UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

Graduado en Ingeniería del Software

EMERGY: Una plataforma para la gestión de manuales de emergencias

EMERGY: A platform for emergency manual management

Realizado por
Francisco de Paula Arjona Jiménez

Tutorizado por
Eduardo Guzmán de los Riscos

Departamento
Lenguajes y Ciencias de la Computación
UNIVERSIDAD DE MÁLAGA

MÁLAGA, mayo de 2023

UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADO EN INGENIERÍA DEL SOFTWARE

EMERGY: Una plataforma para la gestión de manuales de emergencias

EMERGY: A platform for emergency manual management

Realizado por
Francisco de Paula Arjona Jiménez

Tutorizado por
Eduardo Guzmán De los Riscos

Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA
MÁLAGA, MAYO DE 2023

Fecha defensa: mayo de 2023

Agradecimientos

Agradezco sinceramente a mi familia por su apoyo incondicional durante toda mi carrera. Desde el principio, me dieron la oportunidad de estudiar lo que quisiera, sin importar cuál fuera mi elección. A lo largo de estos años su comprensión y motivación constante han sido un pilar fundamental en mi camino académico.

También quiero expresar mi agradecimiento a Eduardo Guzmán por haber aceptado ser mi tutor. Gracias a su disposición, pude desarrollar un proyecto final acorde a mis intereses y expectativas. Siempre estuvo disponible para resolver cualquier duda o consulta que surgiera.

Resumen

Los manuales de emergencias suelen ser extensos y poco prácticos en situaciones de crisis. Por lo tanto, es crucial contar con herramientas que aborden este problema proporcionando una interfaz que brinde instrucciones claras para cada tipo de emergencia. El objetivo de este proyecto es desarrollar una plataforma que facilite la consulta de manuales de emergencia. Además, esta plataforma también incorpora la funcionalidad de permitir a los usuarios reportar incidentes a los servicios de emergencia de manera rápida y eficiente, basándose en el estándar de interoperabilidad JESIP del Reino Unido, el cual enfatiza la importancia de colocar a las personas en el centro de los incidentes.

Para conseguir esto, se ha desarrollado una aplicación móvil con interfaz simple e intuitiva para el usuario. A nivel tecnológico se ha hecho uso de dos *frameworks* de javascript, React Native para el *frontend*, y Express.js para el *backend*.

Emergy ofrece una solución integral para la gestión de emergencias, proporcionando acceso rápido y sencillo a manuales de emergencia, así como la capacidad de reportar incidentes. Los usuarios de Emergy pueden crear perfiles de usuario, almacenar contactos de emergencia y utilizar características como llamadas y envío de mensajes de texto. Esta plataforma móvil se centra en dar una respuesta rápida y efectiva antes situaciones críticas.

En resumen, Emergy es una plataforma móvil que aborda las limitaciones de los manuales de emergencia convencionales, proporcionando instrucciones claras y dando la opción de reportar dichas situaciones.

Palabras clave: Emergencias, JESIP, manuales de emergencias, React Native, Express.js.

Abstract

Emergency manuals are typically extensive and impractical during crisis scenarios. Therefore, it is essential to have tools that tackle this issue by offering a user interface with clear instructions for various types of emergencies. The main objective of this project is to create a user-friendly platform that enables easy access to emergency manuals. Additionally, the platform incorporates a feature that allows users to swiftly and efficiently report incidents to emergency services, following the JESIP interoperability standard from United Kingdom, which emphasizes prioritizing individuals in emergency situations.

To accomplish this, a mobile applications has been developed with a straightforward and intuitive interface. Technologically, the Project utilizes two JavaScript frameworks: React Native for the frontend and Express.js for the backend.

Emergy presents a comprehensive solution for emergency management, facilitating quick and effortless access to emergency manuals, along with the capability to report incidents. Users of Emergy can create personal profiles, store emergency contacts, and utilize functionalities such as calls and text messaging. The mobile platform is designed to deliver prompt and effective responses to critical situations.

In summary, Emergy is a mobile platform that addresses the limitations of traditional emergency manuals by providing clear instructions and empowering users to report incidents promptly.

Keywords: Emergency, JESIP, Manual, Native, Report.

Índice

1

Introducción

1.1 Motivación

La motivación detrás de este proyecto surge de la necesidad de mejorar la gestión de emergencias y reducir los impactos de los accidentes o situaciones de crisis en la sociedad. Los accidentes y eventos imprevistos pueden tener consecuencias devastadoras, afectando la seguridad, la salud y el bienestar de las personas, así como el entorno y la infraestructura.

La falta de acceso rápido y práctico a información relevante durante situaciones de emergencia puede dificultar la toma de decisiones acertadas y retrasar la respuesta efectiva. Los manuales de emergencia tradicionales, debido a su extensión y complejidad, a menudo no son útiles en momentos críticos.

Por lo tanto, este proyecto se enfoca en desarrollar una plataforma que aborde estas limitaciones, proporcionando una interfaz intuitiva y accesible que brinde instrucciones claras y específicas para cada tipo de emergencia. Esta plataforma también incorpora la capacidad de reportar incidentes de manera rápida y eficiente, lo que permite a las autoridades conocer los detalles del incidente y dar una respuesta más ágil a estos incidentes.

Además, la motivación para desarrollar este proyecto se ve reforzada por los avances tecnológicos y las oportunidades que brindan las aplicaciones móviles. En la era digital actual, las aplicaciones móviles se han convertido en herramientas indispensables para acceder rápidamente a información y servicios vitales. Las aplicaciones móviles permiten mayor movilidad y accesibilidad, lo que significa que los usuarios pueden tener los pasos de actuación ante una emergencia determinada y las herramientas de reporte al alcance de sus manos en todo momento.

1.2 Objetivos

El desarrollo de este proyecto de fin de grado tiene como objetivo principal la elaboración de una aplicación móvil para que los usuarios se sientan seguros y preparados frente a situaciones de emergencia. Se trata de empoderar a los usuarios, brindándoles herramientas prácticas y confiables para poder tomar acciones rápidas y adecuadas a cada contexto.

Para lograr ese objetivo final, se establecen los siguientes subobjetivos:

- En primer lugar, se realizará un estudio de las aplicaciones con un carácter similar, para hacer un análisis de cuáles son las necesidades imprescindibles que debería tener la aplicación.
- Investigación de cuáles son los manuales de emergencia fundamentales y de dónde se pueden obtener para su posterior uso, así como cuál es la mejor forma de mostrárselo al usuario de una forma resumida y útil.
- Estudio de uso y almacenamiento de datos más adecuado para cada caso de uso, como pueden ser almacenamiento de datos de usuarios, los manuales de emergencia o los incidentes reportados.
- Desarrollo e implementación del modelo M/ETHANE. Este es un marco de informes establecido que proporciona una estructura común para que los usuarios y las salas de control compartan información sobre incidentes. Se recomienda su uso para todos los incidentes e ir actualizándolo a medida que se desarrolla el incidente. Para los incidentes que no alcanzan el umbral de incidente importante, este se convierte en ETHANE.
- Y para finalizar, el desarrollo de las funcionalidades clave para que esta aplicación tenga sentido:

- Consultar manuales de emergencia: Proporcionar al usuario una lista de manuales de emergencia para su visualización y uso.
- Reportar un incidente: Permitir que el usuario pueda enviar a las autoridades la información de cualquier incidente siguiendo el modelo METHANE, que cuenta con datos como localización, posibles peligros o daños producidos, entre otros.
- Implementación de un registro de usuarios: Un sistema de creación de cuentas y almacenamiento de datos personales, para su uso en caso de emergencia. Obviamente cumpliendo unos requisitos de seguridad.
- Historial de incidentes y lista de contactos con parámetros de búsqueda: Cada usuario registrado contará con un registro de todos los incidentes que haya reportado

a lo largo del tiempo, así como una lista de contactos de emergencia que haya añadido para poder llamar y enviar mensajes de texto. Hay que añadir que podrá consultar ambos contando con una barra de búsqueda.

1.3. Metodologías utilizadas en el desarrollo

La programación extrema (*Extreme Programming* o *XP Programming*) es una metodología de desarrollo que forma parte de las conocidas metodologías ágiles. Su objetivo principal es gestionar y desarrollar proyectos de manera eficaz, flexible y controlada. Aunque esté estrechamente relacionada, es importante destacar que XP y Agile son conceptos diferentes. XP es una metodología que se centra en prácticas concretas dentro del marco de trabajo Agile. Fue la metodología dominante en el mundo ágil en los años 2000, hasta que apareció *Scrum*.

Scrum es un marco de trabajo ágil que se basa en un enfoque colaborativo y flexible, donde los equipos trabajan en iteraciones cortas y regulares llamadas 'sprints'. Se caracteriza por la autogestión del equipo y adaptabilidad a medida que se van obteniendo resultados.

Al tratarse de un proyecto unipersonal, la combinación de estas dos metodologías es perfecta, pues pone más énfasis en la adaptabilidad que en la previsibilidad, permitiendo mucha más flexibilidad para el desarrollo.

1.4 Estructura de la Memoria

Esta memoria se ha dividido en diferentes capítulos para facilitar su lectura y comprensión. A continuación, se exponen cuáles son estos capítulos y que se trata en cada uno de ellos:

1. **Introducción**

   En este primer capítulo se presenta una visión general del proyecto, explicando su contexto y la motivación que lo impulsó, así como los objetivos que se pretenden alcanzar.

2. **Estudio del arte**

   En este capítulo se revisan los conceptos y teorías relevantes relacionados con la gestión de emergencias, los manuales de emergencia, las aplicaciones móviles y los avances tecnológicos en este campo.

3. **Tecnologías y Herramientas**

   En este capítulo se exponen cuáles son las tecnologías y herramientas más apropiadas para el proyecto.

4. **Análisis y Diseño del almacenamiento**

   En este capítulo se aborda la gestión y organización de los datos en el proyecto, viendo así cual es la opción más robusta, escalable y segura.

5. **Especificación y Análisis**

   En este capítulo se enumerarán y explicarán todos los requisitos funcionales y no funcionales que cumple el proyecto.

6. **Implementación y Desarrollo**

   En este capítulo se entra en detalle de como se ha implementado y desarrollado todo el código, tanto en el *backend* como en el *frontend*.

7. **Conclusión y Trabajos Futuros**

   En este último capítulo se tratan las conclusiones obtenidas a lo largo del proyecto, se discuten las limitaciones encontradas y se proponen posibles mejoras y líneas de trabajo futuro.

2

Estudio del arte

2.1 Análisis de la tecnología actual

A continuación, se llevará a cabo una investigación sobre las diferentes aplicaciones actualmente disponibles para el público, que cumplen o podrían cumplir una función similar a la que se busca lograr con este proyecto:

2.1.1 Aplicaciones similares

- **My112**

  Esta aplicación permite, de una forma muy simple, llamar al 112 y mandar automáticamente la localización en caso de emergencia. También se puede recibir avisos en tiempo real de incidencias cercanas. Cuenta con una respuesta bastante rápida, y una interfaz accesible y fácil, ya que está especialmente pensada para gente mayor. Como puntos negativos presenta que es necesaria una configuración inicial para que funcione o bien que pide acceso a algunos ficheros del móvil que no son necesarios en el momento de usarla.

- **AlertCops**

  Aplicación oficial de las Fuerzas y Cuerpos de Seguridad del Estado. Con la asistencia de esta aplicación, los usuarios tienen la capacidad de comunicarse directamente con las comisarías más cercanas. Proporciona la posibilidad de alertar sobre emergencias en las proximidades y es totalmente gratuita. Sin embargo, se requieren configuraciones adicionales para asegurar que la ubicación se actualice correctamente.

- **JESIP App**

  Se trata de la aplicación en la que se ha inspirado el proyecto, pues usa el modelo M/ETHANE para reportar incidentes. Sin embargo, la aplicación no funciona correctamente en dispositivos móviles y tampoco presenta la funcionalidad de consultar manuales de emergencias.

- **Ariadna**

  Aunque no tiene las mismas funcionalidades que las que se buscan en este proyecto, presenta funcionalidades muy interesantes que podrían desarrollarse en el futuro. Es una aplicación desarrollada por la Asociación Española de Cardiología que permite localizar desfibriladores cerca del usuario.
- **Apps predeterminadas**

  Si no se desea instalar ninguna aplicación, los dispositivos móviles o tablets ya incluyen algunas aplicaciones que pueden resultar útiles en caso de emergencia. En Android, por ejemplo, se pueden añadir contactos de emergencia e información médica, como grupo sanguíneo, alergias, etc. Todos estos datos serán accesibles sin tener que desbloquear el móvil. En IOS, existe la aplicación **Emergencia SOS**, que permite al usuario pedir ayuda de forma rápida y sencilla y alertar a contactos de emergencia.

2.2 Manuales de Emergencia actuales

Los manuales de emergencia tradicionales con frecuencia pasan desapercibidos debido a su extensión y falta de practicidad. Estos documentos de cientos de páginas resultan abrumadores y difíciles de consultar en situaciones de crisis. La información vital se diluye entre las páginas y se vuelve muy complicado encontrar instrucciones claras y concisas para cada tipo de emergencia. Como resultado, la atención y utilidad de estos manuales se ven limitadas, lo que puede impactar negativamente en la capacidad de respuesta y gestión eficiente de emergencias.

2.3 Conclusiones

Tras investigar sobre los manuales de emergencias que hay en la web y sobre aplicaciones similares, su uso y funcionalidades, se puede afirmar que una aplicación sencilla puede desempeñar un papel crucial en situaciones de emergencia, brindando respuestas más rápidas y enviando información más detallada en comparación con una llamada al servicio de emergencias tradicional. Un diseño intuitivo y fácil de usar la hace accesible para las personas mayores, eliminando cualquier dificultad en su manejo. Este tipo de aplicaciones demuestran ser herramientas prácticas y efectivas. Con el continuo avance tecnológico, es probable que se vea un mayor desarrollo y adopción de estas aplicaciones en el futuro, fortaleciendo aún más nuestra capacidad de respuesta ante situaciones críticas.

3

Tecnologías y herramientas

3.1 Introducción

Ya realizado el estudio inicial de lo que el proyecto exige, se van a desglosar las tecnologías y herramientas que se han usado y por qué, así como la arquitectura del mismo.

3.2 Tecnologías

En el desarrollo de este proyecto se ha utilizado los siguientes lenguajes de programación y tecnologías:

- **JavaScript**

  Es un lenguaje de programación ampliamente utilizado y estandarizado en el ámbito de las aplicaciones web. Permite a los desarrolladores controlar y programar la interfaz del usuario en respuesta a diversas interacciones y eventos lanzados por el usuario.

- **Express.js**

  Es un framework muy popular utilizado para crear APIs de manera rápida y sencilla. Express simplifica el manejo de las solicitudes HTTP entrantes y las respuestas salientes, permitiendo definir fácilmente cómo debe responder la aplicación a diferentes rutas. También ofrece un sistema de middleware flexible que permite agregar funcionalidades adicionales, como autenticación, manejo de errores y registro de solicitudes.

- **React Native**

  Es un framework de desarrollo de aplicaciones móviles que permite construir aplicaciones nativas para IOS y Android utilizando JavaScript y la biblioteca de componentes de React. La gran ventaja que presenta es que permite compartir la mayor parte del código entre las plataformas, lo que resulta en un desarrollo mucho más eficiente y rápido.

3.3 Herramientas

En este proyecto se han empleado las siguientes herramientas:

- **Visual Studio Code**

  VS Code es un editor de código muy popular desarrollado por Microsoft. Está diseñado para ser liviano, personalizable y altamente extensible, y proporciona una amplia gama de características y funcionalidades.

  Una de las extensiones usadas en el proyecto es **ThunderClient**, que sirve para probar y depurar APIs. Proporciona una interfaz fácil de usar para enviar solicitudes HTTP a una API y ver las respuestas correspondientes.

- **Expo**

  Es una plataforma que junto con React Native nos permite desarrollar aplicaciones para dispositivos móviles. Algunas de las funcionalidades que ofrece:

  - Desarrollo multiplataforma.
  - Live Reload y Hot Reloading. Proporciona ver los cambios realizados en la aplicación en tiempo real.
  - Acceso a APIs nativas.
  - Comunidad activa y buena documentación.

  A estos puntos hay que añadir que gracias a su aplicación móvil **Expo Go**, permite simplemente con escanear un código QR ver la aplicación en tiempo real en el dispositivo del usuario, sin necesidad de usar emuladores.

- **GitHub**

  Sitio web que usa Git, un sistema de control de versiones, para facilitar la creación, compartición y seguimiento de proyectos software, ya sea de forma individual o colaborativa.

  En este proyecto, se aprovechará GitHub para tener una copia de seguridad de la aplicación y para mantener un control de versiones.

- **MySQL**

  Es una plataforma de gestión de bases de datos relacional de código abierto muy popular. Permite almacenar, organizar y recuperar datos de manera segura y confiable.

- **AWS Cognito**

  Es un sercivio de Amazon Web Services que ofrece características de autenticación y autirzación para aplicaciones. Proporciona una capa de seguridad que garantiza la protección de los datos y el control de acceso a las aplicaciones.

3.4 Librerías

Librerías que se han añadido al proyecto:

- **React Navigation**

  Es una biblioteca muy popular en React Native. Su función principal es proporcionar una forma fácil y eficiente de gestionar la navegación entre pantallas dentro de una aplicación. Algunas de las características que ofrece son:

  - Navegación entre pantallas. Permite al usuario moverse entre diferentes pantallas de manera intuitiva.
  - Diferentes tipos de navegación. Ofrece varios tipos de navegación, como por ejemplo *stack* (flujo lineal entre pantallas), *tab navigation* (navegación de pestañas) o *drawer* (muestra un menú deslizable).
  - Gestión del historial de navegación. Mantiene un historial de navegación para facilitar el seguimiento de las pantallas visitadas.
  - Navegación anidada y enrutamiento dinámico. Esto significa que se puede usar una navegación dentro de otra navegación, por ejemplo, podemos usar un *stack* dentro de una pantalla del *tab navigator*.
  - Personalización y transiciones animadas.

- **Redux Toolkit**

  Es una biblioteca oficial de Redux que ofrece unas funcionalidades similares, pero con una implementación mucho más sencilla. Esta biblioteca proporciona exactamente una solución centralizada para gestionar el estado de una aplicación, permitiendo así tener un estado global accesible desde todas los componentes y pantallas de la aplicación. Algunas de las características que presenta son:

  - Configuración simplificada. Proporciona una función que simplifica el store de Redux y reduciendo el código necesario a escribir.
  - Sintaxis reducida.
  - Permite definir “porciones”. Pequeños estados donde cada uno representa un objeto diferente.

- Proporciona una función que automáticamente genera los *actions* y *reducers* correspondientes, disminuyendo la cantidad de código repetitivo.

- **AWS Amplify**

  Es una biblioteca y conjunto de herramientas que simplifica la creación de aplicaciones escalables y seguras. Mediante la CLI (Command Line Interface) simplifica la configuración inicial y el despliegue de la infraestructura necesaria de la aplicación.

- **React Hook Form**

  Es una biblioteca de gestión de formularios que utiliza los hooks de React y que proporciona un enfoque sencillo y eficiente. Otra de sus características es que permite la validación de los campos de los formularios.

- **Expo Location**

  Es una biblioteca proporcionada por Expo que ofrece funcionalidades para acceder a la ubicación del dispositivo.

- **MySQL2**

  Es un controlador de base de datos para Node.js que permite interactuar con bases de datos MySQL. Ofrece:

  - Conexiones rápidas y eficientes
  - Compatibilidad con promesas, permitiendo el uso de funciones asíncronas
  - Manejo de múltiples consultas en una sola llamada
  - Soporte para pool de conexiones

- **Nodemon**

  Es una herramienta de desarrollo que ayuda a mejorar el flujo de trabajo durante el desarrollo de aplicaciones. Su función principal es reiniciar automáticamente la aplicación cuando se detectan cambios en los archivos, permitiendo una actualización instantánea sin necesidad de reiniciar el servidor.

4

Análisis y diseño del almacenamiento

4.1 Análisis

En este capítulo se analizarán los requisitos de almacenamiento de la aplicación y se tomarán decisiones sobre las mejores opciones para implementar un sistema eficiente y seguro. Se realizará un diseño detallado del esquema de almacenamiento, teniendo en cuenta aspectos como la estructura de la base de datos, la selección del sistema de gestión de bases de datos y las estrategias de seguridad y respaldo de datos. El objetivo principal es garantizar un almacenamiento óptimo que cumpla con las necesidades de la aplicación.

4.2 Servicios de bases de datos

Para este proyecto se usan dos servicios de almacenamiento de datos complementarios:

4.2.1 MySQL

Es un sistema de gestión de bases de datos relacional muy utilizado, que ofrece una estructura robusta y confiable para almacenar y administrar datos. Destaca por su escalabilidad y rendimiento, convirtiéndolo en una opción popular para aplicaciones web.

Una de las principales fortalezas que posee es su motor de almacenamiento, que permite realizar consultas rápidas y complejas, siendo especialmente importante en entornos donde la velocidad de acceso a los datos es crucial, cosa que es esencial para el proyecto del que se trata en esta memoria. Otro aspecto que destacar es su amplia compatibilidad con diversos lenguajes y plataformas, lo que facilita su integración en diferentes entornos de desarrollo.

Otro aspecto importante por considerar es la compatibilidad con archivos JSON, lo cual simplifica la gestión de los manuales de emergencia tanto para su almacenamiento como para su recuperación a través de la API y su presentación al usuario. Esto permite una integración fluida entre la base de datos y la aplicación, facilitando el acceso rápido y eficiente a la información.

4.2.2 AWS Cognito

A pesar de todos los puntos favorables que se han presentado en el punto anterior, MySQL tiene algunas limitaciones de seguridad, en especial si no se configura adecuadamente. Es por ello por lo que se ha optado por usar también AWS Cognito.

Amazon Cognito es una solución integral para la autenticación, autorización y gestión de usuarios en aplicaciones móviles y web. Ofrece una integración sencilla y una fácil configuración, lo que permite implementar rápidamente funciones de inicio de sesión seguras y personalizadas. Además, proporciona flexibilidad al usuario pudiendo iniciar sesión también a través de terceros como Facebook, Google o Apple.

Su facilidad de uso se refleja en su interfaz intuitiva y su documentación detallada, lo que facilita el proceso de desarrollo y reduce el tiempo de implementación. También ofrece opciones avanzadas, como la administración de perfiles, confirmación de registro mediante códigos enviados al correo o móvil del usuario, recuperación de contraseñas y posibilidad de personalización de flujos de inicio de sesión.

4.3 Diseño

A continuación, se describen los diseños utilizados en ambos sistemas más a fondo:

4.3.1 Diseño en MySQL

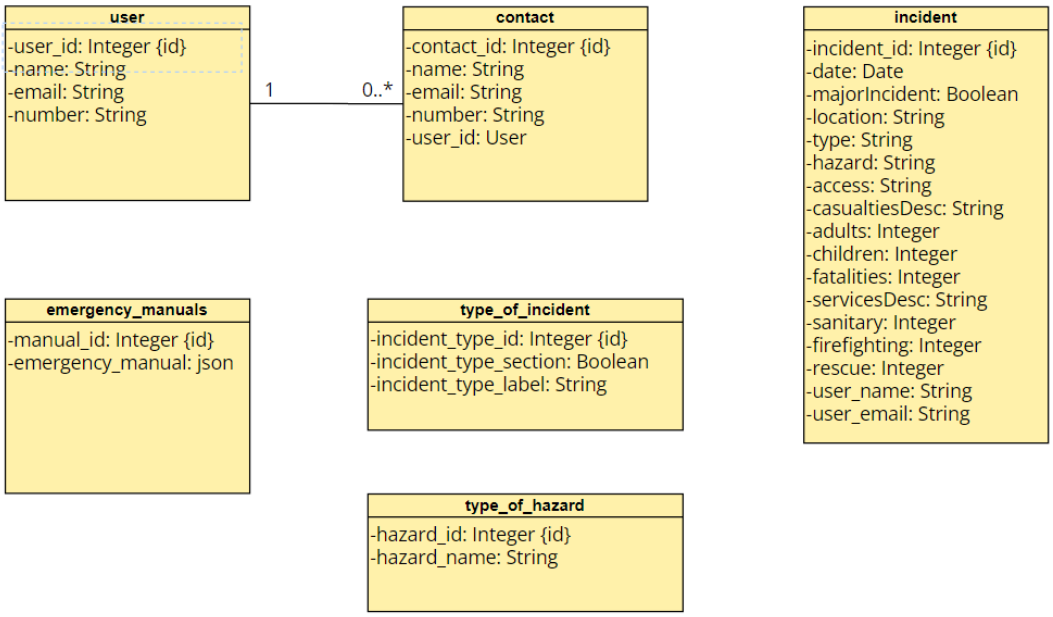


Ilustración 1 *Diagrama UML de emergydb (MySQL)*

Como se describe en la Ilustración 1, la base de datos consta de 6 tablas:

- **User**

  Esta tabla describe al usuario con sus campos e identificador único. Aunque la lógica de las cuentas de usuario esté integrada con Amazon Cognito, en cada registro se guardan los datos del usuario para poder asociarlo a sus contactos o a sus incidentes reportados.

- **Contact**

  Se describe el contacto añadido por el usuario. Cuenta con sus propios campos e identificador único, así como con un identificador del usuario que le añadió. Con esta relación asociamos a un usuario con cero o más contactos.

- **Emergency_Manuals**

  Se describe un objeto json con un identificador único, permitiendo cargar así de una forma más cómoda los manuales completos, con sus textos, imágenes, etc.

- **Incident**

  Se describe el incidente reportado por el usuario. Si este tiene la sesión iniciada en el momento del reporte, se enviarán juntos los datos proporcionados sobre el accidente, los datos del usuario (nombre y email). Si por lo contrario tiene la sesión cerrada, se enviará como anónimo.

- **Type_of_incident**

  Esta tabla se encarga de almacenar los valores que se presentan en un elemento *select* dentro del formulario destinado a reportar un incidente, exactamente el campo: tipo de incidente.

- **Type_of_hazard**

  Esta tabla se encarga de almacenar los valores que se presentan en un elemento *select* dentro del formulario destinado a reportar un incidente, exactamente el campo: posibles peligros que puede ocasionar el incidente.

4.3.2 Diseño en AWS Cognito

La característica distintiva de Cognito es su enfoque en el control de acceso mediante autenticación y autorización. Esto implica la verificación de dos elementos esenciales: la identidad de la persona que intenta acceder al sitio web y si dicha persona tiene los permisos adecuados para acceder. Ambos aspectos son cruciales para garantizar la seguridad de los usuarios.

De hecho, se considera como una solución de Administración de Identidad y Acceso del Cliente: CIAM (por sus siglas en inglés).

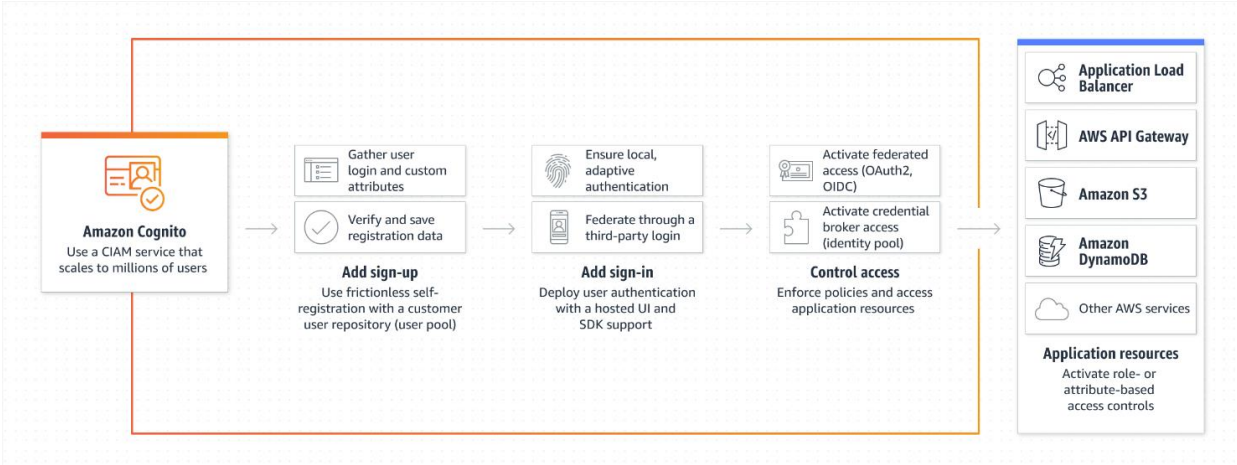


Ilustración 2 *Pasos que sigue Amazon Cognito*

4.4 Formato de los manuales de emergencia

A continuación, se describe de dónde se obtienen los manuales de emergencia y cómo se almacenan en la base de datos para luego obtenerlos en el *frontend* y mostrarlos de la forma más eficiente y sencilla posible.

4.4.1 Fuente

Tras muchas horas de investigación y revisión de páginas que ofrecían manuales de emergencia, se decidió por usar la página oficial del gobierno de los Estados Unidos dedicada a la preparación y respuesta ante emergencias y desastres. Esta proporciona información valiosa sobre cómo estar preparada para diferentes situaciones, como desastres naturales, emergencias médicas y eventos terroristas. Ofrece recursos, guías y consejos prácticos para que los individuos, las familias y las comunidades puedan planificar y tomar medidas para mantenerse seguros y protegidos en caso de una crisis. Se adjunta dirección del sitio web en la bibliografía.

4.4.2 Formato de almacenamiento

La página mencionada en el apartado anterior no dispone de una API para obtener manuales directamente, sino que simplemente proporciona información y recursos en formato de texto y documentos descargables. Para obtener manuales y recursos específicos, se recomienda visitar dicha página y descargar los documentos relevantes de manera manual.

Tras esto, se decidió por crear una estructura propia que permitiera almacenar y gestionar los manuales de emergencia con comodidad y rapidez.

Dicha estructura es un archivo de datos *json*:

Ilustración 3 *JSON manuales de emergencia*

Consta de:

- **manual_title.** Este es el título del manual de emergencias
- **manual_content.** En este array de datos se almacenan todos los textos e imágenes que contiene el manual. Se divide en secciones, y cada sección a su vez se divide en:
    - **section_id.** Identificador único de la sección de un manual.
    - **section_image.** Enlace a imagen de la sección.
    - **section_title.** Título de la sección.
    - **section_description.** Descripción de la sección.
    - **section_elements.** Elementos adicionales que contiene una sección:
        - **element_id.** Identificador único del elemento de una sección.
        - **text.** Texto del elemento.
        - **type.** Tipo de elemento, que puede ser un título, un paso a realizar, o un punto a tener en cuenta.
        - **textBold.** Texto del elemento en negrita, si lo hubiera.
        - **steps.** Número de pasos que pertenecen al título de una subsección.

5

Especificación y análisis

5.1 Requisitos funcionales

De acuerdo con las funcionalidades mencionadas anteriormente, en este apartado se abordarán los requisitos funcionales necesarios para el desarrollo de la aplicación:

- RF1. El usuario podrá crear un usuario.
- RF2. El usuario podrá iniciar sesión.
- RF3. El usuario podrá cerrar sesión.
- RF4. El usuario podrá añadir contactos.
    - RF4.1. El usuario podrá llamar a sus contactos.
    - RF5.1. El usuario podrá enviar mensajes de texto a sus contactos.
- RF5. El usuario podrá buscar contactos por su nombre.
- RF6. El usuario podrá consultar manuales de emergencia.
- RF7. El usuario podrá reportar incidentes.
    - RF7.1. El usuario podrá reportar incidentes con sesión iniciada
    - RF7.2. El usuario podrá reportar incidentes sin sesión iniciada
- RF8. El usuario podrá ver sus incidentes reportados.
- RF9. El usuario podrá buscar incidentes por su localización.

5.2 Requisitos no funcionales

A continuación, los requisitos no funcionales:

- RNF1. El sistema deberá usar protocolos de seguridad necesarios para el control de cuentas, tales como OAuth 2.0, SAML (Security Assertion Markup Language) y JWT (JSON Web Tokens).
- RNF2. El sistema deberá simplemente almacenar los incidentes reportados, ya que al tratarse de un proyecto puramente académico no se van a alertar a las autoridades como en una situación real.
- RNF3. El sistema deberá funcionar en IOS, Android y web.

5.3 Casos de uso

Nombre	Descripción
Caso de uso	Crear un usuario
Actores	Usuario estándar
Descripción	El usuario podrá registrarse en la aplicación
Precondiciones	El usuario se encuentra en la pantalla de 'Cuenta' y nunca se ha registrado
Escenario Principal	1. El usuario pulsa sobre el apartado de 'Cuenta' en la barra de navegación. 2. El usuario pulsa sobre el botón de 'Obtener cuenta' 3. El usuario introduce sus datos 4. El usuario introduce el código de confirmación recibido
Escenario Alternativo	4.b. El usuario cierra la página de confirmación

Tabla 1 *Caso de uso: Registro en la aplicación*

Nombre	Descripción
Caso de uso	Iniciar sesión
Actores	Usuario estándar
Descripción	El usuario podrá iniciar sesión en la aplicación
Precondiciones	El usuario no ha iniciado sesión
Escenario Principal	1. El usuario pulsa sobre el apartado de 'Cuenta' en la barra de navegación. 2. El usuario introduce sus datos 3. El usuario pulsa sobre el botón de 'Iniciar Sesión
Escenario Alternativo	2.b. El usuario no introduce sus datos

Tabla 2 *Caso de uso: Inicio de Sesión en la aplicación*

Nombre	Descripción
Caso de uso	Cerrar sesión
Actores	Usuario estándar
Descripción	El usuario podrá cerrar sesión en la aplicación
Precondiciones	El usuario ha iniciado sesión
Escenario Principal	1. El usuario pulsa sobre el apartado de 'Cuenta' en la barra de navegación. 2. El usuario pulsa sobre el botón de 'Cerrar Sesión
Escenario Alternativo	

Tabla 3 *Caso de uso: Cierre de sesión en la aplicación*

Nombre	Descripción
Caso de uso	Ver contactos
Actores	Usuario estándar
Descripción	El usuario podrá ver sus contactos
Precondiciones	El usuario ha iniciado sesión
Escenario Principal	1. El usuario pulsa sobre el apartado de ‘Contactos’ en la barra de navegación.
Escenario Alternativo	

Tabla 4 *Caso de uso: Ver lista de contactos*

Nombre	Descripción
Caso de uso	Añadir contacto
Actores	Usuario estándar
Descripción	El usuario podrá añadir contactos
Precondiciones	El usuario ha iniciado sesión
Escenario Principal	1. El usuario pulsa sobre el apartado de ‘Contactos’ en la barra de navegación. 2. El usuario pulsa el ‘+’ 3. El usuario añade los datos del contacto 4. El usuario pulsa sobre el botón de ‘Añadir’
Escenario Alternativo	3.b. El usuario sale de la página

Tabla 5 *Caso de uso: Añadir Contacto*

Nombre	Descripción
Caso de uso	Enviar SMS a un contacto
Actores	Usuario estándar
Descripción	El usuario podrá enviar mensaje de texto a un contacto
Precondiciones	El usuario ha iniciado sesión
Escenario Principal	1. El usuario pulsa sobre el apartado de ‘Contactos’ en la barra de navegación. 2. El usuario pulsa sobre el contacto 3. El usuario pulsa sobre el botón de ‘SMS’
Escenario Alternativo	3.b. El usuario cierra la página de la aplicación de mensajería

Tabla 6 *Caso de uso: Envío SMS a Contacto*

Nombre	Descripción
Caso de uso	Llamar a contacto
Actores	Usuario estándar
Descripción	El usuario podrá enviar llamar a un contacto
Precondiciones	El usuario ha iniciado sesión
Escenario Principal	1. El usuario pulsa sobre el apartado de 'Contactos' en la barra de navegación. 2. El usuario pulsa sobre el contacto 3. El usuario pulsa sobre el botón de 'Llamar'
Escenario Alternativo	3.b. El usuario cierra la página de teléfono

Tabla 7 *Caso de uso: Llamada a Contacto*

Nombre	Descripción
Caso de uso	Buscar un contacto
Actores	Usuario estándar
Descripción	El usuario podrá buscar a un contacto
Precondiciones	El usuario ha iniciado sesión
Escenario Principal	1. El usuario pulsa sobre el apartado de 'Contactos' en la barra de navegación. 2. El usuario pulsa sobre la barra de búsqueda 3. El usuario escribirá el nombre
Escenario Alternativo	2.b. El usuario cancela la búsqueda

Tabla 8 *Caso de uso: Buscar Contacto por nombre*

Nombre	Descripción
Caso de uso	Consultar manual de emergencia
Actores	Usuario estándar
Descripción	El usuario podrá consultar manuales de emergencia
Precondiciones	El usuario se encuentra en 'Home'
Escenario Principal	1. El usuario pulsa sobre el apartado de 'Home' en la barra de navegación si no está ahí 2. El usuario visualiza la lista de manuales 3. El usuario pulsa sobre el manual que le interese
Escenario Alternativo	

Tabla 9 *Caso de uso: Consultar Manual de Emergencias*

Nombre	Descripción
Caso de uso	Reportar incidente en anónimo
Actores	Usuario estándar
Descripción	El usuario podrá reportar incidentes
Precondiciones	El usuario no ha iniciado sesión
Escenario Principal	1. El usuario pulsa sobre el apartado de 'Home' en la barra de navegación si no está ahí 2. El usuario pulsa en '+' 3. El usuario rellena los campos que vea convenientes 4. El usuario le da a 'OK' y se enviará un incidente con usuario: anónimo
Escenario Alternativo	

Tabla 10 *Caso de uso: Reportar incidente como anónimo*

Nombre	Descripción
Caso de uso	Reportar incidente
Actores	Usuario estándar
Descripción	El usuario podrá reportar incidentes
Precondiciones	El usuario ha iniciado sesión
Escenario Principal	1. El usuario pulsa sobre el apartado de 'Home' en la barra de navegación si no está ahí 2. El usuario pulsa en '+' 3. El usuario rellena los campos que vea convenientes 4. El usuario le da a 'OK' y se enviará un incidente con los datos del usuario automáticamente

Tabla 11 *Caso de uso: Reportar incidente como usuario registrado*

Nombre	Descripción
Caso de uso	Ver incidentes
Actores	Usuario estándar
Descripción	El usuario podrá ver sus incidentes reportados
Precondiciones	El usuario ha iniciado sesión
Escenario Principal	1. El usuario pulsa sobre el apartado de 'Incidentes en la barra de navegación.
Escenario Alternativo	

Tabla 12 *Caso de uso: Ver historial de incidentes*

Nombre	Descripción
Caso de uso	Buscar un incidente
Actores	Usuario estándar
Descripción	El usuario podrá buscar un incidente
Precondiciones	El usuario ha iniciado sesión
Escenario Principal	1. El usuario pulsa sobre el apartado de 'Incidentes' en la barra de navegación. 2. El usuario pulsa sobre la barra de búsqueda 3. El usuario escribirá la dirección
Escenario Alternativo	2.b. El usuario cancela la búsqueda

Tabla 13 *Caso de uso: Buscar Incidente por dirección*

4

Arquitectura e implementación de la aplicación

6.1 Introducción

En este capítulo se abordará en detalle toda la parte relacionada con el diseño arquitectónico de la aplicación y su implementación. Se describirán las decisiones tomadas en cuanto a la arquitectura y estructura del código. En primer lugar, se explicará el desarrollo del backend de la aplicación que gestiona la información para su almacenamiento en la base de datos, y la API REST y los *endpoints* que han sido desarrollados para su comunicación con el frontend. Igualmente, en la segunda parte de este capítulo se describe la estructura arquitectónica del *frontend*. Nótese que, para facilitar el diseño en diversas capas, desde el punto de vista de la implementación, cada una de estas se hará corresponder con una carpeta en el código fuente facilitando la organización del código.

6.3 Arquitectura y Desarrollo del backend

Una vez explicado el funcionamiento y la importancia del almacenamiento de datos en el proyecto, es el momento de describir en detalle el *backend*.

Esta es la parte del proyecto que se encarga de recibir y validar las solicitudes del usuario, interactuar con la base de datos para recuperar o modificar los datos necesarios y enviar las respuestas correspondientes.

A continuación, se describen las capas principales que componen la arquitectura del backend y que corresponden a su vez con carpetas de la aplicación:

- **Controllers.** Aquí están desarrollados los controladores, la lógica de las consultas que existe detrás de los *endpoints*. Estos se encargan de ejecutar el código para obtener de la base de datos lo que el usuario solicita. Hay uno por cada tabla de la base de datos.
- **Routes.** En esta capa se encuentran las rutas utilizadas por el *frontend* para comunicarse con el *backend* y pasarle los parámetros necesarios. Dentro de esta capa hay a su vez un archivo '.routes.js' por cada tabla de la base de datos.
  Adicionalmente se encuentran los archivos de configuración, tanto para conectar el código con la base de datos de MySQL como para desplegar el servidor de datos para poder ejecutar esas llamadas a la API.

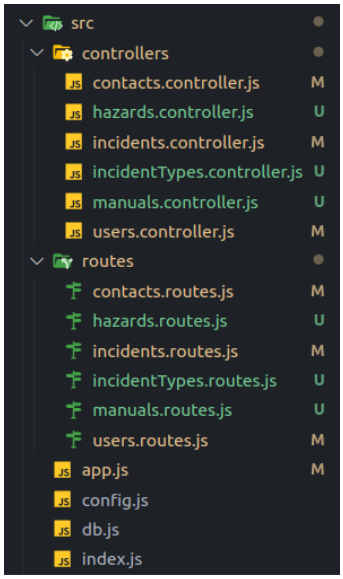


Ilustración 4 *Capas de backend organizadas en carpetas y archivos*

- **Endpoints.** Esta capa representa la interfaz de comunicación con el frontend y contiene, por ello, las URL específicas a las cuales se envían las solicitudes desde el *frontend*. Cada *endpoint* representa una operación o función que el *backend* puede realizar, y está asociado a un método HTTP, que índica la acción a realizar. En el proyecto estos *endpoints* son:
  1. **{url}/api/users.** Soporta GET y POST. Sirve para obtener todos los usuarios de la base de datos y para crear un nuevo usuario, respectivamente.
  2. **{url}/api/users/:id.** Soporta GET y DELETE. Sirve para obtener y eliminar un usuario en particular, obteniéndolo mediante su identificador único.
  3. **{url}/api/contacts.** Soporta GET y POST. Sirve para obtener todos los contactos de la base de datos y para crear un nuevo contacto, respectivamente.

4. **{url}/api/ contacts/:id.** Soporta GET y DELETE. Sirve para obtener y eliminar un contacto en particular, obteniéndolo mediante su identificador único.
5. **{url}/api/ contacts_by_email/:email.** Soporta GET. Sirve para obtener y eliminar un contacto en particular, obteniéndolo mediante su campo de email.

6. **{url}/api/type_of_hazards.** Soporta GET. Sirve para obtener todos los 'peligros que puede ocasionar el incidente' almacenados y poder mostrarlos como opciones al usuario.
7. **{url}/api/type_of_incidents.** Soporta GET. Sirve para obtener todos los 'tipos de incidente' almacenados y poder mostrarlos como opciones al usuario.
8. **{url}/api/incidents.** Soporta GET y POST. Sirve para obtener todos los incidentes de la base de datos y para crear un nuevo incidente, respectivamente.
9. **{url}/api/incidents/:id.** Soporta GET. Sirve para obtener un incidente en particular, obteniéndolo mediante su identificador único.
10. **{url}/api/incidents_by_user_email/:email.** Soporta GET. Sirve para obtener un incidente en particular, obteniéndolo mediante su campo de 'user_email'
11. **{url}/api/manuals.** Soporta GET. Sirve para obtener todos los manuales de emergencias de la base de datos.
12. **{url}/api/manuals/:id.** Soporta GET. Sirve para obtener un manual de emegencias en particular, obteniéndolo mediante su identificador único.

Hay que añadir que para poder desarrollar y probar estos *endpoints* durante esta etapa de implementación, como se ha mencionado anteriormente, se ha hecho uso de la extensión 'ThunderClient':

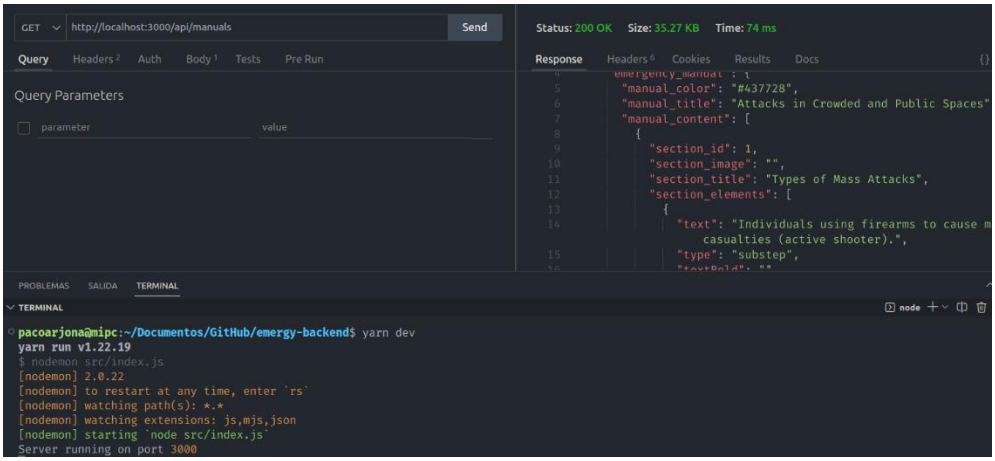


Ilustración 5 *Captura de pantalla usando 'ThunderClient'*

6.4 Arquitectura y Desarrollo del frontend

Finalmente, una vez que se ha elaborado y completado el funcionamiento interno de la aplicación, se inició el diseño arquitectónico y desarrollo del *frontend*.

El *frontend* es el componente arquitectónico de la aplicación con la que interactúan los usuarios finales, y tiene como objetivo principal de proporcionar una interfaz intuitiva, atractiva y fácil de usar para que los usuarios puedan interactuar con las funcionalidades del sistema.

Debido al uso de 'React Native', lo primero que se diseñó fue la estructuración arquitectónica del frontend. A continuación, se describirán brevemente las capas del frontend:

- **components.** Esta capa contiene todos los componentes (ítems) de la aplicación que serán utilizadas en más de una pantalla, lo que permite poder reutilizarlos en donde se desee.
- **data.** En esta capa se almacena la información de los textos planos de la aplicación en formato json.
- **hooks.** En esta capa se almacenan los *custom hooks*. Estos son pequeñas porciones de lógica que se guardan en un archivo aparte para poder reutilizarse y para mantener el código más limpio y ordenado. Suelen tener lógica como *fetching* de datos, actualización de páginas, renderizado, etc.
- **redux.** En esta capa se almacena la lógica desarrollada mediante la librería de *Redux Toolkit*. Se encuentran los archivos donde se declara el estado global, junto con sus *actions* y *reducers*, permitiendo así su modificación en cada página del proyecto.
- **screens.** En esta capa se almacenan todas las páginas del proyecto, donde se desarrolla la interfaz y la experiencia del usuario.
- **stacks.** En esta capa se almacenan todas las navegaciones que usa el proyecto:
    - **TabNavigator.** Contiene la lógica y diseño del 'tab navigator' o navegación por pestañas. Dentro de cada pestaña, se ubica a su vez cada uno de los 'Native Stack' de cada apartado, permitiendo una paginación lineal, fluida y cómoda para el usuario.
    - **NativeStackHome.** Contiene las páginas de Home, donde se puede reportar un incidente accediendo a sus formularios o bien consultar un manual de emergencia.

    - **NativeStackIncidents.** Contiene las páginas de Incidentes, mostrando una página u otra dependiendo de si el usuario ha iniciado sesión, ya que si lo ha hecho podrá ver una lista de sus incidentes reportados.
    - **NativeStackContacts.** Contiene las páginas de Contactos, mostrando una página u otra dependiendo de si el usuario ha iniciado sesión, ya que si lo ha hecho podrá ver una lista de sus contactos añadidos. El usuario también podrá interactuar con la lista, pinchando en un contacto para ver sus detalles, llamarle o enviarle un mensaje, o bien para eliminarlo.
    - **NativeStackAccount.** Contiene las páginas de Cuenta, mostrando una página u otra dependiendo de si el usuario ha iniciado sesión, ya que si lo ha hecho podrá ver sus datos, y en caso contrario, aparecerá la pestaña de iniciar sesión, donde enlaza con todo el flujo de páginas como crear cuenta, contraseña olvidada, etc.
- **styles.** En esta capa se almacenan los ficheros de estilos, tales como el tema de la aplicación y los diferentes estilos de los contendore. Es importante añadir que en esta tecnología no se usa CSS como lenguaje para personalizar los componentes, si no 'StyleSheet', un objeto del propio *framework* que se utiliza para definir y aplicar estilos a los componentes.

- **utils.** A través de esta capa se configura la URL para llamar a la API ({url}/api/…/). A diferencia de otros desarrollos, este proyecto al utilizar las tecnologías antes mencionadas implica que el hecho de poder visualizar los cambios a tiempo real en un dispositivo móvil no sea compatible con llamar al API desde *localhost*, como suele ser en la mayoría de los proyectos. Esto se debe a que el *backend* está desplegado en un dispositivo, y el *frontend* en otro, por lo que se ha de usar como URL la dirección IP del primero.

Todas las capas mencionadas están estructuradas en carpetas dentro a su vez de la carpeta 'src' o *source*, pero fuera de esta existe otra capa (y carpeta) independiente denominada **amplify.** Esta carpeta se crea en el directorio raíz del proyecto, y contiene los archivos y configuraciones relacionados con la infraestructura y servicios de AWS que se utilizan en la aplicación. El contenido de esta puede variar dependiendo de la configuración y los servicios que se estén usando en el proyecto, en este en particular, Amazon Cognito.

Es importante tener en cuenta que esta carpeta y sus respectivos archivos son generados automáticamente por *AWS Amplify*, por lo que se recomienda evitar realizar modificaciones manuales a estos archivos.

Por último, el archivo que se ejecuta al iniciarse la aplicación y que llama al resto de componentes se denomina **App.js**. En este, simplemente se llama al *store* de 'Redux', y al componente *navigator* de 'React Navigation'.

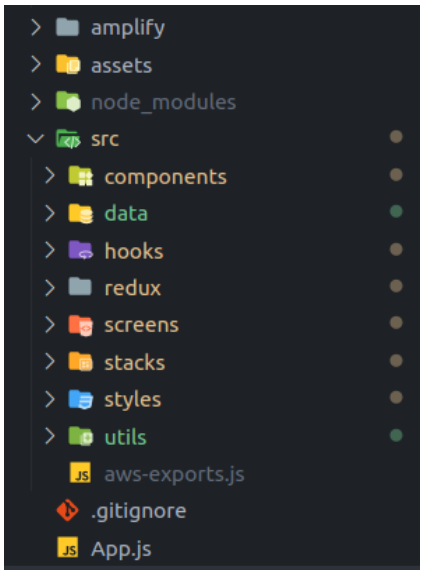


Ilustración 6 *Carpetas del frontend*

7

Conclusión y trabajos futuros

7.1 Introducción

Para finalizar la redacción de esta memoria, se dejará una reflexión final sobre las conclusiones y objetivos logrados, así como una visión de lo que podría llegar a ser el proyecto modificando y extendiendo algunas funcionalidades.

7.2 Conclusiones y objetivos logrados

Tal y como se explicó en las motivaciones, los manuales de emergencia son muy poco útiles por norma general cuando debería ser al revés, permitiendo a la sociedad aprender cómo actuar ante una situación de emergencia porque puede suponer cuestión de vida o muerte en circunstancias extremas. Algunos ejemplos recientes que se pueden mencionar son los terremotos en Turquía, o los conflictos con armas de fuego en Estados Unidos, donde quizás el conocimiento del protocolo de actuación podría haber salvado a alguien o evitar algún mal mayor. Por otra parte, llamar a emergencias suele ser un proceso que requiere bastantes minutos, a lo que hay que sumar que, si alguien se encuentra en una situación límite, dificulta aún más la comunicación con los servicios de emergencia.

Por tanto, se considera que el proyecto reúne las características y funcionalidades necesarias para contemplar estos casos, aportando seguridad y confianza al usuario, y que puede llegar a todas a las edades dada su facilidad de uso.

Desde el punto de vista académico, se ha aprendido a desarrollar un proyecto completo pasando por todas sus etapas, desde el problema a resolver hasta la implementación del código, pasando por diferentes análisis y especificaciones. Se han aplicado muchos de los conocimientos adquiridos en la carrera, tanto a nivel de organización y gestión de tareas como a nivel técnico.

Para concluir, cabe decir que las tecnologías empleadas en este proyecto están a la orden del día, lo que el saber usarlas y trabajar con ellas es de gran utilidad en el mundo laboral, así como el hecho de que hay muchísimos recursos donde documentarse y poder seguir aprendiendo.

7.3 Dificultades encontradas

Al tratarse de un proyecto unipersonal y de gran escala, durante el desarrollo del mismo se han encontrado varias dificultades.

Inicialmente, una vez elegido el tema a tratar en la aplicación, se tenía que decidir que tecnologías se iban a usar, valorando aspectos tales como flexibilidad, escalabilidad y utilidad, además de que se trataba de una aplicación móvil. A esto se ha de añadir, que el autor no tenía experiencia con dichas tecnologías, por lo que previamente a empezar a desarrollar el proyecto, tuvo que aprender cómo funcionaban y a cómo diseñar la arquitectura correctamente, ya que esta es la base de todo el proyecto, así como tampoco tenía ninguna experiencia con el diseño web y con la experiencia de usuario.

Durante el propio desarrollo de la aplicación, se encontraron varias dificultades a resolver, como el uso y manejo de los datos de geolocalización para obtener la posición actual del usuario, o por qué unas librerías son mejoras que otras. Una de las grandes dificultades encontradas, es de dónde se iban a obtener los recursos de los manuales de emergencia, ya que tenían que tratarse de manuales actualizados, no muy extensos y útiles. Otra de las dificultades es cómo se iba a orientar la interfaz de la aplicación de cara al control de usuarios, es decir, los inicios de sesión, registros, etc., ya que se trata de una aplicación que, si en un momento en particular el usuario necesita usarla para reportar un incidente, debía detectar si la sesión esta iniciada con anterioridad para obtener los datos del usuario rápidamente, o bien si está cerrada, aunque los datos del usuario se envíen como anónimos, que siga existiendo la posibilidad de reportar un incidente.

Por último, como otra gran dificultad encontrada, es la gestión de las tareas y el tiempo a dedicar en cada una de ellas, puesto que tareas que parecían las más sencillas y que se estimó en un principio que en poco tiempo se podían desarrollar, resultaron ser las tareas más tediosas y complicadas.

7.4 Trabajos futuros

Tal y como se ha mencionado antes, el tema que trata el proyecto es de actualidad además de hacer uso de tecnologías que están a la orden del día, por lo que hay muchas opciones para poder seguir desarrollando este proyecto.

Algunas de ellas son:

- Añadir alguna funcionalidad nueva, como por ejemplo la que se mencionó en la aplicación de ‘Ariadna’ de localizar desfibriladores cerca de su posición.
- Rediseño de la aplicación, añadiendo una página de bienvenida que explique al usuario para que sirve cada aplicación, así como implementar unos nuevos diseños desarrollados por algún profesional con la experiencia de usuario.
- Despliegue del código en la nube, para no tener que hacer uso de un servidor local.
- El estudio de mercado y lanzamiento de la aplicación en las tiendas de aplicaciones de Apple y Android, para que llegue al máximo número de usuarios posibles.
- Por último, también se podría añadir un entorno de pruebas del código, así como algún tipo de control de calidad, asegurando de esta manera el funcionamiento de la aplicación bajo cualquier circunstancia.

Bibliografía

JESIP Website. (2022b, septiembre 20). *Home - JESIP Website*. https://www.jesip.org.uk/

JESIP Website. (2022, 16 junio). *M/ETHANE - JESIP Website*. https://www.jesip.org.uk/joint-doctrine/m-ethane/

Ready.gov. (s. f.). https://www.ready.gov/es

React Native · Learn once, write anywhere. (s. f.). https://reactnative.dev/

React. (s. f.). https://es.react.dev/

React Navigation | React Navigation. (s. f.). https://reactnavigation.org/

Expo Documentation. (s. f.). Expo Documentation. https://docs.expo.dev/

Location. (s. f.). Expo Documentation. https://docs.expo.dev/versions/latest/sdk/location/

Redux Toolkit | Redux Toolkit. (s. f.). https://redux-toolkit.js.org/

AWS | Gestión de identidades y autenticación de usuario en la nube. (s. f.). Amazon Web Services, Inc. https://aws.amazon.com/es/cognito/

Desarrollo de pila completa - Aplicaciones web y móviles - AWS Amplify. (s. f.). Amazon Web Services, Inc. https://aws.amazon.com/es/amplify/

MySQL :: MySQL Documentation. (s. f.). https://dev.mysql.com/doc/

GitHub: Let's build from here. (s. f.). GitHub. https://github.com/

Home. (s. f.). https://react-hook-form.com/

React-Native-Checkbox. (s. f.). *GitHub - re*act-native-checkbox/react-native-checkbox: *Checkbox component for React Native*. GitHub. https://github.com/react-native-checkbox/react-native-checkbox

npm: @react-native-community/datetimepicker. (s. f.). npm. https://www.npmjs.com/package/@react-native-community/datetimepicker

npm: mysql2. (s. f.). npm. https://www.npmjs.com/package/mysql2

npm: nodemon. (s. f.). npm. https://www.npmjs.com/package/nodemon

Hedkvist, P. (2022, 30 marzo). React Native Authentication Using AWS Cognito & Amplify. *Medium*. https://medium.com/swlh/react-native-authentication-using-aws-cognito-amplify-2ead9f2ee544

Bello, E. (2021, 29 abril). *Descubre qué es el Extreme Programming y sus características*. Thinking for Innovation. https://www.iebschool.com/blog/que-es-el-xp-programming-agile-scrum/

Qué es SCRUM. (2021, 20 septiembre). Proyectos Ágiles. https://proyectosagiles.org/que-es-scrum/

Wells, D. (s. f.). *Extreme Programming: A Gentle Introduction.* http://www.extremeprogramming.org/

Naranjo, M. (2022, 20 marzo). Aplicaciones de emergencia para teléfonos Android. *Computer Hoy*. https://computerhoy.com/reportajes/tecnologia/mejores-aplicaciones-emergencia-telefonos-android-1029625

Bleta. (2023, 15 mayo). *Las 5 mejores aplicaciones para emergencias - Bleta*. https://bleta.io/blog__trashed/mejores-aplicaciones-emergencias/

Ricca, P. (2022). Te puede salvar la vida: todo lo que puedes hacer con la app Emergencias en Android. *Xataka Android*. https://www.xatakandroid.com/tutoriales/te-puede-salvar-vida-todo-que-puedes-hacer-app-emergencias-android

Apéndice A

Manual de Instalación

A.1 Requerimientos

Para poder usar la herramienta en su dispositivo es necesario contar con:

- MySQL instalada por consola, aunque luego se puede utilizar MySQL WorkBench como versión gráfica. (versión 8.0.32 usada)
- Cuenta en AWS Services y una aplicación de autenticación creada en Amazon Cognito. En el siguiente punto se describe la configuración que ha de seguir.
- Node.js instalado en su sistema, a ser posible actualizado. (versión 19.7.0 usada)
- Emulador Android o IOS, o bien usar 'Expo Go' en su propio dispositivo móvil.
- Visual Studio Code o cualquier entorno. Esta instalación da por hecho que se usa este.

A.2 Instalación del backend

Deberá tener previamente una base de datos en MySQL operativa y una aplicación creada en Amazon Cognito. Para tener esta última se deberán seguir los siguientes pasos:

1. Ir a https://eu-west-3.console.aws.amazon.com/console y crearse una cuenta si no se tiene. En caso contrario, iniciar sesión.
2. En la barra de búsqueda, buscar: 'Amplify' y darle a la opción de todas las aplicaciones
3. Seleccionar la opción de 'Build an app' en 'New app'
4. Darle un nombre a la aplicación
5. Una vez haya cargado y se ubique en el menú de configuración de Amplify Studio, en la barra de navegación de la izquierda buscar 'Authentication'
6. Una vez encontrado, configurar el *login* y *sign up*:

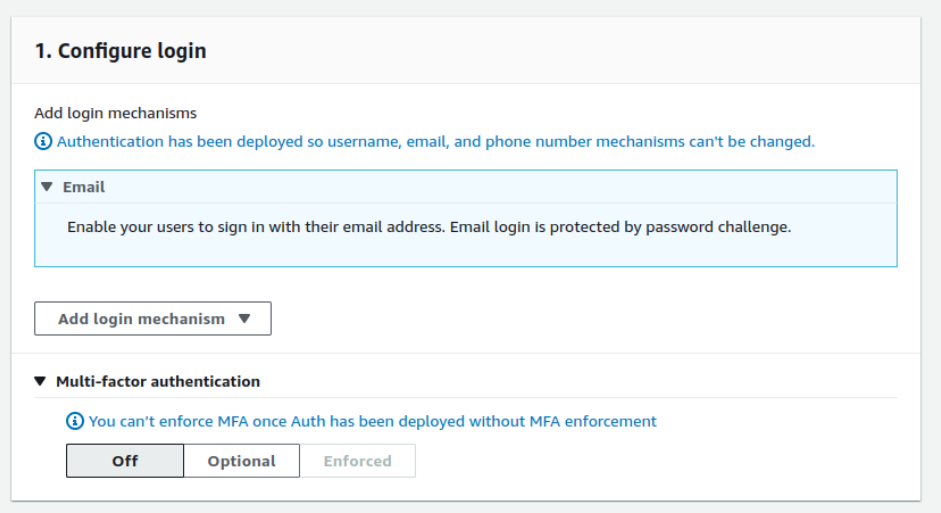


Ilustración 7 *Configuración Login en AWS Amplify*

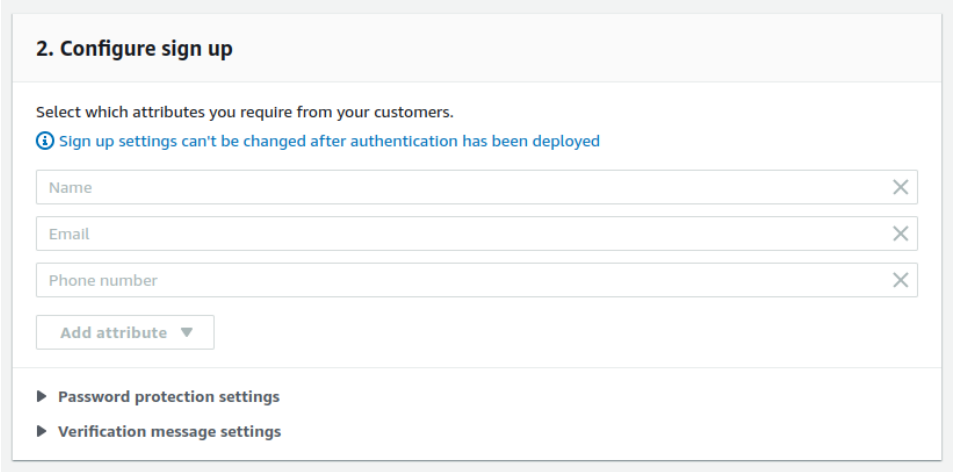


Ilustración 8 *Configuración Sign Up en AWS Amplify 1*

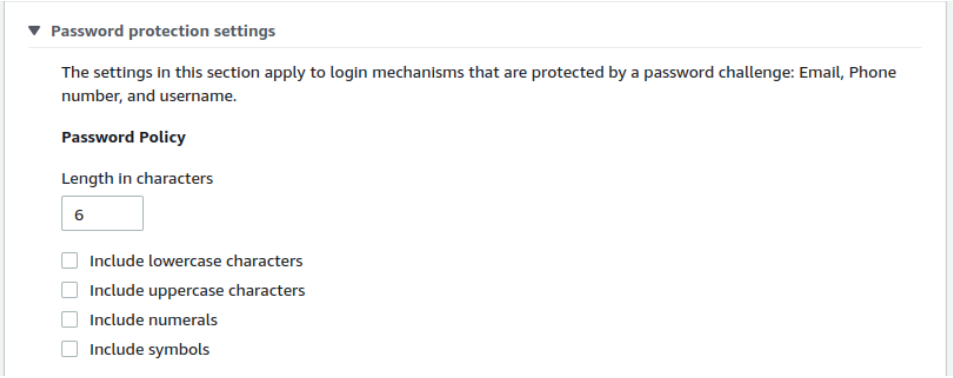


Ilustración 9 *Configuración Sign Up en AWS Amplify 2*

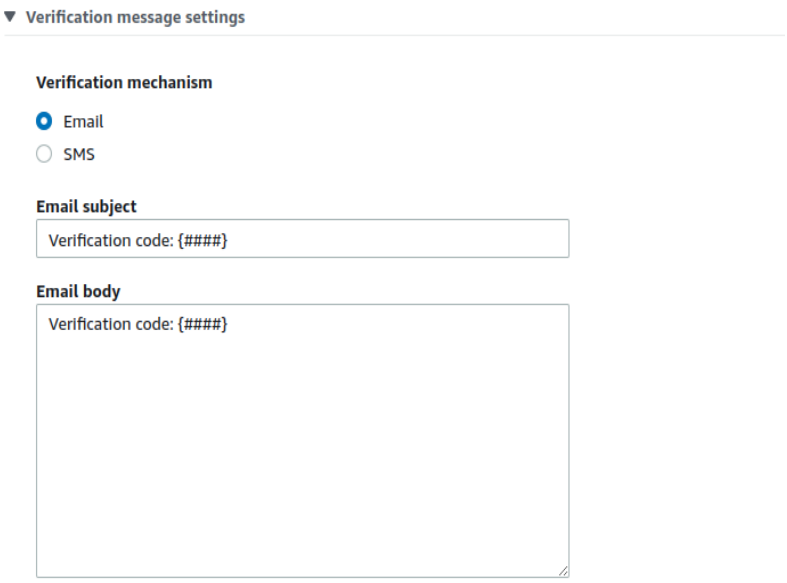


Ilustración 10 Configuración Sign Up en AWS Amplify 3

Una vez hecho esto, se abrirá la carpeta 'emergy-backend' en el editor, y en la terminal ejecutará 'yarn install' para instalar todas las dependencias.

A continuación, se necesita completar este tutorial para configurar Cognito con el proyecto:

https://docs.amplify.aws/start/q/integration/react-native/

(se recomienda la opción 1, en el apartado 'Configure the Amplify CLI versión < 10.8.0')

Por último paso, se creará un archivo '.env' en la raíz del proyecto, y se rellenará con las siguientes variables de entorno:

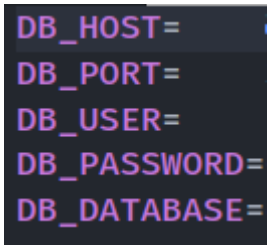


Ilustración 11 *Variables de entorno del backend*

A.3 Instalación del frontend

Para la instalación del *frontend* simplemente se tendrá que ejecutar 'yarn install' en la terminal de la raíz del proyecto para que se instalen las dependencias.

Acto seguido, se configurará la variable de entorno:

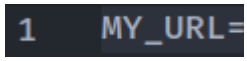


Ilustración 12 Variable de entorno del frontend

Dicha variable será de la forma: http://**{dirección_ip_dispositivo}:{puerto_que_usa}**

A.4 Despliegue de la aplicación

Una vez instalados ambos directorios, simplemente se ejecutarán los comandos:

- **yarn dev** en el backend: Deberá indicarse por consola que todo ha ido correctamente y que el servidor está desplegado en el puerto 3000 (o el que se haya configurado en la base de datos).
- **yarn expo start** en el frontend: Deberá indicarse por consola que todo ha ido correctamente y que puede desplegarlo en el dispositivo que desee. Siguiendo las pautas de este proyecto, se deberá escanear el código QR que aparezca en pantalla con la cámara del móvil (IOS) o con la aplicación de Expo Go (Android) para poder ver la aplicación en tiempo real en su dispositivo, si no siempre está la opción de usar cualquier emulador.

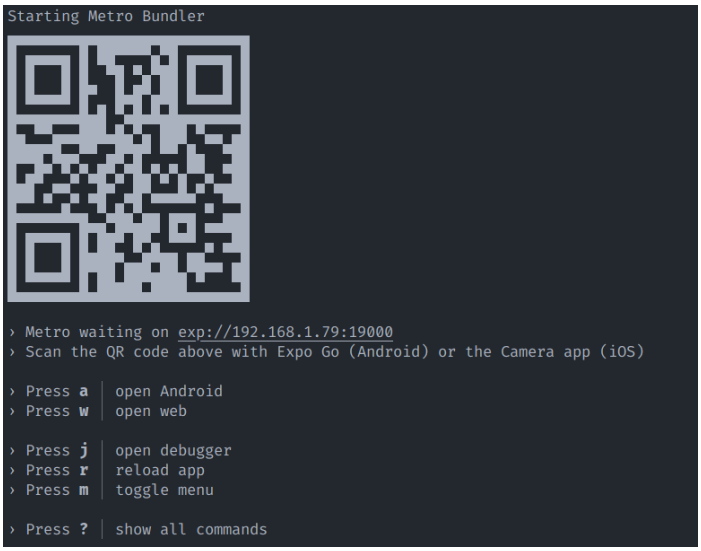


Ilustración 13 *Salida de terminal al ejecutar el frontend*

Apéndice B

Manual de Usuario

A.1 Introducción

A continuación, se describe paso a paso por funcionalidades y pestañas cómo usar la aplicación:

A.2 Crear una cuenta de usuario

Para crear una cuenta de usuario, hay que seguir los siguientes pasos:

1. **Situarse en la pantalla de 'Account'**

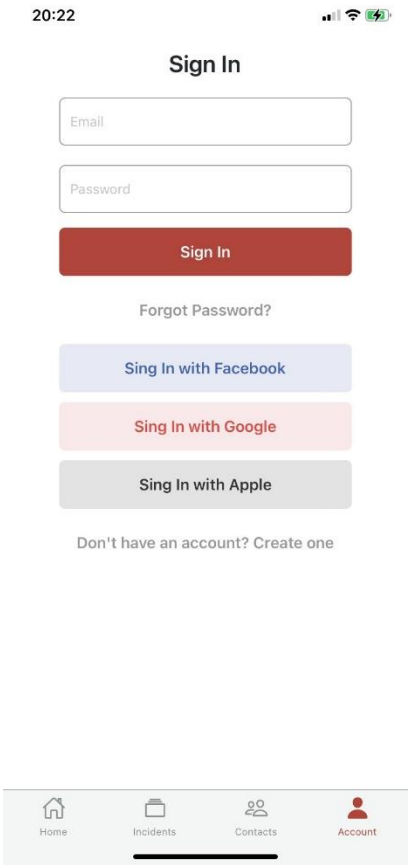


Ilustración 14 *Página de Cuenta*

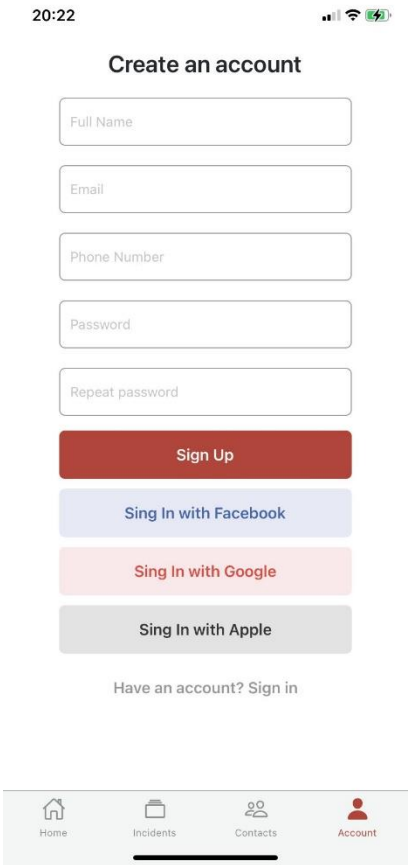
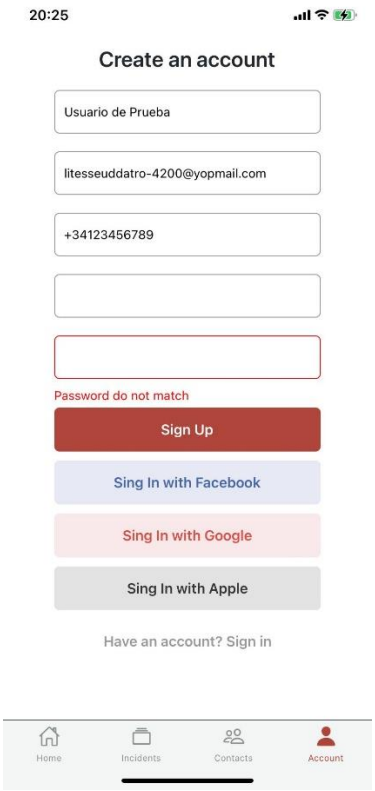


Ilustración 15 *Registrarse*

2. Introducir los datos correctamente y darle al botón de ‘Sign Up’



En esta captura se muestra un error por no introducir bien el campo: repetir contraseña

Ilustración 16 *Registrarse: Introducir datos correctamente*

3. Confirmar email mediante código de verificación

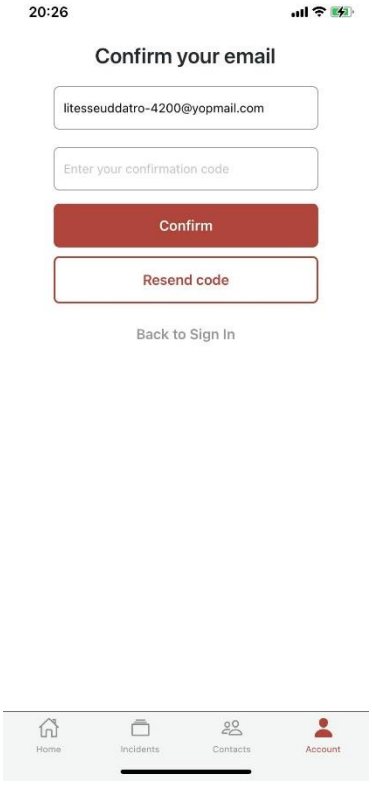


Ilustración 17 *Registrarse: Código de verificación*

Ilustración 18 *Registrarse: Email con código de verificación*

4. **Iniciar sesión con los datos de su cuenta creada**

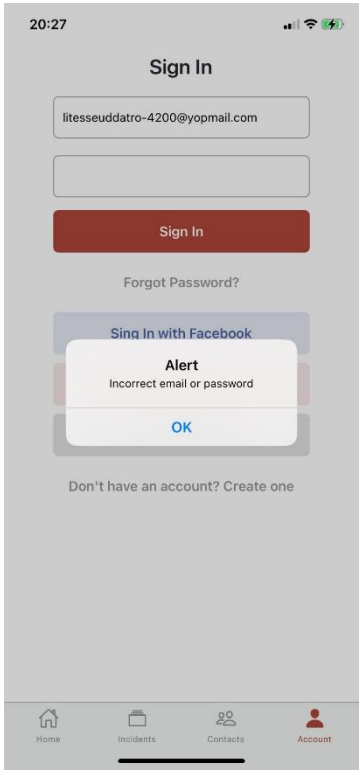


Ilustración 19 *Introducir datos correctamente*

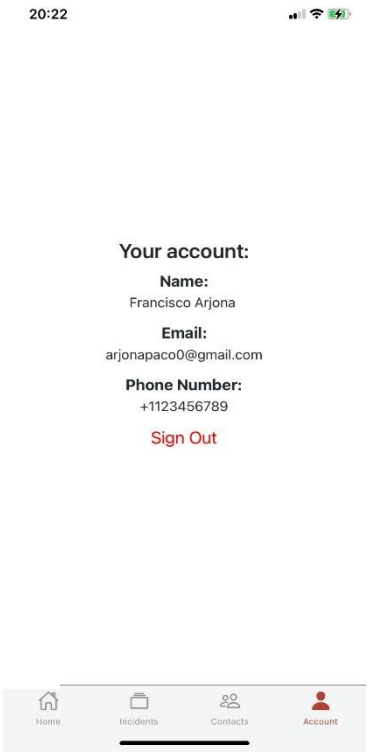


Ilustración 20 *Pantalla de Cuenta con sesión iniciada*

A.3 Funcionalidades de contacto

A continuación, se detallan cuáles son las páginas de contactos y qué funcionalidades ofrece la aplicación. Hay que aclarar que para usar todas estas funcionalidades la sesión debe estar iniciada:

1. **Ver lista de contactos y acceder a los datos de un contacto**

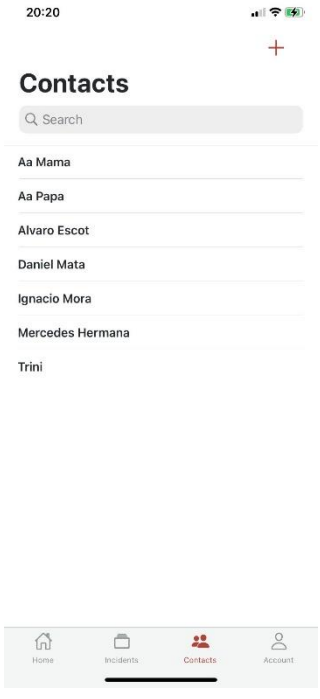


Ilustración 21 *Lista de contactos*

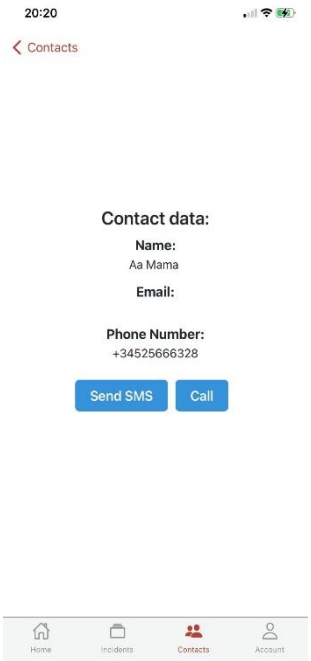


Ilustración 22 *Datos de contacto*

2. Llamar o mandar mensaje de texto a un contacto

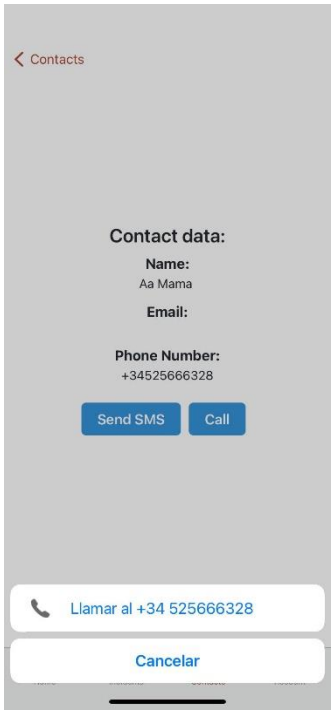


Ilustración 23 *Llamar contacto*

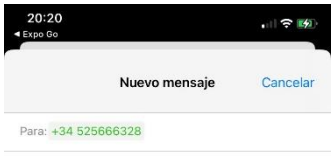


Ilustración 24 *Mandar mensaje a contacto*

3. Agregar un nuevo contacto

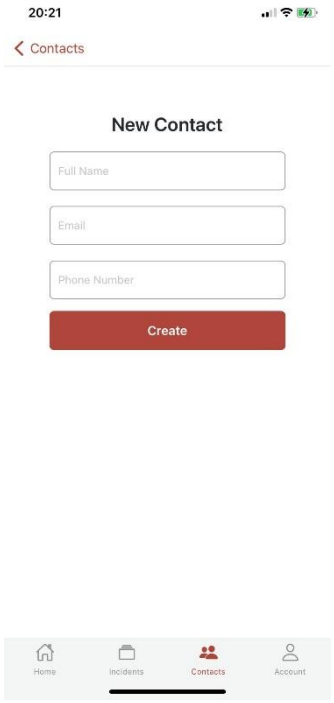


Ilustración 25 *Nuevo contacto*

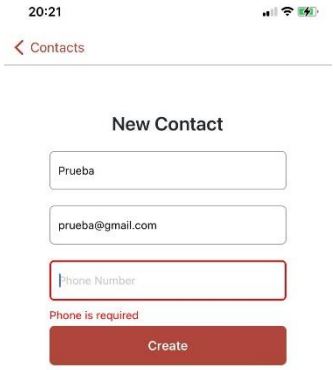


Ilustración 26 *Error de formulario*

4. Búsqueda por nombre de contacto

Ilustración 27 *Resultados de búsquedas*

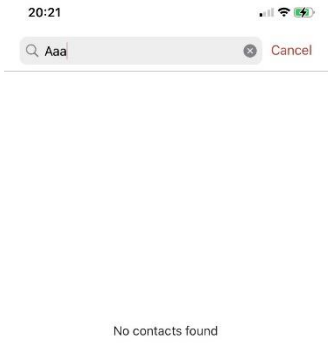


Ilustración 28 *Sin coincidencias*

A.4 Funcionalidades en pestaña de incidente

En este apartado se detallan cuáles son las páginas de incidentes y qué funcionalidades ofrece la aplicación. Para estas funcionalidades la sesión debe estar iniciada:

1. Ver lista de incidentes reportados

Ilustración 29 *Lista de incidentes*

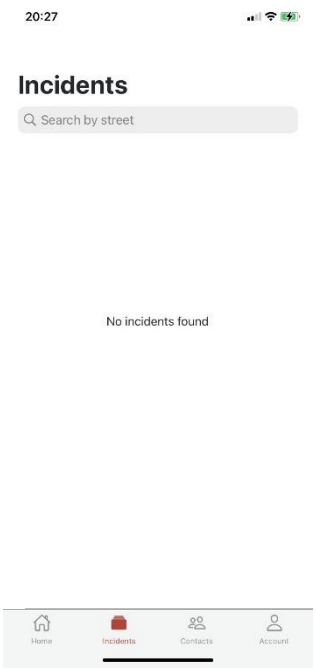


Ilustración 30 *Lista sin incidentes*

2. **Búsqueda por dirección de incidente**

Ilustración 31 *Resultado de búsqueda de incidente*

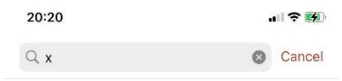


Ilustración 32 *Sin coincidencias de incidente*

A.5 Consultar manuales de emergencias

En este apartado se describe cómo se pueden consultar los manuales de emergencias:

1. **Situarse en la pantalla de 'Home'**

Ilustración 33 *Página de Home*

2. **Pulsar sobre el manual a leer**

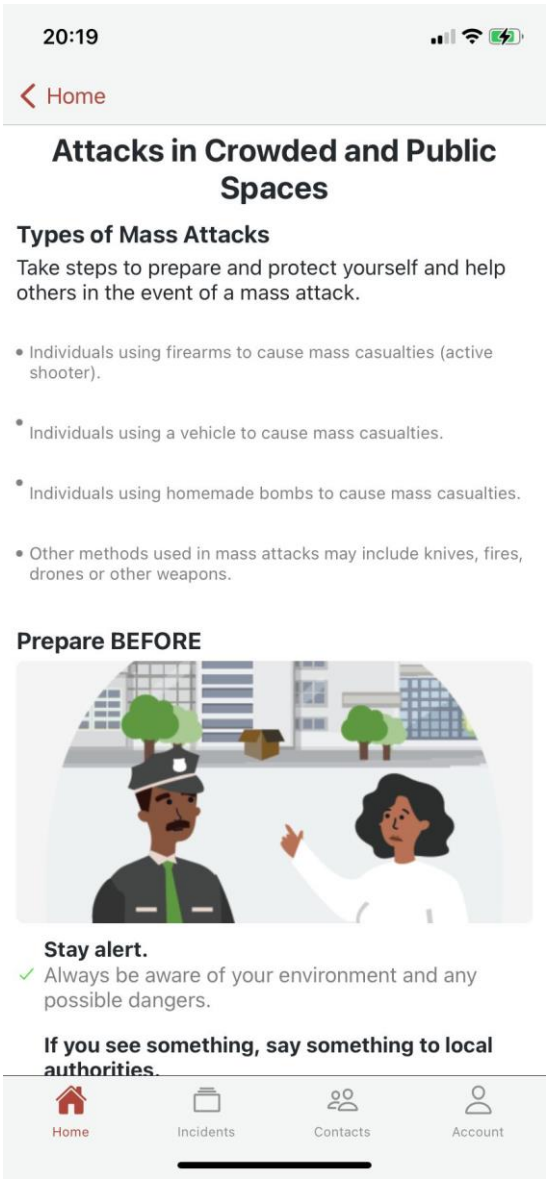


Ilustración 34 *Pantalla de manual de emergencia 1*

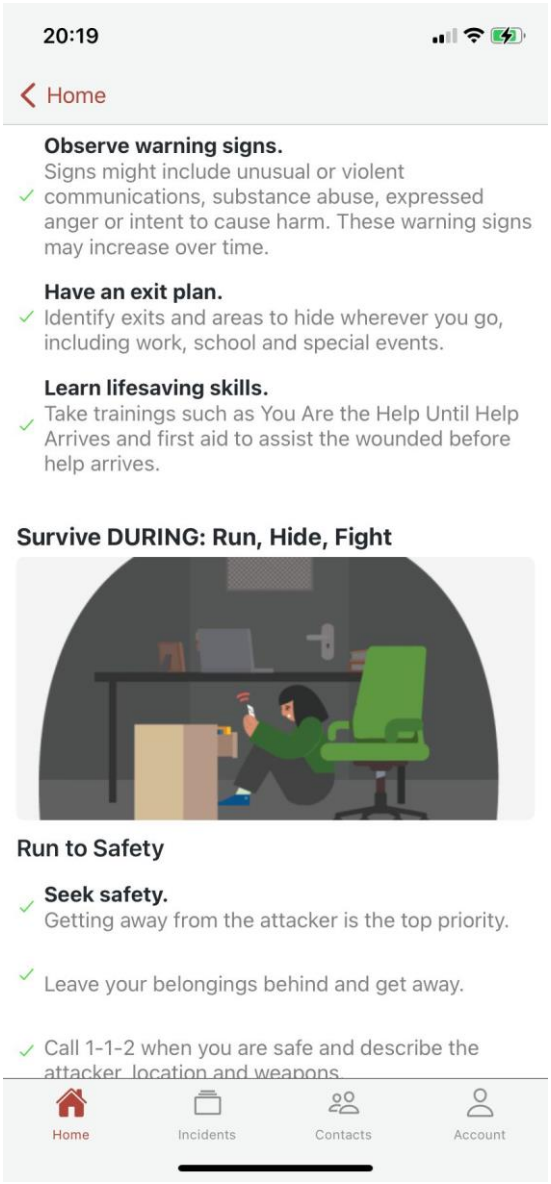


Ilustración 35 *Pantalla de manual de emergencia 2*

A.6 METHANE / Reportar incidente

A continuación, se detalla paso por paso cómo reportar un incidente. Se aclara antes de ver los pasos que todas las partes del formulario tienen un botón al final de la pantalla que dice ‘OK’; pulsar si se ha rellenado ese campo. Una vez pulsado el botón cambiará de color y comenzará el procedimiento para reportar un incidente.

1. **Pulsar sobre el botón de ‘REPORT INCIDENT’**

Ilustración 36 *Pantalla Home*

2. Declarar si es un incidente de alto riesgo o no

*Ilustración **37** Incidente de algo riesgo*

3. Guardar ubicación en tiempo real

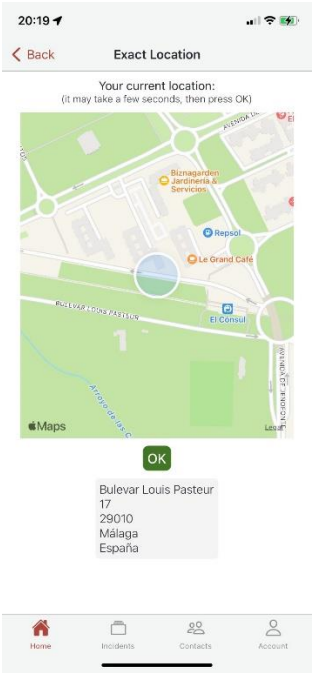


***Ilustración 38** Ubicación*

4. Definir qué tipo de incidente es

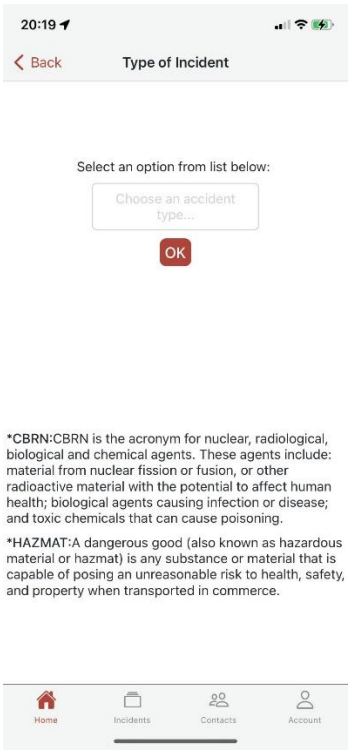


Ilustración 39 *Tipo de incidente*

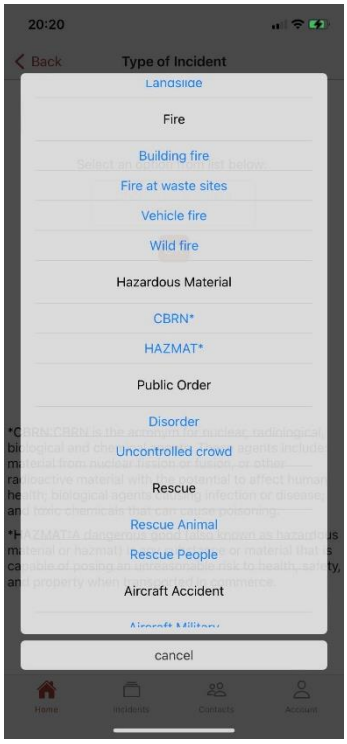


Ilustración 40 *Selección de tipo de incidente*

5. Definir si existe algún tipo de riesgo

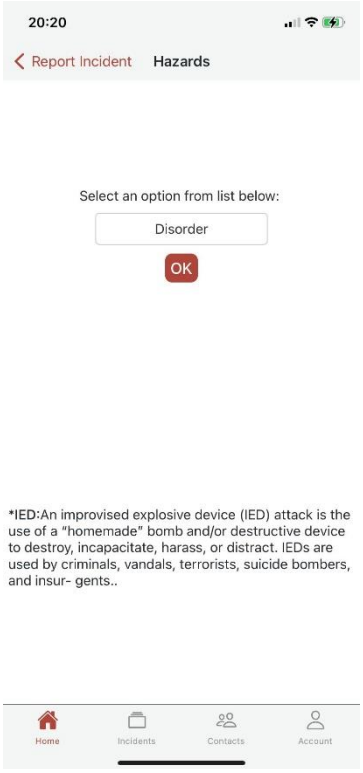


Ilustración 41 *Posibles riesgos*

6. Describir si hay algún problema de acceso a la zona

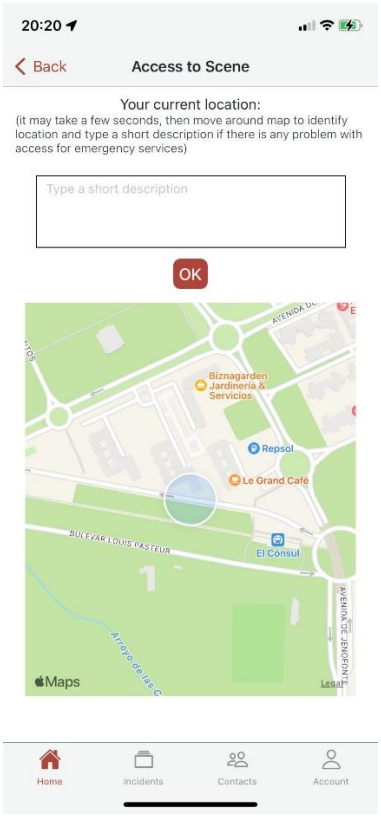


Ilustración 42 *Acceso a la escena*

7. Describir si hay algún herido

Ilustración 43 *Pestaña de accidentados*

8. Describir qué servicios de emergencias requiere el escenario

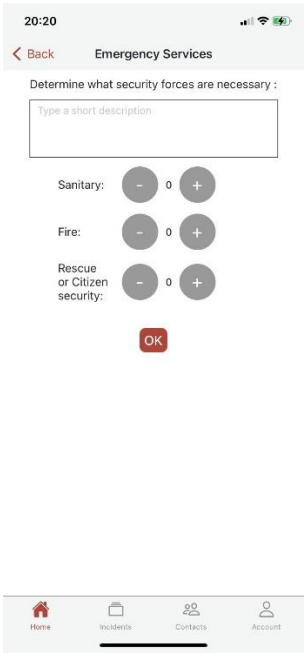


Ilustración 44 *Información sobre el servicio de emergencias*

9. Enviar incidente

Como se ha mencionado en esta memoria, al tratarse de un proyecto puramente académico esta funcionalidad no reporta un incidente realmente a los servicios de emergencia, sino que simplemente muestra un mensaje al usuario y se añade a la su lista de incidentes:

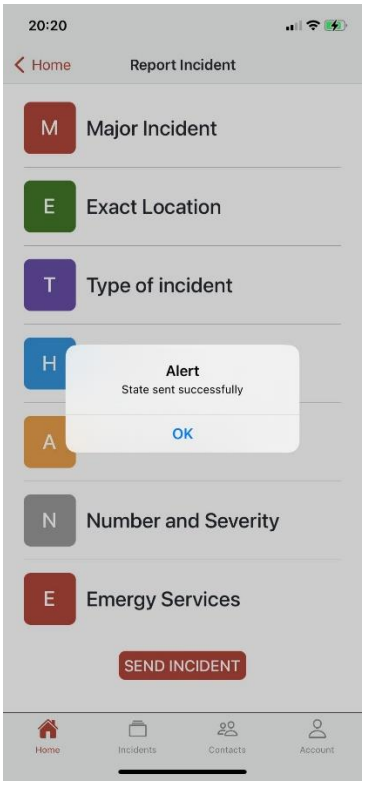


Ilustración 45 *Reportar incidente*

UNIVERSIDAD DE MÁLAGA | uma.es

E.T.S. DE INGENIERÍA INFORMÁTICA

E.T.S de Ingeniería Informática
Bulevar Louis Pasteur, 35
Campus de Teatinos
29071 Málaga