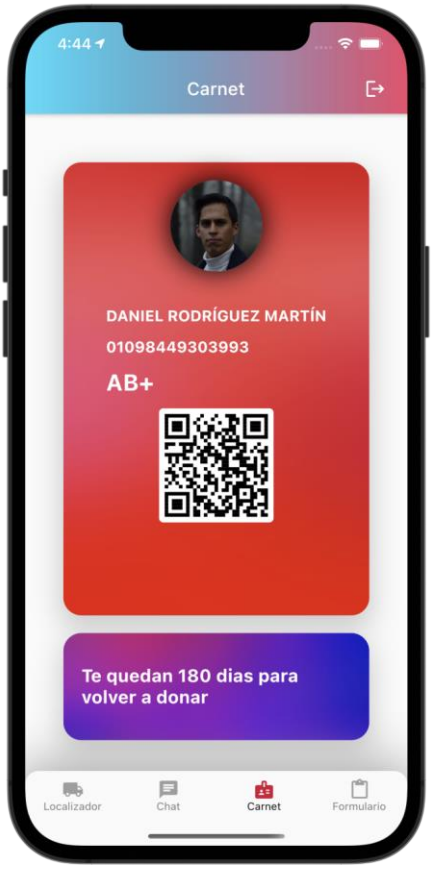

Figura 22. *Carnet de donante*

Asimismo, justo por debajo de dicho carnet encontramos la sección de aviso al donante, el cual muestra un contador anexo a un mensaje informativo para volver a donar en la fecha restante indicada. Para este caso, con el fin de hacer un pequeño contraste con el diseño mostrado en la zona superior del cuerpo de la página, se opta por utilizar distintos matices del color violeta, simulando como en ocasiones anteriores un pequeño gradiente.

El funcionamiento de este apartado se basa en obtener la fecha de la última donación que ha hecho el donante, fecha que queda registrada en la base de datos al rellenar correctamente el formulario de donación y enviarlo al sistema, y restarle dicha fecha sumada con **6 meses**, en el caso de que el usuario sea hombre, o **3 meses**, en caso de tratarse de una mujer.

Finalmente, se formatea el resultado en días para proporcionar una mayor legibilidad al usuario.

2.3.6. Formulario de donación

El **formulario de donación** abarca el último apartado de este proyecto. Consiste en la parte más sencilla, y a la vez más importante, del proceso de donación, ya que a través de página se informará al usuario si está capacitado para aportar su donación a la comunidad.

***Figura 23**. Formulario de donación (botón para el escáner)*

Tal y como se observa en la figura 23, observamos lo que anticipamos previamente en la parte de diseño de la aplicación. En este caso, se muestra en el cuerpo de la aplicación el botón para habilitar la cámara y escanear el código QR del puesto de donación pertinente. Para el diseño de este apartado, se opta por seguir el estilo explicado en ocasiones previas, basado en una serie de gradientes con colores que guardan cierta relación con los colores utilizados tanto en la *Splashscreen* de la aplicación como en el Appbar de la misma.

Al pulsar sobre dicho botón, y escanear el código comentado anteriormente, se consulta en la base de datos si la información almacenada en el código QR corresponde con algunos de los puestos de donación registrados en el sistema. En caso contrario, se mostrará un mensaje al usuario informándole de que el código no se corresponde con un puesto de donación vigente.

En caso de validarse correctamente dicho QR, se proporciona al donante el formulario, como el que se observa en la figura 24.

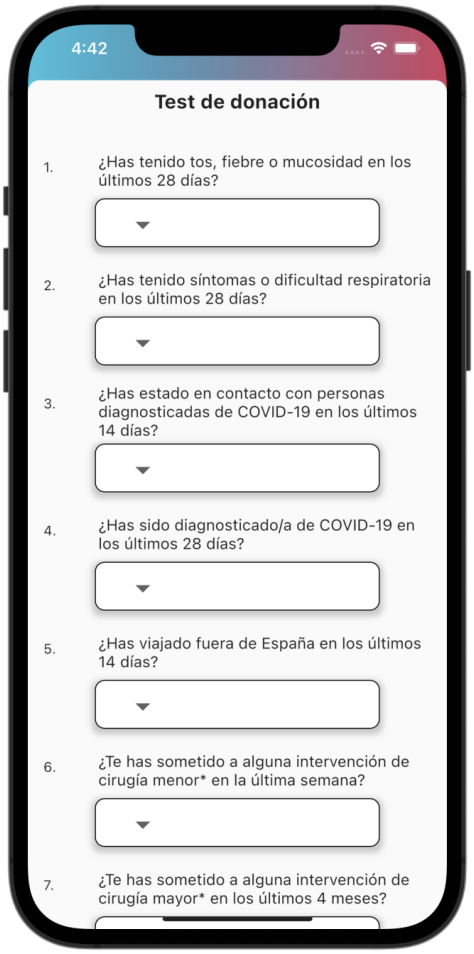

Figura 24. *Formulario de donación (preguntas)*

Es imprescindible que éste rellene todas las preguntas, que se componen de una pequeña lista con las posibles respuestas. En este caso basta con responder '*Si*' o '*No*'.

En el supuesto caso de que el/la usuario responda a cualquiera de las preguntas de forma afirmativa, automáticamente se cerrará el cuestionario indicando al usuario que no es apto para realizar la donación, tal y como se observa en la figura 25.

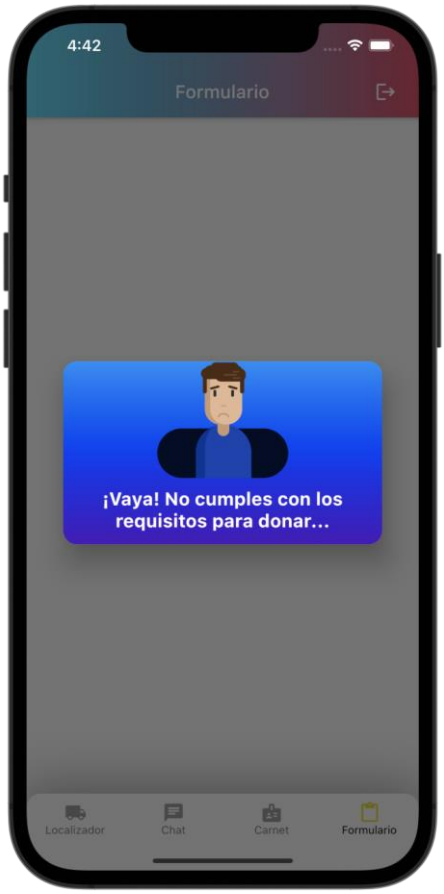

Figura 25. *Mensaje de alerta del formulario de donación*

Capítulo 3

Desarrollo del backend

3.1. Herramientas

En esta sección, se muestra la metodología utilizada para gestionar los datos de la aplicación. En esta ocasión se ha optado por utilizar una base de datos no relacional, en concreto se utiliza la plataforma de **Firebase**, la cual nos dota de distintas alternativas con las que poder manejar toda la información de nuestro proyecto.

3.1.1. Firebase

Firebase no es más que una plataforma para el desarrollo de aplicaciones web y aplicaciones móviles que se caracterizan por funcionar en la nube y que usa un conjunto de herramientas para la creación y sincronización de proyecto, haciendo posible el crecimiento del número de usuarios y dando resultado también a la obtención de una mayor monetización.

3.1.2. ¿Qué ventajas ofrece Firebase?

***Figura 26**. Logotipo de Firebase*

En comparación con otras alternativas, esta herramienta nos provee de ciertos beneficios como la fácil sincronización de los datos de un proyecto, sin tener que administrar conexiones o escribir lógica de sincronización compleja.

Además utiliza un conjunto de **herramientas multiplataforma**: se integra fácilmente para plataformas web como en aplicaciones móviles y además es compatible con grandes plataformas, como IOS, Android, aplicaciones web, etc. Por no hablar su infraestructura, ya que al estar desarrollada por Google permite hacer una escala automáticamente para cualquier tipo de aplicación, desde las más pequeñas hasta las más potentes.

No obstante, el factor decisivo por el que se ha escogido esta base de datos es la facilidad de poder crear proyectos **sin necesidad de un servidor**, ya que las herramientas se incluyen en los SDK para los dispositivos móviles y web, por lo que no es necesario la creación de un servidor para el proyecto.

Es gracias a todas estas funcionalidades por las que cualquier desarrollador puede combinar y adaptar la plataforma a medida de sus necesidades.

3.1.3. Firebase Firestore Database

Firebase ofrece distintas alternativas dentro de sus funcionalidades, adaptadas a las necesidades concretas de cada desarrollador. Actualmente existen dos posibles formas de definir nuestra base de datos si contamos con estos servicios:

→ **Realtime Database**
→ **Firestore Database**

3.1.4. ¿Qué caracteriza a cada una?

Las dos ofrecen prácticamente los mismos servicios, y son proveer SDK centrados en el cliente, sin servidores que implementar ni mantener, actualizaciones en tiempo real y proporcionar un pequeño nivel gratuito, donde posteriormente se paga por las funcionalidades extras que se deseen agregar.

Lo que realmente las diferencia es que la primera es la base de datos original de Firebase, y es una solución eficiente y de baja latencia destinada a las apps para dispositivos móviles que necesitan estados sincronizados entre los clientes en tiempo real. Mientras que la segunda es la base de datos más reciente de Firebase para el desarrollo de apps para dispositivos móviles, y aprovecha lo mejor de Realtime Database con un modelo de datos nuevo y más intuitivo.

De esta manera, conseguimos que con Cloud Firestore también se puedan realizar consultas más ricas y rápidas, y el escalamiento se ajusta a un nivel más alto que Realtime Database, motivo más que suficiente para decantarse por esta alternativa.

3.2. Funcionalidades

Pasemos a centrarnos ahora en las funcionalidades del servidor de nuestra aplicación. Lo primero que procedemos a implementar es la lógica necesaria para la autenticación de los usuarios de Blooxum.

3.2.1. Autenticación

Tal y como estamos observando en la figura 27, en la sección de autenticación de usuarios de Firebase encontramos todas las cuentas registradas en la aplicación, y donde a cada una de ellas se les ha asignado un identificador de usuario UID.

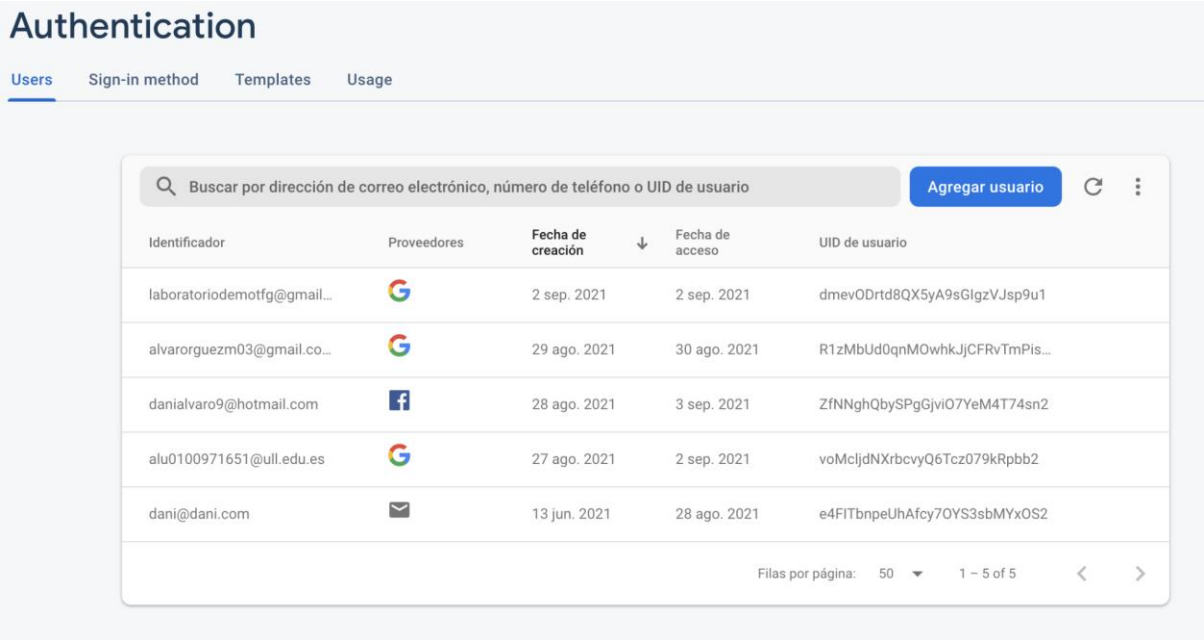

***Figura 27**. Usuarios registrados en la base de datos de la aplicación*

La pregunta que surge ahora es cómo se consigue almacenar esas cuentas. Para ello, lo primero que se hace es invocar al método *_register*, cuya llamada se hace a la hora de hacer click sobre el botón de registro de nuestra aplicación. Esto permite crear una instancia de un usuario nuevo, tal y como se muestra en la figura 28

```
_register(BuildContext context, LoginBloc bloc) async {

    final info = await usuarioProvider.nuevoUsuario(bloc.email, bloc.password);

    if (info['ok']) {
        // Avoid error "Bottom overflowed by 99909 pixels"
        SchedulerBinding.instance.addPostFrameCallback((_) {

                Navigator.pushReplacementNamed(context, 'tracker');
            }
        );
    }
    else {
        mostrarAlerta(context, info['mensaje']);
    }
}
```

***Figura 28**. Método de registro de usuarios*

Lo siguiente que se hace es comprobar que la respuesta a esa petición de crear un nuevo usuario no ha tenido ningún inconveniente, ya que, en caso de no ser así, se muestra un mensaje de alerta al usuario indicándole que esa cuenta ya ha sido registrada anteriormente.

Ahora, al observar la figura 29, podemos ver con detalle el funcionamiento del método *nuevoUsuario* que mencionamos anteriormente. Lo primero que debemos tener en cuenta en la petición POST que debemos hacer a la base de datos Firebase, esto es, insertando un enlace por defecto para la petición junto con nuestro token privado generado en el momento de crear nuestro proyecto en el sistema. Posteriormente, la respuesta que se obtiene de esa petición POST se guarda en formato JSON en un dato de tipo Map, permitiéndonos desglosar esa información en formato clave/valor. Si de esa petición podemos obtener el *idToken* que se ha debido crear al hacer la petición POST, se devuelve dicho token.

```
Future<Map<String,dynamic>> nuevoUsuario(String email, String password) async {

    final authData = {
        'email'             : email,
        'password'          : password,
        'returnSecureToken' : true
    };

    final resp = await http.post(
        Uri.parse('https://identitytoolkit.googleapis.com/v1/accounts:signUp?key=$_firebaseToken'),
        body: json.encode(authData)
    );

    Map<String, dynamic> decodedResp = json.decode(resp.body);

    if(decodedResp.containsKey('idToken')) {

        _prefs.token = decodedResp['idToken'];
        return {'ok' : true, 'token' : decodedResp['idToken']};
    }
    else
        return {'ok' : false, 'token' : decodedResp['error']['message']};

}
```

Figura 29. *Método nuevoUsuario*

Si nos fijamos ahora en la figura 30 en la forma en la que inician sesión los usuarios ya registrados, podemos destacar que apenas existen diferencias notables respecto al método de registro de nuevos usuarios, y es tan simple como el hecho de que lo único que es importante cambiar en este caso es el enlace que se guardará en la respuesta *resp*.

```dart
Future<Map<String,dynamic>> login(String email, String password) async {

    final authData = {
        'email'            : email,
        'password'         : password,
        'returnSecureToken' : true
    };

    final resp = await http.post(
        Uri.parse('https://identitytoolkit.googleapis.com/v1/accounts:signInWithPassword?key=$_firebaseToken'),
        body: json.encode(authData)
    );

    Map<String, dynamic> decodedResp = json.decode(resp.body);

    print(decodedResp);

    if(decodedResp.containsKey('idToken')) {

        _prefs.token = decodedResp['idToken'];
        return {'ok' : true, 'token' : decodedResp['idToken']};
    }
    else
        return {'ok' : false, 'token' : decodedResp['error']['message']};
}
```

Figura 30. *Método login*

3.2.2. Donaciones realizadas

A la hora de que un donante envíe su formulario de donación y éste sea apto, automáticamente se podrá guardar en la base de datos. Como vemos en la figura 31, cada donación tiene asociado una serie de datos que son vitales para distinguirla de otras donaciones. En concreto, cada donación consta de 3 datos elementales:

1. Fecha
2. Puesto de donación
3. Nombre del donante

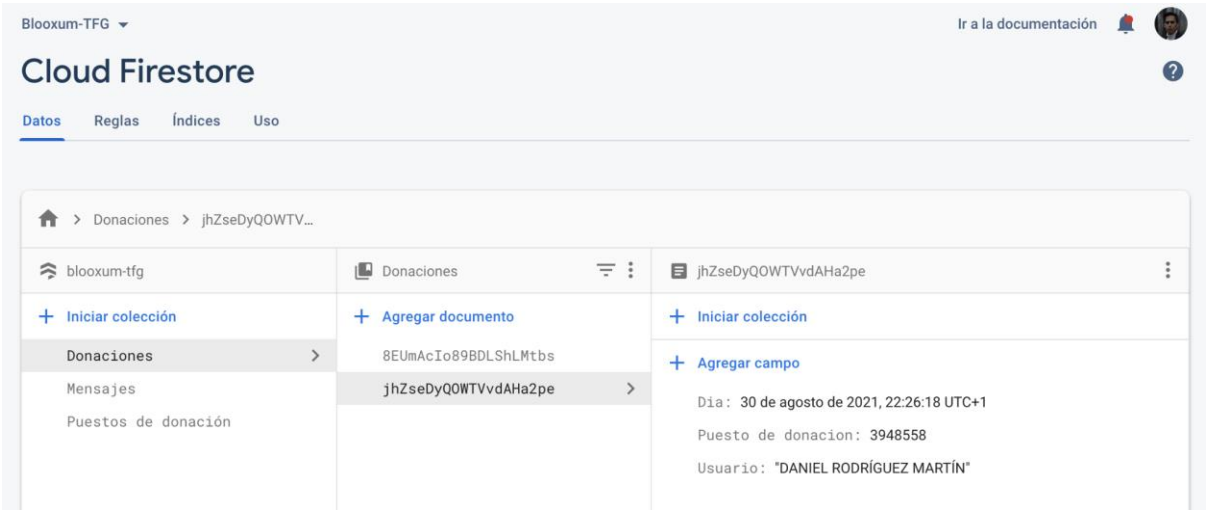

***Figura 31**. Registro de las donaciones realizadas*

De esta forma, cada vez que se cree una instancia de donación, como vemos en la figura 32, al obtener la instancia de Firebase del usuario validado en ese momento, se crea un nuevo documento en la colección de Firebase *Donaciones* donde se guardan los tres datos comentados anteriormente. Es importante señalar que en caso de que ya exista dicha colección, no se vuelve a crear, sino que se añaden tantos documentos como donaciones distintas se hagan.

```
MaterialButton(
    height: 50.0,
    color: Colors.green[600],
    onPressed: () {
        var loginUser = FirebaseAuth.instance.currentUser;
        FirebaseFirestore.instance.collection("Donaciones").add({"Dia" : DateTime.now(), "Puesto de donacion" : 3948558, "Usuario": loginUser.displayName });
        Navigator.pop(context);
    },
    child: Text("Enviar", style: TextStyle(fontSize: 20.0, color: Colors.white),),
), // MaterialButton
```

***Figura 32**. Widget del botón para enviar los resultados del test a la base de datos*

3.2.3. Mensajes enviados

Por último, pasamos a la información de los mensajes enviados guardados en nuestra base de datos. Tal y como vemos en la figura 33, en la colección de mensajes se guardan tantos documentos como mensajes son enviados entre el usuario y los laboratorios. De la misma manera que ocurre con las donaciones, cada mensaje consta de una serie de campos obligatorios, que constituyen los siguientes:

1. Fecha del mensaje
2. Mensaje
3. Usuario que envía el mensaje

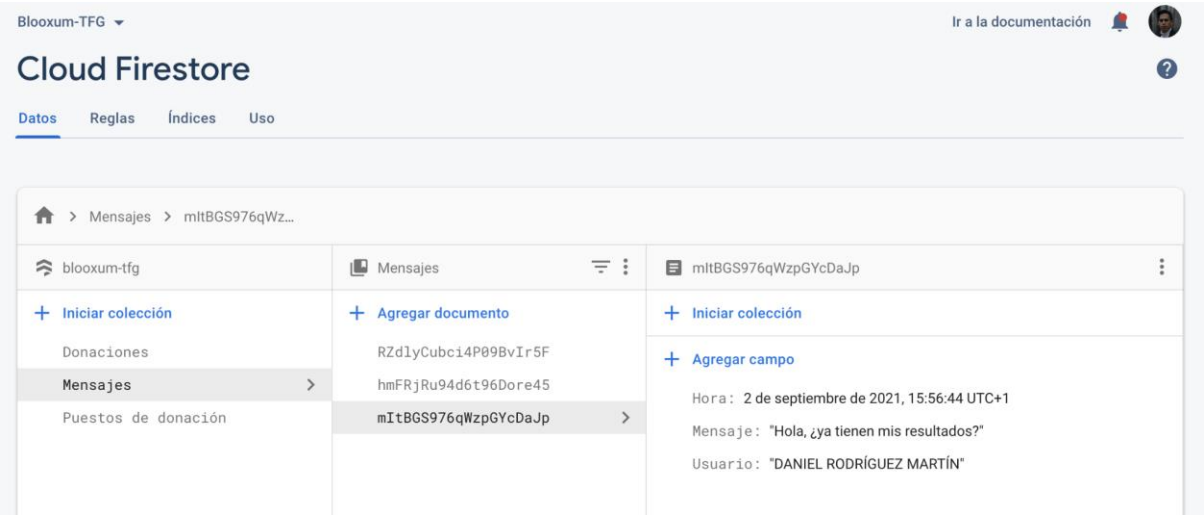

Figura 33. *Registro de los mensajes enviados*

Señalar que la forma en la que se envían los datos al sistema es idéntica a la explicada en anteriores ocasiones.

Capítulo 4

Conclusiones y futuras aproximaciones

Trabajar en este proyecto ha sido un proceso bastante largo, intuitivo y, en algunas ocasiones, bastante tedioso por distintas razones. Previo al comienzo de esta iniciativa, se realizó una formación en Flutter bastante focalizada en ser capaz de abarcar cualquier tipo de proyecto. Al principio las razones por las que se había cogido este opción se basaban en la sencillez que otorga desarrollar aplicaciones con esta tecnología, no obstante y como se comentaba anteriormente, aparecieron en determinados momentos del desarrollo de la aplicación algunas complicaciones cuya resolución fue bastante ardua, no solo por no lograr en determinados momento que la aplicación funcionase, sino por la casi escasa documentación que en muchas ocasiones provocó una tergiversación en el entendimiento de las complicaciones que aparecian.

La mayor complicación que surgió durante el desarrollo de la aplicación fue la definición de los datos compartidos dentro de las distintas secciones de la aplicación. A la hora de insertar un widget a la aplicación que nos capacitaba para trabajar con dicha información, aparecian problemas derivados que no guardaban relación entre ellos. Todos estos problemas casualmente ocurrían a la hora de intentar desplegar la aplicación en dispositivos iOS.

Por otro lado, otro pequeño inconveniente que surgía dada la poca experiencia con el lenguaje era la adaptación del contenido de la aplicación para cualquier dispositivo móvil, que con el tiempo se fue mejorando en base a la documentación consultada.

Además, también surgieron algunos pequeños inconvenientes relacionados con la autenticación de usuarios que no pudo ser resuelta, que consistía en cerrar la sesión de usuario de Google y Facebook formalmente en la base de datos. Tras realizar distintas consultas en la documentación aportada por la comunidad de Flutter, se llegó a la conclusión de que consistía en un problema generalizado cuya solución no estaba preestablecida.

Para finalizar, es importante ver los avances que se han producido hasta el momento en el desarrollo de la aplicación frente a los objetivos que se planteaban en un principio. Se habían planteado 6 funcionalidades, repartidas en dos generales o 4 específicas, de las cuales solo 1 no se pudo completar.

El principal motivo de su descarte fue la imposibilidad de garantizar un correcto funcionamiento dada las expectativas iniciales que no se cumplieron, ya que se habían planteado a distintos representantes del Instituto Canario de Hemodonación y Hemoterapia la idea de contar con estos servicios, y dada la nula comunicación posterior a dichas reuniones,

no era posible acceder a las bases de datos y poder realizar notificaciones reales de carencia de sangre.

Para el futuro, sería interesante mejorar algunos aspectos de la aplicación, así como agregar nuevas funcionalidades que podrían ser realmente interesantes. Algunas de ellas son las siguientes:

- **Mejorar la implementación del Chat** → como se comentó anteriormente, es importante mejorar este apartado para garantizar una correcta autenticación de los usuarios y mostrar debidamente el contenido de las conversaciones correspondientes al usuario con sesión validada.

- **Mejorar la integración de los datos de aquellos usuarios que no opten por el uso de Redes Sociales para registrarse en la aplicación** → Al ser la primera vez que se utilizaba este lenguaje para desarrollar un proyecto real, se optó hacer bastante hincapié en funcionalidades que no se habían utilizado antes en ningún otro proyecto hasta el momento, de modo que los datos utilizados por aquellos usuarios que no utilizaba este método de autenticación no estaban tan centralizados como los otros, lo cual da pie a mejorar bastante este apartado.

- **Mejorar el diseño del Splashscreen en otros dispositivos** → A pesar de no afectar al correcto funcionamiento de la aplicación, es sumamente importante mejorar este apartado con el fin de lograr una correcta experiencia de usuario, sin discriminar a ningún dispositivo.

- **Realizar una modificación en la sección del carnet de donante para una posible lectura de los datos a través de NFC** → Gracias a esta tecnología se podría implementar la lectura de los datos del donante por parte del personal autorizado con simplemente pasar el dispositivo del donante por un lector habilitado para ello.

- **Definir encriptado end-to-end para los mensajes del chat** → actualmente y como se ha podido observar, los mensajes que se intercambian en el apartado de Chat son visibles desde la base de datos. Esto por supuesto no es algo habitual, ya que los datos que se almacenan en la base de datos deben ser encriptados anteriormente desde el cliente, es decir, desde el dispositivo del donante. Pero teniendo en cuenta que este proyecto se había realizado para fines didácticos, no se llevó a cabo.

Capítulo 5

Summary and future approaches

Working on this project has been quite a long, intuitive and, sometimes, quite tedious process for various reasons. Prior to the beginning of this initiative, a Flutter training was carried out, which was very focused on being able to cover any type of project. At first, the reasons why this option had been taken were based on the simplicity of developing applications with this technology, however, and as mentioned above, some complications appeared at certain times during the development of the application, the resolution of which was quite arduous. not only because of not getting the application to work at certain times, but also because of the almost scant documentation that on many occasions caused a misrepresentation in the understanding of the complications that appeared.

The biggest complication that arose during the development of the application was the definition of the shared data within the different sections of the application. When inserting a widget to the application that enabled us to work with said information, derived problems appeared that were not related to each other. All these problems happened by chance when trying to deploy the application on iOS devices.

On the other hand, another small inconvenience that arose given the little experience with the language was the adaptation of the content of the application for any mobile device, which over time was improved based on the documentation consulted.

In addition, there were also some small problems related to user authentication that could not be resolved, which consisted of formally closing the Google and Facebook user session in the database. After various consultations on the documentation provided by the Flutter community, it was concluded that it consisted of a generalized problem whose solution was not pre-established.

Finally, it is important to see the progress that has occurred so far in the development of the application compared to the objectives that were raised at the beginning. 6 functionalities had been proposed, divided into two general or 4 specifics, of which only 1 could not be completed.

The main reason for its dismissal was the impossibility of guaranteeing proper functioning given the initial expectations that were not met, since the idea of having these services had been raised with different representatives of the Canarian Institute of Hemodonation and Hemotherapy, and given the null communication after these meetings, it was not possible to access the databases and be able to make real notifications of blood shortages.

For the future, it would be interesting to improve some aspects of the application, as well as add new functionalities that could be really interesting. Some of them are the following:

- **Improve the implementation of Chat** → as mentioned above, it is important to improve this section to ensure correct user authentication and properly display the content of the conversations corresponding to the user with a validated session.

- **Improve the integration of the data of those users who do not choose to use Social Networks to register in the application** → As this was the first time that this language was used to develop a real project, it was decided to place a lot of emphasis on functionalities that were not They had been used before in any other project so far, so the data used by those users who did not use this authentication method were not as centralized as the others, which leads to a considerable improvement in this section.

- **Improve the design of the Splashscreen on other devices** → Despite not affecting the correct operation of the application, it is extremely important to improve this section in order to achieve a correct user experience, without discriminating against any device.

- **Make a modification in the donor card section for a possible reading of the data through NFC** → Thanks to this technology, the reading of the donor data by authorized personnel could be implemented by simply passing the donor's device through a reader enabled for it.

- **Define end-to-end encryption for the chat messages** → currently and as has been seen, the messages exchanged in the Chat section are visible from the database. This of course is not something usual, since the data that is stored in the database must be previously encrypted from the client, that is, from the donor's device. But considering that this project had been carried out for educational purposes, it was not carried out.

Capítulo 6

Presupuesto

En base a las propiedades de este proyecto, el presupuesto que se desglosa a continuación está formalizado en base a la duración del desarrollo, que se estima en una duración de 4 meses.

6.1. Software

Aquí se muestran los distintos costes relacionados con los programas necesarios para el correcto desarrollo del proyecto.

Visual Studio Code	0 €
XCode	0 €
Android Studio	0 €

***Tabla 1**: Presupuesto del software utilizado*

6.2. Hardware

En este apartado, se desglosan los precios en base al equipamiento utilizado para el desarrollo.

Ordenador portátil	1100 €
Segundo monitor	90 €
Tablet	520 €

***Tabla 2**: Presupuesto del hardware utilizado*

6.3. Servicios

A lo largo de esta tabla, se muestran los costes de los distintos servicios necesarios para la implementación de la app.

	Precio/Mes	**Total**
Internet (Fibra óptica)	50 €	200 €
Hosting de la aplicación	0 €	0 €

***Tabla 3**: Presupuesto de los servicios utilizados*

6.4. Salario

Los costes del salario del desarrollador se han estimado en base a la experiencia de éste.

	Horas	**Precio/hora**	**Total**
Programación	320 horas	10 €	3200 €

***Tabla 4**: Presupuesto de los costes del desarrollador en base a la experiencia del mismo*

6.5. Presupuesto final

En base a los distintos costes desglosados anteriormente, a continuación, se muestran los costes globales del desarrollo de este proyecto.

	Total
Software	0 €
Hardware	1710 €
Servicios	200 €
Salario	3200 €
Total	**5110 €**

***Tabla 5**: Presupuesto final*

Bibliografía

[1] - https://flutter.dev/?gclid=Cj0KCQjw-NaJBhDsARIsAAja6dM8H4gc0VMAGGdYa0GCekL0oTKpLc0O8mpxmLYn9jdr3CXQF_2Y5ckaAh_vEALw_wcB&gclsrc=aw.ds

[2] - https://dart.dev/

[3] - https://code.visualstudio.com/

[4] - https://developer.android.com/studio?hl=es

[5] - https://apps.apple.com/es/app/xcode/id497799835?mt=12

[6] - https://developers.google.com/identity/sign-in/web/sign-in

[7] - https://developers.facebook.com/docs/facebook-login/

[8] - https://developer.apple.com/sign-in-with-apple/

[9] - https://www.instagram.com/bydanielson/

[10] - https://www.mapbox.com/

[11] - https://es.stackoverflow.com/