Sistema para la visualización, control y prevención de aglomeraciones de personas en lugares públicos mediante dispositivos móviles de tipo smartphone

TITULACIÓN: Grado en Ingeniería Informática

AUTOR: Javier Díaz Domínguez

TUTORIZADO POR:

Pablo Carmelo Fernández López

21 de mayo de 2021

Agradecimientos

A mis padres, que siempre han sido un pilar incondicional en mi vida; a mi hermano, que me alegra la vida desde hace dos años, y a mis amigos, con los que sé que puedo contar para cualquier cosa.

Resumen ejecutivo

Es conocida por todos la situación excepcional que ha protagonizado las noticias y conversaciones alrededor de todo el mundo desde hace algo más de un año ya, y es que la pandemia global provocada por el coronavirus ha afectado a las sociedades en algunos de los aspectos pilares que las definen: ha menguado la interacción social, ha limitado la libre circulación, ha provocado situaciones sin precedentes que han afectado profundamente al estado mental de las personas y ha tambaleado los cimientos de la economía mundial, entre otras muchas cosas.

Puestos a analizar los principales retos o desafíos que esta situación nos ha impuesto como sociedad, se pueden señalar como principales el control del contacto físico y la creación de entornos seguros donde se reduzca la probabilidad de contagio. Para ello, los diferentes gobiernos han impuesto limitaciones o restricciones para tratar de lograr los objetivos mencionados, intentando minimizar el impacto a la economía o la interacción social entre las personas.

Sin embargo, se han desaprovechado algunos recursos con los que contamos como sociedad y que podrían haber contribuido notablemente a facilitar las labores de control y gestión requeridas por la pandemia mundial.

Según la Agencia Española de Protección de Datos (Agencia Española de Protección de Datos, 2020), "Conocer los patrones de movilidad de la población, ver dónde se desplazan las personas cuando trabajan o los fines de semana, puede ser beneficioso para una administración en cualquier tiempo." (p. 5).

En base a esta premisa, *Where to go App* trata de contribuir a la lucha contra el COVID-19 mediante el desarrollo de una aplicación para dispositivos móviles de tipo smartphone. Con la creación de este producto se pretende implementar un sistema de información que permita a los usuarios conocer predicciones de los niveles de ocupación de las diferentes localizaciones, además de la seguridad de ellas en cuanto a las medidas tomadas para limitar la difusión del virus, y poder elegir un lugar seguro para el propósito que crean conveniente (ocio, compras...).

Para la implementación del sistema descrito anteriormente, se propone el desarrollo de una aplicación móvil junto con un soporte de información, que trabajarán en conjunto para desarrollar las funcionalidades necesarias para cumplir los requisitos impuestos. El propio desarrollo de la aplicación se ha estructurado tomando algunos conceptos de metodologías ágiles como SCRUM, pero aplicadas a un equipo de trabajo unipersonal, donde la figura de *product owner* ha sido delegada en el tutor de TFG.

Además de una metodología de desarrollo ágil, también se ha tratado de seguir la filosofía TDD, con el fin de crear una aplicación robusta y persistente, y una base de código preparada para la evolución progresiva que le permita adaptarse a nuevos requisitos que puedan aparecer. Por último, cabe destacar que el código de la aplicación desarrollada ha sido estructurado y ordenado en base a patrones de diseño modernos, en favor de los principios SOLID y la reusabilidad y escalabilidad del proyecto.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1: Introducción

Desde aproximadamente marzo de 2020, el mundo se ha tenido que enfrentar a lo que ha sido probablemente uno de los mayores desafíos de la edad contemporánea: la pandemia mundial provocada por la expansión descontrolada del virus SARS-Cov-2. Durante lo que ha durado esta situación sin precedentes para los gobiernos e instituciones que han tenido que enfrentarse a la gestión de la crisis sanitaria, se han aplicado diferentes medidas y estrategias a la hora de tratar de reducir los contagios y sanear el sistema sanitario, siempre tratando de minimizar los contagios y el impacto a la economía.

De entre las diferentes medidas aplicadas, unas más restrictivas que otras, muy pocas han aprovechado efectivamente los recursos tecnológicos de los que dispone la sociedad actualmente, y es que el contexto social en el que se produce la pandemia goza de uno de los recursos más aprovechables para el desarrollo de soluciones que puedan contribuir a superar los retos que impone una crisis sanitaria de esta magnitud. Este recurso no es otro que la información.

En este documento se plantearán los diferentes retos que ha presentado la pandemia del COVID-19, analizando y planteando una solución tecnológica que puede facilitar las tareas necesarias para superarlos.

Recursos

Para la elaboración del conjunto de materiales que componen la entrega, ha sido requerido el uso de los diferentes recursos listados a continuación.

Recursos software

Para la elaboración de la memoria, se ha empleado el editor de documentos Microsoft Office Word, cuya licencia ha sido suministrada por la ULPGC. Además, se ha tomado como referencia para la elaboración de los estilos, la disposición de las páginas y la estructura del documento la plantilla suministrada en la web institucional en LATEX. Para su edición se ha utilizado el software de edición LATEX *OneLeaf*.

En cuanto al código de la aplicación desarrollada, se ha elaborado utilizando principalmente como editor de texto *Visual Studio Code*. Otros recursos destacables durante el desarrollo han sido la instalación de Android Studio, requerida para la ejecución del entorno de emulación Android donde se testeaba y probaba la aplicación.

El *framework* de desarrollo elegido ha sido Flutter (se justificará más adelante). Para el soporte de información de la aplicación (*back-end*), se ha elaborado un módulo software ejecutable en Node.js. Cabe destacar también la utilización de GraphQL como lenguaje de comunicación entre la aplicación móvil y el soporte de información del sistema.

Recursos hardware

Durante el desarrollo tanto de la aplicación como de la memoria, sólo ha sido necesario la utilización de un computador con conexión a internet. Los dispositivos móviles empleados para el desarrollo han sido emulados.

Recursos documentales

Para la elaboración del código de la aplicación, fue necesaria la lectura y comprensión de una serie de recursos documentales para conocer la correcta utilización de los lenguajes y tecnologías empleados en el proyecto. Esta documentación está encabezada por:

- Documentación sobre Flutter [1]

Esta documentación fue empleada para adquirir el conocimiento necesario para elaborar la base de código que daría soporte a la aplicación desarrollada en el proyecto.

- Documentación sobre MapBox[2]

La utilización de una API para mostrar mapas y trabajar con datos de geolocalización requirió un periodo de aprendizaje sobre las diferentes estructuras de datos que se utilizan para manejar la geolocalización, así como los diferentes recursos disponibles para obtener la información deseada.

- Documentación sobre GraphQL[3]

Para implementar correctamente las diferentes ventajas que ofrece GraphQL como lenguaje de peticiones sobre el back-end, fue necesaria la utilización de los recursos documentales ofrecidos en su página web oficial.

- Stack Overflow[4]

Para la resolución de muchos de los contratiempos, fallos y *bugs* encontrados durante el desarrollo de la aplicación, se acudió al foro de desarrolladores Stack Overflow en busca de otros usuarios que hubieran encontrado problemas similares.

Por otro lado, la elaboración de algunos de los aspectos de la aplicación requirió la realización de una pequeña investigación sobre los diferentes métodos de contagio y las medidas tomadas por las diferentes instituciones gubernamentales para su contención. Esta investigación se realizó sobre documentación oficial del Ministerio de Sanidad del Gobierno de España, del Ministerio de Educación del Gobierno de España y de la Organización Mundial de la Salud. Todos estos documentos serán referenciados adecuadamente en apartados posteriores del documento.

Tecnologías empleadas

A continuación, se realizará una explicación más detallada del conjunto de tecnologías empleadas para la composición del sistema de información que da soporte a la solución planteada. Se describirán las tecnologías agrupadas por su ámbito de operación, y se justificará su uso frente a las distintas alternativas existentes.

[1] https://flutter.dev/docs
[2] https://docs.mapbox.com/
[3] https://graphql.org/learn/
[4] https://stackoverflow.com/

Back end

El back end está compuesto por dos elementos principales: el soporte de almacenamiento físico y la capa de software encargada de atender las peticiones.

El soporte de almacenamiento de los datos de la aplicación consiste en una base de datos para la que se ha elegido el software de base de datos MongoDB.

Mongo DB es una base de datos basada en documentos, de propósito general y código abierto. Está clasificada como NoSQL, debido a su estructura de almacenamiento en forma de documentos con estructura JSON y con esquemas opcionales.

La elección del uso de una base de datos no relacional frente a una base de datos relacional depende de diversos factores. Los que se han tenido en consideración para el caso de la aplicación desarrollada en concreto, han sido los siguientes:

- Flexibilidad

Una de las principales ventajas que presentan las bases de datos no relacionales frente a las relacionales es la flexibilidad que aportan al desarrollo. Permiten gestionar entidades de almacenamiento de información sin tener que fijar un modelo predefinido, facilitando los cambios en las entidades en el momento, sin tener que cambiar la configuración de almacenamiento de la base de datos. Esto es posible gracias a su estructura de documentos, que permite categorizar los documentos por colecciones, sin tener que contar con los mismos campos.

- Facilidad de uso

La facilidad de uso de MongoDB también ha supuesto un aspecto que lo ha posicionado por encima de otras opciones, y es que después de la instalación del software de MongoDB, no se requiere configuración adicional para poder ejecutar directamente un sistema de información que haga uso de la base de datos. Esto es ideal para realizar un despliegue, puesto que minimiza la configuración del entorno de producción.

Otras bases de datos (SQL) requieren la definición y ejecución de un esquema que defina las entidades que se emplearán para almacenar los datos. Esto en MongoDB no es necesario, pero como se mencionará más adelante se ha empleado una librería JavaScript que permite la definición de un esquema de entidades de almacenamiento que será gestionado directamente por Node.js.

- Naturaleza de los datos con los que la aplicación trabaja

Por último, para el caso concreto de la aplicación desarrollada, el uso de una base de datos no relacional encajaba a la perfección con la naturaleza de los datos con los que la aplicación trabajaría, pues los datos de geolocalización son, por lo general, bastante desestructurados. Además, MongoDB cuenta con la posibilidad de indexar los documentos almacenados utilizando campos de geolocalización, permitiendo así la optimización de las búsquedas por puntos geográficos o incluso operaciones como búsqueda de puntos próximos o puntos dentro de una región determinada. Todas estas características son aprovechadas en la implementación de la aplicación.

Por otro lado, dentro del conjunto de capas que conforman lo que se denomina el back end de la aplicación, se encuentra la capa de software encargada de la gestión de las peticiones que llegan desde el front end, siendo esta capa la que coordina el funcionamiento de la base de datos con las entidades y estructuras de datos con las que trabaja el front end de la aplicación. Dicha capa no es más que un módulo software ejecutado en Node.js y escrito en JavaScript, que se debe de ejecutar en la máquina donde se reciban las peticiones enviadas por el front end.

- Node.js

Node.js es un software de código abierto, multiplataforma, orientado a la capa de servidor basado en el lenguaje de programación JavaScript y el motor de ejecución de Google V8.

V8 es el entorno de ejecución para JavaScript creado para Google Chrome. Está escrito en C++ y compila el código JavaScript en código máquina en lugar de interpretarlo en tiempo real.

Node.js se registra con el sistema operativo y realiza una escucha activa de las conexiones entrantes, ejecutando un *callback* cuando se registran nuevas conexiones.

Este módulo a su vez hace uso de algunos *frameworks* y librerías de desarrollo para Node.js, que serán detalladas a continuación.

- Mongoose

Es una solución JavaScript basada en esquemas para modelar los datos de una aplicación, y que utiliza el controlador de MongoDB para Node.js para comunicarse con la instancia de la base de datos. Este controlador facilita y estandariza el uso de MongoDB en las aplicaciones JavaScript, pero manteniendo la flexibilidad y facilidad de uso mencionadas anteriormente. Se podría decir que Mongoose transforma la experiencia de uso de una base de datos no relacional en una experiencia a medio camino entre las bases de datos no relacionales y las relacionales. Esta "transformación" la consigue mediante el uso de esquemas que permiten la definición de una serie de entidades o estructuras de datos que predefinen los campos de los documentos almacenados en la base de datos. De esta manera se facilita y estandariza el uso de los datos para aplicaciones desarrolladas con lenguajes de programación tipados, como es el caso de *GraphQL* y *Flutter*.

Otro aspecto destacable es que las entidades definidas en los esquemas son manejadas por Mongoose y no por MongoDB, es decir, en las colecciones donde se almacenan estas entidades se podrán seguir almacenando documentos que no cumplan con este esquema, si se diera el caso en el que otras aplicaciones accedieran a la base de datos simultáneamente.

- Express.js

Express es un *framework* para Node.js que ofrece funcionalidades diversas como el enrutamiento o la gestión de cookies y sesiones para facilitar la creación de páginas web. Se apoya en otras librerías ya implementadas que gestionan el protocolo http.

- GraphQL

GraphQL es un lenguaje de consulta para API's además de un módulo de ejecución para resolver las consultas con los datos almacenados. Ofrece la posibilidad a los clientes de una API

hacer peticiones de datos precisas, obteniendo sólo y exclusivamente lo que necesitan. Implementa una capa de abstracción sobre la API que clarifica y simplifica su uso.

En el caso de la aplicación desarrollada, escrita en JavaScript y ejecutada en Node.js, la librería en concreto que ofrece soporte a la utilización de GraphQL sobre una API es Apollo Server. Apollo es una librería de software libre que permite la creación de servidores que implementan el lenguaje GraphQL para la resolución de consultas. Los principales casos de uso son:

- Un servidor **stand-alone**, incluyendo un entorno **serverless**.
- En adición al middleware existente de una aplicación, como *express*.

Esta librería permite escribir código limpio y bien estructurado, contribuyendo a la escalabilidad del código y facilitando su mantenimiento.

Como se ha mencionado anteriormente, la elección de Apollo está justificada para las características de la aplicación desarrollada, y su implementación con el **middleware express** se justifica por la posibilidad de combinar la resolución de rutas tradicional de express con la resolución de peticiones GraphQL que implementa Apollo. Con esta combinación de librerías, se permite la configuración de un servidor web con diferentes puntos de escucha (como por ejemplo `/home`) junto con un punto de escucha concreto para la resolución de peticiones de datos, comúnmente se emplea `/graphql` . Además, Apollo implementa un entorno de pruebas por defecto en la misma ruta que, al ser accedido desde un navegador, sirve una interfaz gráfica que permite realizar peticiones escritas en GraphQL directamente sobre la API, realizando corrección de código y documentación sobre los modelos definidos en el esquema.

Front end

Sobre el conjunto de tecnologías que componen la interfaz con la que el usuario final va a interactuar, el primer aspecto a aclarar es que ha sido desarrollado como una aplicación para dispositivos móviles. A continuación, se detallarán algunos detalles sobre las tecnologías que se emplearon para logar una capa de interacción con el usuario que cumpliera con los requisitos establecidos por las historias de usuario que se plantearon al inicio del desarrollo.

- **Flutter**

Flutter es un kit de desarrollo software de código abierto elaborado por Google para la elaboración de aplicaciones multiplataforma (IOS, Linux, Android, Windows, Web) a partir del mismo código fuente. Las principales ventajas que aporta Flutter al desarrollo de aplicaciones para dispositivos móviles son:

- Documentación

Es uno de los frameworks de desarrollo multiplataforma más empleados en los últimos años, y cuenta con una documentación excelente. Esto lo convierte en uno de los mejor mantenidos y con el respaldo de una de las mayores comunidades de desarrolladores.

- Rapidez

Flutter está construido sobre Skia, que es la librería gráfica 2D sobre la que se desarrollaron Google Chrome y Android. El lenguaje de desarrollo empleado es Dart, que permite la compilación a código máquina ARM de 32 o 64 bits así como JavaScript para la web y la arquitectura x64 de los computadores tradicionales.

- Experiencia de usuario

Permite a los diseñadores la utilización de su visión creativa al máximo, eliminando por completo las limitaciones derivadas de la utilización de un framework. Utiliza una estructura por capas, concediendo al desarrollador el control de cada uno de los pixeles de la pantalla.

- Desarrollo productivo

Flutter ofrece la posibilidad a los desarrolladores de ver los cambios realizados en el código fuente inmediatamente sin necesidad de reinicio de la aplicación, ofreciendo una experiencia de desarrollo fluida y rápida.

- MapBox

Para el renderizado de mapas interactivos en tiempo real y la obtención de datos de geolocalización de sitios aprovechables para el uso en el sistema desarrollado, era necesaria la inclusión de una API de geolocalización. Después de una pequeña etapa de análisis y comparación entre las diferentes alternativas existentes (que será mostrada posteriormente), se decidió que MapBox es la opción que más se ajusta a las características de la aplicación desarrollada.

Mapbox es un proveedor online de mapas personalizados. Desde 2011 ha expandido considerablemente su catálogo de mapas personalizados, en respuesta a la limitada oferta de proveedores de mapas como Google maps.

Mapbox toma sus datos de geolocalización de fuentes de datos abiertas como Nasa u Open Street Maps, así como de fuentes de datos propietarias como DigitalGlobe. Basan su funcionamiento en Node.js, Mapnik, GDAL y Leaflet.

Justificación de las competencias específicas cubiertas

Durante la elaboración de este trabajo, se han desarrollado algunas de las principales competencias de la mención cursada por el alumno. En concreto, estas competencias son:

Competencia	Justificación
CII013: Conocimiento y aplicación de las herramientas necesarias para el almacenamiento, procesamiento y acceso a los Sistemas de información, incluidos los basados en web.	El uso de diversas herramientas, descritas en esta memoria, como
SI01: Capacidad de integrar soluciones de Tecnologías de la información y las comunicaciones y procesos empresariales para satisfacer las necesidades de información de las organizaciones, permitiéndoles alcanzar sus objetivos de forma efectiva y eficiente, dándoles así ventajas competitivas.	Concibiendo la sociedad como una organización, y el sistema planteado para cubrir los objetivos descritos como una necesidad de información para la misma, se considera el sistema desarrollado una solución, haciendo uso de las tecnologías de la información, que le otorga una ventaja no competitiva y que además supone una contribución de cara a lograr unos objetivos globales.
SI03: Capacidad para participar activamente en la especificación, diseño, implementación y mantenimiento de los sistemas de información y comunicación	A partir de una serie de retos obtenidos del contexto global actual, se ha planteado un sistema que pretende ayudar en el cumplimiento de estos. El sistema planteado ha sido diseñado e implementado íntegramente por el alumno, y este proceso ha quedado plasmado en la memoria desarrollada.
SI06: Capacidad para comprender y aplicar los principios y las técnicas de gestión de la calidad y de la innovación tecnológica en las organizaciones.	El uso de tecnologías punteras en el desarrollo del proyecto como Flutter, GraphQL o Node.JS justifican la aplicación de los principios de innovación tecnológica en las organizaciones. Además, el uso de metodologías de desarrollo ágiles y la inclusión de pruebas que aseguran la calidad del producto sirven de justificación de la aplicación de los principios de gestión de la calidad tecnológica.

Capítulo 2: Análisis

Durante este capítulo de la memoria, se procederá a explicar y comentar las etapas y pasos previos a la implementación de la aplicación, así como otros aspectos relevantes sobre el proceso desarrollo.

Contexto

Como se ha mencionado anteriormente, el concepto principal del proyecto surge del planteamiento de una posible solución a algunos de los problemas que plantea la aparición de la pandemia mundial del coronavirus, como el control de las aglomeraciones en espacios públicos o privados, o la comprobación del cumplimiento de las medidas anti-COVID planteadas por las diferentes instituciones de la salud. Es por ello, que se emprende una etapa de investigación superficial sobre los aspectos principales para tener en cuenta en el desarrollo de tal solución.

Mecanismos de transmisión del COVID-19

Se pretende elaborar una solución tecnológica que contribuya en la reducción de la propagación del COVID-19 en la medida de lo posible. Por ello, conocer los mecanismos de transmisión y lo que estos implican es clave para lograr dicho objetivo.

Según la OMS (*Transmission of SARS-Cov-2: implications for infection prevention precautions, 9 julio 2020*) las principales vías de transmisión de las infecciones respiratorias por SARS-CoV-2 ocurren por:

- Transmisión por gotas y contacto

El contagio de SARS-CoV-2 puede ocurrir tras el contacto directo, indirecto o estrecho con personas infectadas a través de secreciones infectadas como la saliva o secreciones respiratorias. La transmisión mediante estas secreciones puede ocurrir durante contactos estrechos (menos de 1 metro) con una persona infectada con síntomas respiratorios mientras ésta esté cantando o hablando. Esto ocurre porque estas secreciones infectadas pueden alcanzar la boca, los ojos o la nariz de una persona susceptible y provocar su infección.

La transmisión indirecta a través de objetos o superficies contaminados con estas secreciones es posible, aunque no existe evidencia que la haya demostrado.

- Transmisión aérea

La transmisión por aerosoles se produce por partículas que permanecen suspendidas en el aire durante un tiempo variable, a una distancia mayor de 2 metros y especialmente en lugares cerrados con ventilación escasa. Los aerosoles pueden generarse a partir de la evaporación de gotas mayores y cuando se habla o se respira.

- Otras formas de transmisión

El ARN de SARS-Cov-2 ha sido detectado en otras muestras biológicas, incluyendo la orina y las heces de algunos pacientes, y las investigaciones en el campo de la transmisión por sangre e intrauterina siguen en proceso. Sin embargo, estos métodos de transmisión no serán considerados en la elaboración de los criterios de evaluación del protocolo de protección contra

el SARS-Cov-2 debido a la baja probabilidad de ocurrencia de estos escenarios en espacios públicos o con alta concurrencia de personas.

Recomendaciones

De igual manera que el conocimiento de los métodos de transmisión es clave para la implementación de la solución planteada, las recomendaciones de las principales instituciones sanitarias también deberán de estar reflejadas de alguna manera.

Partiendo de las vías de transmisión identificadas hasta el momento, la OMS (*Transmission of SARS-Cov-2: implications for infection prevention precautions*, 9 julio 2020) realiza las siguientes recomendaciones:

El método más efectivo de romper cadenas de transmisión del virus que causa el COVID-19 es **evitar el contacto cercano entre personas infectadas y el resto de las personas.** La rápida detección e insolación de los casos de infección con el SARS-Cov-2 limita la transmisión de este y frena las cadenas de contagio.

Teniendo en cuenta que los infectados asintomáticos pueden ser transmisores del virus de igual manera, el **uso de mascarillas en lugares públicos** donde hay transmisión comunitaria y donde otros métodos de prevención como el distanciamiento social no son posibles es crucial.

Las mascarillas deberán de ser usadas como complemento de un paquete de medidas preventivas, junto con:

1. **Limpieza frecuente de manos**
2. **Distanciamiento social** siempre que sea posible
3. Limpieza y desinfección del entorno

Objetivos

Una vez realizado un análisis sobre los detalles más relevantes del contexto sobre el que se pretende elaborar una solución a los problemas planteados inicialmente, se procede con el planteamiento de una serie de objetivos, que serán los que articulen el proceso de diseño y desarrollo del proyecto.

- Desarrollo de un sistema software para plataforma móvil tipo smartphone, con el adecuado nivel funcional.
- El sistema software debe permitir registrar, visualizar y controlar las aglomeraciones de personas.
- El sistema software debe permitir conocer las medidas de seguridad asociadas al protocolo COVID adoptadas por cada lugar o espacio.
- El sistema software debe permitir denunciar el incumplimiento de dichas medidas en virtud de la realidad producida.

Historias de usuario

La definición de estos objetivos generales permite la elaboración de una serie de historias de usuario que serán las que detallen más específicamente las funcionalidades que la aplicación deberá de tener para lograr los objetivos desde los que se parten.

ID	Nombre
1	Registro
Descripción	
Como usuario no registrado necesito registrarme para identificarme como usuario, para que el sistema reconozca mi identidad en ocasiones futuras.	
Criterios de evaluación	
1. Dado un usuario no registrado y no identificado, cuando se trata de registrar sin introducir datos entonces se le requiere que introduzca datos válidos 2. Dado un usuario no registrado y no identificado, cuando se trata de registrar rellenando todos los campos del formulario entonces se realizará comprobación de los formatos de email y contraseña. 3. Dado un usuario no registrado y no identificado, cuando se trata de registrar rellenando todos los campos del formulario con datos correctamente formateados, entonces se realizará el registro del usuario y pasará a ser usuario registrado.	

ID	Nombre
2	Identificación
Descripción	
Como usuario registrado no identificado quiero identificarme para acceder a las funcionalidades de la aplicación. No es la primera vez que el usuario utiliza la aplicación.	
Criterios de evaluación	
1. Dado un usuario, cuan do trata de introducir datos incorrectamente formateados entonces se le indicará que los datos introducidos no están correctamente formateados. 2. Dado un usuario no registrado, cuando introduce los datos de inicio de sesión correctamente formateados entonces se le indica que los datos no corresponden con ningún usuario y se le recomienda el registro. 3. Dado un usuario registrado, cuando introduce los datos de inicio de sesión correctamente formateados entonces se realiza el inicio de sesión, pasa a ser usuario identificado y se realiza la transición a la página principal. 4. La información del usuario registrado debe almacenarse en un lugar accesible para todos los módulos de la aplicación.	

ID	Nombre
3	Navegación
Descripción	
Como usuario quiero poder navegar en el mapa para seleccionar una localización concreta. Una vez registrado, el usuario pasa a una pantalla donde se mostrará un mapa. El usuario podrá navegar libremente por el mapa, y podrá seleccionar lugares de interés en el mapa para visualizar información sobre los lugares.	
Criterios de evaluación	
1. Dado un usuario registrado, cuando realiza la identificación exitosamente entonces se le mostrará la página de navegación en el mapa. 2. Dado un usuario identificado y situado en la página de navegación de mapa, entonces se le mostrará un mapa interactivo que le permitirá buscar, seleccionar e interactuar con las diferentes localizaciones.	

ID	Nombre
4	Indicar asistencia
Descripción	
Como usuario quiero indicar mi asistencia a una localización para permitir el control de las aglomeraciones	
Criterios de evaluación	
1. Dado un usuario registrado y una localización del mapa seleccionada, entonces se mostrará un botón accionable para indicar la asistencia a la localización. 2. Dado un usuario registrado y una localización del mapa seleccionada, al accionar el botón de indicar asistencia se añadirá un registro de asistencia y se actualizará el número de posibles asistentes. 3. Dado un usuario registrado y una localización del mapa seleccionada a la que el usuario ya ha indicado su asistencia, al accionar el botón de asistencia entonces se mostrará un mensaje indicando que el usuario ya ha indicado su asistencia para dicha localización.	

ID	Nombre
5	Valorar las medidas anti-COVID de una localización
Descripción	
Como usuario quiero poder valorar y comentar las medidas anti-COVID de una localización para que los demás usuarios conozcan el cumplimiento de estas.	
Criterios de evaluación	
1. Dado un usuario registrado y una localización seleccionada, entonces se mostrará un botón para realizar una valoración de las medidas anti-COVID de una localización. 2. Dado un usuario registrado y una localización seleccionada, al accionar el botón para realizar una valoración entonces se mostrará una página con un formulario de estrellas para valorar las medidas anti-COVID de una localización. 3. Dado un usuario registrado y localizado en la página para realizar una valoración sobre una localización, al rellenar el formulario y accionar el botón de enviar el formulario, entonces se actualizará la información del lugar para incluir las valoraciones incluidas.	

ID	Nombre
6	Visualización de las valoraciones de otros usuarios
Descripción	
Como usuario quiero poder visualizar las valoraciones de otros usuarios sobre las medidas anti-COVID de una localización para conocer su cumplimiento.	
Criterios de evaluación	
1. Dado un usuario registrado y una localización seleccionada, entonces se mostrarán las medias de las valoraciones, por categoría, realizadas por los demás usuarios. 2. Dado un usuario registrado y una localización seleccionada, entonces se mostrará el número de valoraciones realizadas por los demás usuarios y la media total de las valoraciones realizadas.	

ID	Nombre
7	Visualización de la asistencia
Descripción	
Como usuario quiero poder visualizar los datos de asistencia a una localización concreta para decidir si mi asistencia a dicho lugar es apropiada.	
Criterios de evaluación	
1. Dado un usuario situado en la página de navegación del mapa, cuando selecciona una localización registrada en el sistema entonces se le mostrarán los datos de asistencia de dicha localización para ese día. 2. Dado un usuario situado en la página de navegación del mapa, cuando selecciona una localización no registrada en el sistema, entonces se le indicará que la localización no está registrada y se le dará la opción de registrarla.	

Formulario de evaluación de las medidas anti-COVID 19

Una vez establecido el contexto, los objetivos y realizado el análisis correspondiente sobre las medidas recomendadas por las instituciones sanitarias más importantes a nivel mundial, europeo y estatal, es posible proceder con la definición de la serie de criterios que se permitirá evaluar a los usuarios de cada una de las localizaciones dentro de la aplicación.

Considerando la **transmisión aérea y la transmisión por gotas y contacto** las principales vías de transmisión más relevantes en localizaciones donde puede ocurrir la transmisión comunitaria y las recomendaciones especificadas por la OMS (*Transmission of SARS-Cov-2: implications for infection prevention precautions*, 9 julio 2020) para limitar la transmisión del SARS-Cov-2, se identifican los siguientes criterios para evaluar las medidas de protección para frenar la transmisión del SARS-Cov-2.

- **Calidad de la ventilación**

Se pedirá a los usuarios que introduzcan una valoración estimada de la calidad de la ventilación como medida preventiva a la **transmisión aérea**, basados en los aspectos evaluables según la OMS (*Roadmap to improve and ensure good indoor ventilation in the context of COVID-19*) que son **la velocidad mínima del sistema de ventilación mecánico, la velocidad mínima de ventilación natural, y la dirección de la corriente.** Naturalmente, para evaluar estos aspectos correctamente se requiere información y exploraciones tomadas con instrumentos de medición. Los usuarios de la aplicación no contarán con este tipo de herramientas, por lo que se pedirá una valoración estimada basada en el criterio personal de cada usuario, ofreciendo un **apartado de información** donde se detallarán los aspectos evaluables mencionados anteriormente.

- **Distanciamiento social**

Se pedirá a los usuarios que realicen una evaluación subjetiva de las medidas que se toman en las localizaciones para mantener la distancia de seguridad de los transeúntes en todo momento: vías delimitadas para cada dirección, control de aforos en espacios cerrados, indicaciones del personal para mantener la distancia de seguridad... Teniendo todos estos aspectos en cuenta, se solicitará una evaluación de entre 0 y 5 estrellas de las medidas de distanciamiento social para cada localización.

- **Facilidad del saneamiento de manos**

Se solicitará a los usuarios que introduzcan una valoración estimada de las medidas tomadas en las localizaciones para facilitar el saneamiento de manos a los asistentes, como medida preventiva para la transmisión por gotas y contacto directo. Para esta medición, se pedirá a los usuarios que se tengan en cuenta aspectos como la existencia de dispensadores de gel alcohólico, la frecuencia de éstos y la disposición de los encargados del lugar a incitar a los transeúntes al higienizado de las manos. Teniendo estos aspectos en cuenta, los usuarios serán los que introduzcan su valoración entre 0 y 5 estrellas de las medidas de saneamiento de manos para cada localización.

- **Uso de mascarillas**

Se solicitará a los usuarios de la aplicación que realicen una evaluación estimada del uso de mascarillas generalizado de las personas de dicha localización, como medida preventiva para frenar la transmisión aérea. Los aspectos a tener en cuenta para la realización de la valoración serán el uso de la mascarilla por parte de los transeúntes, los empleados de la localización (si los hubiere) y las indicaciones del uso de mascarilla por parte de los encargados o empleados de la localización en caso de haberlos. Teniendo estos aspectos en cuenta, los usuarios serán los que introduzcan su valoración entre 0 y 5 estrellas del uso de mascarillas en cada localización

- Limpieza y desinfección del entorno

Se solicitará a los usuarios de la aplicación que realicen una evaluación estimada de la calidad en la limpieza y desinfección del entorno como medida preventiva para la transmisión indirecta a través de objetos o superficies, que, aunque no está demostrada como vía de transmisión del SARS-Cov-2, se tendrá en cuenta como criterio de evaluación, con una ponderación menor para la puntuación global de la localización. Los criterios para tener en cuenta en la elaboración de la evaluación de este aspecto serán la frecuencia de limpieza por parte del personal de la localización si lo hubiere y la limpieza aparente del lugar. Teniendo estos aspectos en cuenta, los usuarios serán los que introduzcan su valoración entre 0 y 5 estrellas de las medidas de limpieza y desinfección de cada localización.

La transmisión comunitaria se define como "experimentar grandes brotes de transmisión a los que se les puedan atribuir a características como: grandes números de casos no vinculables entre sí, una agrupación de casos identificados mediante la vigilancia centinela, y/o un conjunto de brotes no relacionados en un territorio." (Organización Mundial de la Salud, 2020).

Riesgos

Durante la operación de la aplicación desarrollada, pueden ocurrir inconvenientes, ataques o sucesos que pongan en peligro tanto la integridad del funcionamiento de la aplicación como la seguridad de la información almacenada. Se realiza la siguiente identificación de los principales riesgos:

- Acceso no autorizado a la información de los usuarios
- Suplantación de la identidad de los usuarios
- Tratamiento de datos personales de geolocalización de los usuarios
- Ataques de denegación de servicio
- Avería o caída de la máquina de alojamiento del back end.

Mitigaciones

En correspondencia con los riesgos identificados en el apartado anterior, se idean una serie de acciones y se establecen unas medidas con tal de mitigar o eliminar la existencia de estos.

- Implantación de un método de autenticación basado en el uso de tokens JSON, que identificará a los usuarios de forma inequívoca y solo les permitirá las operaciones correspondientes.
- Información de asistencia anónima

Para paliar el inconveniente de tratar datos de geolocalización de los usuarios, que implicaría el cumplimiento de las medidas establecidas en la LOPD y la correspondiente penalización por su tratamiento inadecuado, se opta por anonimizar la información de los usuarios, es decir, registrar asistencias anónimas a las diferentes localizaciones. De esta manera, se evita el registro de la localización de los usuarios, y se utilizan los datos de geolocalización estrictamente bajo el consentimiento de los usuarios. Nunca se utilizarán datos extraídos de los sensores GPS de los dispositivos móviles, por lo que tampoco se deberá de justificar el uso de la geolocalización de cara a la interfaz del sistema operativo del dispositivo móvil.

- En cuanto a los ataques de denegación de servicio y el mantenimiento de las máquinas en las que se alojará el back end, existen dos escenarios posibles:
    - En caso de alojar la ejecución del servidor web en una máquina administrada localmente, se deberá de establecer una estructura de red y de máquinas virtuales con la debida configuración requerida para asegurar la disponibilidad y restringir los accesos a los puertos adecuados. Esto requeriría la utilización de redes internas y firewalls.
    - En caso de alojar la ejecución del servidor web en un servicio de alojamiento externo, esta responsabilidad será delegada al proveedor del servicio, que deberá de asegurar que se cumplen los requisitos de seguridad y disponibilidad acordados (generalmente en un contrato).

Estado del arte

Para tratar el estado del arte relacionado con la aplicación que se pretende realizar, se deben de analizar las diferentes soluciones disponibles actualmente que compartan el objetivo de colaborar, contribuir o aportar en la lucha contra el COVID-19 de alguna manera. Después de haber hecho una búsqueda en los principales repositorios de aplicaciones disponibles para dispositivos móviles (Play Store y App Store), se identifican las aplicaciones más similares.

Radar COVID

Esta aplicación, diseñada y dirigida por la secretaría de Estado de Digitalización e Inteligencia Artificial del Gobierno de España, tiene como objetivo contribuir en la detención de la propagación del coronavirus. Cumple esta función mediante la detección de contactos con personas que hayan sido diagnosticadas de COVID 19 haciendo uso de la tecnología bluetooth de los dispositivos móviles. Basa su función en la publicación anónima del diagnóstico positivo dentro de la aplicación, para que los dispositivos móviles detecten los contactos con usuarios que hayan comunicado su diagnóstico.

Ilustración 1 Funcionamiento de Radar COVID

Tabla comparativa de las funcionalidades de Radar COVID-19 y Dónde ir App.

	Radar COVID	Dónde ir App
Funcionalidades	Informa a los usuarios de los contactos con positivos	Permite a los usuarios conocer una estimación de personas que acudirán a un lugar
Método	Emplea la tecnología bluetooth, que debe estar activada en todo momento	No emplea tecnologías de geolocalización ni de intercomunicación
Contribución	Reduce la propagación del COVID 19 mediante el diagnóstico de posibles positivos.	Pretende limitar la propagación del virus reduciendo las aglomeraciones y

Asistencia COVID-19

Asistencia COVID-19 es una aplicación oficial del Gobierno de España que permite a los usuarios autoevaluarse sobre los posibles síntomas del COVID-19, e informarse sobre las recomendaciones a seguir. El objetivo final de este aplicativo es ayudar a los ciudadanos en la autoevaluación de los ciudadanos con el fin de reducir el volumen de llamadas al número de emergencias sanitarias, informar a la población, permitir un triaje inicial de posibles casos y un seguimiento posterior por parte de las comunidades autónomas.

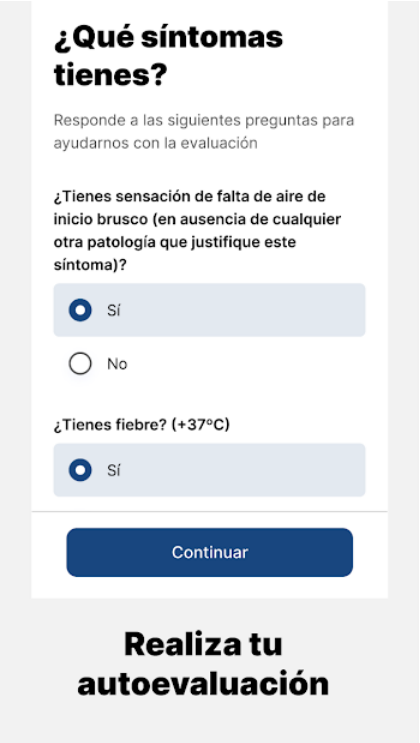

Ilustración 2 Funcionamiento de Asistencia COVID-19

Tabla comparativa de las funcionalidades de Asistencia COVID-19 y Dónde ir App.

	Asistencia COVID-19	Dónde ir App
Objetivo	Es una herramienta para el diagnóstico autónomo los síntomas del COVID-19.	Contribuye en la lucha contra el COVID-19 tratando de sanear el servicio de llamadas de emergencia sanitarias.
Contribución	Contribuye en la lucha contra el COVID-19 tratando de sanear el servicio de llamadas de emergencia sanitarias.	Su contribución a la lucha contra el COVID consiste en la reducción de las aglomeraciones.

Crush COVID Rhode Island

Crush COVID es una respuesta a la pandemia del COVID-19 que proporciona a los habitantes de Rhode Island acceso a todos los recursos necesarios durante la crisis sanitaria, incluyendo un diario de geolocalización y síntomas.

El diario de geolocalización utiliza la tecnología GPS de los dispositivos móviles para realizar un seguimiento de los lugares visitados en los últimos 20 días. Cuando se diagnostica un positivo, se emplea esta información de geolocalización para notificar a los usuarios con los que ha estado en contacto y los lugares visitados.

Otra de las funcionalidades proporcionadas por la aplicación es el diario de síntomas, donde introduciendo el código postal y un formulario diario se puede realizar un seguimiento de los posibles casos por códigos postales.

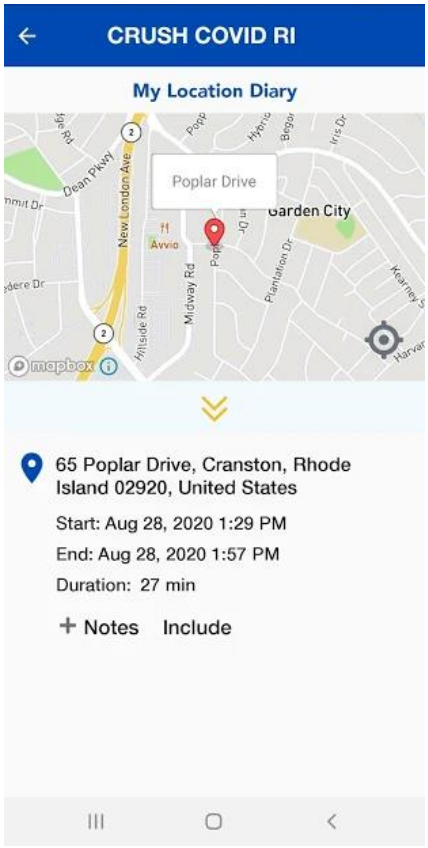

Ilustración 3 Funcionamiento de Crush Covid RI

Tabla comparativa entre Crush COVID-19 y Dónde ir App

	Crush COVID-19 App	Dónde ir App
Funcionalidad	Realiza seguimiento de los sitios en los que el usuario ha estado utilizando la tecnología GPS. Permite detectar y controlar los casos de COVID-19 por código postal	No requiere el uso de la geolocalización
Contribución	La aportación a la lucha contra el COVID de esta aplicación es el control y seguimiento de los casos positivos, facilita el rastreo de posibles nuevos positivos.	Su contribución a la lucha contra el COVID consiste en la reducción de las aglomeraciones y fomentar el cumplimiento de las medidas COVID.

Google popular times

Es interesante incluir esta funcionalidad de Google Maps dentro de esta comparativa, pues su objetivo es similar al de la aplicación desarrollada. Sin embargo, es posible identificar algunas

diferencias claves que posicionan a la aplicación desarrollada como una mejor opción para contribuir a la lucha contra el COVID.

En primer lugar, esta funcionalidad de Google Maps utiliza como fuente de datos la geolocalización anonimizada de usuarios que hayan voluntariamente permitido el uso de esta información, agregándola e interpretándola mediante algoritmos de inteligencia artificial para elaborar una predicción basada en los datos recogidos durante un periodo de tiempo previo a la fecha de consulta.

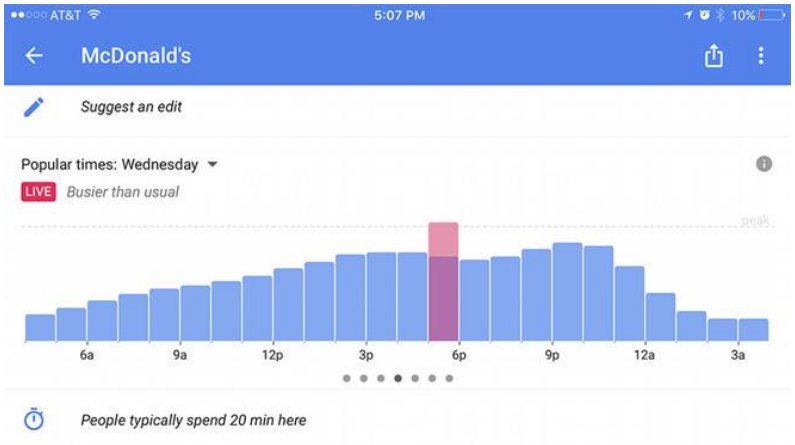

Ilustración 4 Ejemplo de funcionamiento de "Google popular times"

Sin embargo, la intención y el uso que se les da a estos datos no está relacionada con la prevención de aglomeraciones, sino más bien con el ahorro de tiempo de los usuarios. La aplicación desarrollada pretende recoger los datos de manera explícita, realizando una predicción más exacta y variable, además de incluir más funcionalidades que la diferencian de una función extra de Google Maps.

Tecnologías

De igual manera que se realiza un estudio sobre el contexto y el estado del arte, antes de decidir sobre qué tecnologías se pretende realizar el desarrollo de la aplicación es necesario una correspondiente exploración de las diferentes alternativas existentes en el mercado.

Framework para la aplicación móvil

A la hora de seleccionar una librería o un framework de desarrollo multiplataforma sobre el que articular el proceso de creación de un proyecto, es de vital importancia tener en cuenta algunos aspectos que influirán considerablemente en la experiencia del programador y en el rendimiento de la aplicación, como el lenguaje de programación, el estado actual de uso, el soporte, la comunidad...

Por ello, se realiza una pequeña comparativa entre las 3 principales opciones contempladas para el desarrollo, seleccionadas por su popularidad y su extenso uso, que son Flutter, Ionic y React Native.

Flutter

Flutter es un kit de desarrollo software de código abierto elaborado por Google para la elaboración de aplicaciones multiplataforma (IOS, Linux, Android, Windows, Web) a partir del mismo código fuente.

Principales ventajas

- Documentación

Es uno de los frameworks de desarrollo multiplataforma más empleados en los últimos años, y cuenta con una documentación excelente. Esto lo convierte en uno de los frameworks de desarrollo mejor mantenidos y con el respaldo de una de las mayores comunidades de desarrolladores.

- Rapidez

Futter está construido sobre Skia, que es la librería gráfica 2D sobre la que se desarrollaron Google Chrome y Android. El lenguaje de desarrollo empleado es Dart, que permite la compilación a código máquina ARM de 32 o 64 bits, así como JavaScript para la web y la arquitectura x64 de los computadores tradicionales.

- Experiencia de usuario

Permite a los diseñadores la utilización de su visión creativa al máximo, eliminando por completo las limitaciones derivadas de la utilización de un framework. Utiliza una estructura por capas, concediendo al desarrollador el control de cada uno de los pixeles de la pantalla.

- Desarrollo productivo

Flutter ofrece la posibilidad a los desarrolladores de ver los cambios realizados en el código fuente inmediatamente sin necesidad de reinicio de la aplicación, ofreciendo una experiencia de desarrollo fluida y rápida.

Ionic

Ionic es un framework de desarrollo de aplicaciones móviles híbridas[5] de código abierto que permite el desarrollo de aplicaciones web nativas de alta calidad con tecnologías web. Este framework está basado en Web Components[6] , y cuenta con gran rendimiento y usabilidad.

[5] El desarrollo híbrido de aplicaciones móviles consiste en la creación y ejecución de un proyecto de aplicación en múltiples plataformas (iOS, Windows, Android...). Suponen una pérdida de rendimiento en comparación con las aplicaciones nativas, pero a cambio ofrecen mejor integración y rapidez en el desarrollo

[6] Los Componentes Web son un paquete de diferentes tecnologías que te permiten crear elementos personalizados reutilizables — con su funcionalidad encapsulada apartada del resto del código — y utilizarlos en las aplicaciones web.

Principales ventajas

- Integración con frameworks de desarrollo web populares

Ionic permite el desarrollo de aplicaciones haciendo uso de frameworks de desarrollo web con uso muy extendido como Angular, React y Vuejs. Ionic utiliza Cordova para permitir el uso de los controladores nativos de los diferentes dispositivos.

- Comunidad

Es uno de los frameworks multiplataforma más antiguos, por lo que cuenta con una gran comunidad y soporte.

- Rendimiento

El uso de tecnologías web avanzadas como HTML, CSS y JavaScript permite que las aplicaciones desarrolladas con Ionic tengan un alto rendimiento.

React Native

React Native, al igual que Ionic y Flutter, es un framework de desarrollo móvil multiplataforma, que permite el desarrollo de aplicaciones para diferentes dispositivos.

- Integra las ventajas de React al desarrollo móvil

Una de las principales ventajas de React para el desarrollo de aplicaciones web es que está basado en componentes, basando el desarrollo en la creación de diferentes módulos encapsulados con sus propios estados y realizar composiciones de los componentes para crear interfaces de usuario complejas.

- Desarrollo ágil

Permite la compilación en vivo de las aplicaciones durante el desarrollo.

- Portabilidad

Permite la reutilización del código en diversas plataformas

Tabla comparativa

	React Native	Ionic	Flutter
Lenguaje de programación	React Native utiliza JavaScript como lenguaje de programación, haciendo uso de React. También permite escribir componentes en Swift, Objective-C o Java cuando es necesario.	Utiliza múltiples tecnologías web (HTML5, CSS, JS).	Flutter emplea Dart, que es un lenguaje creado específicamente para el desarrollo de aplicaciones web, Android e iOS.
Rendimiento	React Native utiliza API y componentes nativos, ofreciendo un rendimiento casi nativo.	Ionic requiere el uso de plugins y paquetes de terceros que envuelven la aplicación en una capa nativa.	Flutter ofrece el mejor rendimiento de los 3 frameworks gracias a su propio motor de renderizado y un amplio rango de widgets y plugins.
Estado actual de uso	Gracias a estar basado en JavaScript ha sido empleado por grandes compañías como Facebook o Airbnb.	Ionic ha sido empleado por compañías como McLaren o McDonalds.	Flutter también cuenta con un amplio abanico de grandes empresas que lo emplean: ebay, Groupon, Tencent...

Después de este contraste entre las principales opciones consideradas, se decide utilizar Flutter como framework de desarrollo para la aplicación.

API de geolocalización

De igual manera que se realiza una comparativa entre las características más relevantes de los frameworks de desarrollo contemplados para el proyecto, se debe de seguir el mismo procedimiento con las distintas opciones disponibles para obtener un servicio de geolocalización que permita dotar a la aplicación de la capacidad de mostrar mapas y trabajar con datos geográficos. Los principales proveedores de servicios de este tipo son Google, Mapbox, Here y Open Street Maps.

> *La comparación de los precios se realizará teniendo en cuenta los principales servicios necesarios para la implementación de las funciones de la aplicación:*
>
> *Servicios de dibujado de mapas: necesario para implementar la navegación de los usuarios en el mapa*
>
> *Servicios de búsqueda de lugares (Geocoding): se requiere el uso de este tipo de servicios para permitir a los usuarios realizar búsquedas de lugares concretos y transformar dichas búsquedas en vectores geográficos*

Google Maps

Google Maps es un servicio de mapeado web desarrollado por Google. Ofrece imágenes satélites, aéreas, imágenes interactivas de calle, condiciones en tiempo real del tráfico y planificación de rutas.

Google Maps para Android e iOS fue lanzado en 2008, incorporando la tecnología GPS de los dispositivos móviles para permitir la navegación en tiempo real.

La API de Google Maps, denominada Plataforma Google Maps, aloja alrededor de 17 APIs diferentes, recogidas bajo los términos de Lugares, Mapas y Rutas. Inicialmente se trataba de una API JavaScript, fue expandida para contener una API para aplicaciones desarrolladas en Adobe Flash (actualmente deprecada), un servicio para proveer imágenes estáticas y servicios web para permitir el Geocoding generando indicaciones de rutas, así como obtener perfiles de elevación.

Open Street Maps

Open Street Maps es un proyecto colaborativo con el objetivo de crear un mapa gratuito y editable del mundo. El principal producto del proyecto son los datos de geolocalización.

Los datos son tomados por voluntarios que realizan exploraciones sistemáticas del terreno empleando herramientas como unidades GPS de mano, un cuaderno, cámaras digitales o grabadores de voz. Estos datos son introducidos posteriormente a la base de datos de Open Street Maps.

HERE

HERE WeGo es un servicio de mapeado y navegación web, operado por Here Tecgnologies. Originalmente desarrollado por Nokia bajo en nombre de HERE Maps, fue lanzada en 2013 como una versión mejorada de Nokia Maps. Es el principal proveedor de mapas para las tabletas y smartphones de Amazon.

Mapbox

Mapbox es un proveedor online de mapas personalizados. Desde 2011 ha expandido considerablemente su catálogo de mapas personalizados, en respuesta a la limitada oferta de proveedores de mapas como Google Maps.

Mapbox toma sus datos de geolocalización de fuentes de datos abiertas como Nasa o Open Street Maps, así como de fuentes de datos propietarias como DigitalGlobe. Basan su funcionamiento en Node.js, Mapnik, GDAL y Leaflet.

Aplicaciones que usan Mapbox

- Facebook

La aplicación nativa de Facebook emplea Mapbox para mostrar los lugares de interés, restaurantes...

- Snapchat

Snap utiliza Mapbox para mostrar *Snap Map*, un mapa de calor en tiempo real que muestra qué está pasando ahora mismo. Los usuarios pueden navegar por todo el globo para encontrar historias relevantes para ellos.

- Skyscanner

Skyscanner ayuda diariamente a miles de usuarios a encontrar su viaje perfecto gracias al despliegue de un mapa que muestra en directo a qué lugares del mundo pueden viajar, mostrando restricciones de entrada, requisitos de cuarentena...

Comparativa de precios

	Google Maps	Open Street Maps	Here	Mapbox
Dibujado de mapas	Gratis para móviles[7]	Gratis	Gratis hasta 250 mil transacciones	Gratis hasta 25000 usuarios
Geocoding	5 USD/1000 solicitudes	A través de módulos externos	Gratis hasta 250 mil transacciones	Gratis hasta 100000 transacciones

Porqué Mapbox

Finalmente, la decisión de emplear Mapbox para el desarrollo de la aplicación por encima de las demás opciones se debe a 4 aspectos fundamentales:

- **Integración con Flutter**. Gracias a flutter-mapbox-gl[8], una librería que ofrece integración directa del SDK de Mapbox para Android y iOS en forma de Widgets para Flutter, el desarrollo y la inclusión de mapas interactivos en la aplicación se simplifica considerablemente.
- **Soporte**. Mapbox debe en gran parte su popularidad a la confianza que grandes compañías como Facebook, Snapchat o CNN han depositado sobre sus servicios, y esto le ha convertido en uno de los principales proveedores de mapas a nivel mundial.
- **Personalización**. Mapbox ofrece una herramienta de edición online de mapas, donde se permite la edición de aspectos de los mapas como la visualización de puntos de interés, qué capas de información geográfica se desea mostrar (ríos, altitud, mares...) así como los colores y el aspecto del mapa.
- **Facturación.** En este apartado, se destaca de los demás frameworks que se asimilan más en los demás apartados, concretamente **Google Maps,** pues la diferencia de precio en la rama para desarrolladores es bastante considerable, por no mencionar los siguientes niveles, donde Mapbox ofrece opciones más flexibles y escalables que las de Google Maps.

[7] Google Maps ofrece un crédito de 200 USD al mes para las aplicaciones en desarrollo
[8] Librería para Flutter (https://github.com/tobrun/flutter-mapbox-gl)

Cabe destacar que la elección de Mapbox frente a la opción gratuita, Open Street Maps, se debe principalmente a la **experiencia de usuario** mejorada que ofrece Mapbox y a la cantidad de **funciones con las que cuenta Mapbox** de serie que ahorran horas de desarrollo y que serían imposibles de abarcar en un tiempo tan limitado.

Plan de negocio

Pese a los costes de despliegue que presenta el sistema de información desarrollado, el objetivo principal del desarrollo de esta aplicación no es la obtención de beneficio económico mediante la oferta de un servicio a sus clientes. La aplicación ha sido ideada y desarrollada con el fin de contribuir de manera desinteresada en la lucha contra el COVID-19.

No obstante, es posible la inclusión de un sistema de impresión de anuncios en pantalla, lo que naturalmente permitiría explotar económicamente el uso de la aplicación.

Sin embargo, la razón de ser principal de la aplicación no es, en mi opinión, explotable económicamente. Es por ello, que la intención principal con la que se desarrolló la aplicación fue la de que la fase de despliegue de la aplicación, en caso de que la hubiere, fuera asumida por alguna administración pública, estatal o regional, ya que serían este tipo de entidades las principales interesadas en el aprovechamiento de las funcionalidades que la aplicación ofrece y los objetivos que persigue.

Capítulo 3: Diseño

La fase siguiente al análisis de los diferentes aspectos que engloba el desarrollo de una aplicación es la de diseño. Se han planteado las historias de usuario, los objetivos, los riesgos y mitigaciones correspondientes, y se han identificado los aspectos claves para diferenciar el producto de su competencia. Consecuentemente, se procede con el diseño de la aplicación, que implica el conjunto de la aplicación y sus diferentes partes, incluyendo el diagrama del sistema de información, los mockups de la interfaz de usuario y más específicamente el diagrama de clases que compondrán los diferentes módulos de la aplicación.

Diagrama de flujo

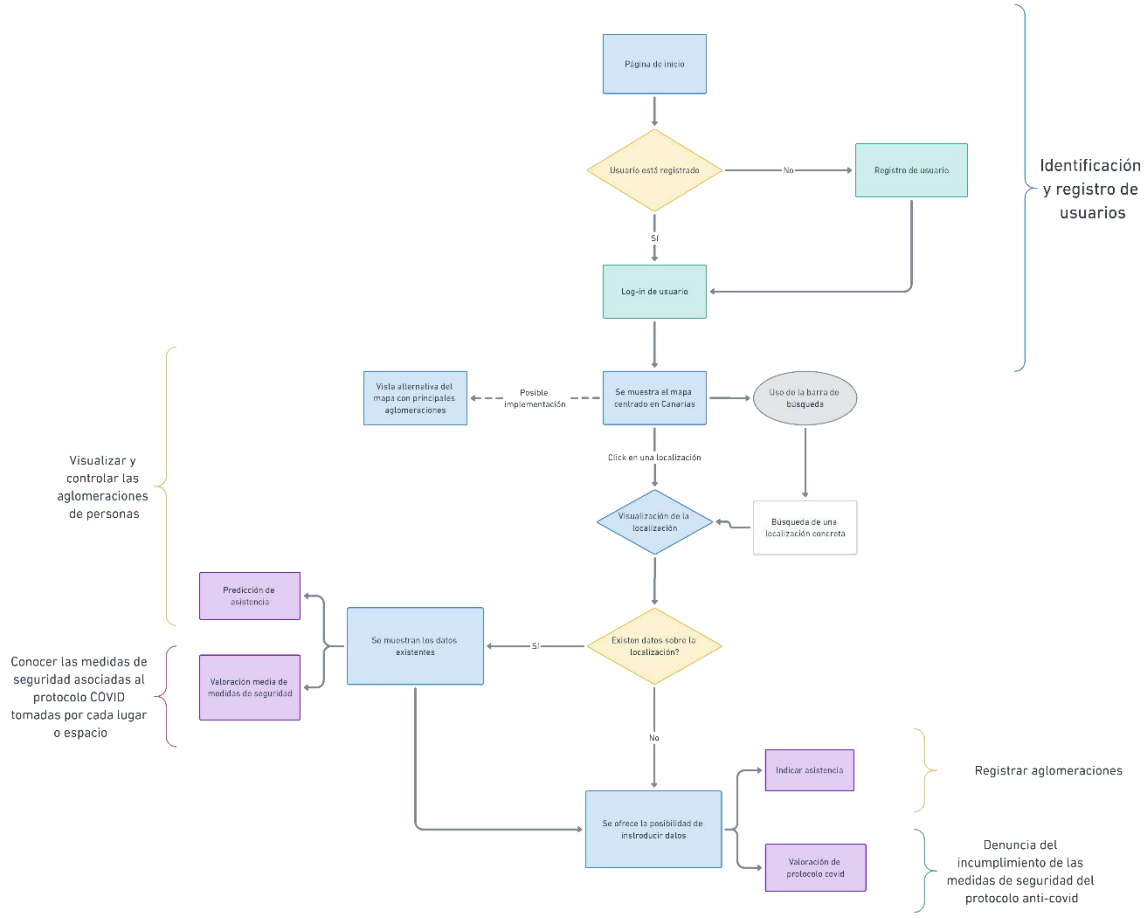

Ilustración 5 Diagrama de flujo

En la ilustración se describe el flujo de las acciones de los usuarios, y se identifican los diferentes objetivos que se pretenden logar con la elaboración del proyecto. Por un lado, se identifican los pasos previos a la utilización de la aplicación, el registro y la identificación de usuarios. Estos pasos son necesarios para permitir la realización de las siguientes tareas indicadas en el diagrama, que se corresponden con la visualización y control de las aglomeraciones de personas, el conocimiento de las medidas de seguridad asociadas al protocolo COVID tomadas por cada localización y la denuncia del incumplimiento de estas. El registro e identificación de los usuarios es necesario para poder llevar un seguimiento de la

actividad que estos realicen y poder contabilizar las asistencias y las valoraciones sobre los diferentes protocolos de manera correcta y precisa.

Una vez el usuario se encuentre registrado e identificado dentro de la aplicación, se le permitirá realizar las acciones que cubren los objetivos planteados durante la fase de análisis (mencionadas anteriormente). Con este flujo de acciones se establece el primer diagrama, sobre el que se articularán otras fases del diseño y que incrementalmente supondrá las bases para definir un modelo de aplicación concreto y especificado concretamente.

Layout del sistema

Para establecer una estructura general de las diferentes partes que compondrán el Sistema de Información que se pretende implementar, es bastante útil elaborar un diagrama en el que se representen las diferentes capas de software que compondrán la aplicación.

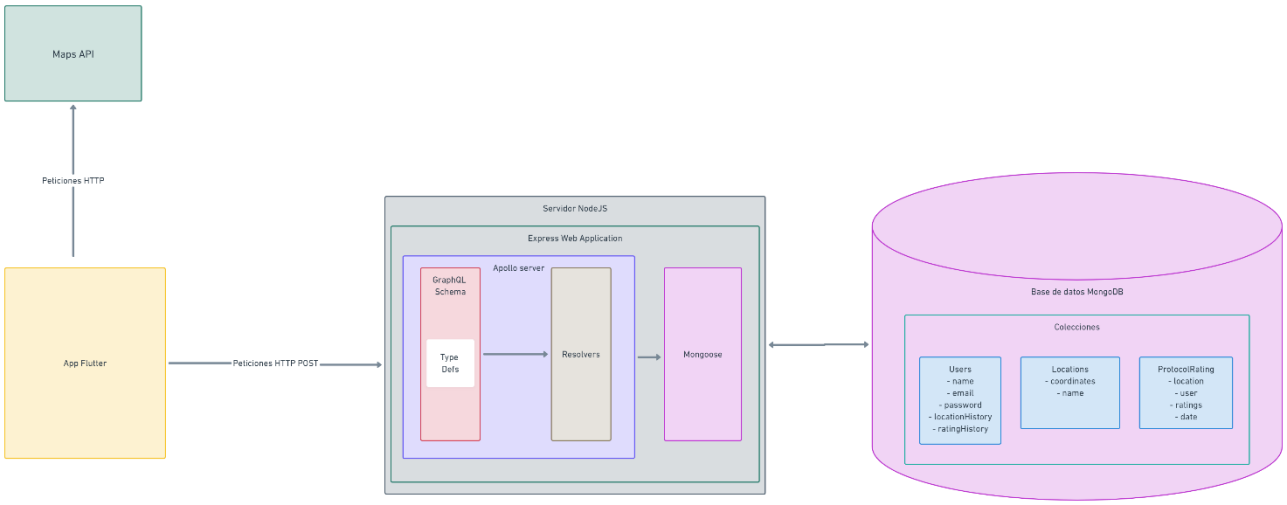

Ilustración 6 Layout del sistema – Completo

El diagrama elaborado (Ilustración 6 Layout del sistema – Completo) pretende establecer una estructura general del sistema de información implementado, diferenciando los diferentes módulos. En él se pueden distinguir dos grandes bloques, uno que se puede identificar como el conjunto de módulos que componen el back end y el otro que se corresponde con los módulos que compondrán el front end de la aplicación.

Los bloques de la aplicación (izquierda, amarillo) y la API de mapas (derecha, verde) se desarrollarán y explicarán en apartados posteriores (diagrama de clases).

Los bloques del servidor Node.js (centro, verde oscuro) y la Base de datos MongoDB (derecha, morado) son los que compondrán el back-end de la aplicación, y merecen ser explicados con más detalle.

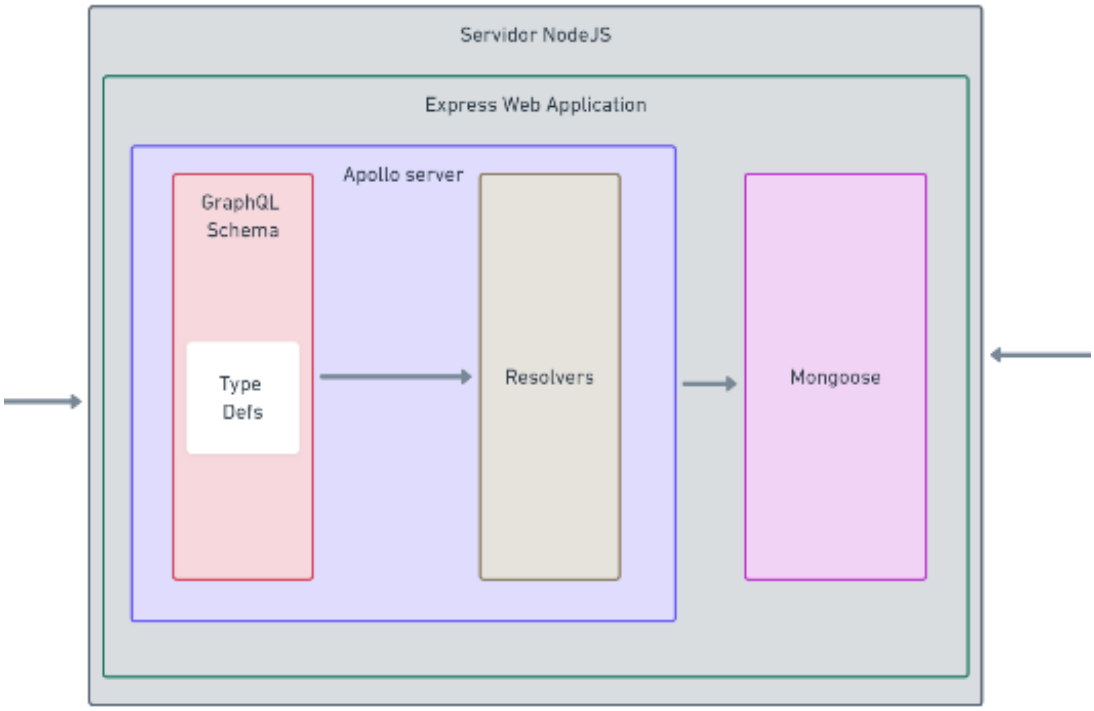

Ilustración 7 Layout del sistema - Detalle

En el detalle del layout del sistema se aprecia cómo el módulo de software que será ejecutado en Node.js hará uso de otras tecnologías internamente para estructurar y componer la capa de software encargada de responder las peticiones del Front end en forma de peticiones HTTP. Esto se realizará haciendo uso de la librería para manejar peticiones HTTP de Node.js Express.js.

Las peticiones que llegan al sistema serán procesadas por los *resolvers,* que son una serie de funciones que se ejecutan para atender las peticiones de los clientes. Estas funciones deben estar definidas previamente en un esquema de definición de tipos y clases contenido en el fichero *Type Defs*. Este fichero sirve a su vez de interfaz para los clientes que quieran conocer el funcionamiento de la API, pues pueden utilizarlo como guía para conocer los diferentes tipos y operaciones que permite. Toda esta estructura de archivos es procesada y ejecutada por una librería software denominada Apollo Server, que es la encargada de dotar a la aplicación de la capacidad de procesar peticiones HTTP con formato GraphQL.

Una vez las peticiones son recibidas y los *resolvers* son ejecutados, se hace uso de una librería de acceso a base de datos denominada *mongoose*, que como se ha mencionado en apartados anteriores, permite el acceso mediante esquemas a las colecciones de la base de datos MongoDB. Este driver abstrae el uso de colecciones e índices a la interacción con una serie de objetos definidos previos a la ejecución de la aplicación.

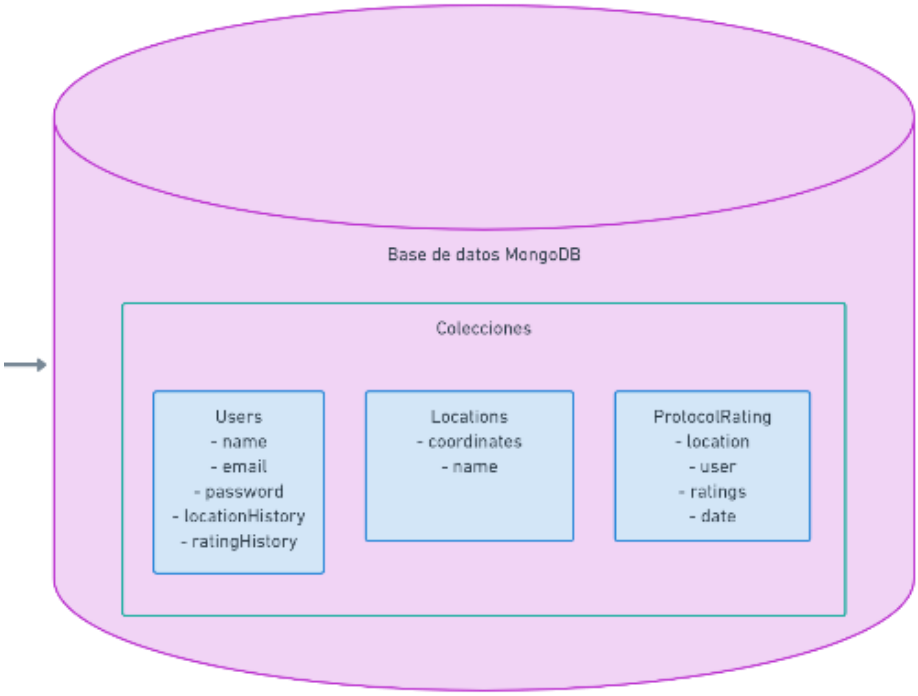

Ilustración 8 Layout del sistema - Detalle 2

Por último, para lograr un mejor entendimiento de la capa de software que se encarga de la persistencia de la información dentro de la aplicación, se realiza un análisis de la representación gráfica del módulo que se encarga de ello. Se aprecia en el detalle (Ilustración 7 Layout del sistema – detalle 2) cómo el almacenamiento se estructura en diferentes colecciones, y estas tendrán una serie de campos. Estos campos son especificados en el esquema de mongoose previa a la ejecución del sistema, y es el propio mongoose el encargado de su lectura, carga, actualización y mantenimiento.

Mockups de la aplicación

Otra fase importante de cara a la creación de una interfaz de usuario correcta y que sea capaz de plasmar eficazmente las funcionalidades y las intenciones de la aplicación, es la del diseño de los mockups. La realización de esta fase permite agilizar considerablemente la fase posterior de implementación.

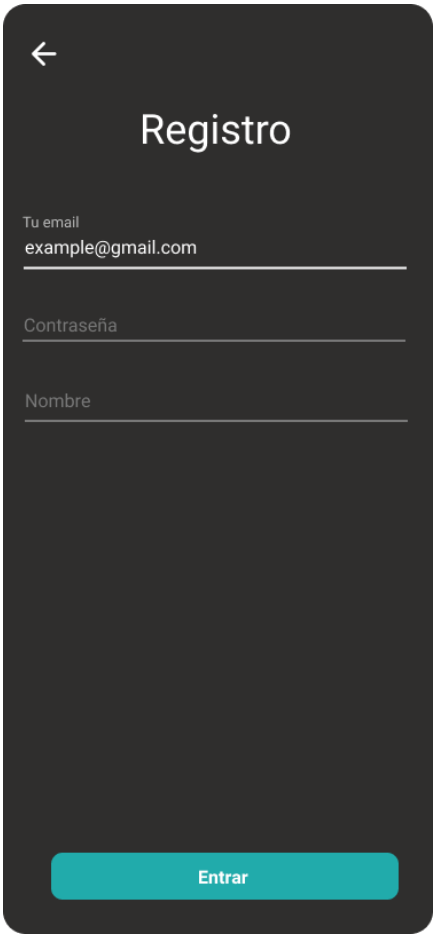

Ilustración 9 Mockup de la página de registro

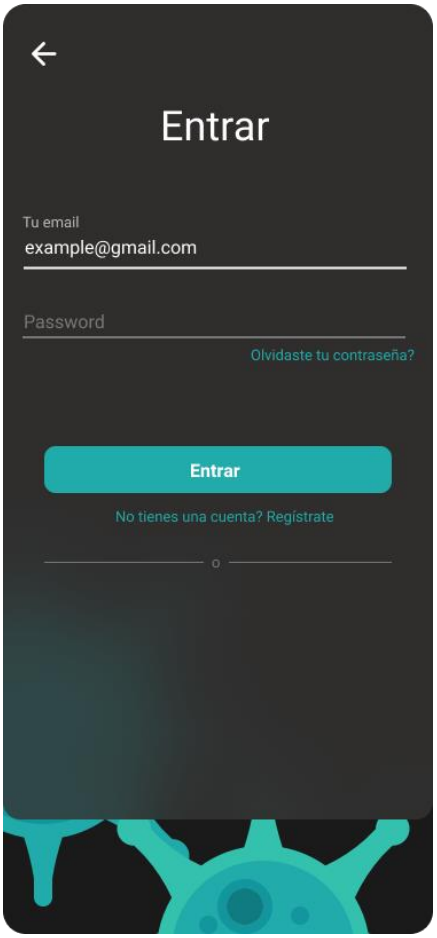

Ilustración 10 Mockup de la página de identificación

Estos diseños pretenden implementar las funcionalidades básicas para permitir el **registro** y la **identificación** de usuarios. Se incluirán una serie de formularios para permitir a los usuarios introducir sus datos de registro e identificación, para poder acceder a las funcionalidades de la aplicación.

Ilustración 11 Mockup de la página de navegación

Ilustración 12Mockup de la página de localización seleccionada

Estas ilustraciones muestran el diseño ideado para implementar las funcionalidades que permitirán a los usuarios navegar y seleccionar las diferentes localizaciones disponibles dentro de la aplicación. De esta manera, los usuarios de la aplicación tendrán una interfaz funcional que les permitirá buscar su lugar de ocio y seleccionarlo para visualizar la información sobre las diferentes características que almacena el sistema (Asistentes, valoraciones medias...)

Ilustración 13 Información sobre una localización

Ilustración 14 Mockup de la página para valorar una localización

Por último, estos son los mockups que muestran la interfaz que será presentada a los usuarios para realizar las diferentes acciones relacionadas con cada una de las localizaciones de la aplicación: una vez seleccionada una localización que esté registrada en la base de datos, se permitirá a los usuarios visualizar la información sobre las valoraciones de otros usuarios o la asistencia para ese día, como realizar una valoración sobre los diferentes protocolos anti-COVID que se siguen en ese lugar.

Diagrama de clases de la aplicación

El siguiente paso, una vez diseñada la interfaz con la que los usuarios interactuarán directamente, es establecer la estructura de clases y entidades que permitirá implementar las funcionalidades descritas por las historias de usuario y los objetivos de la aplicación. Para el desarrollo de este diagrama de clases, se ha utilizado un enfoque de diseño basado en la arquitectura hexagonal.

Arquitectura hexagonal

La arquitectura hexagonal es un patrón de diseño de software basado en la separación de los diferentes módulos en capas, permitiendo su evolución de manera aislada y responsabilizando a cada entidad de una funcionalidad única.

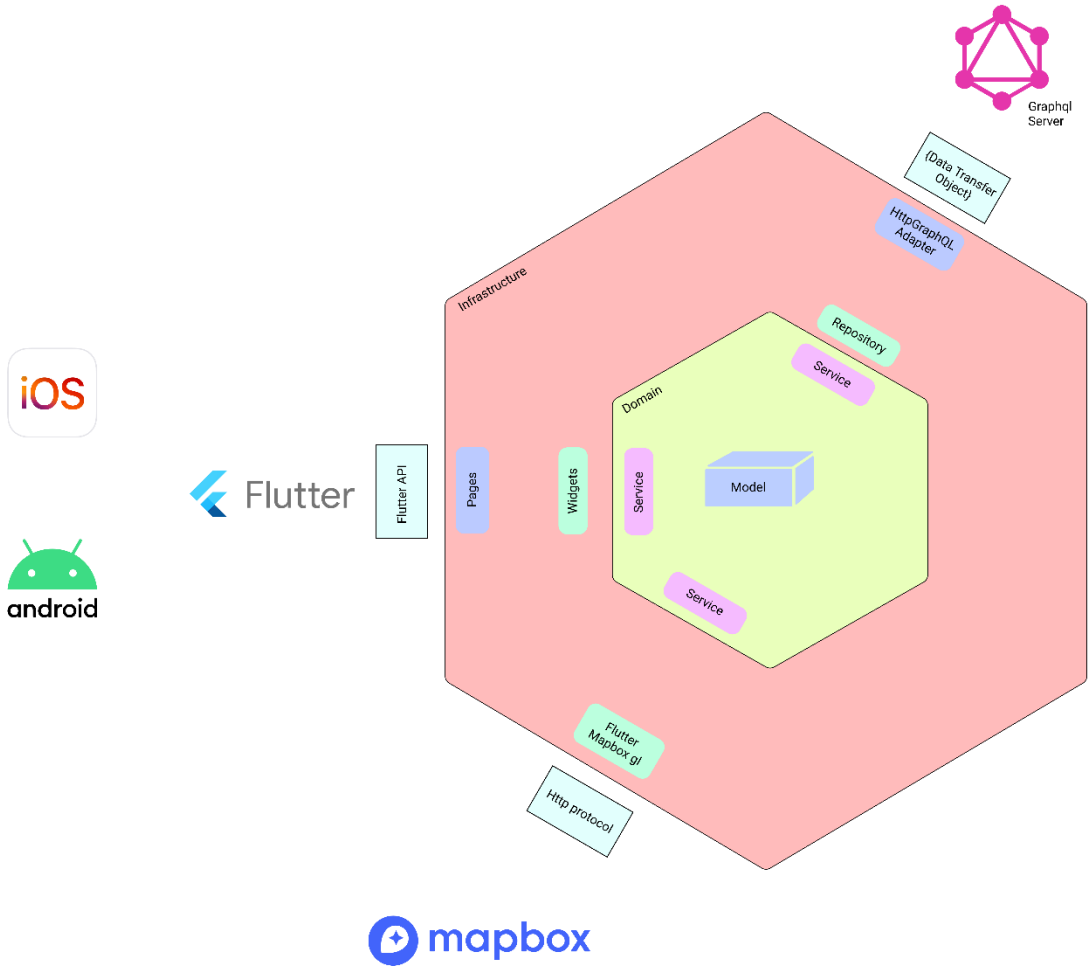

Ilustración 15 Arquitectura hexagonal en la aplicación desarrollada

El término hexagonal viene de la facilidad que representa asociar el concepto teórico con el concepto visual, pues se plantea la distribución de las capas de manera que en el interior del hexágono se encuentra el código base, denominado dominio, y en cada uno de sus laterales se sitúan las interacciones con los servicios externos. En el caso de la aplicación desarrollada, se pueden localizar las capas de interacción externas como los widgets de Flutter, la API de mapas o el propio Back End de la aplicación. La idea principal es **aislar** los módulos de software que interaccionan con estas entidades de manera que la parte del código que gobierne el funcionamiento de la aplicación se sitúe en la capa de dominio.

Front end

Siguiendo el concepto definido en el apartado anterior, se procede con la elaboración de un diagrama de clases que cumpla con la estructura arquitectónica planteada en el apartado anterior.

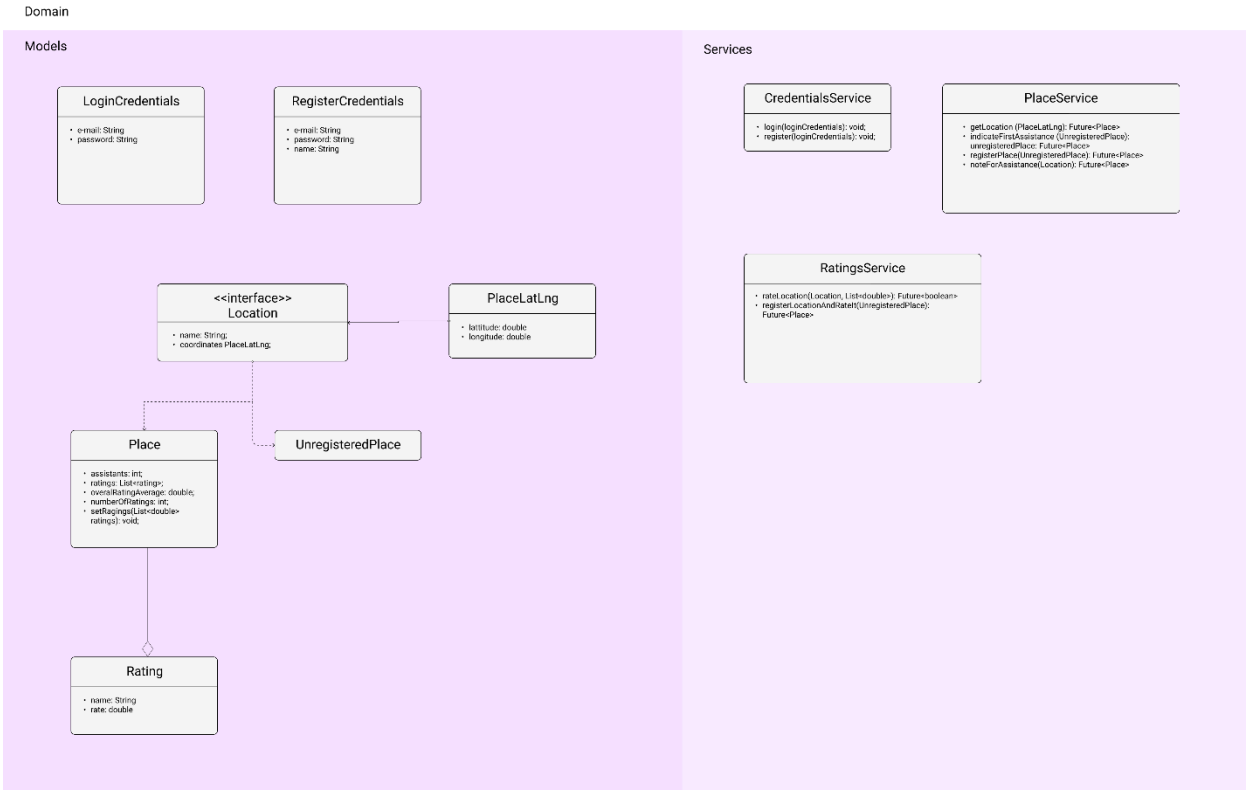

Ilustración 16 Diagrama de clases del Front End

Este diagrama de clases se corresponde con la capa del front end que contiene la lógica y las entidades bases sobre las que se estructura la aplicación. El resto de las capas son específicas de las tecnologías utilizadas para visualizar e interactuar con los datos, como Flutter o GraphQL, y sus estructuras y clases están ceñidas a lo establecido por los frameworks.

Back end

El uso de GraphQL y Mongoose en el desarrollo del Back-end de la aplicación, provoca que arquitectónicamente, la distribución de clases y de ficheros esté dictada por la estructura predefinida por los frameworks de desarrollo. No obstante, dentro de la relativa libertad de estructuración que resta, se ha seguido la filosofía de distribución y aislamiento de dependencias para lograr un código que respete los principios SOLID. Como resultado, se realiza la distribución de los diferentes ficheros en directorios independientes, diferenciando por módulos, y se centraliza la ejecución en un fichero denominado `API.js` que es el punto de partida de la aplicación cuando se ejecuta en Node.js.

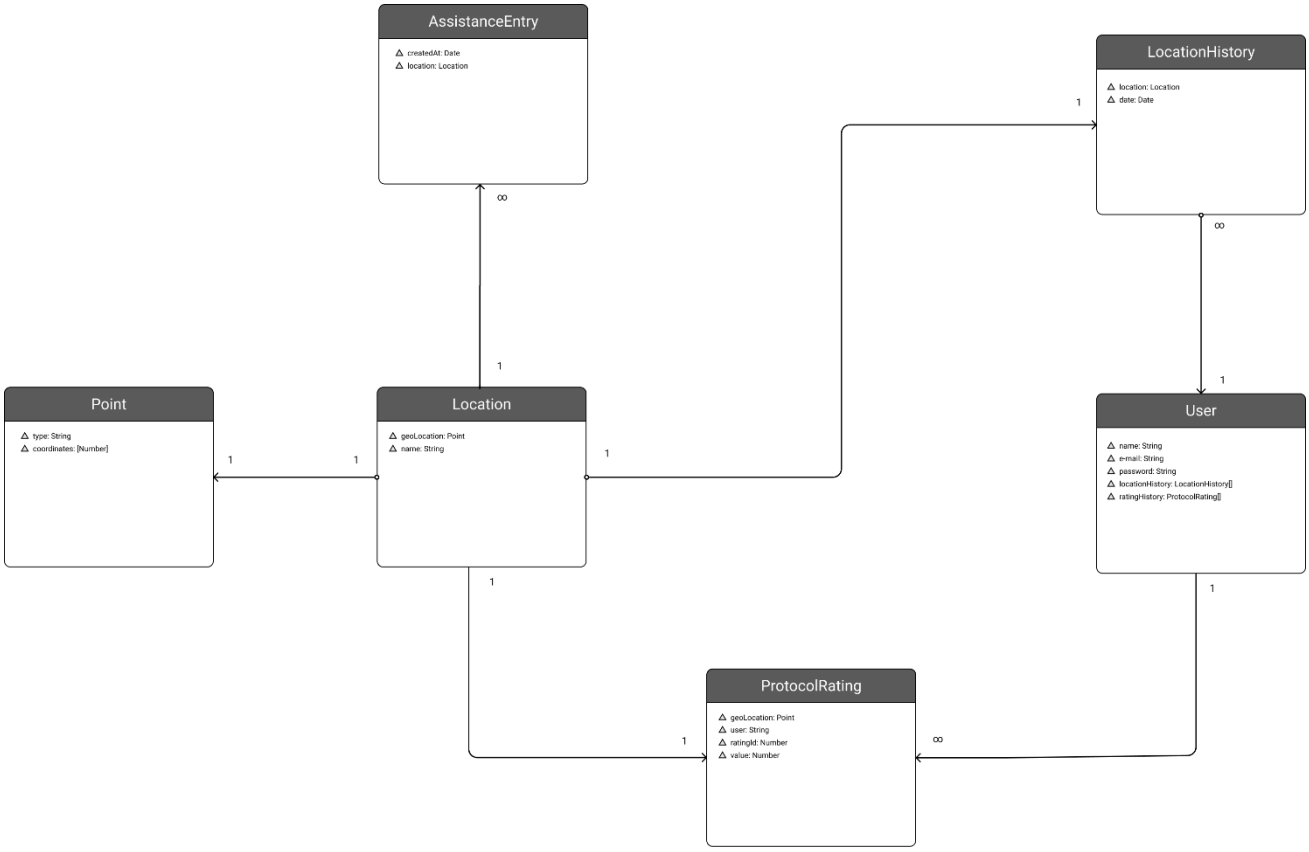

Ilustración 17 Diagrama de clases del almacenamiento

Esta figura pretende representar la estructura de clases empleada para estructurar el almacenamiento de información en la base de datos. Como se ha mencionado anteriormente, pese a que se emplea una base de datos no relacional para dotar de persistencia al almacenamiento de información en el sistema, el uso de una capa de software (Mongoose) que concede al sistema la capacidad de definir un esquema sobre el que se gobernará el almacenamiento de documentos en MongoDB, permite la implementación de este esquema en forma de clases estáticas en Mongoose.

Capítulo 4: Implementación

Una vez se disponen los recursos y la base documental necesarias para respaldar el proceso de desarrollo, es posible comenzar a describir la fase de implementación del proyecto.

Metodología

El primer aspecto por destacar de la fase de desarrollo es la instauración de una metodología de desarrollo inspirada en SCRUM, para un equipo de desarrollo unipersonal, basada en el uso de las historias de usuario y los Sprint para guiar el desarrollo.

En primer lugar, como se menciona en el capítulo anterior, durante la fase de diseño se elaboraron una serie de historias de usuario, que son las que recogen el conjunto de funcionalidades que se pretenden implementar. Una vez establecidas las historias de usuario y establecidos los requerimientos de la aplicación, se procede con la implementación de manera incremental, manteniendo reuniones semanales con el profesor de TFG para llevar un seguimiento del avance.

Se establecieron una serie de Sprint, en los cuales se marcaron objetivos a cumplir en un periodo de tiempo determinado (generalmente 2 semanas). Estos objetivos se corresponden con los requisitos de validación de las historias de usuario, es decir, después del periodo de tiempo preestablecido se deberá de comprobar que la aplicación cumple con los criterios de validación impuestos por las historias de usuario designadas para ese Sprint.

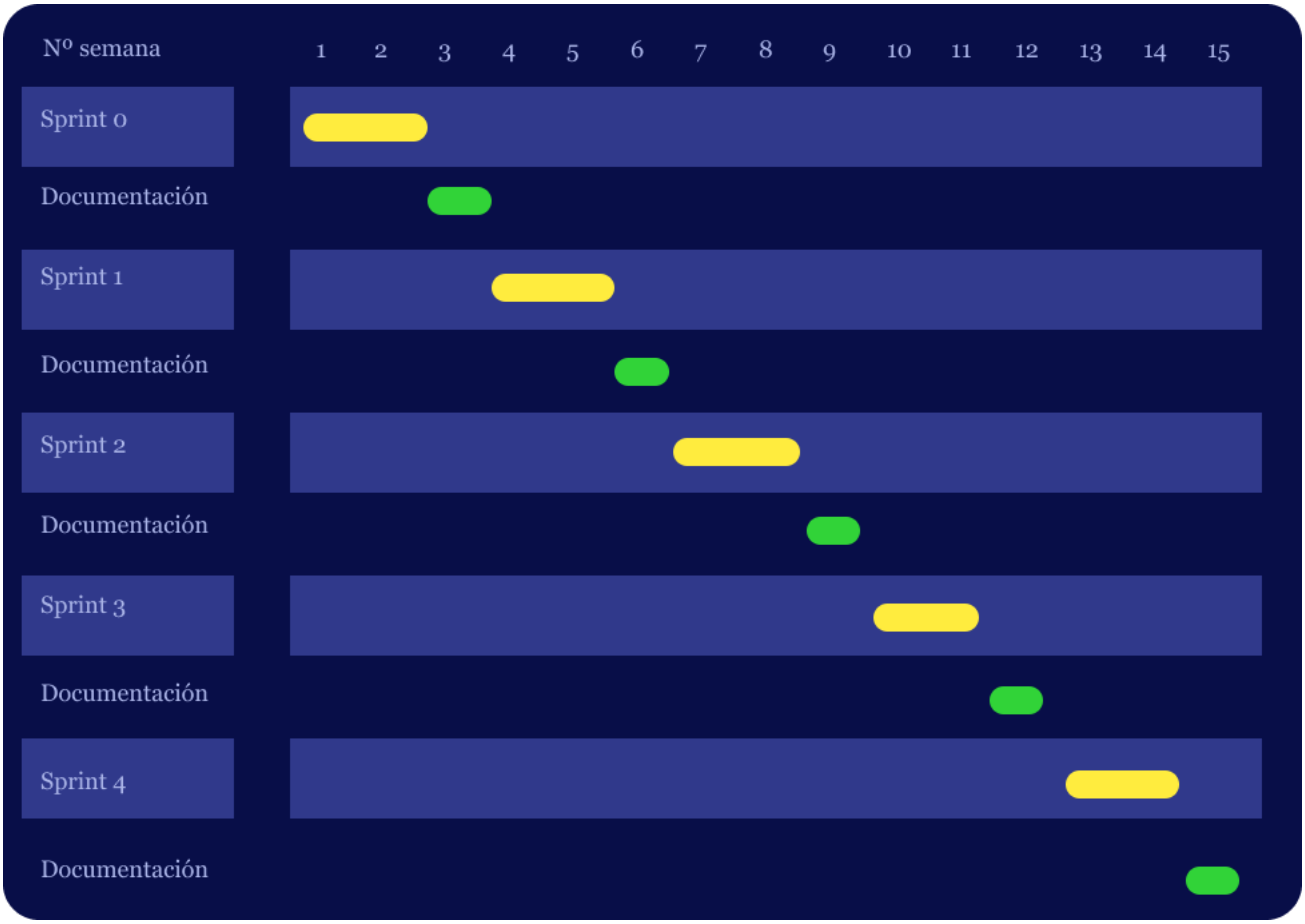

Ilustración 18 Diagrama de Gantt

Como se muestra en Ilustración 18 Diagrama de Gantt, la distribución de las tareas durante el periodo de desarrollo del proyecto fue cíclica: se alternaban periodos de desarrollo de las historias de usuario designadas para cada sprint con periodos de elaboración de la memoria, donde se documentaba incrementalmente las tareas y pasos seguidos para satisfacer los requisitos de las historias de usuario implementadas durante esa semana. La duración de los periodos descritos era de 2 y 1 semana correspondientemente. La distribución de las historias de usuario en los diferentes Sprint fue la siguiente:

Historias de usuario designadas a cada Sprint

	Historias de usuario designadas
Sprint 0	Configuración inicial (mockups, entorno de trabajo...)
Sprint 1	Registro, Identificación
Sprint 2	Navegación, Indicar Asistencia
Sprint 3	Visualizar asistencia, Valoración del protocolo COVID de una localización
Sprint 4	Visualización de las valoraciones de otros usuarios

Naturalmente, al tratarse de un equipo de desarrollo unipersonal, no es necesario mantener reuniones de retrospectiva, y la parte de SCRUM dedicada a la organización y coordinación de los equipos de desarrollo se omite por completo. El único rol que se traslada desde SCRUM es el de *product owner,* representado simbólicamente por el tutor de TFG.

GraphQL: entidades y detalles de implementación

De cara a la ejecución del back end cumpliendo con la especificación de GraphQL, se debe de configurar el servidor Apollo con una serie de parámetros, que dictarán la estructura de la API y definirá las entidades de datos que los clientes podrán obtener.

Type Defs

Type Defs o definición de tipos en castellano es un esquema de clases, en el caso de Apollo una ristra, que se le aporta a la configuración del servidor durante su inicialización, que sirve para configurar el modelo de resolución de objetos que implementa GraphQL y definir *qué datos* se pueden obtener.

Esta ristra tiene un formato específico, consistido en unas palabras reservadas (como `type`, `input`, `String` o `float`) y una sintaxis concreta. Por ejemplo, en el caso de la definición de la estructura de datos que compone un usuario en la aplicación, su definición en el esquema de GraphQL se realiza de la siguiente manera:

```
type Location {
  geoLocation: GeoLocation
  name: String
  predictedAssistance: Int
  protocolRatings: [ProtocolRating]
}
```

Ilustración 19 Ejemplo de definición de tipo GraphQL

Otro paso importante en la configuración del esquema de Apollo es la definición de las peticiones (queries) y las mutaciones. Ambas se definen dentro de la ristra descrita anteriormente (typeDefs) junto con los demás tipos.

Queries

Las queries o consultas son un tipo por defecto de GraphQL que se utiliza para definir las diferentes operaciones de lectura que se pueden realizar sobre la API que implementa el servidor. Por ejemplo, para el caso que nos ocupa, si queremos implementar una consulta sobre los datos que devuelva todos los usuarios que almacena el sistema, la definición del tipo Query contendría lo siguiente:

```
type Query {
  location(latlng: [Float]): Location
  locationRatings(location: LocationInput!): LocationRatings
}
```

Ilustración 20 Definición de queries GraphQL.

Mutations

Las mutaciones, de igual manera que las queries, son un tipo por defecto de GraphQL, pero que se emplea para definir las operaciones de escritura sobre el sistema. Análogamente que con las queries, si se desea implementar una operación de registro de un usuario, por ejemplo, la definición del tipo *Mutation* deberá contener lo siguiente

```
type Mutation {
    register(email: String!, password: String!, name: String!): Boolean
    login(email: String!, password: String!): LoginOutput
    addLocation(location: LocationInput!): Location
    noteForAssistance(location: LocationInput!): Location
    rateLocation(location: LocationInput!, ratings: [Float]!): StatusResponse
  }
```

Ilustración 21 Definición de mutation en Graphql

De esta manera, se le estará indicando al cliente que quiera utilizar la API que para añadir un usuario debe aportar 3 ristras de manera obligatoria (el carácter `!` lo indica), y que como resultado de esta operación recibirá un booleano que indicará si la operación se ha realizado con éxito. Además, al implementar el método que se encargará de la resolución de esta operación, el desarrollador deberá ceñirse a lo definido en el esquema.

Resolvers

Como se menciona en el apartado anterior, una vez definidas las operaciones que se pueden realizar sobre la API es necesario implementar los métodos y rutinas que se encargarán de resolver dichas peticiones. Son precisamente los *resolvers* los que se encargan de esto.

Para definirlos, en el caso de Apollo, se debe definir un objeto compuesto por dos campos (Query y Mutation), y cuya definición debe de ceñirse estrictamente a lo descrito en la definición de tipos (type Defs).

Siguiendo este protocolo, el contenido de este objeto para las queries y mutaciones descritas anteriormente sería el siguiente:

```
const resolvers = {
  Query: {
    location: async (_, { latlng }, { auth }) => {
      return await authController.requireAuth(
        auth,
        locationController.getLocation,
        latlng
      );
    },
  },
  Mutation: {
     register: async (_, { email, password, name }) => {
      return await authController.register(email, password, name);
    },
  },
};
```

Ilustración 22 Definición del objeto resolvers en JavaScript

Los métodos referenciados realizan la implementación concreta de las acciones necesarias para acceder a la base de datos, realizar la autenticación del usuario, transformar los datos al formato correcto y devolver el resultado definido por los Type Defs.

De esta manera se podrán realizar consultas escritas en Graphql solicitando un subconjunto (o todos) de los campos definidos en el esquema de location enviando el siguiente cuerpo en la petición HTTP dirigida al *path* de escucha de Apollo.

```
query ($latlng:[Float]!) {
  location(latlng: $latlng) {
    geoLocation{
      coordinates
    }
    name
    predictedAssistance
    protocolRatings {
      ratings {
        rating
      }
    }
  }
}
```

Ilustración 23 Petición de location en GraphQL

Esta petición obtendría los campos referenciados de la localización cuyas coordenadas sean más cernada a las pasadas como argumento (`$LatLng`).

Flutter: estructura y detalles de la implementación

Para seguir eficazmente la distribución de clases según lo establecido por la arquitectura hexagonal, se ha decidido implantar una organización del árbol de directorios fiel a los conceptos de dominio e infraestructura.

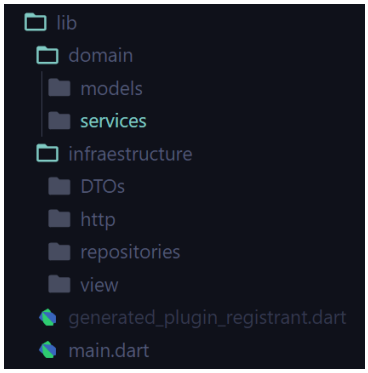

Ilustración 24 Distribución de ficheros y diretorios en Flutter

Dentro de la capa de dominio, se encuentran las implementaciones lógicas de las entidades nucleares del proyecto, y sus acciones principales. Para una localización, por ejemplo, se dispone de una definición de clase con los atributos necesarios para la representación y las operaciones pertinentes de una localización localizada dentro del directorio `models`, y una capa superior al modelo que ofrece una interfaz a las capas superiores de la arquitectura con las operaciones aplicables a una localización, identificada como servicio y localizada dentro del directorio `services.`

Por otro lado, el directorio infraestructura contiene las implementaciones concretas requeridas para el uso de cada una de las tecnologías descritas anteriormente, que son los Widgets de Flutter y la interacción con el back end mediante peticiones HTTP.

Toda la lógica de visualización e interacción con el usuario se encuentra dentro del directorio `view`, a su vez estructurada por funciones y clases. Un alto porcentaje del código que se encuentra en este directorio es configuración de vistas mediante Widgets, aprovechando en la medida de lo posible los widgets suministrados por la librería nativa de Flutter `flutter-material`.

El código encargado de realizar las peticiones HTTP formateadas correctamente según lo especificado por GraphQL y de procesar las respuestas, realizando las conversiones entre JSON (formato de las respuestas del Back End) y los objetos definidos en la capa de dominio, se localiza en los directorios `repositories` y `http.` El directorio `DTOs` contiene lógica necesaria para la traducción de los objetos de la capa de dominio a JSON (Data transfer Object).

Testeo

Para el desarrollo de la aplicación se ha seguido la metodología TDD, por lo que la elaboración de baterías de pruebas suficientemente robustas y completas ha sido crucial para el éxito del proyecto, además de para el cumplimiento de los requisitos definidos en la fase de diseño y en especial para la mantenibilidad del código.

Se han elaborado pruebas para las diferentes partes del proyecto, haciendo uso de diferentes frameworks de testeo.

Back end

Para el testeo de la implementación de la API en NodeJS se ha utilizado el framework de testeo Mocha.

Mocha

Mocha es un framework de testeo Javascript ejecutable en NodeJS y en navegador, que cuenta con numerosas funcionalidades. La combinación de este framework con la librería [supertest](https://www.npmjs.com/package/supertest) permite la ejecución de test haciendo uso del protocolo HTTP, emulando casos de uso reales del código de la aplicación.

Chai

Chai es una librería de aserciones open-source para NodeJS y el navegador, que permite la escritura de pruebas siguiendo las filosofías BDD y TDD, resultando en construcciones sintácticas cercanas al lenguaje común, como, por ejemplo:

```
expect(res.body.errors[0].message).to.equal('User already exists!');
```

Ilustración 25 Ejemplo de test hecho con Mocha Chai

La combinación de las librerías descritas anteriormente permite la ejecución de una batería de test que comprueba el comportamiento del protocolo HTTP y de las respuestas del servidor.

```
it('register users', (done) => {
  request
    .post('/graphql')
    .send({
      query:
        'mutation { register (email: "test1@gmail.com", password: "1234") }'
    })
    .expect(200)
    .end((err, res) => {
      if (err) return done(err);
      done();
    });
});
```

Ilustración 26 Ejemplo de test utilizando Mocha, Chai y Supertest

Con pruebas como esta, se pueden realizar tests *end-to-end* sobre el *back end* de la aplicación facilitando la refactorización del código y favoreciendo la robustez y estabilidad de la aplicación.

Istambul

Istanbul es un pack de herramientas de testing que permite conocer la medida en la que los tests de una aplicación abarcan y ejecutan el código de una aplicación. Permiten conocer numéricamente la cobertura que los tests ofrecen al proyecto. Una cobertura del 100% indicará que la aplicación es robusta y está probada frente a la mayoría de los casos posibles.

File	% Stmts	% Branch	% Funcs	% Lines
All files	98.56	85.27	96.43	100
src	100	76.67	100	100
resolvers.js	100	75	100	100
typeDefs.js	100	100	100	100
src/controllers	100	88.31	100	100
auth.controller.js	100	89.66	100	100
location.controller.js	100	89.29	100	100
protocolRating.controller.js	100	85	100	100
src/models	92	100	0	100
AssistanceEntries.js	100	100	100	100
Location.js	87.5	100	0	100
ProtocolRating.js	87.5	100	0	100
User.js	100	100	100	100
src/tests	100	86.36	100	100
__utils.js	100	86.36	100	100

Ilustración 27 Cobertura de la ejecución de las pruebas

La ejecución de la batería de pruebas desarrolladas para el Back End de la aplicación resultan en la cobertura total mostrada (Ilustración 27 Cobertura de la ejecución de las pruebas). Estos resultados muestran una cobertura de declaraciones (*Statements*, primera columna) del 98.56% para la aplicación, lo que significa que el 98.56% de las instrucciones del código son ejecutadas durante las pruebas de la aplicación. La cobertura mostrada para la ejecución de las ramas es mucho menor, y esto podría ser identificado como carencias en la batería de pruebas. Sin embargo, ejecutando la herramienta de cobertura de las pruebas (*istanbul*) e indicándole que muestre la salida de la ejecución en formato HTML, se puede observar claramente cuál es el problema exacto.

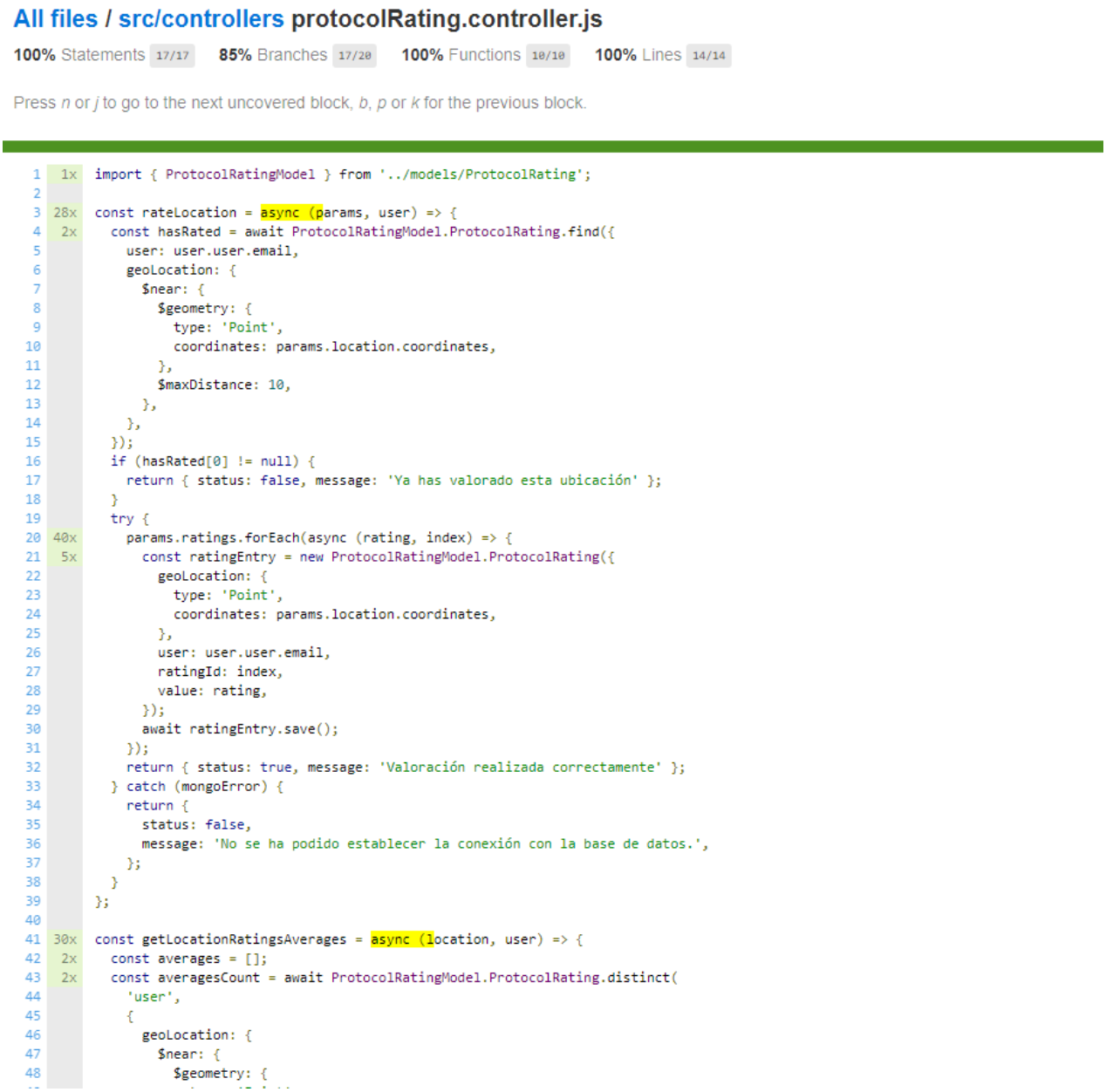

Ilustración 28 Visualización de la cobertura en formato HTML

Como se aprecia en la Ilustración 28 Visualización de la cobertura en formato HTML, los casos donde la herramienta de cobertura indica que existen ramas sin explorar (resaltado en

amarillo) son las líneas donde se indica que las funciones son asíncronas. Esto indica que la herramienta de cobertura detecta que, de las posibles opciones que existen para las promesas en JavaScript, no se exploran todas, es decir, que la batería de tests no comprueba la correcta ejecución en caso de que las promesas indicadas no se resuelvan correctamente. Si no fuera por esta peculiaridad, el resultado de la cobertura de las pruebas sería 100%.

Front End

En el front-end se han empleado la librería de *testing* suministrada por Flutter, denominada `test`.

Su sintaxis es bastante similar a la de los frameworks de testing JavaScript. Su estructura soporta la elaboración de 2 tipos de tests:

- **Tests unitarios:** es la unidad básica de testing, permite comprobar el funcionamiento de una función o una clase en concreto. En el caso concreto de la aplicación desarrollada, muchas de las funciones creadas se centran en la gestión de librerías internas de Flutter para realizar peticiones empleando el protocolo HTTP. Estas funciones son las encargadas de la comunicación con el *back-end* de la aplicación. Sin embargo, el entorno de ejecución de tests de Flutter tiene una peculiaridad que complica considerablemente el testeo de estas funciones, y es que, durante la ejecución de los tests, se deshabilitan las peticiones HTTP para favorecer la eficiencia y la velocidad en la ejecución de los tests. Es por ello, que para poder realizar tests de los métodos que manejan estas peticiones, es necesario realizar un *mockeo* de la librería interna de Flutter que realiza las peticiones HTTP.
- **Tests de Widgets**: el objetivo de los *widgets tests* es comprobar que la interfaz de usuario del widget en cuestión funciona correctamente. Esto es posible gracias a la interfaz de *testing* de widgets que proporciona Flutter. Esta interfaz ofrece al desarrollador la posibilidad de manejar el ciclo de vida de un widget, automatizando acciones de usuario sobre los diferentes elementos de un Widget, y realizando comprobaciones de su aspecto visual.

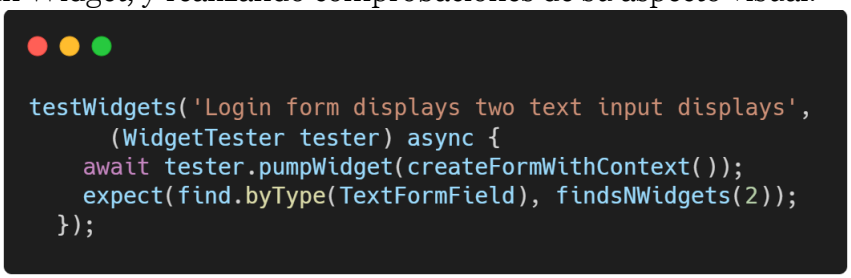

Ilustración 29 Ejemplo de test de Widget

Finalmente, la batería de tests elaborada para la parte del sistema desarrollada en Flutter cuenta con dos secciones diferenciadas: pruebas de Widgets y pruebas de páginas.
Esta diferenciación se puede realizar gracias a la estructuración de clases que se ha seguido durante el desarrollo de la interfaz de usuario, diferenciando entre los módulos de código encargados de la composición de los widgets de los de las páginas. De esta manera, se puede decir que los módulos que renderizan las páginas son composiciones de los que componen los widgets, fomentando de esta manera la reutilización de código y simplificando las pruebas de la aplicación.

Capítulo 5: Resultado

Como producto de salida de la fase de implementación del proyecto, se obtiene finalmente un sistema de información completo, que permite el registro de las asistencias de manera voluntaria y el seguimiento de las posibles aglomeraciones, así como un sistema de valoración y crítica del cumplimiento de las medidas anti-COVID tomadas en las diferentes localizaciones registradas por el sistema. Además, cabe destacar que se trata de un sistema incremental completamente autónomo, que no requiere acción alguna de administradores para su funcionamiento, es decir, los propios usuarios son los que van registrado las nuevas localizaciones y generando información sobre su asistencia y sus valoraciones.

Funcionamiento

A continuación, se muestran algunas ilustraciones del funcionamiento de la aplicación móvil y la API ejecutadas en un entorno local.

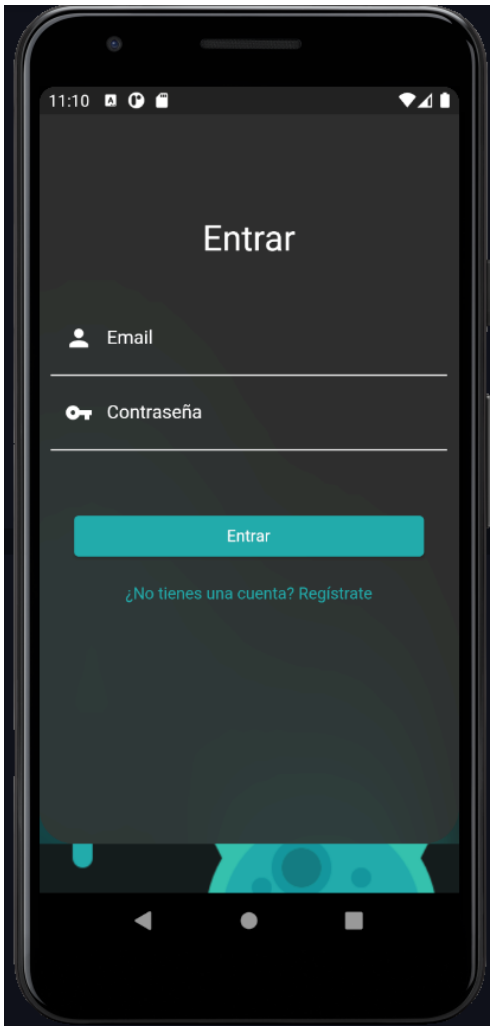

Ilustración 30 Resultado de la ejecución: Página de identificación

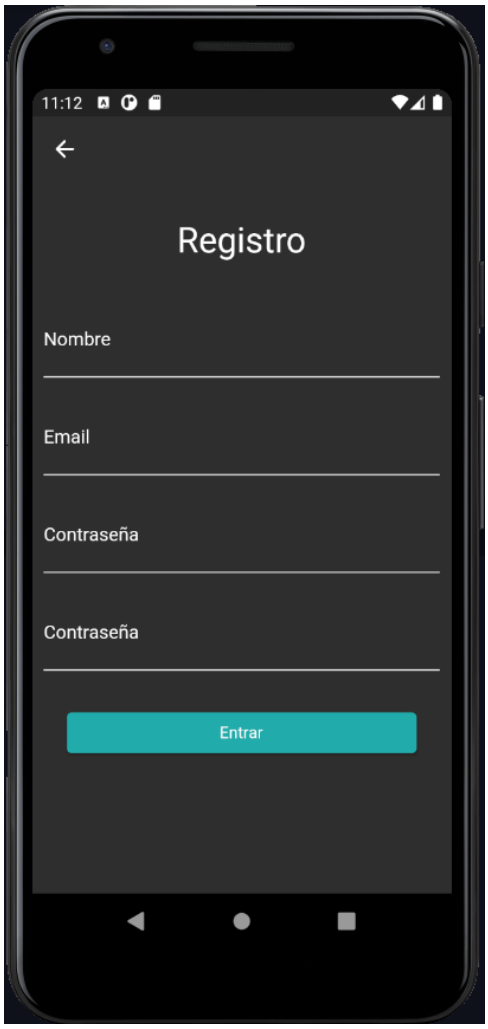

Ilustración 31 Resultado de la ejecución: Página de registro

Ilustración 32 Resultado de la ejecución: Página de navegación en el mapa

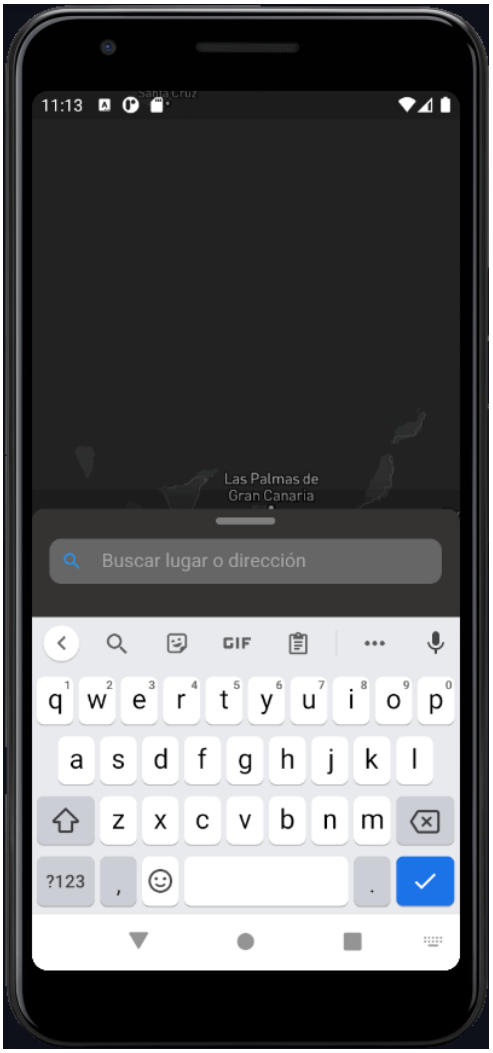

Ilustración 33 Resultado de la ejecución: Barra de búsqueda

Ilustración 34 Resultado de la ejecución: Visualización de información sobre localización

Ilustración 35 Resultado de la ejecución: Visualización de información sobre localización 2

Ilustración 36 Resultado de la ejecución página de valoración de localización

Ilustración 37 Resultado de la ejecución página de valoración de localización 2

Despliegue

Para realizar un despliegue del sistema en un entorno de producción de cara a la explotación de la aplicación móviles, son necesarias algunos ajustes y configuraciones previas, que no se han llevado a cabo para la realización de este trabajo tanto por la limitación de tiempo dedicado al desarrollo como por motivos económicos.

Implementación del protocolo HTTPS

Uno de los pasos requeridos previos al despliegue es la implementación del protocolo HTTPS para el servidor que aloja el Back end de la aplicación, lo que requeriría de la obtención de un certificado SSL firmado por una autoridad certificada. Este paso es requerido para poder pasar los requisitos de seguridad que impone Flutter para sus aplicaciones, pues las peticiones por HTTP están desactivadas por defecto. Para poder realizar un despliegue completo, es necesario que las peticiones realizadas por el dispositivo móvil se realicen mediante HTTPS.

Despliegue del Back End en un servicio de alojamiento

Además de requerir la implementación del protocolo HTTPS, la ejecución del Back End deberá de alojarse en una máquina debidamente protegida contra ataques de denegación de servicio y accesos no autorizados. Además, dicha máquina deberá de poder ser accedida por todos los dispositivos móviles que descarguen la aplicación, por lo que la opción más viable será la obtención de un dominio y la asociación de este dominio a la máquina que ejecute el software del soporte de almacenamiento del sistema. Cabe destacar que esta máquina puede ser una máquina configurada manualmente y a la que se tenga acceso físico, lo que aumentaría el tiempo de preparación del sistema, dificultaría el mantenimiento y aumentaría el coste de ejecución. Por estos motivos, se considera la opción de alojar la ejecución del Back End en una máquina mantenida por un proveedor de servicios en la nube como Amazon o Google.

Carga de la aplicación en las diferentes *Stores*

El último paso necesario para el despliegue a un entorno de producción de la aplicación es la carga del software de la aplicación móvil en los diferentes repositorios de los diferentes sistemas operativos en los que Flutter compila la aplicación. Estos son App Store para iOS y Play Store para Android.

Una vez realizada esta carga y los pasos descritos anteriormente, la aplicación podría ser distribuida y utilizada por cualquier usuario de dispositivos móviles interesados en la aplicación.

Capítulo 6: Perspectiva

Posibles mejoras

Debido a la limitación de tiempo y de velocidad de desarrollo que impone la inconveniencia de que el equipo de desarrollo esté compuesto por una sola persona, se han abarcado las funcionalidades mínimas para crear un producto mínimo viable, que sea capaz de mostrar la intención detrás del proyecto, pero no con todo el detalle que se hubiera deseado imprimir. Por este motivo, se añade este apartado en el que se indicarán las posibles adiciones al proyecto y que pueden favorecer o abarcar por completo el cumplimiento de todos los objetivos descritos durante la fase de análisis.

Sistema de registro de asistencias con fecha y hora

La implementación realizada en el sistema para realizar el seguimiento de la asistencia de los usuarios les permite indicar que van a acudir a un lugar creando una entrada en el sistema que dura 24 horas desde el momento de su creación. De esta manera, las asistencias pueden ser engañosas o no representar fielmente el objetivo que se persigue. Para corregir esto, se puede añadir al sistema actual información horaria, permitiendo a los usuarios indicar de manera más exacta la fecha y la hora a la que pretenden acudir al lugar, además de visualizar la asistencia para un día y una hora concretas. De esta manera, se lograría una representación más exacta de la asistencia a los lugares, pudiendo realizar un control mucho más preciso de las aglomeraciones.

Permitir los registros mediante el servicio de autenticación de Google

Esta característica permitiría a los usuarios registrarse automáticamente en el sistema mediante el uso de los datos de su cuenta de Google. Para ello, se añadiría un botón en la página de identificación de la aplicación para que el usuario pueda “continuar con Google”. De esta manera, se agiliza considerablemente el proceso de registro de los usuarios, contribuyendo a la mejora de la experiencia de los usuarios en la aplicación.

Creación de una aplicación web.

De igual manera que la aplicación móvil consume los datos de la base de datos mediante el Back End, se podría implementar una página web, haciendo uso de tecnologías web JavaScript para visualizar la información alojada en el sistema desde el navegador, llegando así a más usuarios.

Inclusión de un sistema de notificaciones

Con el fin de que los usuarios de la aplicación estén al tanto de la predicción de asistencia a una localización a la que hayan indicado su asistencia, se pueden añadir notificaciones mediante la implementación del protocolo de notificaciones PUSH (MDN Contributors, s.f.).

Mejora general de la experiencia de usuario de la App móvil.

Las limitaciones de tiempo y personal durante el proceso de desarrollo han reducido considerablemente la atención en el detalle en cuanto a la interfaz de usuario. Flutter ofrece diferentes herramientas para personalizar y crear una experiencia de usuario fluida y atractiva,

como animaciones, integraciones con otras aplicaciones, widgets…. Todas estas funcionalidades podrían ser incorporadas en la aplicación, haciéndola mucho más vistosa y llegando así a más usuarios.

Visión

Para finalizar, la previsión de este proyecto, como se ha mencionado previamente, es que sea adoptada por alguna administración pública o similares que estén interesados en el control y visualización de aglomeraciones, y que estén dispuestos a asumir el coste asociado al proceso de despliegue y mantenimiento de la aplicación. Además, en caso de que una administración recoja la administración y despliegue de esta aplicación beneficiaría considerablemente la operación de la aplicación, pues su funcionamiento se basa en que los propios usuarios sean proactivos en su uso.

Conclusión

El proceso de análisis e implementación de la aplicación ha evidenciado la importancia de la información en el modelo de sociedad en el que vivimos actualmente, y que además bautiza a la misma como la "sociedad de la información". Por esta misma razón, resulta trivial llegar a la conclusión de que el aprovechamiento de toda esta información puede ser infinitamente beneficioso y explotable para el desarrollo de soluciones útiles para todos. En este caso, tratándose de una situación excepcional, como lo ha sido una pandemia mundial que, como se ha mencionado anteriormente, ha propuesto nuevos e insólitos retos a las diferentes instituciones a nivel mundial, el desarrollo de respuestas tecnológicas haciendo uso de todos los recursos disponibles es una responsabilidad que recae en las instituciones y organismos que cuentan con las herramientas y recursos necesarios para crear este tipo de sistemas.

Sin embargo, como se ha demostrado durante la redacción de este documento y con la elaboración de este trabajo, con los conocimientos adecuados y los recursos suficientes, la creación de soluciones que contribuyan desinteresada y parcialmente para lograr los objetivos planteados por situaciones adversas como esta es posible.

El concepto de idear un sistema que plantee métodos eficaces de resolución de problemas globales mediante la utilización eficiente de la información surge de la concepción de la sociedad como una organización, que genera información a partir de su actividad, y que, al igual que las organizaciones, se enfrenta a retos y problemas para los que debe plantear soluciones eficaces y eficientes haciendo uso de los recursos disponibles. Como resultado de este planteamiento, se pueden proponer métodos para sobrevenir y superar dichos problemas con mayor agilidad y eficiencia. Esta idea es extrapolable a las administraciones públicas, que disponen de inmensurables cantidades de datos y que pueden ser aprovechados en virtud de los procesos que se plantean.

En esta línea, también es interesante analizar cómo muchas de las instituciones públicas y encargadas de la creación de este tipo de soluciones al servicio de la ciudadanía, no cumplen con los estándares de calidad de los procesos de desarrollo establecidos por las metodologías de desarrollo ágil modernas. Estas metodologías pueden ser adaptadas a este tipo de organizaciones para incrementar su efectividad y agilidad de respuesta frente a las adversidades.

La utilización de tecnologías punteras y novedosas como Flutter, GraphQL o Node.js para la creación de sistemas de información suponen un salto de calidad, tanto en el producto final a nivel de rendimiento y características, como en el proceso de desarrollo. El uso de lenguajes de programación y entornos de ejecución modernos facilita considerablemente el proceso de creación, prueba y ejecución de los sistemas. Es por ello que, aun teniendo en cuenta las barreras de modernización que presentan habitualmente las instituciones en sus infraestructuras tecnológicas, ya sea por regulación o por desconocimiento, que existen en las administraciones públicas se considera que la actualización de las tecnologías empleadas en los sistemas de información que suelen dar soporte a la administración electrónica y otras interfaces tecnológicas que ofrecen, podrían resultar en una mejora considerable de la calidad de los servicios, por no mencionar las numerosas ventajas que aportarían a los desarrolladores.

Con la pila de componentes implementados durante este trabajo, se ha pretendido dar soporte al control y a la reducción de aglomeraciones de cara a la gestión eficiente de la pandemia global del COVID 19. Naturalmente esta propuesta cuenta con inconvenientes y limitaciones evidentes en cuanto a la precisión y correctitud de las mediciones con las que se trabajan, por lo que lo que se pretende con este proyecto es establecer las bases y ejemplificar las infinitas posibilidades que la recolección e interpretación de datos ofrecen para la creación de soluciones eficientes y efectivas contra cualquier situación adversa que se pueda presentar.

Bibliografía

Agencia Española de Protección de Datos. (Mayo de 2020). *El uso de las tecnologías en la lucha contra el COVID19. Un análisis de costes y beneficios.* Obtenido de Agencia Española de Protección de Datos: https://www.aepd.es/sites/default/files/2020-05/analisis-tecnologias-COVID19.pdf

Apollo. (2021). *Documentación oficial sobre Apollo GraphQL.* Obtenido de https://www.apollographql.com/docs/

Facebook, Inc. (2021). *Documentación oficial de React Native.* Obtenido de https://reactnative.dev/docs/getting-started

Ferrera, A. (Diciembre de 2020). *Software Crafters*. Obtenido de Arquitectura hexagonal en el FrontEnd: https://softwarecrafters.io/react/arquitectura-hexagonal-frontend

Foundation, O. (2021). *Documentación oficial de OpenStreetMaps*. Obtenido de https://wiki.openstreetmap.org/wiki/API

Google. (2021). *Documentación oficial de Google Maps*. Obtenido de https://developers.google.com/maps/documentation

Google. (s.f.). *Documentación oficial de Flutter*. Obtenido de https://flutter.dev/docs

Ionic. (04 de 03 de 2021). *Documentación oficial de Ionic.* Obtenido de https://ionicframework.com/docs

Mapbox. (2021). *Documentación oficial de MapBox*. Obtenido de https://docs.mapbox.com/

MDN Contributors. (s.f.). *Notifications API*. Obtenido de https://developer.mozilla.org/en-US/docs/Web/API/Notifications_API

Ministerio de Sanidad del Gobierno de España. (2020). *Recomendaciones de operación y mantenimiento de los sistemas de climatización y ventilación de edificios y locales para la prevención de la propagación del SARS-Cov-2.*

Organización Mundial de la Salud. (16 de Mayo de 2020). *Cleaning and disinfection of environmental surfaces in the context of COVID-19.* Obtenido de https://www.who.int/publications/i/item/cleaning-and-disinfection-of-environmental-surfaces-inthe-context-of-covid-19

Organización Mundial de la Salud. (2020). *Global surveillance for COVID-19 caused by human infection with COVID-19 virus.*

Tecnologies, H. (2021). *Documentación oficial de HERE*. Obtenido de https://developer.here.com/

The GraphQL Foundation. (2021). *Documentación oficial de GraphQL*. Obtenido de https://graphql.org/learn/

tobrun. (Abril de 2021). *Flutter Mapbox GL repositorio oficial*. Obtenido de https://pub.dev/packages/mapbox_gl