UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

Grado en Ingeniería del Software

DISEÑO Y DESARROLLO DE UNA APLICACIÓN MÓVIL PARA LA GESTIÓN COLABORATIVA DE EMERGENCIAS EN TIEMPO REAL BASADA EN GEOLOCALIZACIÓN

DESIGN AND DEVELOPMENT OF A MOBILE APPLICATION FOR COLLABORATIVE MANAGEMENT REAL TIME EMERGENCY BASED ON GEOLOCATION

Realizado por
Sandra Vázquez Pérez

Tutorizado por
Eduardo Guzmán de los Riscos

Departamento
Lenguajes y Ciencias de la Computación
UNIVERSIDAD DE MÁLAGA

MÁLAGA, julio de 2025

UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADO EN INGENIERÍA DEL SOFTWARE

DISEÑO Y DESARROLLO DE UNA APLICACIÓN MÓVIL PARA LA GESTIÓN COLABORATIVA DE EMERGENCIAS EN TIEMPO REAL BASADA EN GEOLOCALIZACIÓN

DESIGN AND DEVELOPMENT OF A MOBILE APPLICATION FOR COLLABORATIVE MANAGEMENT REAL TIME EMERGENCY BASED ON GEOLOCATION

Realizado por
Sandra Vázquez Pérez

Tutorizado por
Eduardo Guzmán De los Riscos

Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA

MÁLAGA, JUNIO DE 2025

Fecha defensa: julio de 2025

Resumen

En un mundo cada vez más expuesto a riesgos y situaciones de emergencia, la rápida difusión de información precisa y confiable se convierte en un elemento clave para la prevención de desastres y la protección de la población. Las catástrofes recientes, como inundaciones, terremotos, incendios y tormentas, han puesto de manifiesto la necesidad de contar con herramientas tecnológicas que permitan a las personas actuar con rapidez y eficacia.

Este trabajo de fin de grado aborda el diseño y desarrollo de una aplicación móvil innovadora, centrada en la gestión colaborativa de emergencias mediante el uso de geolocalización. La aplicación busca proporcionarles a los usuarios una herramienta que permita no sólo reportar situaciones de emergencia en tiempo real, sino también recibir alertas personalizadas que incluyan información relevante para garantizar su seguridad. El modelo planteado promueve la colaboración activa entre distintos perfiles de usuarios. Cada uno de estos roles desempeña funciones específicas que contribuyen a garantizar la credibilidad y utilidad de las alertas.

El principal objetivo de esta herramienta es mejorar la gestión de emergencias facilitando la comunicación eficiente y confiable entre los usuarios, promoviendo la prevención de riesgos y una actuación informada. Fomentando una respuesta más efectiva en tiempo real. Además, se pretende ofrecer una experiencia de usuario accesible e intuitiva, que permita a cualquier persona contribuir activamente a la seguridad de su comunidad. Pretendiendo promover la difusión de recomendaciones específicas de seguridad, contribuyendo así a minimizar riesgos y salvar vidas.

Palabras clave: *Emergencias, Prevención de Riesgos, React Native, Spring Boot, MySQL*

Abstract

In a world increasingly exposed to risks and emergency situations, the fast dissemination of accurate and reliable information becomes a key element in disaster prevention and the protection of the population. Recent catastrophes such as floods, earthquakes, wildfires and storms have highlighted the need for technological tools that enable people to act speed and effectively.

This final degree project focuses on the design and development of an innovative mobile application based on the collaborative management of emergencies through the use of geolocation. The application aims to provide users with a tool that not only allows them to report emergency situations in real time, but also to receive custom alerts containing relevant information to ensure their safety. The proposed model encourages active collaboration between different user roles, each of which carries out specific functions that contribute to the credibility and usefulness of the alerts.

The main objective of this tool is to improve emergency management by facilitating efficient and trustworthy communication between users, promoting risk prevention and informed action, and encouraging a more effective real-time response. Moreover, it seeks to offer an accessible and intuitive user experience, enabling anyone to actively contribute to the safety of their community. Additionally, it aims to promote the dissemination of specific safety recommendations, helping to minimise risks and save lives.

Keywords: *Emergencies, Risk Prevention, React Native, Spring Boot, MySQL*

Índice

Resumen 1

Abstract 2

Índice 1

Índice de figuras 5

Índice de Cuadros 10

Introducción 1
1.1 Motivación 1
1.2 Objetivos 2
1.3 Estudio de aplicaciones similares 4
1.4 Metodología 5
1.5 Estructura de la memoria 6

Tecnologías y Herramientas utilizadas 9
2.1 Tecnologías del Frontend 9
2.2 Tecnologías del Backend 12
2.3 Base de Datos 14
2.4 Herramientas y Software utilizados 14

Especificación y análisis 17
3.1 Descripción general de la aplicación 17
3.1.1 Acceso a la aplicación 17
3.1.2 Mapa Interactivo 18
3.1.3 Emergencias 19
3.1.4 Notificaciones 29
3.1.5 Cuenta de Usuario y Roles 31
3.1.6 Historial de Emergencias 32
3.1.7 Solicitudes 33
3.2 Actores del sistema 34
3.3 Requisitos Funcionales 35
3.4 Requisitos No Funcionales 37
3.5 Modelo de Comportamiento 37
3.5.1 Diagrama de Casos de Uso 37
3.5.2 Diagramas de Flujo de Procesos 68
3.6 Modelo de interacción 74

3.6.1 Diagrama de Secuencia ... 74

Modelado y Diseño del sistema ... 83
4.1 Base de Datos ... 83
4.1.1 Modelo E/R ... 83
4.2 Diseño de la interfaz de usuario ... 85
4.3 Diagrama de arquitectura y tecnologías de la aplicación ... 85

Implementación y pruebas ... 87
5.1 Diseño del modelo relacional y módulo de carga de datos ... 88
5.2 Desarrollo del backend ... 88
5.2.1 Estructura del proyecto ... 88
5.2.2 Base de datos en el Backend ... 91
5.2.3 Endpoints destacados ... 92
5.3 Desarrollo del frontend ... 98
5.3.1 Estructura ... 98
5.3.2 Partes de componentes destacadas ... 99
5.4 Pruebas de la aplicación ... 106

Despliegue de la aplicación ... 111
6.1 Frontend ... 111
6.2 Backend ... 112
6.3 Base de datos ... 113

Conclusiones y trabajos futuros ... 115
7.1 Evaluación del cumplimiento de los objetivos ... 115
7.2 Dificultades encontradas y soluciones adoptadas ... 116
7.3 Líneas futuras de mejora ... 117

Referencias ... 119

Manual de Instalación ... 121
A.1 Requisitos previos ... 121
A.2 Ejecución de la base de datos en local ... 122
A.3 Instalación y ejecución del backend ... 122
A.4 Instalación y ejecución del frontend ... 123
A.5 Ejecución del emulador ... 123

Manual de usuario ... 125
B.1. Registro de un usuario ... 125
Funcionalidades comunes: ... 126
B.2. Creación de una emergencia ... 126
B.3. Edición de datos Mi cuenta ... 128
B.4. Solicitud nuevo rol ... 129
B.5. Visualización historial de emergencias ... 129
Funcionalidades como ciudadano ... 130
B.6. Solicitud edición de emergencia ... 130
B.7. Reporte de emergencia ... 131

B.8. Visualización mis solicitudes de Edición de Emergencias 132
B.9. Visualización mis solicitudes de cambio de rol 132
Funcionalidades como experto y autoridad: 133
B.10. Edición y verificación de una emergencia 133
B.11. Resolución de una emergencia 134
B.12. Gestión solicitudes de Edición de Emergencias 134
Funcionalidades como administrador: 135
B.13. Gestión solicitudes de Cambio de Rol 135

Índice de figuras

Figura 1. Símbolos de Inundación 21
Figura 2: Símbolos de Incendio forestal 22
Figura 3: Símbolos de Terremoto 22
Figura 4: Símbolos de Tsunami 22
Figura 5: Símbolos de Tornado 22
Figura 6: Símbolos de Huracán 23
Figura 7: Símbolos de Tormenta 23
Figura 8: Símbolos de Deslizamientos de Tierra 23
Figura 9: Símbolos de Erupción Volcánica 23
Figura 10: Símbolos de Ola de calor 24
Figura 11: Símbolos de Ola de frío 24
Figura 12: Símbolos de Granizada 24
Figura 13: Símbolos de Nevada 24
Figura 14: Símbolos de Derrumbe de edificio 25
Figura 15: Símbolos de Bloqueo de carretera 25
Figura 16: Símbolos de Fuga de gas 25
Figura 17: Símbolos de Corte de energía 25
Figura 18: Símbolos de Epidemia 26
Figura 19: Símbolos de Brote de enfermedad 26
Figura 20: Símbolos de Intoxicación 26
Figura 21: Símbolos de Protesta violenta 26
Figura 22: Símbolos de Evacuación 27
Figura 23: Símbolos de Refugiados 27
Figura 24: Símbolos de Fuga química 27
Figura 25: Símbolos de Explosión industrial 27
Figura 26: Símbolos de Colapso de infraestructuras 28
Figura 27: Símbolos de Accidente de coche 28
Figura 28: Símbolos de Accidente aéreo 28
Figura 29: Símbolos de Accidente ferroviario 28
Figura 30: Símbolos de Hundimiento marítimo 29
Figura 31: Símbolos de Ataque de fauna peligrosa 29
Figura 32: Símbolos de Plaga de animales 29
Figura 33: Diagrama de casos de uso como ciudadano de la aplicación 38
Figura 34: Visualización parte Mapa Interactivo Diagrama Caso de Uso Ciudadano 38
Figura 35: Visualización parte Crear Emergencia Diagrama Caso de Uso Ciudadano 39

Figura 36: VIsualización parte Gestión Datos Personales Diagrama Caso de Uso Ciudadano 39
Figura 37: Visualización parte más funciones Diagrama Caso de Uso Ciudadano..... 40
Figura 38: Diagrama de casos de uso como Experto o Autoridad de la aplicación.... 41
Figura 39: Visualización parte visualizar detalles Emergencia Diagrama Caso de Uso Experto/Autoridad 41
Figura 40: Visualización parte Crear Emergencia Diagrama Caso de Uso Experto/Autoridad 41
Figura 41: Visualización parte Solicitudes Edición de Emergencias Diagrama Caso de Uso Experto/Autoridad..... 42
Figura 42: Diagrama de casos de uso como administrador de la aplicación..... 42
Figura 43: Visualización parte Solicitudes pendientes cambio de Rol Diagrama Caso de Uso Administrador..... 42
Figura 44: Diagrama de flujo de Autenticación..... 68
Figura 45: Diagrama de flujo de Creación de Emergencia..... 69
Figura 46: Diagrama de flujo de Historial de Emergencias..... 69
Figura 47: Diagrama de flujo de resolución de Emergencias 71
Figura 48: Diagrama de flujo de edición de Emergencias 71
Figura 49: Diagrama de flujo de Gestión de solicitudes de edición de emergencias... 72
Figura 50: Diagrama de flujo de gestión de datos personales..... 72
Figura 51: Diagrama de flujo de solicitud de cambio de rol..... 73
Figura 52: Diagrama de flujo de gestión de solicitudes de cambio de rol..... 74
Figura 53: Diagrama de secuencia Autenticación..... 75
Figura 54: Diagrama de secuencia Crear emergencia..... 76
Figura 55: Diagrama de secuencia Historial de emergencias 76
Figura 56: Diagrama de secuencia Resolución de emergencias..... 77
Figura 57: Diagrama de secuencia Edición de emergencias 78
Figura 58: Diagrama de secuencia de Gestión solicitudes edición de emergencias..... 79
Figura 59: Diagrama de secuencia de Gestión de datos personales 80
Figura 60: Diagrama de secuencia de Solicitud de cambio de rol..... 80
Figura 61: Diagrama de secuencia de Gestión de solicitudes de cambio de rol 81
Figura 62: Modelo Entidad Relación de la aplicación 84
Figura 63: Diagrama de arquitectura de la aplicación..... 86
Figura 64: Estructura Backend 89
Figura 65: Endpoint carpeta controllers 89
Figura 66: Método de la capa de Servicios..... 89
Figura 67: Capa Repository..... 90
Figura 68: Entidad base de datos en backend 90
Figura 69: Ejemplo clase DTO..... 91
Figura 70: Endpoint Emergencias No resueltas..... 92
Figura 71: Método en la capa servicio Emergencias no resueltas 92
Figura 72: Consulta Base de datos Emergencias no resueltas..... 92
Figura 73: Endpoint para crear una emergencia 93
Figura 74: Método en la capa servicio para crear una emergencia 93

Figura 75: Endpoint para editar una emergencia 93
Figura 76: Método en la capa servicio para editar una emergencia 94
Figura 77: Endpoint para crear una nueva localización 94
Figura 78: Método en la capa servicio para crear una nueva localización 95
Figura 79: Endpoint para crear una nueva notificación 95
Figura 80: Método en la capa de servicio para crear una nueva notificación 95
Figura 81: Método que utiliza fórmula de Haversine 96
Figura 82: Lista de usuarios a notificar 97
Figura 83: Endpoint para que un usuario se registre o inicie sesión 97
Figura 84: Método en la capa de servicio para que un usuario se registre o inicie sesión 98
Figura 85: Estructura del frontend 98
Figura 86: Petición de servicio inicio de sesión 100
Figura 87: Obtención localización usuarios 100
Figura 88: Petición permisos obtención ubicación 100
Figura 89: Obtención token de notificación usuario 101
Figura 90: Utilización de almacenamiento persistente 101
Figura 91: Método para la recuperación de contraseñas 101
Figura 92: Creación de un contexto 102
Figura 93: Listener para notificar sobre cambios en el estado de autenticación 102
Figura 94: Obtención contexto de autenticación 102
Figura 95: Componente MapView 103
Figura 96: Llamada al servicio de geocodificación inversa 104
Figura 97: Obtención dirección mediante geocodificación inversa 104
Figura 98: Método para obtener las sugerencias de localizaciones 105
Figura 99: Obtención de sugerencias de localizaciones 105
Figura 100: Método para cerrar sesión 105
Figura 101: Cierre de sesión Firebase 106
Figura 102: Gestión navegación entre pantallas 106
Figura 103: Edición parámetros base de datos 122
Figura 104: Pantalla de inicio de sesión 125
Figura 105: Pantalla de registro 126
Figura 106: Pantalla principal 126
Figura 107: Pantalla para crear una emergencia 127
Figura 108: Pantalla para crear una emergencia 2 127
Figura 109: Pantalla de sugerencias al introducir una dirección 127
Figura 110: Pantalla tras crear una emergencia 127
Figura 111: Pantalla de notificación de creación de una emergencia 127
Figura 112: Pantalla con menú desplegable 128
Figura 113: Pantalla de Mi cuenta 128
Figura 114: Pantalla de edición de datos personales 128
Figura 115: Pantalla Mi cuenta con datos actualizados 129
Figura 116: Pantalla Mi cuenta con desplegable solicitud de rol 129
Figura 117: Pantalla tras solicitar el cambio de rol 129

Figura 118: Pantalla Historial de emergencias 130
Figura 119: Pantalla visualización detalles de una emergencia 130
Figura 120: Pantalla solicitud edición de emergencia 131
Figura 121: Pantalla visualización detalles de una emergencia deslizada hacia abajo 131
Figura 122: Pantalla visualización detalles de una emergencia cuando ha reportado ya 131
Figura 123: Pantalla Mis solicitudes de edición de emergencias 132
Figura 124: Pantalla Mis solicitudes de edición de emergencias con solicitud resuelta 132
Figura 125: Pantalla Mis solicitudes de cambio de rol 133
Figura 126: Pantalla visualización detalles de una emergencia como experto o autoridad 133
Figura 127: Pantalla edición de emergencia como experto o autoridad 133
Figura 128: Pantalla resolución de emergencia como experto o autoridad 134
Figura 129: Pantalla confirmación resolución de emergencia 134
Figura 130: Pantalla Solicitudes de edición de emergencias pendientes 134
Figura 131: Pantalla Solicitudes de edición de emergencias tras rechazar solicitud 135
Figura 132: Pantalla Solicitudes pendientes de cambio de rol 135

Índice de Cuadros

Cuadro 1: Caso de uso: Registrarse 43
Cuadro 2: Caso de uso: Iniciar Sesión 44
Cuadro 3: Caso de uso: Restablecer contraseña 44
Cuadro 4: Caso de uso: Cerrar Sesión 45
Cuadro 5: Caso de uso: Interactuar con el mapa interactivo 45
Cuadro 6: Caso de uso: Diferenciación emergencias verificadas 46
Cuadro 7: Caso de uso: Visualización detalles emergencia 46
Cuadro 8: Caso de uso: Emergencia verificada 47
Cuadro 9: Caso de uso: Notificaciones Emergencia 47
Cuadro 10: Caso de uso: Crear una emergencia 48
Cuadro 11: Caso de uso: Añadir descripción 49
Cuadro 12: Caso de uso: Añadir recomendaciones 49
Cuadro 13: Caso de uso: Seleccionar tipo de Emergencia 50
Cuadro 14: Caso de uso: Seleccionar nivel de Riesgo 51
Cuadro 15: Caso de uso: Seleccionar ubicación de la Emergencia 51
Cuadro 16: Caso de uso: Verificar Emergencias 52
Cuadro 17: Caso de uso: Historial de Emergencias 52
Cuadro 18: Caso de uso: Reportar Emergencias 53
Cuadro 19: Caso de uso: Resolver Emergencias 54
Cuadro 20: Caso de uso: Añadir Recomendaciones Emergencias Existentes 55
Cuadro 21: Caso de uso: Modificar nivel de Riesgo Emergencia Existente 56
Cuadro 22: Caso de uso: Solicitar añadir recomendaciones 56
Cuadro 23: Caso de uso: Solicitar modificar nivel de Riesgo 57
Cuadro 24: Caso de uso: Acceder solicitudes edición emergencias 58
Cuadro 25: Caso de uso: Aceptar solicitudes edición emergencias 59
Cuadro 26: Caso de uso: Denegar solicitudes edición emergencias 59
Cuadro 27: Caso de uso: Acceder Mis Solicitudes edición Emergencias 60
Cuadro 28: Caso de uso: Agregar o modificar nombre 61
Cuadro 29: Caso de uso: Agregar o modificar número de teléfono 61
Cuadro 30: Caso de uso: Modificar radio 62
Cuadro 31: Caso de uso: Solicitar cambio de rol 63
Cuadro 32: Caso de uso: Aceptar solicitudes cambio de rol 64
Cuadro 33: Caso de uso: Denegar solicitudes cambio de rol 64
Cuadro 34: Caso de uso: Visualizar solicitudes cambio de Rol 65
Cuadro 35: Caso de uso: Recibir notificaciones creación de Emergencias 66
Cuadro 36: Caso de uso: Recibir notificaciones modificación nivel de Riesgo 66
Cuadro 37: Caso de uso: Recibir notificaciones verificación de Emergencias 67

Cuadro 38: Caso de uso: Recibir notificaciones resolución de Emergencias 67

1

Introducción

En este apartado se expone qué ha llevado al origen de este proyecto, detallando los objetivos que se persiguen y presentando la metodología empleada durante su desarrollo. Además, se ofrece una visión general sobre la organización del documento con el fin de comprenderlo globalmente.

1.1 Motivación

La aplicación desarrollada en este proyecto tiene como objetivo principal mejorar la gestión colaborativa de emergencias mediante el uso de tecnologías móviles y geolocalización en tiempo real. Esta propuesta busca no solo optimizar la difusión de alertas, sino también construir un entorno colaborativo donde la información fluya de manera fiable, precisa y útil para todos los actores implicados en la gestión del riesgo. El desarrollo de esta herramienta responde a una serie de motivaciones clave que han impulsado su creación y diseño, las cuales se detallan a continuación:

-Incremento en la ocurrencia y gravedad de situaciones de emergencia:
Se ha incrementado con frecuencia diferentes tipos de emergencias como emergencias naturales, meteorológicas, urbanas, sanitarias y sociales entre otras. Estos eventos generan grandes pérdidas humanas y materiales, requiriendo respuestas rápidas y efectivas. La aplicación busca abordar esta necesidad,

ofreciendo una herramienta que mejore la capacidad de reacción de las comunidades afectadas.

-Importancia de la participación ciudadana en la prevención de riesgos:
La involucración de la comunidad es esencial para la detección temprana y la respuesta a emergencias. Sin embargo, pocas soluciones tecnológicas aprovechan el potencial de la colaboración ciudadana para mejorar la gestión de riesgos.

-Necesidad de aprovechar la tecnología de geolocalización:
Las tecnologías basadas en geolocalización ofrecen grandes oportunidades para proporcionar información contextualizada y precisa. Aunque su potencial en el ámbito de la gestión de emergencias es significativo, aún está muy poco utilizado. Esta aplicación integra la geolocalización como núcleo de su diseño, asegurando que las alertas se distribuyan de manera relevante y específica para cada usuario.

-Relevancia de la colaboración entre expertos, autoridades y ciudadanos:
Este proyecto busca promover un modelo colaborativo en el que ciudadanos, expertos y autoridades trabajen juntos para garantizar la precisión y utilidad de las alertas. Este enfoque no sólo aumenta la confianza en la información, sino que también fomenta la cooperación en momentos críticos.

-Contribuir a salvar vidas y minimizar daños:
Este proyecto se desarrolla bajo la motivación de crear una herramienta que contribuya significativamente a este propósito, proporcionando no sólo alertas, sino también recomendaciones de seguridad específicas que puedan ayudar a las personas a actuar correctamente en situaciones críticas.

Por lo tanto, cada una de estas motivaciones refuerza la relevancia y necesidad del proyecto, destacando cómo la aplicación puede abordar desafíos clave en la gestión de emergencias mediante el uso de tecnología y la colaboración activa de todos los actores involucrados.

1.2 Objetivos

El principal objetivo de este proyecto es desarrollar una herramienta tecnológica que facilite la gestión y notificación de emergencias de manera colaborativa, mejorando la capacidad de respuesta y prevención ante situaciones de riesgo. Este sistema busca que ciudadanos, expertos y autoridades trabajen juntos en la identificación, validación y difusión de alertas, maximizando su efectividad mediante el uso de geolocalización y notificaciones personalizadas.

Por lo tanto, los objetivos específicos existentes son los siguientes:

- ***Facilitar la creación de emergencias y gestión de notificaciones en tiempo real:***
  Desarrollar una plataforma que permita a los usuarios crear emergencias de forma ágil y sencilla, reportando situaciones de riesgo de manera inmediata. Además, al utilizar un sistema de notificaciones en tiempo real basado en geolocalización la aplicación asegura que los usuarios estén siempre informados de las emergencias relevantes para ellos.

- ***Fomentar la colaboración activa y organizada entre ciudadanos, expertos y autoridades:***
  Establecer un sistema de roles y permisos que permita a los ciudadanos visualizar y crear emergencias, mientras que los expertos y autoridades validan y enriquecen la información, proporcionando recomendaciones de seguridad y ajustando el nivel de riesgo.
  Este enfoque colaborativo promoverá una respuesta más eficiente y precisa ante situaciones de emergencia.

- ***Garantizar la fiabilidad y la relevancia de la información compartida:*** Implementar un sistema robusto de verificación que permita diferenciar las emergencias reportadas por ciudadanos generales de aquellas confirmadas por expertos. Esta verificación reforzará la confianza de los usuarios en las alertas recibidas y mejorará la toma de decisiones ante situaciones críticas.

- ***Ofrecer información contextualizada y recomendaciones específicas para cada emergencia:*** Ofrecer una visualización clara de las emergencias mediante un mapa interactivo, donde los usuarios puedan acceder a información detallada sobre cada situación de riesgo, incluyendo tipo de emergencia, nivel de gravedad, ubicación y recomendaciones de seguridad específicas para cada caso, facilitando la toma de decisiones informadas.

- ***Facilitar el seguimiento de la evolución de las emergencias a través de actualizaciones en tiempo real:*** Permitir a los usuarios seguir la evolución de las emergencias mediante notificaciones sobre cambios en el nivel de riesgo, estado de verificación o resolución de la emergencia, proporcionando información actualizada y evitando confusión o desinformación en situaciones dinámicas.

1.3 Estudio de aplicaciones similares

Antes de iniciar el desarrollo de esta aplicación, se llevó a cabo un análisis de diversas aplicaciones móviles ya existentes que tienen como objetivo la gestión de emergencias o la comunicación en situaciones críticas. El propósito principal de este estudio fue identificar las funcionalidades más comunes, examinar sus limitaciones y detectar oportunidades de mejora, con el fin de diseñar una herramienta más completa y adaptada a las necesidades reales de los ciudadanos, las autoridades y los expertos.

Entre las aplicaciones más relevantes analizadas se encuentran:

- **My112:** Esta aplicación, vinculada a los servicios de emergencia de varias comunidades autónomas españolas como 112 Madrid o 112 Castilla y León, permite al usuario contactar directamente con el número de emergencias. Una de sus funcionalidades más destacadas es el envío automático de la ubicación exacta del usuario al realizar la llamada, lo que facilita una intervención más rápida por parte de los servicios de emergencia. Además, ofrece la posibilidad de recibir notificaciones personalizadas de avisos emitidos por los centros 112, así como visualizar un histórico de estos avisos desde la propia aplicación.

- **AlertCops:** Desarrollada por el Ministerio del Interior, es una de las aplicaciones más completas en cuanto a reporte ciudadano. Permite compartir la ubicación en tiempo real con contactos de emergencia o con los cuerpos de seguridad, activar un botón SOS para enviar un aviso inmediato acompañado de un audio de 10 segundos, e iniciar un chat con los servicios de emergencia. El usuario puede reportar una gran variedad de situaciones como robos, agresiones, violencia de género, delitos de odio, ocupaciones ilegales o desapariciones, entre otros. Además, se pueden adjuntar fotos, vídeos y especificar si la emergencia afecta al propio usuario o a un tercero.

- **Zello:** Esta aplicación transforma el dispositivo móvil en un walkie-talkie, permitiendo la comunicación por voz en tiempo real. Ha sido especialmente útil en catástrofes naturales como huracanes, donde voluntarios y ciudadanos han utilizado esta herramienta para coordinar esfuerzos de ayuda. Aunque no está orientada específicamente al reporte formal de emergencias, su facilidad de uso y rapidez la convierten en una opción efectiva para la colaboración ciudadana.

- **Disaster Alert:** Esta aplicación ofrece un mapa interactivo global con alertas en tiempo real sobre fenómenos naturales como terremotos, ciclones, tsunamis o erupciones volcánicas. Su enfoque es principalmente informativo, por lo que no permite al usuario reportar emergencias directamente, pero resulta útil para estar al tanto de amenazas activas a nivel mundial.

Tras el análisis de estas herramientas, se concluye que, aunque existen aplicaciones con funcionalidades útiles y específicas, ninguna de ellas ofrece una solución completa que combine el reporte ciudadano en tiempo real con la participación activa de expertos y autoridades dentro de un entorno colaborativo. En muchos casos, las funcionalidades están fragmentadas, enfocadas exclusivamente en la comunicación con cuerpos de seguridad o centradas únicamente en la difusión de información.

Esta carencia evidencia la necesidad de una aplicación como la propuesta en este proyecto, que permita una respuesta más ágil y coordinada en situaciones de emergencia. Casos recientes como el de la DANA en España han demostrado que la falta de comunicación, coordinación y alertas eficaces puede agravar considerablemente los efectos de un desastre. Por ello, dicho trabajo se plantea como una contribución para mejorar la gestión de emergencias, incorporando geolocalización, clasificación de alertas, participación de distintos roles y recomendaciones contextuales que ayuden a reducir riesgos y salvar vidas.

1.4 Metodología

Para el desarrollo del proyecto se ha seguido una metodología ágil, con el objetivo de avanzar de forma incremental, permitiendo realizar mejoras y ajustes continuos a lo largo del proceso. Este enfoque ha favorecido una mayor flexibilidad ante posibles cambios en los requisitos y ha facilitado la incorporación progresiva de nuevas funcionalidades. Esta forma de trabajo ha permitido detectar errores tempranamente y adaptar el desarrollo a las necesidades que iban surgiendo.

Inicialmente, se realizó una especificación de los requisitos funcionales y no funcionales que debía cumplir la aplicación. Posteriormente, se elaboró el script SQL para la creación de la base de datos, teniendo en cuenta las entidades necesarias y sus relaciones.
Una vez definida la estructura de datos, se procedió al desarrollo del backend, creando los endpoints necesarios para la lógica del sistema y la comunicación con la base de datos. A continuación, se inició el desarrollo del frontend, centrado en diseñar una interfaz gráfica intuitiva y accesible para el usuario final.

Durante todo el proceso se ha trabajado de forma incremental, realizando múltiples iteraciones sobre las distintas partes del sistema. Tanto el backend como la base de datos han sido objeto de modificaciones y ampliaciones a medida que se identificaban nuevas necesidades o se optimizaban funcionalidades. Esta evolución continua ha sido clave para mantener la coherencia entre los distintos componentes de la aplicación.
Finalmente, se procedió al despliegue de la aplicación, abarcando la base de datos, el backend y el frontend, lo que permite su utilización desde cualquier dispositivo Android.

Gracias al enfoque ágil adoptado, ha sido posible desarrollar una aplicación completa, robusta y adaptada a los requisitos definidos inicialmente, así como a los que surgieron durante su implementación.

1.5 Estructura de la memoria

A continuación, se describe la estructura de la memoria con el objetivo de proporcionar una visión clara y ordenada del proceso seguido para el desarrollo del proyecto:

- ***Capítulo 2: Tecnologías y herramientas utilizadas.*** En este capítulo se presentan las tecnologías y herramientas empleadas en las distintas fases del proyecto, abarcando tanto la base de datos como el backend y el frontend.

- ***Capítulo 3: Especificación y análisis.*** Se detallan los requisitos de la aplicación, tanto funcionales como no funcionales, así como los diagramas UML correspondientes, con el fin de representar el comportamiento del sistema.

- ***Capítulo 4: Modelado y diseño del sistema.*** Este apartado recoge el diseño completo de la aplicación, incluyendo el modelo Entidad-Relación de la base de datos, la arquitectura general del sistema y el diseño de la interfaz de usuario.

- **Capítulo 5: Implementación y pruebas.** Se expone el proceso de desarrollo del proyecto, acompañado de los casos de prueba utilizados para verificar el cumplimiento de los requisitos previamente definidos.

- ***Capítulo 6: Despliegue de la aplicación.*** En esta parte se muestra cómo se ha desplegado la aplicación y todos los pasos por los que se ha pasado para llegar finalmente al despliegue completo.

- ***Capítulo 7: Conclusiones y trabajo futuro.*** Se presentan los principales objetivos alcanzados con el desarrollo del proyecto, así como posibles líneas de mejora y futuras ampliaciones de la aplicación.

2

Tecnologías y Herramientas utilizadas

En este capítulo se describen las principales tecnologías, frameworks y herramientas empleadas durante el desarrollo del proyecto, elegidad con el objetivo de construir una solución robusta, moderna y adecuada a los requerimientos del sistema.

2.1 Tecnologías del Frontend

- ***React Native:*** Es un framework de desarrollo de aplicaciones móviles de código abierto creado por Meta que permite construir aplicaciones para Android e iOS utilizando un único código en JavaScript o TypeScript. Se basa en React y emplea componentes nativos, lo que garantiza un buen rendimiento y una experiencia fluida. Entre sus principales ventajas destacan la compatibilidad multiplataforma, el desarrollo eficiente, la integración con librerías de terceros y funciones como Hot Reloading, que facilita la visualización de cambios en tiempo real. Además, cuenta con una gran comunidad y una extensa documentación, lo que simplifica el

mantenimiento y la resolución de problemas, convirtiéndose en una opción ideal para el desarrollo de aplicaciones modernas.

- ***Expo:*** Es un framework y plataforma de desarrollo para aplicaciones móviles basado en React Native, que simplifica la creación y gestión de apps sin necesidad de configurar entornos nativos complejos. Permite desarrollar, probar y desplegar aplicaciones de manera rápida gracias a herramientas como Expo Go, que facilita la previsualización en tiempo real, y Expo SDK, que proporciona múltiples APIs y módulos listos para usar, como notificaciones push, acceso a la cámara y geolocalización. Entre sus principales ventajas destacan su facilidad de uso, la eliminación de configuraciones nativas, la compatibilidad multiplataforma y su integración con Expo Application Services (EAS) para la compilación y publicación de aplicaciones sin necesidad de un entorno nativo avanzado.

- ***Expo Location:*** Es un módulo de Expo que permite acceder a la ubicación del dispositivo de manera sencilla en aplicaciones desarrolladas con React Native. Proporciona funcionalidades como la obtención de la ubicación actual, el seguimiento en tiempo real, la geocodificación (convertir coordenadas en direcciones) y la configuración de alertas basadas en la ubicación. Sus principales ventajas incluyen su facilidad de uso, ya que evita configuraciones nativas complejas, su compatibilidad con Android e iOS, y su integración con Expo, lo que permite probar y desarrollar sin necesidad de modificar código nativo. Además, ofrece opciones de optimización de consumo de batería, permitiendo definir la precisión y frecuencia de actualización de la ubicación.

- ***React Navigation:*** Es una biblioteca de navegación para aplicaciones desarrolladas con React Native, que permite gestionar la transición entre pantallas de manera fluida e intuitiva. Ofrece diferentes tipos de navegación, como pilas (stack), pestañas (tab) y cajones laterales (drawer), adaptándose a distintas necesidades de la aplicación. Sus principales ventajas incluyen su facilidad de implementación, su alta personalización, su compatibilidad con Android e iOS, y su integración con el ecosistema de React Native. Además, soporta navegación basada en gestos, mejora la experiencia del usuario y permite la gestión eficiente del estado de las pantallas, optimizando el rendimiento de la aplicación.

- ***Axios:*** Es una biblioteca de JavaScript utilizada para realizar peticiones HTTP en aplicaciones web y móviles, permitiendo la comunicación con APIs y servidores de manera eficiente. Facilita la obtención y envío de

datos, soporta promesas y async/await, y permite gestionar respuestas y errores de forma sencilla. Sus principales ventajas incluyen su facilidad de uso, compatibilidad con JSON y otros formatos, soporte para interceptores (modificación de solicitudes y respuestas), cancelación de peticiones y manejo automático de cabeceras y tokens de autenticación. Además, es compatible con React Native, mejorando la gestión de datos y optimizando la interacción con el backend.

- **Firebase Authentication:** Es un servicio de autenticación de Firebase que facilita la gestión de usuarios en aplicaciones web y móviles. Permite autenticar usuarios mediante correo y contraseña, números de teléfono y proveedores externos como Google, simplificando la implementación de sistemas de inicio de sesión. Entre sus ventajas destacan su facilidad de integración, su alta seguridad mediante autenticación basada en tokens y compatibilidad con OAuth 2.0 y JWT. Además, ofrece gestión automática de sesiones, compatibilidad con React Native y sincronización con otros servicios de Firebase, facilitando el desarrollo de aplicaciones seguras y escalables.

- ***Cloudinary:*** Es una plataforma en la nube especializada en la gestión de archivos multimedia, que permite almacenar, optimizar y entregar imágenes y vídeos de forma eficiente en aplicaciones web y móviles. En el contexto del proyecto, se ha utilizado para subir y gestionar las imágenes de los símbolos de las emergencias. Entre sus principales ventajas destacan su facilidad de integración con frameworks como React Native, su capacidad para transformar y redimensionar imágenes dinámicamente mediante URLs, y su alto rendimiento gracias a una red de entrega de contenidos (CDN). Además, Cloudinary proporciona un panel de control intuitivo, compatibilidad con múltiples formatos y una API RESTful que permite automatizar tareas relacionadas con la gestión multimedia, lo que contribuye a una experiencia de usuario más ágil y optimizada.

- ***React Native Elements:*** Es una biblioteca de componentes UI de código abierto diseñada para React Native, que proporciona una colección de elementos preconstruidos y personalizables como botones, formularios, íconos, tarjetas y más. Permite a los desarrolladores crear interfaces de usuario atractivas y consistentes con menos esfuerzo, aprovechando sus componentes listos para usar y su estilo predeterminado adaptable. Entre sus ventajas destacan su facilidad de integración, la personalización sencilla a través de props y estilos, su compatibilidad multiplataforma (Android e iOS), y el hecho de que está optimizada para rendimiento y accesibilidad.

- ***React Native Maps:*** Es una biblioteca de código abierto que permite integrar mapas interactivos en aplicaciones móviles desarrolladas con React Native. Utiliza Google Maps o Apple Maps dependiendo de la plataforma, y permite agregar funcionalidades como marcadores, rutas, zoom, y geolocalización. Esta herramienta facilita la visualización de ubicaciones y la interacción con mapas, ideal para aplicaciones que requieren información geoespacial. Entre sus ventajas destacan su compatibilidad multiplataforma, su alta personalización, la facilidad de integración con otras librerías y su rendimiento optimizado, permitiendo crear experiencias de usuario fluidas y dinámicas

- ***React Native Safe Area Context:*** Es una biblioteca de código abierto que permite gestionar de manera eficiente las áreas seguras de las pantallas en dispositivos con notch, bordes redondeados o pantallas con barras de estado y navegación. Permite a los desarrolladores crear interfaces que se adaptan a la geometría del dispositivo, asegurando que los elementos de la interfaz no se superpongan con áreas no visibles o interactivas. Entre sus ventajas se incluyen su compatibilidad con dispositivos modernos, su sencilla integración con otras librerías, y la capacidad de gestionar dinámicamente los márgenes y rellenos, mejorando la experiencia de usuario al garantizar que los elementos importantes se mantengan dentro de áreas visibles y accesibles en todas las plataformas.

- ***EAS (Expo Application Services):*** Es un conjunto de servicios proporcionados por Expo que permite compilar, actualizar y distribuir aplicaciones móviles de forma eficiente. En este proyecto se ha utilizado EAS Build para generar el archivo APK del frontend de la aplicación y facilitar su instalación en dispositivos Android, así como EAS Update para subir nuevas versiones sin necesidad de recompilar completamente. Entre sus ventajas destacan la posibilidad de compilar aplicaciones en la nube sin depender de entornos locales complejos, el soporte nativo para proyectos con dependencias personalizadas, y la integración directa con Expo y GitHub.

2.2 Tecnologías del Backend

- ***Spring Boot:*** Es un framework basado en Spring que simplifica el desarrollo de aplicaciones Java, especialmente en el ámbito de servicios web y microservicios. Permite crear aplicaciones independientes y de producción con configuración mínima, ya que viene con una configuración

predeterminada y una estructura de proyecto lista para usarse. Entre sus ventajas destacan su facilidad de uso, su alta integración con herramientas y librerías de Java, su capacidad para crear aplicaciones rápidas y escalables, y la facilidad para implementar y configurar APIs RESTful.

- ***Lombok:*** Es una biblioteca de Java que facilita la escritura de código mediante la generación automática de métodos comunes como getters, setters, constructores, y métodos toString, entre otros, mediante anotaciones. Permite reducir el código boilerplate, simplificando la gestión de clases y mejorando la legibilidad del código. Entre sus ventajas destacan su facilidad de integración con proyectos existentes, su ahorro de tiempo al evitar la escritura repetitiva de métodos estándar y su compatibilidad con diversas herramientas y frameworks, como Spring. Lombok mejora la productividad del desarrollador al generar automáticamente el código necesario, manteniendo la clase limpia y fácil de mantener.

- ***Spring Data JPA:*** Es una parte del framework Spring que facilita el acceso a bases de datos utilizando Java Persistence API (JPA). Permite la creación de repositorios para interactuar con bases de datos de manera sencilla, sin necesidad de escribir código SQL complejo, gracias a la generación automática de consultas basadas en los métodos del repositorio. Entre sus ventajas se incluyen su integración transparente con Spring y su capacidad para manejar operaciones CRUD de manera eficiente, su optimización del acceso a datos, y su compatibilidad con diversos motores de bases de datos.

- ***CORS (cross-origin resource sharing):*** Es un mecanismo de seguridad que permite o restringe el acceso a recursos web ubicados en un dominio diferente al que realiza la solicitud. Se implementa mediante cabeceras HTTP que indican a los navegadores si deben permitir que las solicitudes de un origen distinto accedan a los recursos del servidor.

- ***Railway:*** Es una plataforma de desarrollo en la nube que facilita el despliegue, gestión y escalado de aplicaciones y bases de datos. En este proyecto se ha utilizado para desplegar el backend desarrollado con Spring Boot y la base de datos MySQL, permitiendo un entorno de desarrollo y producción centralizado y accesible. Entre sus principales ventajas destacan su interfaz intuitiva, la facilidad para conectar servicios mediante plantillas preconfiguradas y su integración continua con repositorios como GitHub. Además, Railway permite desplegar con un solo clic, ofrece herramientas de monitoreo y logs en tiempo real, y simplifica la

configuración de variables de entorno, lo que la convierte en una solución eficiente para gestionar infraestructuras backend de manera ágil y sin complicaciones.

2.3 Base de Datos

- ***MySQL:*** Es un sistema de gestión de bases de datos relacional de código abierto, ampliamente utilizado para almacenar y gestionar grandes volúmenes de datos en aplicaciones web y móviles. Utiliza el lenguaje SQL (Structured Query Language) para realizar operaciones de consulta, actualización y gestión de datos en tablas interrelacionadas. Permite crear bases de datos, definir relaciones entre ellas y realizar transacciones de manera eficiente. Entre sus ventajas se incluyen su alta fiabilidad, rendimiento optimizado, facilidad de uso, su comunidad activa y su compatibilidad con diversas plataformas.

2.4 Herramientas y Software utilizados

- ***IntelliJ IDEA:*** Es un entorno de desarrollo integrado (IDE) de JetBrains que está diseñado principalmente para la programación en Java. Ofrece una amplia gama de herramientas, como autocompletado de código, depuración avanzada, refactorización y soporte para frameworks populares. Ha sido el entorno de desarrollo utilizado para el Backend.

- ***Visual Studio Code:*** Es un editor de código fuente de código abierto y multiplataforma desarrollado por Microsoft, diseñado para soportar una gran variedad de lenguajes de programación. Ofrece características como autocompletado inteligente, depuración, integración con control de versiones y una amplia gama de extensiones para personalizar el entorno. Ha sido utilizado para desarrollar el Frontend.

- ***Android Studio:*** Es el entorno de desarrollo integrado (IDE) oficial para la creación de aplicaciones Android, basado en IntelliJ IDEA. Ofrece herramientas avanzadas para desarrollar, depurar y optimizar aplicaciones, destacando su potente sistema de emuladores Android que permite simular dispositivos Android en diferentes configuraciones, tamaños de pantalla y versiones del sistema operativo. Estos emuladores son esenciales para probar las aplicaciones sin necesidad de un dispositivo físico, facilitando el desarrollo y las pruebas en diversas condiciones. Ha sido utilizado para probar la aplicación.

- ***MySQL Workbench:*** Es una herramienta visual de administración y desarrollo para bases de datos MySQL. Proporciona una interfaz gráfica para diseñar, modelar y gestionar bases de datos, permitiendo la creación de esquemas, tablas y consultas SQL de manera eficiente. Ha sido utilizado para crear y gestionar la base de datos.

- ***Expo Go:*** Es una aplicación móvil que permite ejecutar proyectos desarrollados con Expo de manera rápida y sencilla en dispositivos Android e iOS. Funciona como un visor en tiempo real para aplicaciones React Native, permitiendo a los desarrolladores ver y probar sus aplicaciones directamente desde su dispositivo sin necesidad de realizar una compilación. Ha sido utilizado para probar la aplicación en el emulador.

- ***Postman:*** Es una herramienta de desarrollo utilizada para probar, depurar y documentar APIs de manera eficiente. Permite enviar solicitudes HTTP a servidores y visualizar las respuestas, facilitando el proceso de desarrollo y pruebas de servicios web. Se ha utilizado para hacer pruebas a los endpoints del Backend.

- ***GitHub Desktop:*** Es una aplicación de escritorio que facilita la gestión de repositorios Git en el entorno de GitHub sin necesidad de utilizar la línea de comandos. Permite realizar acciones comunes como commit, push, pull y gestionar ramas de manera visual, lo que simplifica el control de versiones y la colaboración en proyectos de desarrollo. Ha sido utilizado para subir los cambios realizado a lo largo del proyecto.

- ***Trello:*** Es una herramienta de gestión de proyectos basada en tableros visuales, que permite organizar tareas mediante listas y tarjetas. Es utilizada para planificar, asignar y seguir el progreso de proyectos facilitando la organización y el seguimiento de tareas. Ha sido utilizado para gestionar y planificar el proyecto.

- ***Canva:*** Es una herramienta de diseño gráfico en línea que permite crear contenido visual de forma sencilla e intuitiva, sin necesidad de conocimientos avanzados en diseño. En este proyecto se ha utilizado para diseñar los símbolos representativos de los diferentes tipos de emergencia, asegurando una comunicación visual clara y coherente en la aplicación. Entre sus ventajas destacan su interfaz amigable, una amplia biblioteca de plantillas e iconos personalizables, y la posibilidad de exportar los diseños en distintos formatos. Además, Canva facilita el trabajo colaborativo, permite mantener una identidad visual consistente y optimiza el proceso

de creación de elementos gráficos, lo que contribuye a una mejor experiencia de usuario dentro de la aplicación.

- ***Visual Paradigm:*** Es una herramienta de modelado y diseño de software que permite crear diagramas UML, modelos de comportamiento y otros elementos visuales para apoyar el análisis y desarrollo de sistemas. En este proyecto se ha utilizado para elaborar el modelo de comportamiento de la aplicación. Entre sus ventajas destacan su interfaz intuitiva, la posibilidad de generar documentación automáticamente, su compatibilidad con estándares UML y su utilidad tanto en fases de análisis como de diseño. Además, Visual Paradigm facilita una visión clara y estructurada del comportamiento del sistema, lo que contribuye a una mejor comprensión del funcionamiento de la aplicación por parte de todos los implicados en el desarrollo.

3

Especificación y análisis

En este capítulo se presenta una visión general del funcionamiento de la aplicación, detallando sus principales características, además de la especificación y análisis del sistema desarrollado. Para ello, se introducen los diferentes actores que interactúan con la aplicación y se definen los requisitos que guían su diseño. Asimismo, se modela el comportamiento del sistema a través de distintos diagramas que permiten representar de forma estructurada las interacciones y procesos claves de la aplicación.

3.1 Descripción general de la aplicación

Tras el estudio de las funcionalidades que ofrecen aplicaciones similares, se plantearon las siguientes características generales que la herramienta debía poseer. La aplicación desarrollada pretende facilitar la gestión de emergencias mediante una interfaz intuitiva y funcionalidades diseñadas para mejorar la comunicación, colaboración y toma de decisiones en situaciones críticas.

3.1.1 Acceso a la aplicación

El acceso a la aplicación se gestiona de manera sencilla y segura a través de Firebase Authentication. Los usuarios pueden iniciar sesión en la aplicación si ya disponen de una cuenta registrada, introduciendo su correo electrónico y

contraseña. En caso de que un usuario no tenga una cuenta, puede registrarse fácilmente proporcionando su correo electrónico y eligiendo una contraseña para crear su perfil.

Además, la aplicación ofrece la funcionalidad de recuperación de contraseña para aquellos usuarios que la hayan olvidado pudiéndose solicitar ingresando su correo electrónico registrado. A continuación, recibirá un correo electrónico con un enlace para establecer una nueva contraseña, garantizando que el proceso sea rápido y seguro.

Firebase Authentication se encarga de la gestión de estos procesos de inicio de sesión, registro y recuperación de contraseña, asegurando la confidencialidad y seguridad de los datos de los usuarios.

3.1.2 Mapa Interactivo

El mapa interactivo es uno de los elementos clave de la aplicación, diseñado para ofrecer una representación visual y geolocalizada de las emergencias reportadas. Este componente permite a los usuarios obtener una visión clara y precisa de las situaciones de riesgo en su entorno o en áreas específicas seleccionadas.

El mapa se centra inicialmente en la ubicación actual del usuario, lo que garantiza que la información mostrada sea relevante para su seguridad personal. Cada emergencia se marca en el mapa con una forma y color asociado a su nivel de riesgo (detallado en la sección 1.4.2) y con un dibujo distintivo que representa su tipo (detallado en la sección 1.4.3). Esta codificación visual facilita la rápida identificación y comprensión de la situación, especialmente en contextos de alta presión donde el tiempo es un factor crítico.

Para aumentar la confianza y credibilidad en la información presentada, las emergencias que han sido verificadas por expertos o autoridades aparecen en el mapa con un efecto de pulso distintivo. Este diseño no solo resalta la importancia de estas alertas, sino que también permite a los usuarios priorizar aquellas que han sido confirmadas como auténticas y relevantes.

El mapa también está diseñado para ser dinámico e interactivo, permitiendo al usuario seleccionar cualquier emergencia marcada para obtener información detallada

3.1.3 Emergencias

A través del mapa interactivo, los usuarios pueden visualizar y acceder a los detalles de cualquier emergencia simplemente seleccionando la emergencia en la interfaz. Al pulsar sobre una emergencia, se abre una nueva pantalla que incluye los siguientes detalles clave:

- Fecha y hora de creación: Proporciona un registro temporal que ayuda a evaluar la actualidad de la emergencia.
- Descripción: Una breve explicación expuesta como título de la emergencia que detalla la naturaleza de la emergencia.
- Recomendaciones de seguridad: Sugerencias específicas añadidas diseñadas para orientar a los ciudadanos sobre las mejores prácticas para protegerse.
- Nivel de riesgo: Clasificación de la emergencia según su gravedad (bajo, moderado, alto o extremo), representada mediante colores y formas para facilitar su identificación.
- Tipo de emergencia: Categoría específica que indica la naturaleza del evento.
- Localización exacta: Localización exacta del incidente.
- Estado de verificación: Indica si la emergencia ha sido validada por expertos o autoridades.
- Notificaciones: Incluye las notificaciones que se han generado para esa emergencia para conseguir un mayor seguimiento sobre ella. De cada notificación se incluye la fecha exacta en que se notificó, el título y una breve descripción.

Además, los expertos o autoridades tienen la capacidad de añadir nuevas recomendaciones después de la creación de una emergencia. Estas actualizaciones garantizan que la información siga siendo útil y relevante, especialmente en situaciones que cambian rápidamente. También pueden actualizar el nivel de riesgo para reflejar con precisión el estado actual de la situación. Y, además, si una emergencia reportada inicialmente por un ciudadano no está verificada, los expertos o autoridades pueden confirmar su autenticidad y actualizar su verificación, lo que incrementa la confianza en la información compartida.

Además, los ciudadanos pueden enviar solicitudes de edición de emergencias, solicitando un cambio en sus recomendaciones y/o en su nivel de riesgo, ayudando así a los expertos y autoridades, los cuales serán los que acepten o denieguen dichas solicitudes.

La funcionalidad de creación de emergencias está disponible para todos los usuarios, promoviendo la participación ciudadana en la identificación de riesgos. Durante el proceso de creación, el usuario puede incluir los siguientes elementos:

- Descripción: Explicación del evento que facilite su identificación por otros usuarios.
- Nivel de riesgo: Clasificación inicial de la gravedad del evento.
- Tipo de emergencia: Categoría específica que describa la naturaleza del incidente.
- Ubicación: Punto exacto en el mapa donde ocurre la emergencia.

No obstante, las emergencias creadas por ciudadanos se marcan inicialmente como no verificadas hasta que un experto o autoridad correspondiente valide su autenticidad. Por otro lado, cuando un experto o autoridad crea una emergencia, este puede incluir desde el inicio el estado de verificación. Este enfoque asegura que las alertas más críticas y confiables se destaquen rápidamente en el sistema.

Además, la aplicación permite que tanto las autoridades como los expertos marquen una emergencia como resuelta, lo que provoca su eliminación del mapa. Esta funcionalidad es fundamental para garantizar que el mapa interactivo refleje únicamente emergencias vigentes y relevantes.
Por otro lado, para mejorar la fiabilidad del sistema, los ciudadanos no pueden marcar directamente una emergencia como resuelta, pero sí pueden reportarla en caso de considerar que ha sido solucionada. Si una emergencia recibe cinco reportes de distintos usuarios, el sistema la marcará automáticamente como resuelta y se eliminará del mapa. Este proceso contribuye a mantener una visualización más clara y actualizada de las situaciones en tiempo real.

Así, la colaboración ciudadana se convierte en un mecanismo adicional para asegurar que el sistema se mantenga actualizado y eficiente, promoviendo la participación activa en la gestión de emergencias.

3.1.3.1 Niveles de Emergencia

La clasificación en distintos niveles es fundamental para priorizar la atención y garantizar una respuesta adecuada. Por ello, se establecen cuatro niveles diferenciados representados por colores y formas.
Este enfoque visual facilita la identificación rápida de la gravedad de la situación y permite una toma de decisiones más efectiva.

Concretamente los niveles definidos son:

- ***Bajo:*** Este nivel indica un riesgo mínimo o controlado. No representan una amenaza inmediata para las personas ni para los bienes materiales, buscan informar sin generar alarma. Se representa con el color verde y la forma de un círculo.

- ***Moderado:*** Corresponde a situaciones que presentan un riesgo intermedio, con la posibilidad de convertirse en un problema mayor si no se gestionan adecuadamente. Requiere precaución, pero no necesariamente una intervención urgente. Se representa con el color amarillo y la forma de un triángulo.

- ***Alto:*** Señala emergencias que ya afectan significativamente a personas, propiedades o el entorno. Requieren una acción inmediata por parte de las autoridades y una mayor precaución por parte de los ciudadanos. Se representa con el color naranja y la forma de un rombo.

- ***Extremo:*** Representa el nivel más crítico de riesgo. Las emergencias en este nivel suponen un peligro inminente para la vida y la seguridad de la población. Demandan una respuesta inmediata y coordinada. Se representa con el color rojo y la forma de un octágono.

3.1.3.2 Tipos de Emergencia

La aplicación distingue diversos tipos de emergencias para optimizar la creación y difusión de alertas. Esta clasificación se basa en las características y el impacto potencial de cada tipo de emergencia, permitiendo a los usuarios identificar rápidamente la naturaleza de la situación. A continuación, se detallan las categorías principales y sus subtipos, por lo tanto, cada subtipo posee 4 símbolos que representan los diferentes niveles (bajo, moderado, alto y extremo) para esa emergencia.

1. **Emergencias naturales:**

Estas emergencias derivan de fenómenos naturales que pueden causar daños significativos a las personas, el medio ambiente o la infraestructura.

- ***Inundación:*** Se produce por la acumulación de agua en áreas habitadas debido a lluvias intensas o desbordamientos de ríos. Se pueden ver los distintos niveles en la figura 1, en el cual el primero representa nivel Bajo, el segundo nivel Moderado, el tercero nivel Alto y el cuarto nivel Extremo.

Figura 1. Símbolos de Inundación

- ***Incendio forestal:*** Incendios que afectan áreas boscosas, representando un peligro para la biodiversidad y las comunidades cercanas. Se pueden ver los distintos niveles en la figura 2.

Figura 2: Símbolos de Incendio forestal

- ***Terremoto:*** Movimientos sísmicos que pueden provocar daños estructurales y riesgos colaterales como deslizamientos de tierra. Se pueden ver los distintos niveles en la figura 3.

Figura 3: Símbolos de Terremoto

- ***Tsunami:*** Ola gigante generada por terremotos o deslizamientos submarinos, que representan un alto riesgo para zonas costeras. Se pueden ver los distintos niveles en la figura 4.

Figura 4: Símbolos de Tsunami

- ***Tornado:*** Fenómeno de viento intenso con forma de espiral que pueden destruir edificios y causar víctimas. Se pueden ver los distintos niveles en la figura 5.

Figura 5: Símbolos de Tornado

- ***Huracán:*** Tormenta tropical con vientos y lluvias extremas, capaces de devastar grandes áreas. Se pueden ver los distintos niveles en la figura 6.

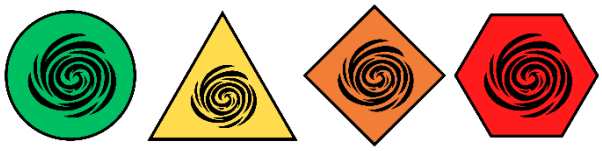

Figura 6: Símbolos de Huracán

- **Tormenta:** Evento meteorológico con fuertes vientos, lluvias, y en ocasiones granizo o rayos. Se pueden ver los distintos niveles en la figura 7.

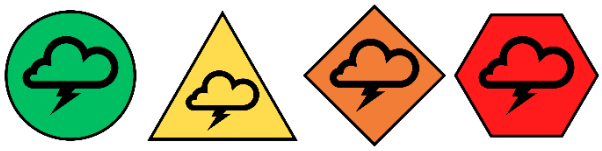

Figura 7: Símbolos de Tormenta

- ***Deslizamientos de Tierra:*** Movimientos de masa terrestre que pueden sepultar caminos, viviendas o zonas agrícolas. Se pueden ver los distintos niveles en la figura 8.

Figura 8: Símbolos de Deslizamientos de Tierra

- ***Erupción Volcánica:*** Expulsión de magma, ceniza y gases volcánicos que amenazan a comunidades cercanas. Se pueden ver los distintos niveles en la figura 9.

Figura 9: Símbolos de Erupción Volcánica

2. **Emergencias Meteorológicas**

Situaciones relacionadas con eventos climáticos extremos que afectan la seguridad de las personas y la infraestructura.

- ***Ola de calor:*** Período prolongado de temperaturas anormalmente altas, con riesgo de golpes de calor y sequías. Se pueden ver los distintos niveles en la figura 10.

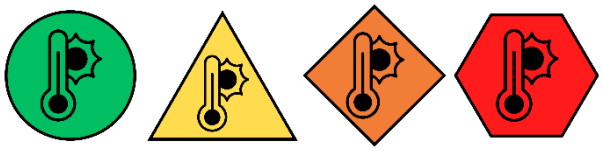

Figura 10: Símbolos de Ola de calor

- ***Ola de frío:*** Descenso extremo de las temperaturas, que puede causar hipotermia y afectar a la actividad cotidiana. Se pueden ver los distintos niveles en la figura 11.

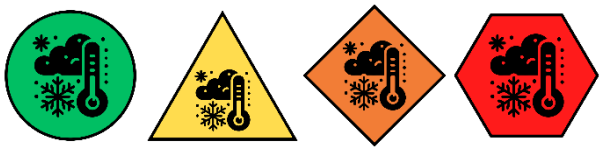

Figura 11: Símbolos de Ola de frío

- ***Granizada:*** Fenómeno donde caen piedras de hielo, que pueden dañar cultivos, vehículos y edificios. Se pueden ver los distintos niveles en la figura 12.

Figura 12: Símbolos de Granizada

- ***Nevada:*** Fuertes acumulaciones de nieve que dificultan el transporte y aumentan el riesgo de accidentes. Se pueden ver los distintos niveles en la figura 13.

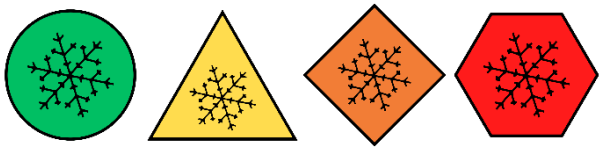

Figura 13: Símbolos de Nevada

3. Emergencias Urbanas

Problemas originados en áreas habitadas, generalmente relacionados con la infraestructura o servicios básicos.

- ***Derrumbe de edificio:*** Colapso de estructuras por fallos constructivos o eventos externos como sismos. Se pueden ver los distintos niveles en la figura 14.

Figura 14: Símbolos de Derrumbe de edificio

- ***Bloqueo de carretera:*** Obstáculos en vías de comunicación que dificultan el tránsito, como deslizamientos o accidentes. Se pueden ver los distintos niveles en la figura 15.

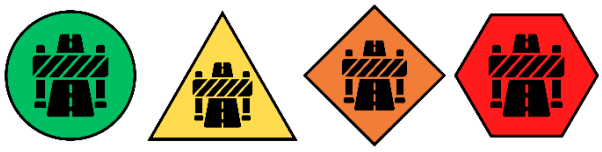

Figura 15: Símbolos de Bloqueo de carretera

- ***Fuga de gas:*** Escape de gases peligrosos que representan riesgos de explosiones o intoxicaciones. Se pueden ver los distintos niveles en la figura 16.

Figura 16: Símbolos de Fuga de gas

- ***Corte de energía:*** Fallo en el suministro eléctrico que afecta la vida diaria y la seguridad de la población. Se pueden ver los distintos niveles en la figura 17.

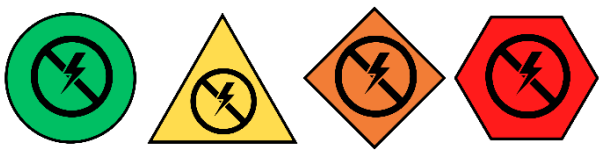

Figura 17: Símbolos de Corte de energía

4. Emergencias Sanitarias

Eventos que ponen en riesgo la salud pública.

- ***Epidemia:*** Propagación rápida de enfermedades infecciosas. Se pueden ver los distintos niveles en la figura 18.

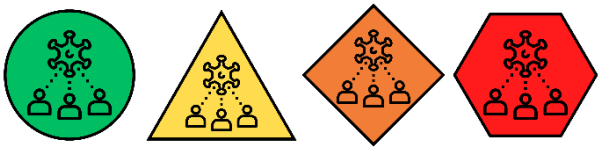

Figura 18: Símbolos de Epidemia

- ***Brote de enfermedad:*** Aparición repentina de casos concentrados de una enfermedad. Se pueden ver los distintos niveles en la figura 19.

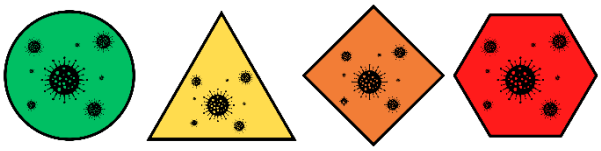

Figura 19: Símbolos de Brote de enfermedad

- ***Intoxicación:*** Consumo o exposición a sustancias tóxicas que afectan a un gran número de personas. Se pueden ver los distintos niveles en la figura 20.

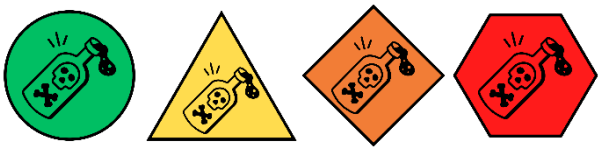

Figura 20: Símbolos de Intoxicación

5. **Emergencias Sociales y Humanas**

Situaciones que implican riesgos para las personas debido a conflictos sociales o migraciones forzadas.

- ***Protesta violenta:*** Manifestaciones que degeneran en violencia, con riesgo para la seguridad pública. Se pueden ver los distintos niveles en la figura 21.

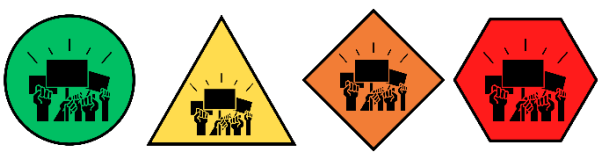

Figura 21: Símbolos de Protesta violenta

- ***Evacuación:*** Desplazamiento de personas de áreas peligrosas debido a emergencias. Se pueden ver los distintos niveles en la figura 22.

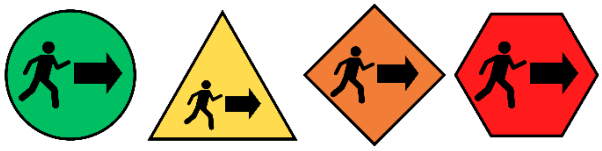

Figura 22: Símbolos de Evacuación

- ***Refugiados:*** Personas desplazadas que buscan seguridad debido a conflictos o desastres naturales. Se pueden ver los distintos niveles en la figura 23.

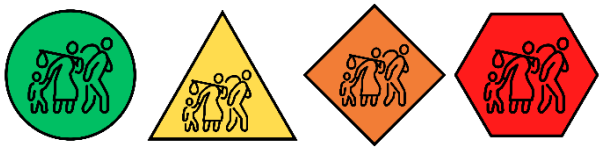

Figura 23: Símbolos de Refugiados

6. Emergencias Tecnológicas

Problemas derivados del uso o fallo de tecnología e infraestructuras industriales.

- ***Fuga química:*** Escape de sustancias peligrosas que contaminan el ambiente y afectan la salud. Se pueden ver los distintos niveles en la figura 24.

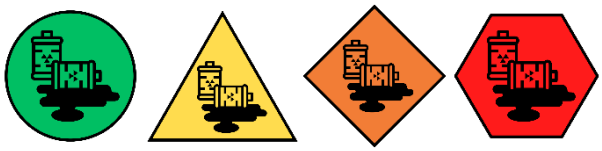

Figura 24: Símbolos de Fuga química

- ***Explosión industrial:*** Accidente en instalaciones industriales que provocan daños significativos. Se pueden ver los distintos niveles en la figura 25.

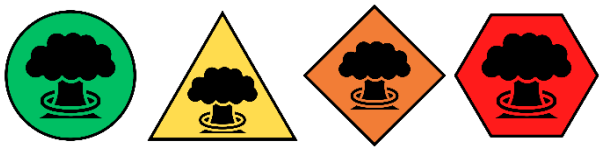

Figura 25: Símbolos de Explosión industrial

- ***Colapso de infraestructura:*** Derrumbe de puentes, carreteras o edificaciones críticas. Se pueden ver los distintos niveles en la figura 26.

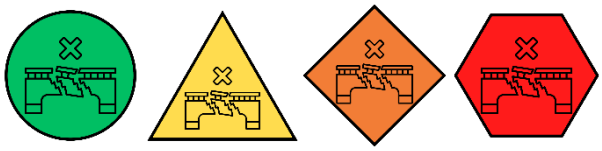

Figura 26: Símbolos de Colapso de infraestructuras

7. Emergencias de Transporte

Incidentes relacionados con sistemas de transporte terrestre, aéreo o marítimo.

- ***Accidente de coche:*** Colisión entre vehículos. Se pueden ver los distintos niveles en la figura 27.

Figura 27: Símbolos de Accidente de coche

- ***Accidente aéreo:*** Emergencias relacionadas con aviones, como accidentes o fallos en vuelo. Se pueden ver los distintos niveles en la figura 28.

Figura 28: Símbolos de Accidente aéreo

- ***Accidente ferroviario:*** Descarrilamientos o colisiones de trenes. Se pueden ver los distintos niveles en la figura 29.

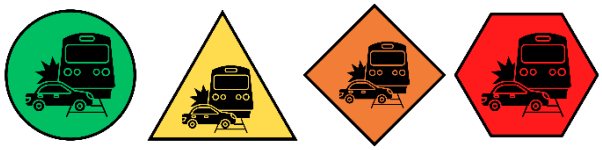

Figura 29: Símbolos de Accidente ferroviario

- ***Hundimiento marítimo:*** Accidentes en embarcaciones. Se pueden ver los distintos niveles en la figura 30.

Figura 30: Símbolos de Hundimiento marítimo

8. **Emergencias Animales**

Eventos que involucran fauna y representan un riesgo directo o indirecto para las personas.

- ***Ataque de fauna peligrosa:*** Agresiones de animales como serpientes, osos o grandes felinos. Se pueden ver los distintos niveles en la figura 31.

Figura 31: Símbolos de Ataque de fauna peligrosa

- ***Plagas de animales:*** Proliferación de insectos o roedores que causan daños a la salud y a los cultivos. Se pueden ver los distintos niveles en la figura 32.

Figura 32: Símbolos de Plaga de animales

3.1.4 Notificaciones

El sistema de notificaciones de la aplicación está diseñado para garantizar que los usuarios reciban alertas rápidas y relevantes sobre las emergencias en su entorno. Las notificaciones se envían directamente al sistema del teléfono del usuario como notificaciones push, asegurando así su visibilidad inmediata. La recepción de estas notificaciones depende del radio de notificación configurado por cada usuario. Si una emergencia ocurre dentro de su radio establecido, el usuario recibirá una notificación correspondiente; en caso contrario, no se enviará ninguna alerta.

Dado que la ubicación de los usuarios se actualiza periódicamente, el sistema ajusta dinámicamente las notificaciones. Esto implica que los usuarios pueden

comenzar a recibir alertas de nuevas emergencias al desplazarse a diferentes ubicaciones, o dejar de recibir notificaciones sobre emergencias que ya no se encuentren dentro de su radio.

El sistema incluye distintos tipos de notificaciones, cada una con una función específica:

- ***Notificación por creación de una emergencia:***
  Cada vez que un usuario, experto o autoridad crea una nueva emergencia, los usuarios que se encuentren en las proximidades de la ubicación recibirán una notificación instantánea. Esta alerta incluye detalles clave, como:
  - *Descripción de la emergencia:* Un resumen breve que permite identificar la naturaleza del evento.
  - Nivel de riesgo: La clasificación inicial del nivel de gravedad (bajo, moderado, alto o extremo).
  - Tipo de emergencia: La categoría correspondiente, como natural, sanitaria, tecnológica, etc.

  Estas notificaciones permiten a los usuarios estar informados desde el momento en que se reporta la emergencia, facilitando la preparación oportuna.

- ***Notificación por cambio de nivel:***
  En situaciones donde un experto o autoridad modifica el nivel de riesgo de una emergencia previamente reportada (por ejemplo, si pasa de moderado a alto) o una solicitud de edición de cambio de nivel de un ciudadano es aceptada, los usuarios cercanos reciben una alerta que notifica el cambio. Esta funcionalidad es esencial para garantizar que las personas tengan acceso a la información más actualizada sobre la gravedad de una emergencia.

- ***Notificación por verificación de emergencia:***
  Cuando una emergencia es revisada y validada por un experto o autoridad, los usuarios cercanos son notificados del cambio en el estado de verificación. Esta notificación resalta que la emergencia ha sido confirmada como auténtica y fiable, lo que incrementa la confianza de los usuarios en la información compartida.

- ***Notificación por emergencia resuelta:***
  Una vez que una emergencia es considerada resuelta (ya sea porque un experto o autoridad lo confirma o porque los usuarios lo reportan), la emergencia desaparece del mapa interactivo. En este caso, se envía una

notificación a los usuarios cercanos indicando que la emergencia ha finalizado, permitiendo a los usuarios saber que la situación ya no representa un riesgo y evita confusiones al navegar por el mapa.

El sistema de notificaciones no solo optimiza la difusión de información en tiempo real, sino que también asegura que los usuarios puedan tomar decisiones informadas con base en datos actualizados. La capacidad de recibir alertas inmediatas en situaciones críticas fomenta la prevención de riesgos y contribuye a una respuesta más coordinada y efectiva.

3.1.5 Cuenta de Usuario y Roles

Dentro de la aplicación, cada usuario tiene la opción de acceder a su cuenta personal, donde puede gestionar y actualizar su información, como su nombre completo, número de teléfono y radio de notificaciones. Este radio, medido en kilómetros, define la distancia del área dentro de la cual el usuario recibirá notificaciones sobre emergencias. De manera predeterminada, al registrarse, cada usuario cuenta con un radio de notificación de 3 kilómetros, el cual puede ser modificado posteriormente según sus preferencias.

Además, los usuarios tienen la posibilidad de solicitar un cambio en su rol dentro de la aplicación. Inicialmente, todos los usuarios se registran con el rol de Ciudadano. Sin embargo, la aplicación permite la asignación de roles específicos que determinan las funcionalidades y privilegios dentro del sistema.

Los roles definidos en la aplicación son los siguientes:

- ***Administrador:*** El administrador es el que posee el control más amplio dentro del sistema. Sus funciones incluyen la gestión de solicitudes de cambio de rol, pudiendo aprobarlas o rechazarlas gestionando de esta forma la autenticación de roles dentro de la aplicación. Asimismo, tiene la capacidad de editar, verificar y resolver emergencias dentro de la plataforma.

- ***Ciudadano:*** Los ciudadanos desempeñan un papel fundamental en la identificación y monitoreo de emergencias. Pudiendo reportar emergencias si consideran que esa emergencia ha desaparecido, visualizar emergencias en el mapa, solicitar un cambio de rol y solicitar cambios en las emergencias ya existentes, como su nivel de riesgo y las recomendaciones existentes.

- ***Autoridad:*** El rol de autoridad está destinado a organismos oficiales como policías, bomberos y otros cuerpos de emergencia. Sus principales responsabilidades son aprobar o rechazar las solicitudes de modificación de

las emergencias creadas por los ciudadanos. Además puede editar directamente dichas emergencias, verificar la existencia de emergencias, resolverlas o solicitar un cambio de rol.

- ***Experto:*** El rol de experto está orientado a profesionales especializados en áreas relevantes, como meteorólogos, sismólogos o especialistas en gestión de desastres. Los expertos también pueden aceptar o denegar las solicitudes de modificación de las emergencias creadas por los ciudadanos. Además de editar, verificar y resolver las emergencias. Y por supuesto pueden solicitar un cambio de rol también.

Cada rol dentro de la aplicación está diseñado para facilitar la cooperación y la efectividad en la gestión de emergencias. Al verificar su rol, un usuario obtiene acceso a las funcionalidades y herramientas que corresponden a sus responsabilidades, contribuyendo de manera más eficiente a la resolución de emergencias y la seguridad general de la comunidad.

3.1.6 Historial de Emergencias

El historial de emergencias es una funcionalidad clave dentro de la aplicación, diseñada para proporcionar a los usuarios un acceso completo y organizado a todas las emergencias gestionadas en la plataforma, tanto las activas como las finalizadas. Este historial facilita el seguimiento de las situaciones de riesgo y promueve un análisis posterior para mejorar la gestión y prevención en el futuro.

Las emergencias dejan de aparecer en el mapa interactivo por los siguientes motivos:

- ***Resolución por un experto o autoridad:***
  Cuando un experto o autoridad marca una emergencia como resuelta, esta se elimina del mapa porque se considera que la situación ya no representa un riesgo activo. Este procedimiento garantiza que solo las emergencias no resueltas permanezcan visibles para los usuarios.

- ***Resolución confirmada por usuarios:***
  Si cinco usuarios reportan que la situación ha sido resuelta, la emergencia se marcará como resuelta y dejará de aparecer en el mapa.
  Este enfoque agiliza la gestión de emergencias menores que pueden resolverse sin la intervención directa de expertos o autoridades.

Para garantizar la transparencia y la posibilidad de realizar un seguimiento completo de las emergencias, la aplicación incluye un apartado dedicado al historial.

Dentro del historial, los usuarios pueden acceder a información detallada sobre cada emergencia, como:

- *Fecha y hora de creación:* Indica el momento en que se reportó la emergencia.
- *Usuario que la creó:* Indica el correo electrónico del usuario que creó la emergencia.
- *Descripción:* Descripción completa de la emergencia.
- *Recomendaciones:* Recomendaciones sugeridas por los usuarios.
- *Nivel y tipo de emergencia:* Clasificaciones clave para comprender la naturaleza y gravedad del evento.
- *Estado de la emergencia:* Especifica si está activa o resuelta.
- *Ubicación:* La localización exacta donde ocurrió la emergencia.

Este historial permite a las autoridades y expertos identificar patrones de riesgo, realizar estudios sobre emergencias frecuentes en determinadas áreas y optimizar los protocolos de respuesta.

3.1.7 Solicitudes

En este apartado se muestran las solicitudes existentes dentro de la aplicación.

- ***Solicitudes de cambio de Rol:***
  Todos los usuarios pueden solicitar un cambio de rol dentro de la aplicación. Esta solicitud tendrá que ser gestionada por los administradores, los cuales a través del apartado Solicitudes pendientes de Cambio de Rol dentro de la aplicación, podrán aceptar o rechazar las peticiones. En cuanto se acepta una petición, el rol del usuario que mandó la solicitud cambiará automáticamente. En estas solicitudes se puede ver el usuario solicitante, la fecha y hora en la que mandó la solicitud y su rol actual y rol solicitado. Además, el usuario solamente puede tener una solicitud en modo Pendiente de cambio de rol. Sin embargo, puede visualizar desde el apartado Mis Solicitudes Cambio de Rol, todas las solicitudes que ha realizado a lo largo del tiempo, tanto las resueltas como la pendiente en el caso de que exista. En este apartado, el usuario puede ver la fecha y hora en la que la realizó, el rol que solicitó, el estado de la solicitud. Y en caso de estar resuelta, el usuario que la resolvió y además la fecha y hora en que la resolvió.

- ***Solicitudes Edición de Emergencias:***
  Los ciudadanos al solicitar editar una emergencia, esta será enviada al apartado de Solicitudes Edición Emergencias, al cual tienen acceso los

expertos y autoridades. En este apartado, dichos usuarios podrán aceptar o rechazar las solicitudes de edición de emergencia realizadas por los ciudadanos. En estas solicitudes se puede visualizar el usuario que la ha realizado, la emergencia que solicita editar, los campos que desea cambiar y además la fecha y hora con la que lo hizo.

Además, los ciudadanos pueden acceder a visualizar todas las solicitudes que han enviado a lo largo del tiempo. En dichas solicitudes pueden ver la emergencia que han solicitado editar, la fecha y hora en que realizaron la solicitud, su solicitud enviada y el estado en el que se encuentra. Y en el caso de estar resuelta, además te muestra el usuario que la resolvió y la fecha y hora en que lo hizo. Con esto, se permite un seguimiento de los cambios en las emergencias viendo quién los sugirió y quién los aceptó.

3.2 Actores del sistema

En la aplicación se identifican cuatro actores, cada uno con roles y responsabilidades específicas dentro del sistema. Estos actores garantizan una gestión eficiente y colaborativa de las emergencias.

- *Ciudadano:*
  Es el usuario común de la aplicación. Las funciones principales que posee es el poder crear nuevas emergencias, solicitar editar emergencias, reportar emergencias y solicitar cambiar de rol. Además de recibir notificaciones sobre emergencias cercanas y poder acceder a los detalles de las emergencias. Un usuario cuando se registra su rol por defecto es el de ciudadano.

- *Experto:*
  El experto es un usuario con conocimientos especializados en determinadas áreas. Su rol es el de añadir valor y credibilidad a la información, por lo que sus funciones principales, aparte de las que tiene el ciudadano, son la verificación, edición y resolución directa de emergencias. Además podrá gestionar las solicitudes de edición de emergencias de los ciudadanos.

- *Autoridad:*
  Las autoridades incluyen cuerpos de seguridad. Permitiendo analizar, verificar y añadir fiabilidad a las emergencias. Por lo tanto sus funciones, aparte de las funciones de los ciudadanos, son las de poder editar emergencias de manera directa, además de verificar y resolverlas. También puede gestionar las solicitudes de edición de emergencias de los ciudadanos.

- ***Administrador:***
  El administrador es el usuario con control total. Su función principal es la de gestionar las solicitudes de cambio de rol, pudiendo aceptarlas o rechazarlas y así modificar las responsabilidades dentro del sistema con respecto a los usuarios.

3.3 Requisitos Funcionales

RF 1: La aplicación debe permitir la autenticación de los usuarios.
RF 1.1. La aplicación debe permitir registrar a los usuarios utilizando su correo electrónico y una contraseña.
RF 1.2. La aplicación debe permitir iniciar sesión utilizando su correo electrónico y contraseña establecido.
RF 1.3: La aplicación debe permitir restablecer la contraseña con el correo electrónico asociado.
RF 1.4: La aplicación debe permitir cerrar sesión.

RF 2: La aplicación debe permitir visualizar un mapa interactivo.
RF 2.1. La aplicación debe permitir interactuar con el mapa interactivo.
RF 2.2. La aplicación debe permitir observar la ubicación de las distintas emergencias.
RF 2.3. La aplicación debe permitir diferenciar las emergencias verificadas.

RF 3: La aplicación debe permitir visualizar una emergencia concreta.
RF 3.1. La aplicación debe permitir acceder a los detalles de la emergencia.
RF 3.2. La aplicación debe permitir comprobar si la emergencia está verificada.
RF 3.3. La aplicación debe permitir visualizar las notificaciones generadas para esa emergencia.

RF 4: La aplicación debe permitir crear una emergencia.
RF 4.1. La aplicación debe permitir añadir una descripción a la emergencia.
RF 4.2. La aplicación debe permitir añadir recomendaciones a la emergencia.
RF 4.3. La aplicación debe permitir elegir el tipo de emergencia que es.
RF 4.4 La aplicación debe permitir elegir el nivel de riesgo.
RF 4.5 La aplicación debe permitir elegir la ubicación de la emergencia.
RF 4.6 La aplicación debe permitir a los expertos y autoridades verificar la emergencia.

RF 5: La aplicación debe permitir realizar un seguimiento de todas las emergencias.

RF 5.1: La aplicación debe permitir a los usuarios acceder al apartado de Historial de Emergencias.
RF 5.2: La aplicación debe permitir en este apartado ver los detalles de todas las emergencias y su estado.

RF 6: La aplicación debe poseer una gestión de resolución de emergencias.
RF 6.1. La aplicación debe permitir a los ciudadanos reportar emergencias para que se resuelvan.
RF 6.2. La aplicación debe permitir a las autoridades y expertos resolver emergencias directamente.
RF 7: La aplicación debe permitir actualizar las emergencias.
RF 7.1. La aplicación debe permitir a las autoridades y expertos añadir recomendaciones a emergencias existentes.
RF 7.2. La aplicación debe permitir a las autoridades y expertos modificar el nivel de riesgo de las emergencias.
RF 7.3 La aplicación debe permitir a los ciudadanos solicitar añadir recomendaciones a emergencias existentes.
RF 7.4 La aplicación debe permitir a los ciudadanos solicitar modificar el nivel de riesgo de las emergencias.

RF 8: La aplicación debe permitir gestionar las solicitudes de edición de las emergencias
RF 8.1. La aplicación debe permitir a las autoridades y expertos acceder al apartado de gestión de solicitudes de edición de emergencias
RF 8.2. La aplicación debe permitir a las autoridades y expertos aceptar solicitudes de edición de emergencias.
RF 8.3 La aplicación debe permitir a las autoridades y expertos denegar solicitudes de edición de emergencias.
RF 8.4 La aplicación debe permitir a los ciudadanos acceder al apartado de mis solicitudes de edición de emergencias.
RF 8.5 La aplicación debe permitir a los ciudadanos visualizar el estado y detalles de sus solicitudes.

RF 9: La aplicación debe permitir gestionar los datos personales de los usuarios.
RF 9.1. La aplicación debe permitir a los usuarios agregar y/o modificar su nombre.
RF 9.2 La aplicación debe permitir a los usuarios agregar y/o modificar su número de teléfono.
RF 9.2. La aplicación debe permitir a los usuarios modificar su radio de notificaciones.

RF 10: La aplicación debe permitir la gestión de roles.
RF 10.1. La aplicación debe permitir a los usuarios solicitar un cambio de rol.
RF 10.2. La aplicación debe permitir al administrador aceptar solicitudes de cambio de rol.
RF 10.3. La aplicación debe permitir al administrador rechazar solicitudes de cambio de rol.
RF 10.4 La aplicación debe permitir a los usuarios acceder a sus solicitudes de cambio de rol.
RF 10.4 La aplicación debe permitir a los usuarios visualizar el estado y detalles de sus solicitudes.

RF 11: La aplicación debe permitir la gestión de notificaciones.
RF 11.1. La aplicación debe permitir a los usuarios recibir notificaciones cuando se crean emergencias dentro de su radio.
RF 11.2. La aplicación debe permitir a los usuarios recibir notificaciones cuando se cambia el nivel de riesgo de una emergencia dentro de su radio.
RF 11.3 La aplicación debe permitir a los usuarios recibir notificaciones cuando se verifica una emergencia dentro de su radio.
RF 11.4. La aplicación debe permitir a los usuarios recibir notificaciones cuando desaparece una emergencia dentro de su radio.

3.4 Requisitos No Funcionales

RNF 1: Usabilidad
RNF 1.1: La interfaz de usuario debe ser intuitiva, clara y adaptada a dispositivos móviles.
RNF 1.2: La aplicación debe estar optimizada para Android y adaptarse a distintas resoluciones de pantalla.
RNF 1.3: La aplicación debe permitir la interacción con el mapa y con las emergencias de forma rápida y sin bloqueos.

RNF 2: Localización.
RNF 2.1: La aplicación debe utilizar la geolocalización del dispositivo para adaptar las notificaciones a la ubicación del usuario.

3.5 Modelo de Comportamiento

3.5.1 Diagrama de Casos de Uso

Se ha realizado tres diagramas de caso de uso de la aplicación correspondientes a los roles, con el fin de mostrar las características y funcionalidades de la aplicación respecto a cada rol.

En el caso de experto y autoridad es el mismo diagrama de caso de uso ya que poseen las mismas funcionalidades.

En la Figura 33 se puede observar el diagrama de caso de uso completo como ciudadano.

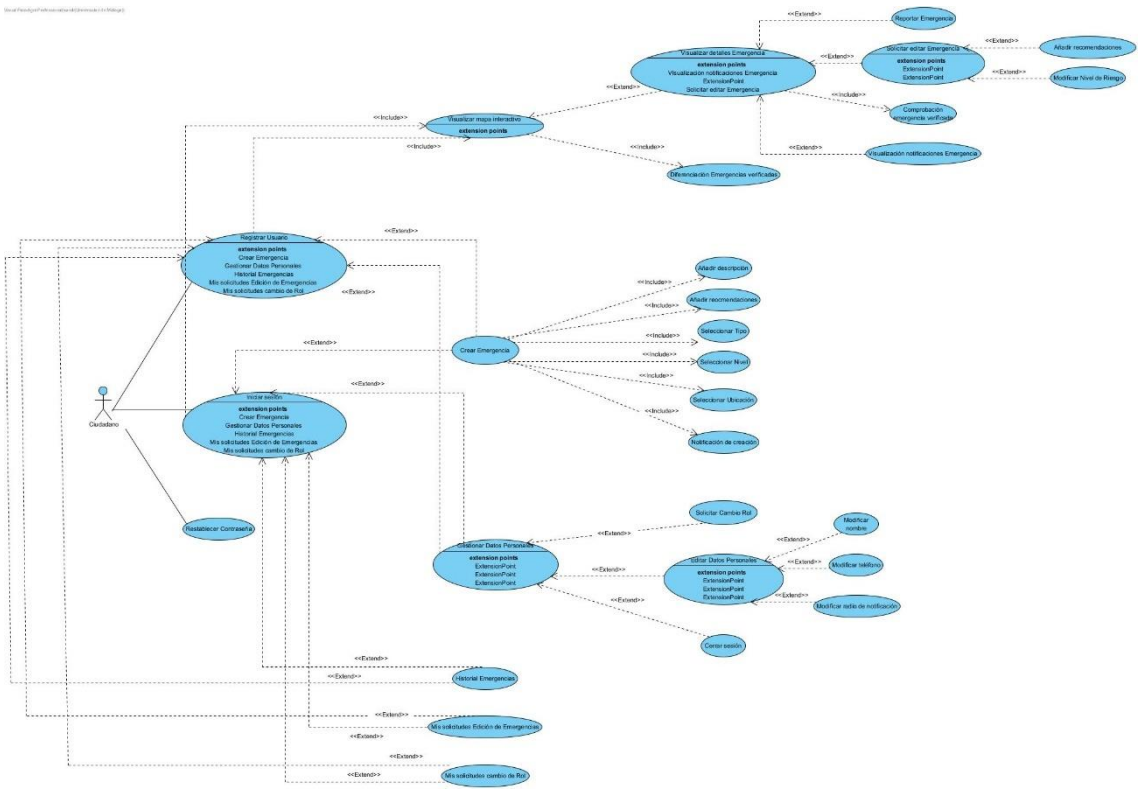

Figura 33: Diagrama de casos de uso como ciudadano de la aplicación

Si se visualiza por partes, en la Figura 34, se puede ver que el ciudadano cuando Inicia Sesión o se Registra, puede acceder al mapa interactivo, pudiendo diferenciar si están verificadas las emergencias de manera visual y en el caso de que lo desee puede visualizar los detalles de las emergencias. En el que desde ahí podrá realizar distintas acciones como reportar la emergencia, solicitar editarla o la visualización de las notificaciones generadas para esa emergencia.

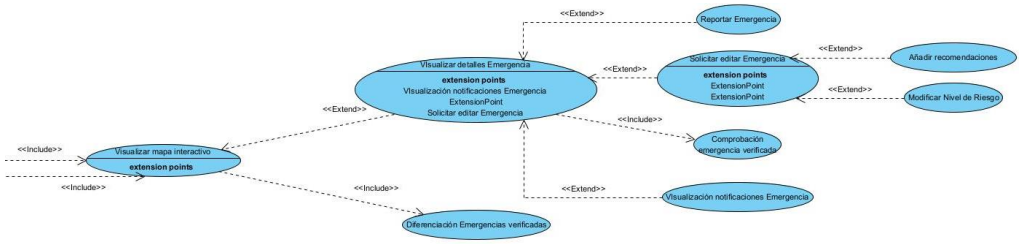

Figura 34: Visualización parte Mapa Interactivo Diagrama Caso de Uso Ciudadano

También, una vez que el usuario se autentica, puede crear una emergencia nueva. Dicho caso de uso incluye los distintos campos necesarios para poder crearla, como se puede ver en la Figura 35.

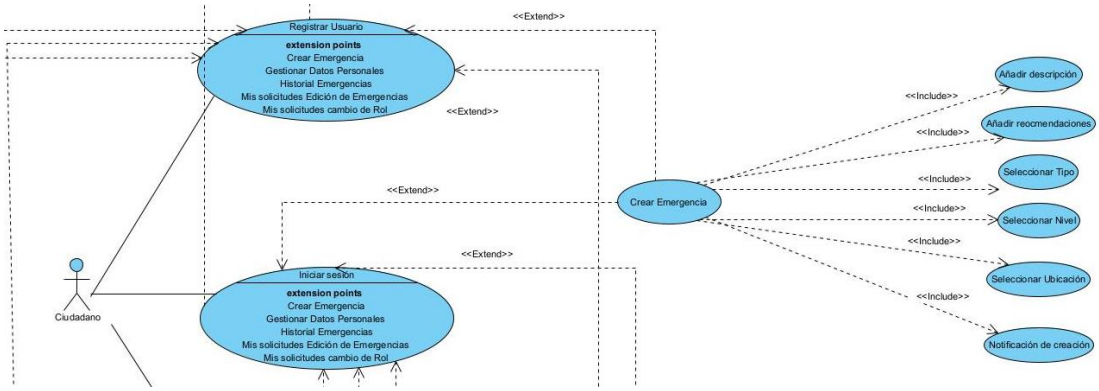

Figura 35: Visualización parte Crear Emergencia Diagrama Caso de Uso Ciudadano

En el caso de uso también se puede apreciar en la Figura 36 que una vez autenticado, el usuario puede gestionar sus datos personales, solicitando un cambio de rol, editar sus datos personales, tales como nombre, teléfono o radio de notificación y también cerrar sesión.

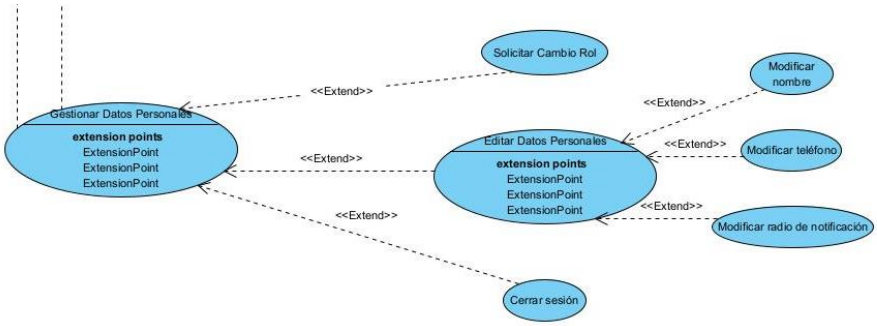

Figura 36: VIsualización parte Gestión Datos Personales Diagrama Caso de Uso Ciudadano

Y por último, se puede ver en el caso de uso de la Figura 37, que una vez que el usuario se ha iniciado sesión o se ha registrado, también se le permite acceder a otras funcionalidades como, por ejemplo, el historial de las emergencias, sus solicitudes de edición de emergencias o sus solicitudes de cambio de rol.

A continuación, también se muestra el diagrama de caso de uso de los expertos y autoridades en la Figura 38.
Si se visualiza detenidamente con respecto al anterior caso de uso y por partes se puede ver que el caso de uso Registrarse desaparece, ya que un usuario cuando se registra tiene por defecto el rol de Ciudadano.

Además, tal y como muestra la Figura 39 en la parte de Visualizar detalles de la Emergencia, se observa en este caso que se pueden resolver y editar Emergencias directamente y verificarla en caso de que no lo esté.

En el caso de la parte de crear emergencia, se puede visualizar en la Figura 40 que además se pueden verificar las emergencias.

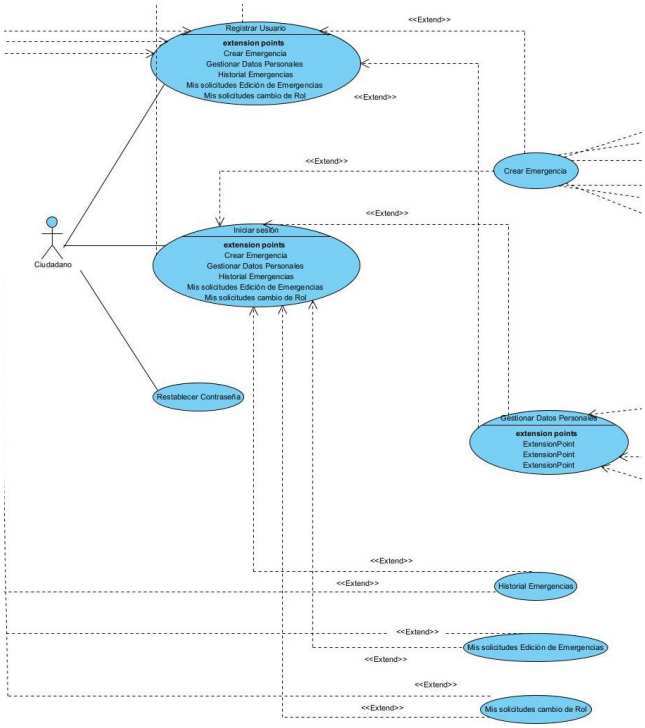

Figura 37: Visualización parte más funciones Diagrama Caso de Uso Ciudadano

Y, por último, en la Figura 41 se puede observar que el usuario posee la gestión de las solicitudes de edición de emergencias realizadas por los ciudadanos, pudiendo aceptarlas o rechazarlas.

A continuación, se va a mostrar el diagrama de caso de uso del administrador en la Figura 42, el cual tiene funcionalidades nuevas.

La diferencia principal es que, en este caso, se pueden gestionar las solicitudes de cambio de rol, pudiendo aceptarlas o rechazarlas. Esta parte se puede visualizar con mayor detalle en la Figura 43.

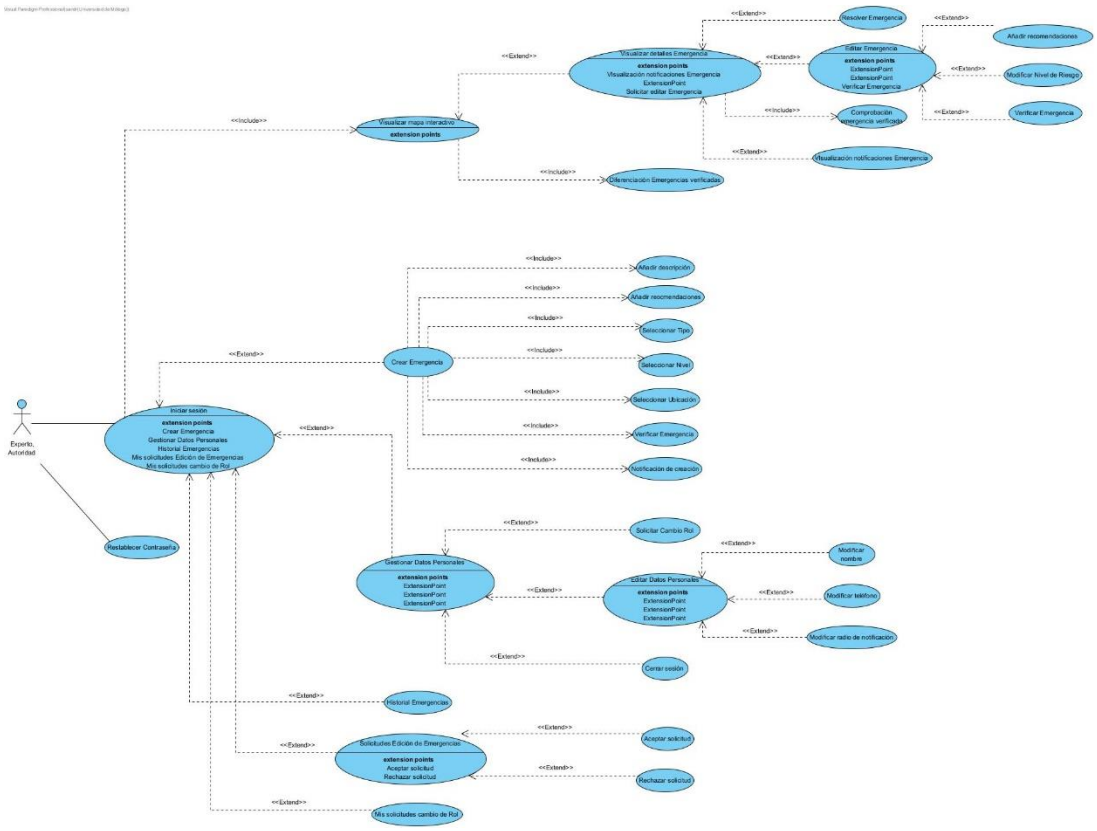

Figura 38: Diagrama de casos de uso como Experto o Autoridad de la aplicación

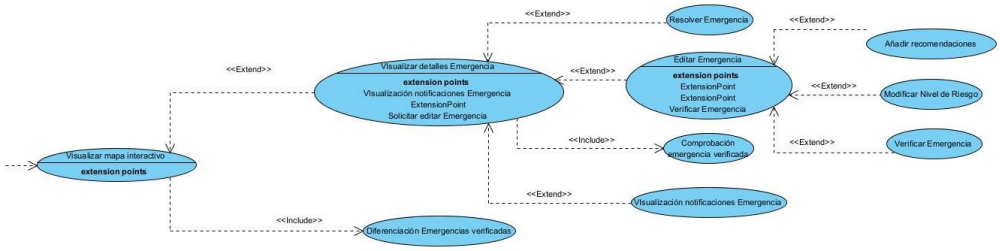

Figura 39: Visualización parte visualizar detalles Emergencia Diagrama Caso de Uso Experto/Autoridad

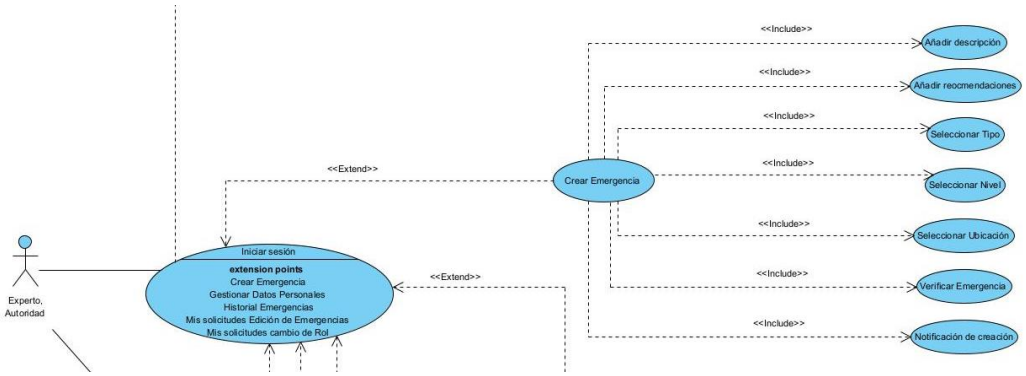

Figura 40: Visualización parte Crear Emergencia Diagrama Caso de Uso Experto/Autoridad

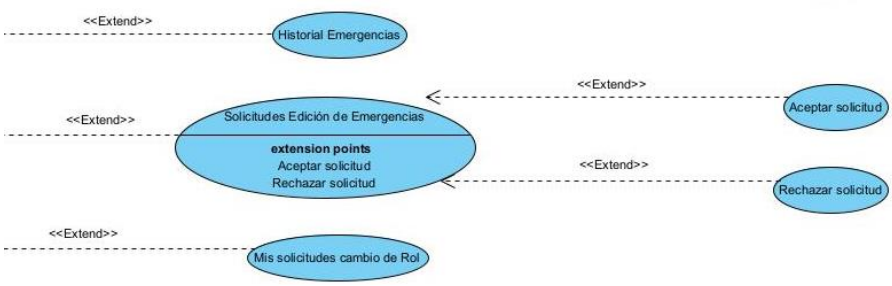

Figura 41: Visualización parte Solicitudes Edición de Emergencias Diagrama Caso de Uso Experto/Autoridad

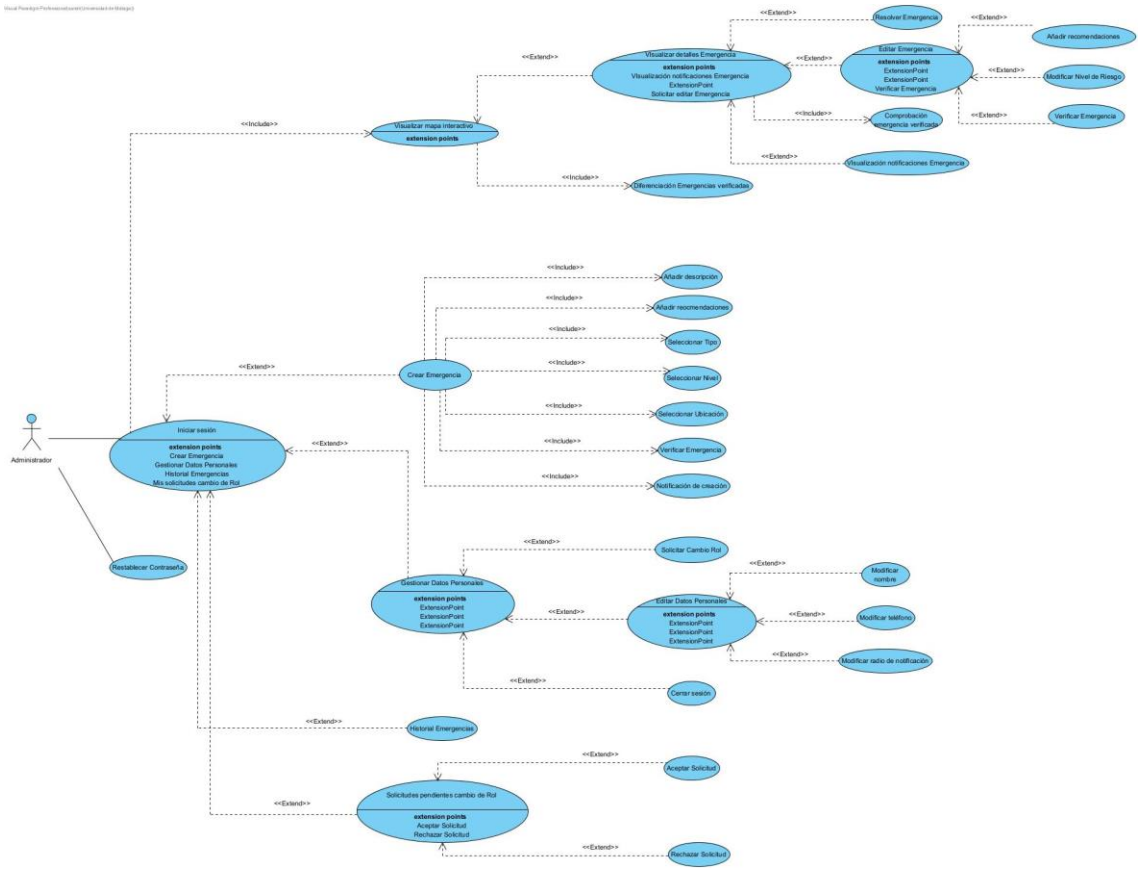

Figura 42: Diagrama de casos de uso como administrador de la aplicación

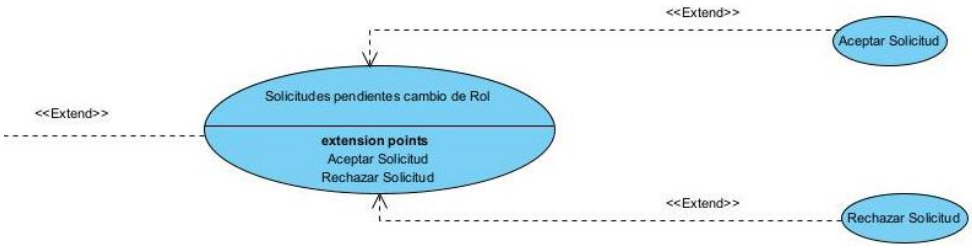

Figura 43: Visualización parte Solicitudes pendientes cambio de Rol Diagrama Caso de Uso Administrador

A continuación se describen en detalle los casos de usos anteriormente mencionados:

Caso de uso	Registrarse

Actores	Ciudadano
Descripción	El usuario podrá registrarse en la aplicación mediante su correo electrónico y contraseña
Precondiciones	El usuario ha accedido a la aplicación y no ha iniciado sesión
Escenario Principal	1. El usuario pulsa sobre el botón de Crear una cuenta 2. El sistema muestra la pantalla de registro 3. El usuario introduce su correo electrónico y contraseña 4. El usuario selecciona el botón de Registrarse 5. El sistema redirige a la página principal de la aplicación
Escenario Alternativo	4b. El usuario al repetir la contraseña para confirmar se confunde y no son iguales 4c. El sistema lanza una alerta sobre que las contraseñas no coinciden

Cuadro 1: Caso de uso: Registrarse

Caso de uso	Iniciar sesión
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario podrá iniciar sesión utilizando el correo electrónico y contraseña con el que se registró
Precondiciones	El usuario ha iniciado la aplicación y no ha iniciado sesión previamente
Escenario Principal	1. El usuario introduce el correo electrónico y contraseña 2. El usuario selecciona el botón de Iniciar Sesión 3. El sistema redirige a la página principal de la aplicación
Escenario Alternativo	2b. El usuario introduce un correo o contraseña incorrecto

	2c. El usuario no es redirigido a la página principal de la aplicación

Cuadro 2: Caso de uso: Iniciar Sesión

Caso de uso	Restablecer contraseña
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario podrá restablecer su contraseña en caso de ser olvidada
Precondiciones	El usuario ha iniciado la aplicación y no ha podido iniciar sesión
Escenario Principal	1. El usuario selecciona el botón de ¿Olvidaste tu contraseña? 2. El sistema muestra la pantalla para restablecer la contraseña 3. El usuario introduce el correo electrónico del que quiere restablecer la contraseña 4. El sistema envía un correo electrónico para restablecer la contraseña 5. El usuario introduce desde el correo electrónico la nueva contraseña 6. El sistema restablece la contraseña
Escenario Alternativo	3b. El usuario introduce un correo electrónico no existente 3c. El usuario no recibe el correo electrónico para restablecer la contraseña 5b. El usuario introduce una contraseña incorrecta nuevamente

Cuadro 3: Caso de uso: Restablecer contraseña

Caso de uso	Cerrar Sesión
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario podrá cerrar sesión
Precondiciones	El usuario ha iniciado sesión previamente

Escenario Principal	1. El usuario accede al apartado de Mi cuenta 2. El sistema muestra la pantalla de Mi cuenta correspondiente al usuario 1. El usuario selecciona el botón de Cerrar sesión 2. El sistema redirige al usuario a la página de inicio de sesión
Escenario Alternativo	

Cuadro 4: Caso de uso: Cerrar Sesión

Caso de uso	Interactuar con el mapa interactivo
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario podrá interactuar con el mapa interactivo
Precondiciones	El usuario ha iniciado sesión previamente
Escenario Principal	1. La aplicación muestra el mapa interactivo de la página principal 2. El usuario puede aumentar/disminuir/seleccionar el mapa
Escenario Alternativo	1b. Por falta de conexión el mapa no carga correctamente

Cuadro 5: Caso de uso: Interactuar con el mapa interactivo

Caso de uso	Diferenciación emergencias verificadas
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario podrá distinguir las emergencias verificadas de las no verificadas
Precondiciones	El usuario ha iniciado sesión previamente
Escenario Principal	1. La aplicación muestra el mapa interactivo de la página principal 2. El usuario puede distinguir las emergencias verificadas en el propio

	mapa debido a la visualización de un pulso
Escenario Alternativo	2b. Por falta de conexión las emergencias no cargan correctamente

Cuadro 6: Caso de uso: Diferenciación emergencias verificadas

Caso de uso	Visualización detalles emergencia
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario podrá visualizar los detalles de las emergencias
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona una de las emergencias 2. El sistema muestra todos los detalles de la emergencia seleccionada
Escenario Alternativo	2b. Por falta de conexión no se carga la emergencia correctamente

Cuadro 7: Caso de uso: Visualización detalles emergencia

Caso de uso	Emergencia verificada
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder ver si una emergencia está verificada visualizando los detalles de la emergencia
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona una emergencia concreta 2. El sistema muestra los detalles de la emergencia

	3. El usuario puede visualizar si la emergencia está verificada mediante el *tooltip* de la pantalla
Escenario Alternativo	2b. Por falta de conexión los detalles de la emergencia no cargan correctamente

Cuadro 8: Caso de uso: Emergencia verificada

Caso de uso	Notificaciones de la Emergencia
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder consultar todas las notificaciones generadas de una emergencia concreta
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona una emergencia concreta 2. El sistema muestra los detalles de la emergencia 3. El usuario selecciona el icono de las notificaciones 4. El sistema muestra todas las notificaciones generadas por la emergencia 5. El usuario puede visualizar todas las notificaciones y sus detalles
Escenario Alternativo	2b. Por falta de conexión los detalles de la emergencia no cargan correctamente

Cuadro 9: Caso de uso: Notificaciones Emergencia

Caso de uso	Crear una emergencia
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder crear una emergencia nueva

Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona el botón de crear emergencia 2. El sistema muestra la pantalla para crear una emergencia nueva 3. El usuario rellena los campos necesarios 4. El usuario selecciona el botón de guardar 5. El sistema redirige a la página principal añadiendo dicha emergencia
Escenario Alternativo	3b. El usuario introduce datos incorrectos

Cuadro 10: Caso de uso: Crear una emergencia

Caso de uso	Añadir una descripción a una emergencia
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder añadir una descripción a una emergencia nueva
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona el botón de crear emergencia 2. El sistema muestra la pantalla para crear una emergencia nueva 3. El usuario rellena los campos necesarios 4. El usuario añade la descripción correspondiente de la emergencia 5. El usuario selecciona el botón de guardar 6. El sistema redirige a la página principal añadiendo dicha emergencia

Escenario Alternativo	3b. El usuario introduce datos incorrectos

Cuadro 11: Caso de uso: Añadir descripción

Caso de uso	Añadir recomendaciones a una Emergencia
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder añadir recomendaciones a una emergencia nueva
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona el botón de crear emergencia 2. El sistema muestra la pantalla para crear una emergencia nueva 3. El usuario rellena los campos necesarios 4. El usuario añade las recomendaciones correspondientes de la emergencia 5. El usuario selecciona el botón de guardar 6. El sistema redirige a la página principal añadiendo dicha emergencia
Escenario Alternativo	3b. El usuario introduce datos incorrectos

Cuadro 12: Caso de uso: Añadir recomendaciones

Caso de uso	Seleccionar tipo de Emergencia
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder seleccionar el tipo de emergencia
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el

	mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona el botón de crear emergencia 2. El sistema muestra la pantalla para crear una emergencia nueva 3. El usuario rellena los campos necesarios 4. El usuario selecciona el tipo de emergencia correspondiente 5. El usuario selecciona el botón de guardar 6. El sistema redirige a la página principal añadiendo dicha emergencia
Escenario Alternativo	3b. El usuario introduce datos incorrectos

Cuadro 13: Caso de uso: Seleccionar tipo de Emergencia

Caso de uso	Seleccionar nivel de Riesgo
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder añadir el nivel de riesgo correspondiente a la emergencia
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona el botón de crear emergencia 2. El sistema muestra la pantalla para crear una emergencia nueva 3. El usuario rellena los campos necesarios 4. El usuario selecciona el nivel de riesgo correspondiente acorde a la emergencia 5. El usuario selecciona el botón de guardar

	6. El sistema redirige a la página principal añadiendo dicha emergencia
Escenario Alternativo	3b. El usuario introduce datos incorrectos

Cuadro 14: Caso de uso: Seleccionar nivel de Riesgo

Caso de uso	Seleccionar ubicación de la Emergencia
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder seleccionar la ubicación exacta de la Emergencia
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona el botón de crear emergencia 2. El sistema muestra la pantalla para crear una emergencia nueva 3. El usuario rellena los campos necesarios 4. El usuario selecciona la ubicación exacta donde se encuentra la emergencia 5. El usuario selecciona el botón de guardar 6. El sistema redirige a la página principal añadiendo dicha emergencia
Escenario Alternativo	4b. El usuario introduce mal la ubicación de la emergencia

Cuadro 15: Caso de uso: Seleccionar ubicación de la Emergencia

Caso de uso	Verificar una emergencia
Actores	Expertos, Autoridades, Administrador
Descripción	El usuario debe poder verificar emergencias
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el

	mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona la emergencia que desea verificar 2. El sistema muestra los detalles de la emergencia correspondiente 3. El usuario selecciona el botón de verificar emergencia 4. El usuario selecciona guardar cambios 5. El sistema redirige a la página principal
Escenario Alternativo	1b. El usuario selecciona una emergencia ya verificada

Cuadro 16: Caso de uso: Verificar Emergencias

Caso de uso	Historial de Emergencias
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder consultar el historial completo de las emergencias
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona el apartado de Historial Emergencias 2. El sistema muestra la pantalla del Historial 3. El usuario puede visualizar los detalles de todas las emergencias
Escenario Alternativo	2b. Por falta de conexión, no se carga correctamente el historial completo

Cuadro 17: Caso de uso: Historial de Emergencias

Caso de uso	Reportar Emergencias
Actores	Ciudadanos
Descripción	El usuario debe poder reportar emergencias y a los 5 reportes en la emergencia se marcará como resuelta

Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona la emergencia que desea reportar 2. El sistema muestra los detalles de la emergencia 3. El usuario selecciona el botón de reportar emergencia 4. El sistema redirige a la página principal añadiendo el reporte a dicha emergencia y eliminándola del mapa en caso de llegar a 5 reportes
Escenario Alternativo	2b. Por falta de conexión la aplicación no carga los detalles de la emergencia correctamente

Cuadro 18: Caso de uso: Reportar Emergencias

Caso de uso	Resolver Emergencias
Actores	Expertos, Autoridades, Administrador
Descripción	El usuario debe poder resolver emergencias directamente y que desaparezcan del mapa
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona la emergencia que desea resolver 2. El sistema muestra los detalles de la emergencia 3. El usuario selecciona el botón de Resolver Emergencia 4. El sistema muestra una alerta sobre si quiere confirmar y resolver la emergencia 5. El usuario selecciona Confirmar

	6. El sistema redirige a la página principal eliminando la emergencia de entre las activas
Escenario Alternativo	2b. Por falta de conexión la aplicación no carga los detalles de la emergencia correctamente 5b. El usuario selecciona Cancelar 6b. El sistema vuelve a redirigir a los detalles de la Emergencia

Cuadro 19: Caso de uso: Resolver Emergencias

Caso de uso	Añadir recomendaciones a Emergencias existentes
Actores	Experto, Autoridad, Administrador
Descripción	El usuario debe poder añadir recomendaciones a las emergencias existentes deseadas
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona la emergencia de la que desea añadir más recomendaciones 2. El sistema muestra los detalles de la emergencia 3. El usuario selecciona el botón de Editar emergencia 4. El sistema muestra la pantalla de edición de emergencia 5. El usuario introduce las recomendaciones deseadas para la emergencia 6. El usuario selecciona el botón de guardar 7. El sistema actualiza y redirige a la página principal

Escenario Alternativo	2b. Por falta de conexión no carga correctamente los detalles de la emergencia 6b. Por un error no se actualiza correctamente los cambios en la emergencia

Cuadro 20: Caso de uso: Añadir Recomendaciones Emergencias Existentes

Caso de uso	Modificar el nivel de Riesgo de una Emergencia existente
Actores	Experto, Autoridad, Administrador
Descripción	El usuario debe poder modificar el nivel de riesgo de las emergencias existentes deseadas
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona la emergencia de la que desea modificar su nivel de riesgo 2. El sistema muestra los detalles de la emergencia 3. El usuario selecciona el botón de editar emergencia 4. El sistema muestra la pantalla de edición de emergencia 5. El usuario selecciona el nuevo nivel de riesgo deseado 6. El usuario selecciona el botón de guardar 7. El sistema actualiza y redirige a la página principal
Escenario Alternativo	2b. Por falta de conexión no carga correctamente los detalles de la emergencia

	6b. Por un error no se actualiza correctamente los cambios en la emergencia

Cuadro 21: Caso de uso: Modificar nivel de Riesgo Emergencia Existente

Caso de uso	Solicitar añadir recomendaciones a Emergencias existentes
Actores	Ciudadano
Descripción	El usuario debe poder solicitar el poder añadir recomendaciones a las emergencias existentes deseadas
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona la emergencia de la que desea solicitar añadir más recomendaciones 2. El sistema muestra los detalles de la emergencia 3. El usuario selecciona el botón de Solicitar editar Emergencia 4. El sistema muestra la pantalla de edición de emergencia 5. El usuario introduce las recomendaciones deseadas para la emergencia 6. El usuario selecciona el botón de Solicitar Cambios 7. El sistema guarda la solicitud y redirige a la página principal
Escenario Alternativo	2b. Por falta de conexión no carga correctamente los detalles de la emergencia 6b. Por un error no se guarda correctamente la solicitud de la emergencia

Cuadro 22: Caso de uso: Solicitar añadir recomendaciones

Caso de uso	Solicitar modificar el nivel de Riesgo de una Emergencia existente
Actores	Ciudadano
Descripción	El usuario debe poder modificar el nivel de riesgo de las emergencias existentes deseadas
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario selecciona la emergencia de la que desea solicitar modificar el nivel de riesgo 2. El sistema muestra los detalles de la emergencia 3. El usuario selecciona el botón de Solicitar editar Emergencia 4. El sistema muestra la pantalla de edición de emergencia 5. El usuario selecciona el nuevo nivel de riesgo deseado 6. El usuario selecciona el botón de Solicitar cambios 7. El sistema guarda la solicitud y redirige a la página principal
Escenario Alternativo	2b. Por falta de conexión no carga correctamente los detalles de la emergencia 6b. Por un error no se guarda correctamente la solicitud de la emergencia

Cuadro 23: Caso de uso: Solicitar modificar nivel de Riesgo

Caso de uso	Acceder a las solicitudes pendientes de edición de emergencias
Actores	Experto, Autoridad
Descripción	El usuario debe poder acceder al apartado de solicitudes pendientes de edición de emergencias

Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario accede al apartado de Solicitudes Edición Emergencias 2. El sistema muestra la pantalla de solicitudes pendientes 3. El usuario visualiza todas las solicitudes pendientes existentes
Escenario Alternativo	2b. Por falta de conexión el sistema no muestra correctamente todas las solicitudes

Cuadro 24: Caso de uso: Acceder solicitudes edición emergencias

Caso de uso	Aceptar solicitudes pendientes de edición de emergencias
Actores	Experto, Autoridad
Descripción	El usuario debe poder acceder al apartado de solicitudes pendientes de edición de emergencias y aceptar las solicitudes deseadas
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario accede al apartado de Solicitudes Edición Emergencias 2. El sistema muestra la pantalla de solicitudes pendientes 3. El usuario visualiza todas las solicitudes pendientes existentes 4. El usuario selecciona el botón Aceptar de la solicitud deseada 5. El sistema recarga la pantalla actualizando las pendientes restantes
Escenario Alternativo	2b. Por falta de conexión el sistema no muestra correctamente todas las solicitudes

	4b. El usuario deniega la solicitud por error

Cuadro 25: Caso de uso: Aceptar solicitudes edición emergencias

Caso de uso	Denegar solicitudes pendientes de edición de emergencias
Actores	Experto, Autoridad
Descripción	El usuario debe poder acceder al apartado de solicitudes pendientes de edición de emergencias y denegar las solicitudes deseadas
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario accede al apartado de Solicitudes Edición Emergencias 2. El sistema muestra la pantalla de solicitudes pendientes 3. El usuario visualiza todas las solicitudes pendientes existentes 4. El usuario selecciona el botón de Rechazar 5. El sistema recarga la pantalla actualizando las pendientes restantes
Escenario Alternativo	2b. Por falta de conexión el sistema no muestra correctamente todas las solicitudes 4b. El usuario acepta la solicitud por error

Cuadro 26: Caso de uso: Denegar solicitudes edición emergencias

Caso de uso	Acceder a mis solicitudes de edición de Emergencias
Actores	Ciudadano
Descripción	El usuario debe poder acceder al apartado de Mis solicitudes Edición de Emergencias y ver sus detalles

Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario accede al apartado de Mis solicitudes Edición de Emergencias 2. El sistema muestra la pantalla de todas las solicitudes realizadas por el usuario 3. El usuario visualiza todas las solicitudes existentes y sus detalles
Escenario Alternativo	2b. Por falta de conexión el sistema no muestra correctamente todas las solicitudes

Cuadro 27: Caso de uso: Acceder Mis Solicitudes edición Emergencias

Caso de uso	Agregar o modificar nombre completo
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder agregar su nombre de la aplicación o modificar el existente
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario accede al apartado de Mi cuenta 2. El sistema muestra la información correspondiente a sus datos 3. El usuario selecciona el botón de Editar Datos 4. El sistema muestra la pantalla de edición de datos personales 5. El usuario introduce el nombre deseado 6. El usuario selecciona el botón de Guardar cambios

	7. El sistema actualiza el nombre del usuario y redirige al apartado de Mi cuenta
Escenario Alternativo	6b. Por un error el sistema no actualiza correctamente el nuevo nombre del usuario

Cuadro 28: Caso de uso: Agregar o modificar nombre

Caso de uso	Agregar o modificar número de teléfono
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder agregar su número de teléfono en la aplicación o modificar el existente
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario accede al apartado de Mi cuenta 2. El sistema muestra la información correspondiente a sus datos 3. El usuario selecciona el botón de Editar Datos 4. El sistema muestra la pantalla de edición de datos personales 5. El usuario introduce su número de teléfono 6. El usuario selecciona el botón de Guardar cambios 7. El sistema actualiza el nombre del usuario y redirige al apartado de Mi cuenta
Escenario Alternativo	6b. Por un error el sistema no actualiza correctamente el nuevo teléfono del usuario

Cuadro 29: Caso de uso: Agregar o modificar número de teléfono

Caso de uso	Modificar radio del usuario

Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder modificar su radio de notificaciones
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario accede al apartado de Mi cuenta 2. El sistema muestra la información correspondiente a sus datos 3. El usuario selecciona el botón de Editar Datos 4. El sistema muestra la pantalla de edición de datos personales 5. El usuario desliza el Slider en función del radio de notificaciones deseado 6. El usuario selecciona el botón de Guardar cambios 7. El sistema actualiza el nombre del usuario y redirige al apartado de Mi cuenta
Escenario Alternativo	6b. Por un error el sistema no actualiza correctamente el nuevo radio de notificaciones del usuario

Cuadro 30: Caso de uso: Modificar radio

Caso de uso	Solicitar cambio de rol del usuario
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder solicitar el cambiar su rol de usuario
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario accede al apartado de Mi cuenta

	2. El sistema muestra la pantalla correspondiente 3. El usuario selecciona el despliegue de Cambiar rol 4. El sistema muestra el despliegue para permitir cambiar el rol 5. El usuario selecciona el rol deseado 6. El usuario pulsa el botón Solicitar cambio de Rol 4. El usuario selecciona el botón de solicitar cambio de rol 5. El sistema guarda la solicitud y redirige a la página principal
Escenario Alternativo	3b. El usuario tiene una solicitud pendiente de cambio de rol 4b. El sistema muestra una frase explicando que no puede solicitar otro cambio de rol hasta que no sea resuelto 5b. Por un error el sistema no guarda correctamente la nueva solicitud de cambio de rol

Cuadro 31: Caso de uso: Solicitar cambio de rol

Caso de uso	Aceptar solicitudes cambio de rol
Actores	Administrador
Descripción	El usuario debe poder aceptar solicitudes de cambio de rol
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario accede al apartado de Solicitudes pendientes de Cambio de Rol 2. El sistema muestra la pantalla correspondiente 3. El usuario selecciona el botón de Aceptar de la solicitud deseada

	4. El sistema recarga la lista y vuelve a mostrar las pendientes
Escenario Alternativo	2b. Por falta de conexión el sistema no muestra correctamente las solicitudes de cambio de rol 4b. El usuario por error deniega la solicitud de cambio de rol

Cuadro 32: Caso de uso: Aceptar solicitudes cambio de rol

Caso de uso	Denegar solicitudes cambio de rol
Actores	Administrador
Descripción	El usuario debe poder denegar solicitudes de cambio de rol
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario accede al apartado de Solicitudes pendientes de Cambio de Rol 2. El sistema muestra la pantalla de solicitudes de cambio de rol 3. El usuario selecciona el botón de Rechazar de la solicitud correspondiente 4. El sistema recarga la lista y vuelve a mostrar las pendientes
Escenario Alternativo	2b. Por falta de conexión el sistema no muestra correctamente las solicitudes de cambio de rol 3b. El usuario por error acepta la solicitud de cambio de rol

Cuadro 33: Caso de uso: Denegar solicitudes cambio de rol

Caso de uso	Visualizar Solicitudes Cambio de Rol
Actores	Ciudadano, Experto, Autoridad

Descripción	El usuario debe poder visualizar todas las solicitudes realizadas para cambiar de Rol
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. El usuario accede al apartado de Mis solicitudes Cambio de Rol 2. El sistema muestra todas las solicitudes de Cambio de Rol realizadas por el usuario 3. El usuario visualiza todas sus solicitudes y sus detalles
Escenario Alternativo	2b. Por falta de conexión el sistema no muestra correctamente las solicitudes de cambio de rol

Cuadro 34: Caso de uso: Visualizar solicitudes cambio de Rol

Caso de uso	Recibir notificaciones cuando se crean Emergencias
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder recibir notificaciones cuando se crean nuevas emergencias dentro de su radio de notificaciones
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. Un usuario ha creado una emergencia con una ubicación que se encuentra dentro del radio de notificaciones del usuario mencionado 2. El sistema envía la notificación correspondiente a los usuarios necesarios 3. El usuario recibe la notificación y la visualiza

Escenario Alternativo	2b. Se produce un error y el sistema no envía la notificación correspondiente

Cuadro 35: Caso de uso: Recibir notificaciones creación de Emergencias

Caso de uso	Recibir notificaciones cuando se modifica el nivel de riesgo de una emergencia
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder recibir notificaciones cuando se modifica el nivel de riesgo de una emergencia existente dentro de su radio
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. Un usuario ha modificado la emergencia cambiándole el nivel de riesgo 2. El sistema envía la notificación correspondiente a los usuarios necesarios 3. El usuario recibe la notificación y la visualiza
Escenario Alternativo	2b. Se produce un error y el sistema no envía la notificación correspondiente

Cuadro 36: Caso de uso: Recibir notificaciones modificación nivel de Riesgo

Caso de uso	Recibir notificaciones cuando se verifican emergencias
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder recibir notificaciones cuando se verifica una emergencia existente dentro de su radio
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el

	mapa interactivo de la página principal
Escenario Principal	1. Un usuario ha verificado una emergencia 2. El sistema envía la notificación correspondiente a los usuarios necesarios 3. El usuario recibe la notificación y la visualiza
Escenario Alternativo	2b. Se produce un error y el sistema no envía la notificación correspondiente

Cuadro 37: Caso de uso: Recibir notificaciones verificación de Emergencias

Caso de uso	Recibir notificaciones cuando se resuelven Emergencias
Actores	Ciudadano, Experto, Autoridad, Administrador
Descripción	El usuario debe poder recibir notificaciones cuando se resuelven emergencias dentro de su radio de notificaciones
Precondiciones	El usuario ha iniciado sesión previamente y la aplicación muestra el mapa interactivo de la página principal
Escenario Principal	1. Un usuario ha resuelto una emergencia con una ubicación que se encuentra dentro del radio de notificaciones del usuario mencionado 2. El sistema envía la notificación correspondiente a los usuarios necesarios 3. El usuario recibe la notificación y la visualiza
Escenario Alternativo	2b. Se produce un error y el sistema no envía la notificación correspondiente

Cuadro 38: Caso de uso: Recibir notificaciones resolución de Emergencias

3.5.2 Diagramas de Flujo de Procesos

Representación del modelo del flujo de eventos para los requisitos identificados. Esto permite visualizar de manera estructurada las distintas etapas de ejecución y toma de decisiones dentro de la aplicación.

Un diagrama de flujo de procesos es una representación gráfica que utiliza símbolos para describir la secuencia de actividades, decisiones y flujos de datos en un sistema. Su uso facilita la comprensión del funcionamiento de la aplicación. En todos los diagramas de flujo excepto en el de autenticación se va a suponer que el usuario ya se ha autenticado y está en la página principal de la aplicación para simplificar los diagramas.

Autenticación: En el diagrama de la Figura 44 se muestra el flujo que puede seguir un usuario para autenticarse en ella. Para ello se distinguen dos casos, si el usuario ya tiene creada la cuenta podrá iniciar sesión con su correo y contraseña, en el caso de que no la haya olvidado. Y en el caso de ser la primera vez que accede a la aplicación, podrá crearse una cuenta con su correo electrónico y contraseña.

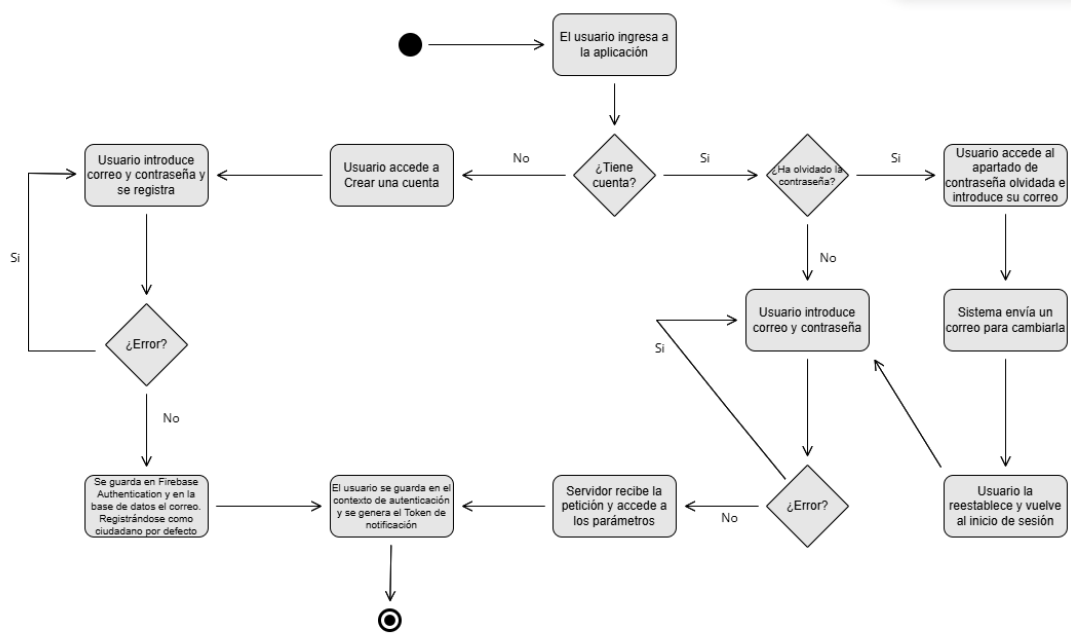

Figura 44: Diagrama de flujo de Autenticación

Crear Emergencia: En la Figura 45 se muestra el flujo de eventos por los que el usuario pasa para crear la emergencia. El usuario introducirá la información necesaria en los campos y se guardará esta información en la base de datos, mostrándose posteriormente en el mapa. Además, los usuarios que se encuentren dentro del área donde se ha creado la emergencia con respecto a su radio de notificación recibirán una notificación correspondiente con todos los detalles.

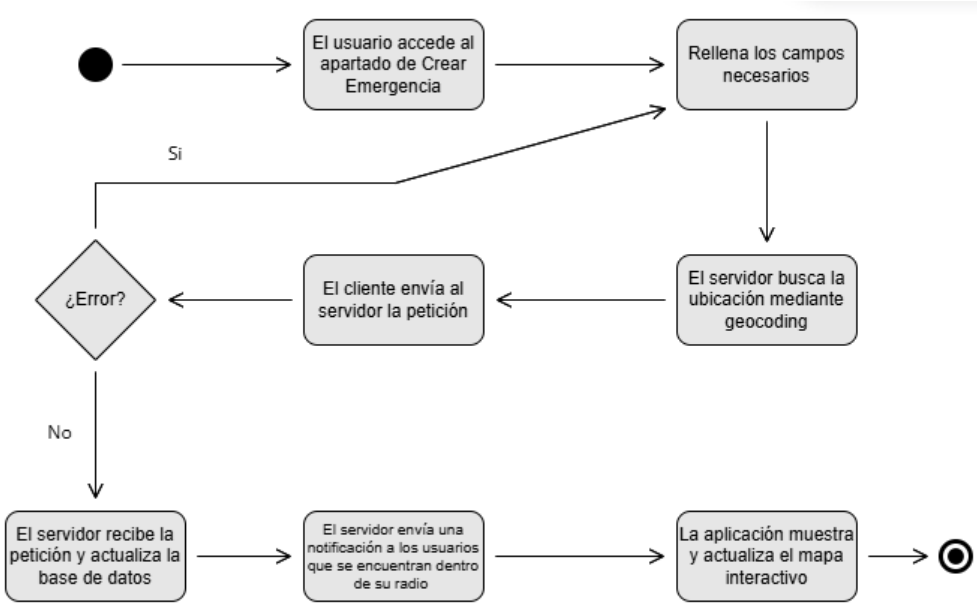

Figura 45: Diagrama de flujo de Creación de Emergencia

Historial de Emergencias: En la Figura 46 se muestran los pasos necesarios para poder acceder y visualizar el historial de todas las emergencias accediendo a su apartado. Con esto se puede conseguir un mayor seguimiento sobre ellas.

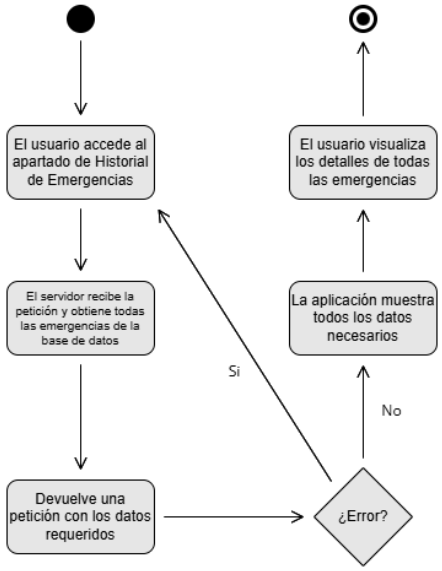

Figura 46: Diagrama de flujo de Historial de Emergencias

Resolución de emergencias: En la Figura 47 se muestran los eventos que se producen para resolver una emergencia, donde difiere la forma de hacerlo según el rol. En el caso de ser ciudadano, se puede reportar la emergencia si consideras que se ha resuelto. Si esa emergencia llega a los 5 reportes de ciudadanos distintos, la emergencia se resolverá automáticamente. Sin embargo, en el caso de no ser

ciudadano, los usuarios podrán resolver la emergencia directamente si lo consideran necesario.

Edición de emergencias: En la Figura 48 se muestra en el diagrama cómo se edita o se solicita editar una emergencia según tu rol distinguiendo de nuevo si el usuario es ciudadano o no. En el caso de ser ciudadano, el usuario puede solicitar editar la emergencia con la información deseada. Esa información será guardada en la base de datos y posteriormente revisada por expertos o autoridades para ser resuelta. Y en el caso de no ser ciudadano, el usuario podrá editar directamente los campos deseados con la información correspondiente. Además, los usuarios que se encuentren dentro del radio de notificación serán notificados de dichos cambios.

Gestión solicitudes edición de emergencias: En la Figura 49 se muestra el diagrama de flujo sobre la gestión de solicitudes de edición de emergencias en el que representa distintos eventos según el rol.
En el caso de ser ciudadano, lo único que se puede hacer respecto a la gestión de solicitudes de edición de emergencias es consultar todas sus solicitudes. Es decir, consultar los detalles y cambios solicitados, además del estado en que se encuentra la solicitud. Y, en el caso de ser experto o autoridad, podrán aceptar o rechazar dichas solicitudes. En el caso de aceptarlas se modificarán los datos correspondientes en la base de datos y además se notificará a los usuarios que se encuentren dentro del radio de notificaciones.

Gestión de datos personales: En la Figura 50 se muestran los eventos en el que el usuario puede gestionar sus datos personales. En este caso los usuarios podrán editar o añadir nueva información sobre ellos en su perfil.

Solicitud de cambio de rol: En la Figura 51 se muestra el diagrama correspondiente a la solicitud de cambio de rol. En el caso de que el usuario ya tenga una solicitud pendiente de cambio de rol no podrá solicitar otra y deberá esperar a que un administrador resuelva esta solicitud. Y en el caso de que no tenga ninguna solicitud pendiente, el usuario podrá guardar la solicitud con el rol deseado.

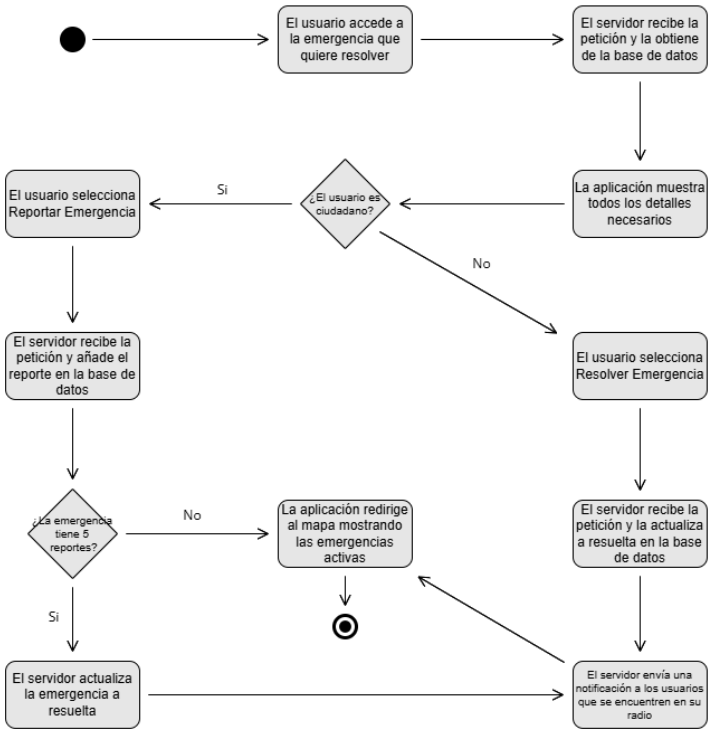

Figura 47: Diagrama de flujo de resolución de Emergencias

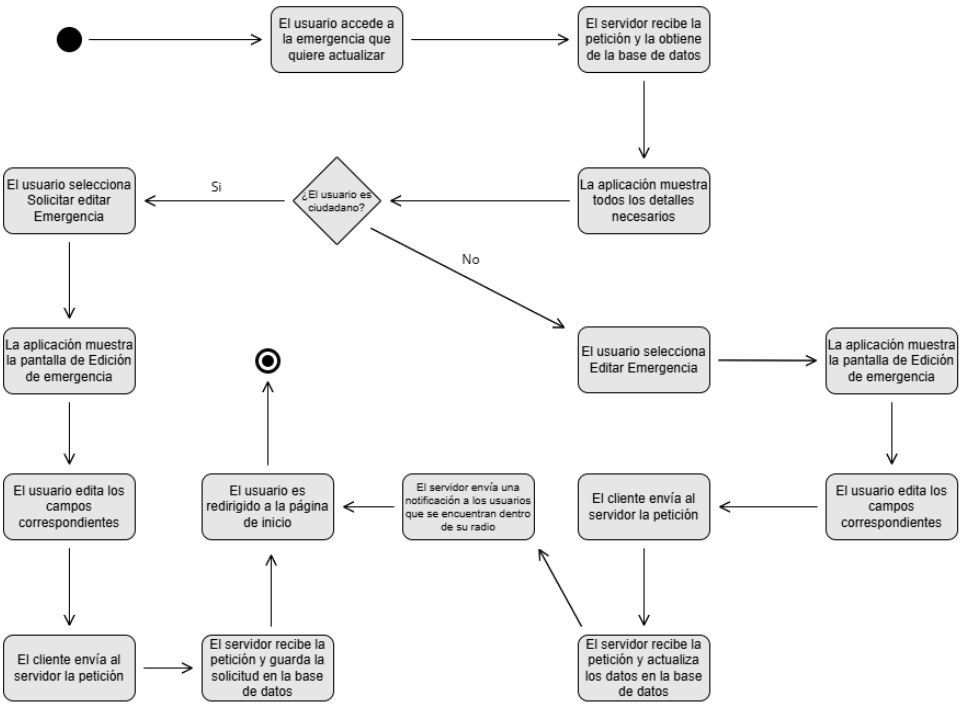

Figura 48: Diagrama de flujo de edición de Emergencias

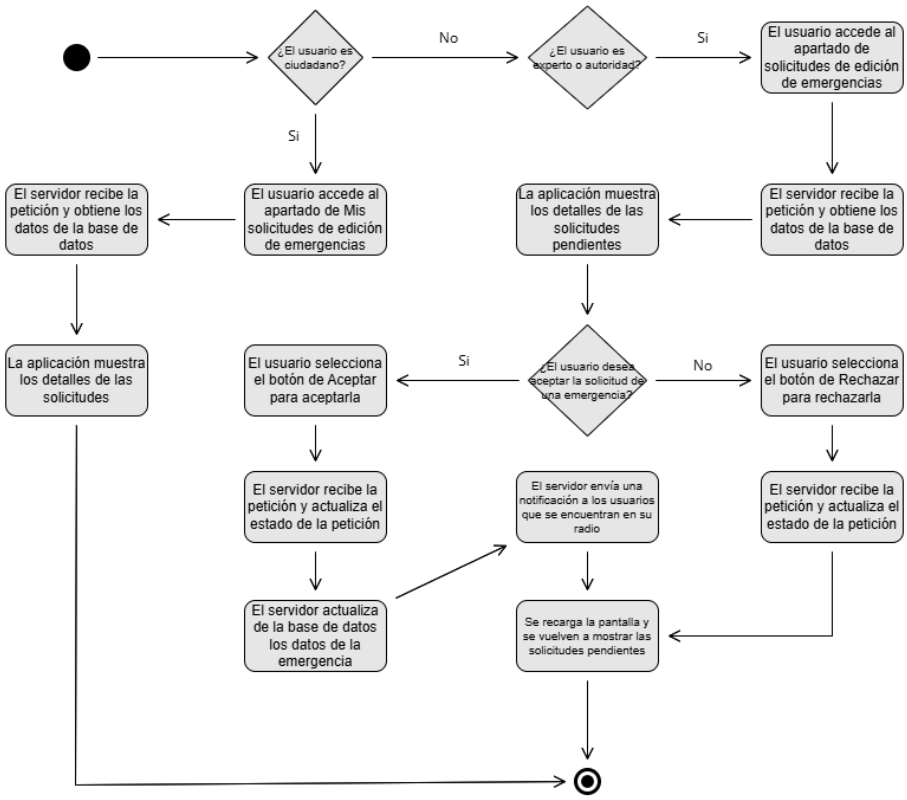

Figura 49: Diagrama de flujo de Gestión de solicitudes de edición de emergencias

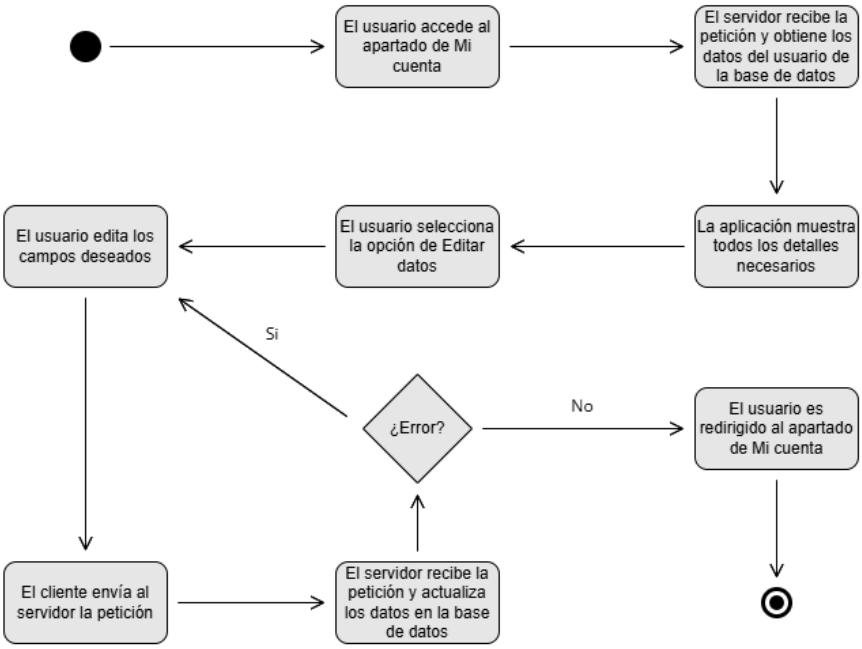

Figura 50: Diagrama de flujo de gestión de datos personales

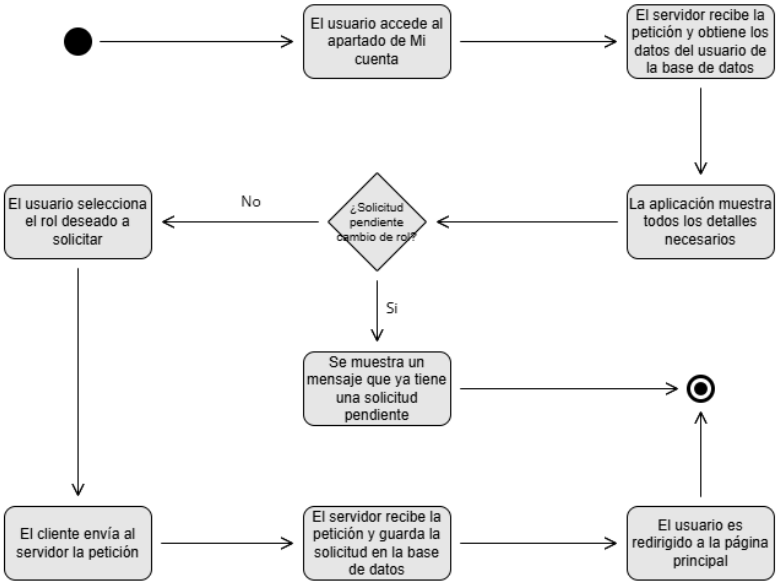

Figura 51: Diagrama de flujo de solicitud de cambio de rol

Gestión solicitudes cambio de rol: En la Figura 52 se muestra el diagrama de flujo sobre la gestión de solicitudes de cambio de rol. En el caso de no ser administrador, lo único que se puede hacer es consultar todas las solicitudes de cambio de rol que ha realizado con su información y estado correspondiente. Sin embargo, si es administrador, podrá aceptar o rechazar dichas solicitudes. En el caso de aceptarla, se modificará la información correspondiente en la base de datos.

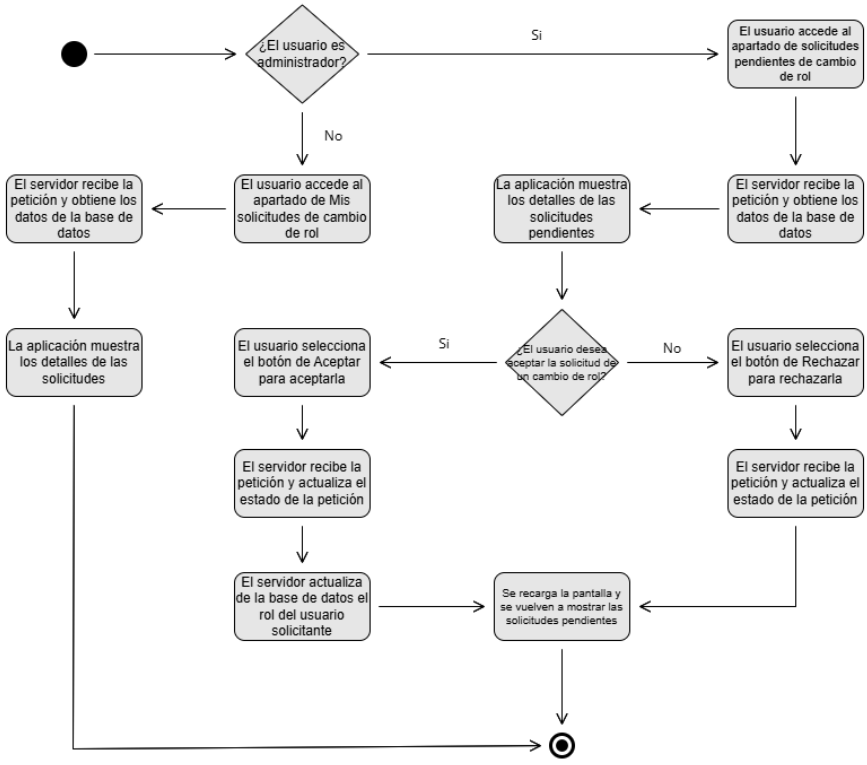

Figura 52: Diagrama de flujo de gestión de solicitudes de cambio de rol

3.6 Modelo de interacción

3.6.1 Diagrama de Secuencia

Con los diagramas de secuencia se va a definir la interacción entre las distintas partes del sistema cuando se realiza alguna función. En todos los diagramas excepto en el de autenticación se supone que el usuario ya se ha autenticado para hacerlo más simple.

Autenticación: En la Figura 53 se puede ver con detalle todas las interacciones entre las diferentes partes del sistema para autenticarse.

Crear emergencia: En la Figura 54 se muestra el diagrama de secuencia correspondiente al proceso de crear una emergencia.

Historial de Emergencias: En la Figura 55 se muestra el diagrama de secuencia correspondiente al historial de emergencias.

Resolución de Emergencias: En la Figura 56 se muestra el diagrama de secuencia sobre la resolución de emergencias.

Edición de emergencias: En la Figura 57 se muestra el diagrama de secuencia correspondiente a la edición de emergencias.

Gestión solicitudes edición de emergencias: En la Figura 58 se muestra el diagrama de secuencia correspondiente a la gestión de solicitudes de edición de emergencias.

Gestión de datos personales: En la Figura 59 se muestra el diagrama de secuencia correspondiente a la gestión de datos personales.

Solicitud de cambio de rol: En la Figura 60 se muestra el diagrama de secuencia correspondiente a la solicitud de cambio de rol.

Gestión solicitudes cambio de rol: En la Figura 61 se muestra el diagrama de secuencia correspondiente a la gestión de solicitudes de cambio de rol.

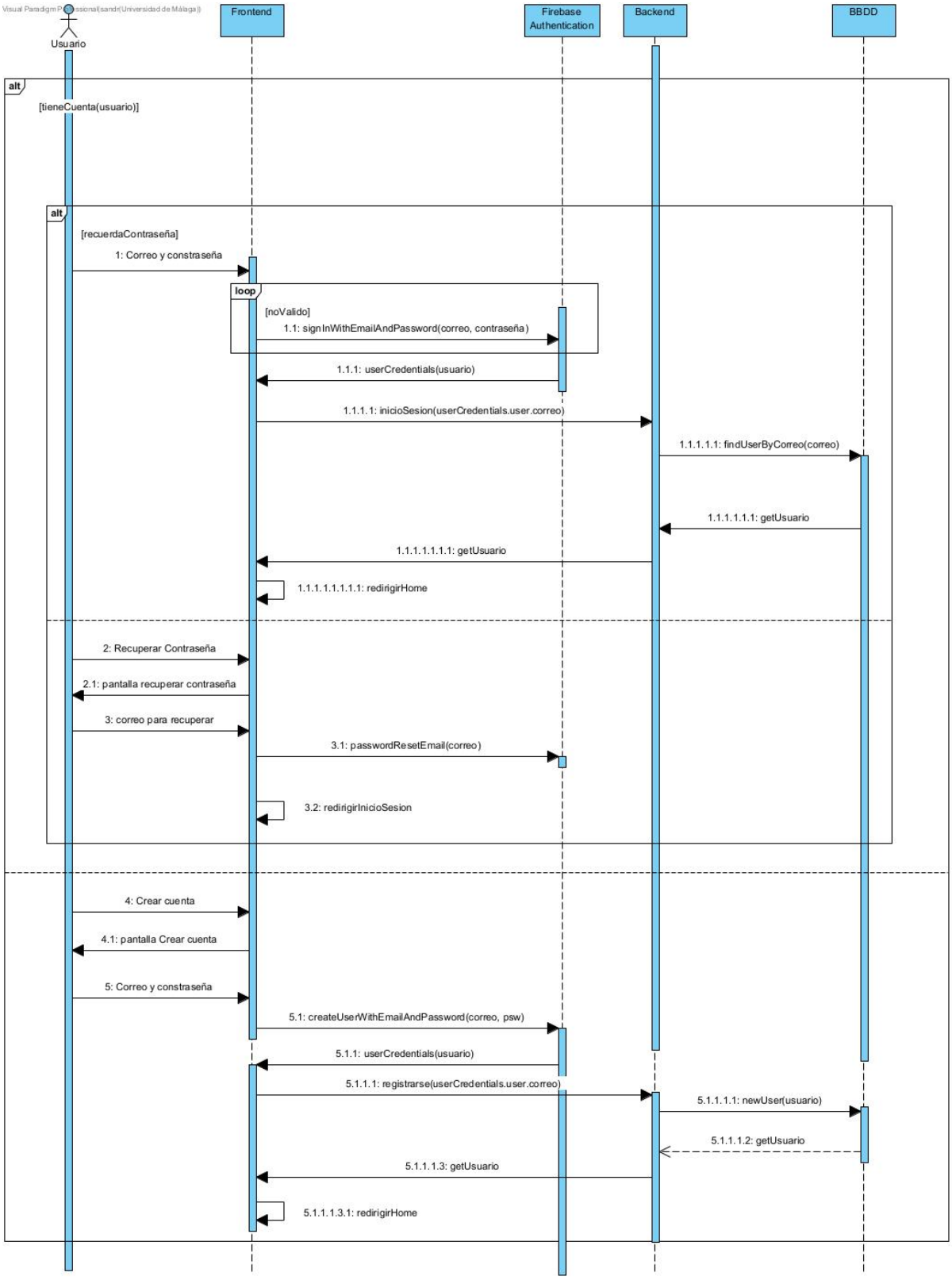

Figura 53: Diagrama de secuencia Autenticación

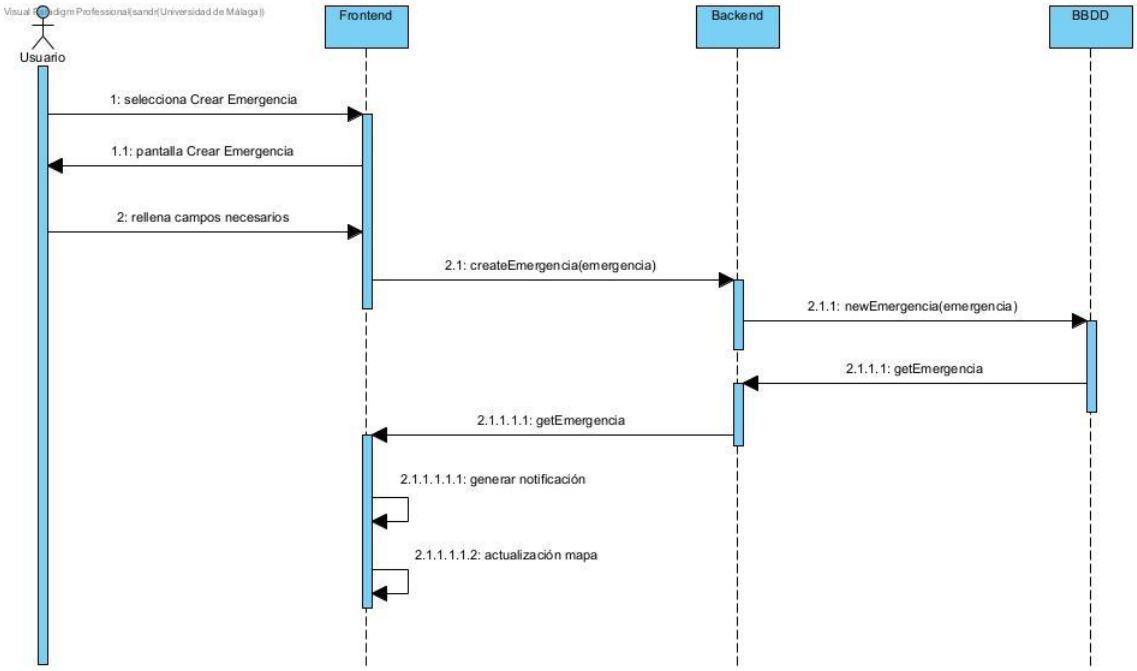

Figura 54: Diagrama de secuencia Crear emergencia

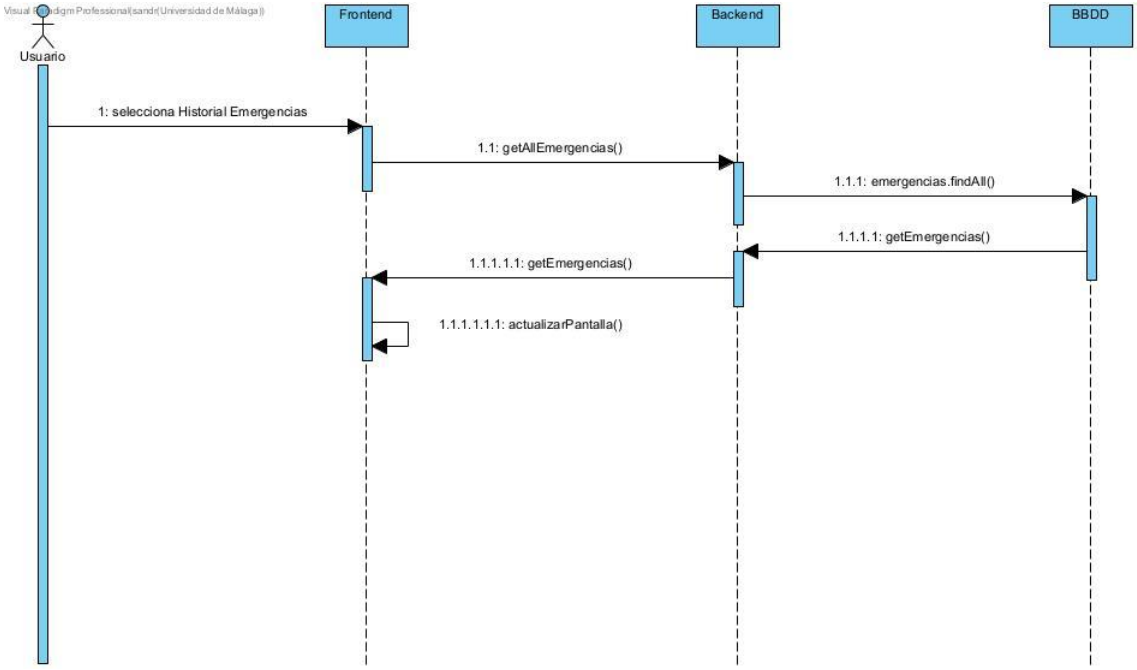

Figura 55: Diagrama de secuencia Historial de emergencias

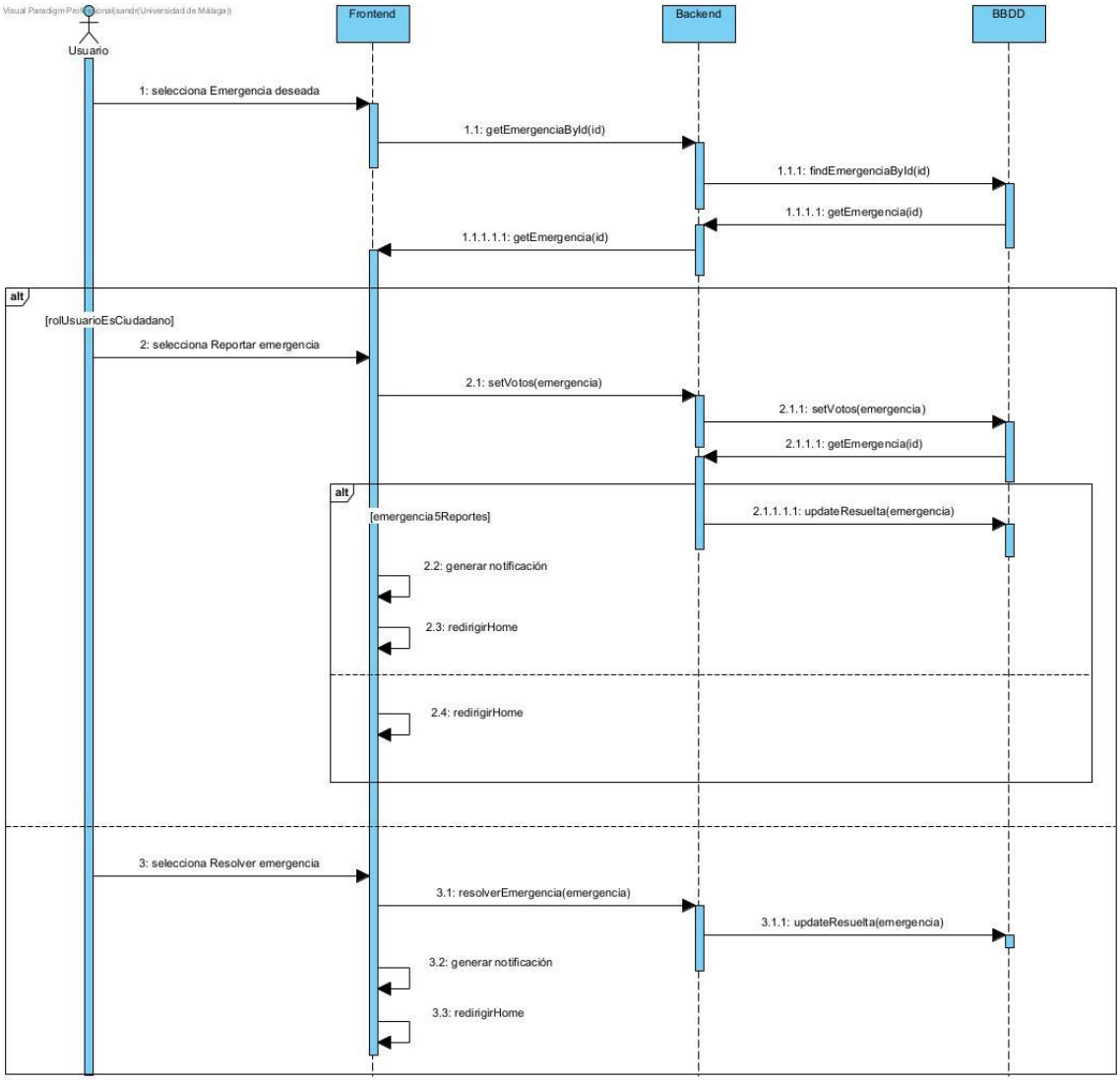

Figura 56: Diagrama de secuencia Resolución de emergencias

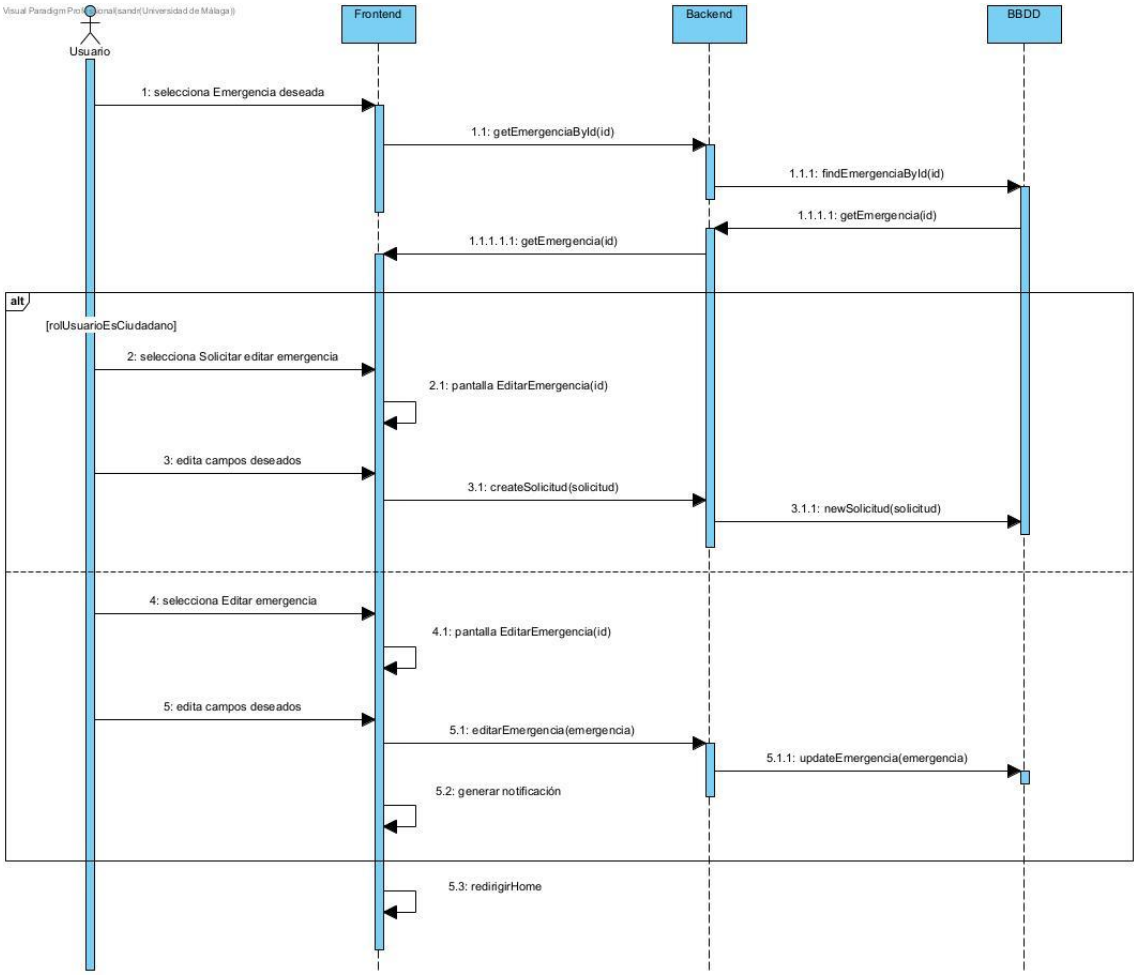

Figura 57: Diagrama de secuencia Edición de emergencias

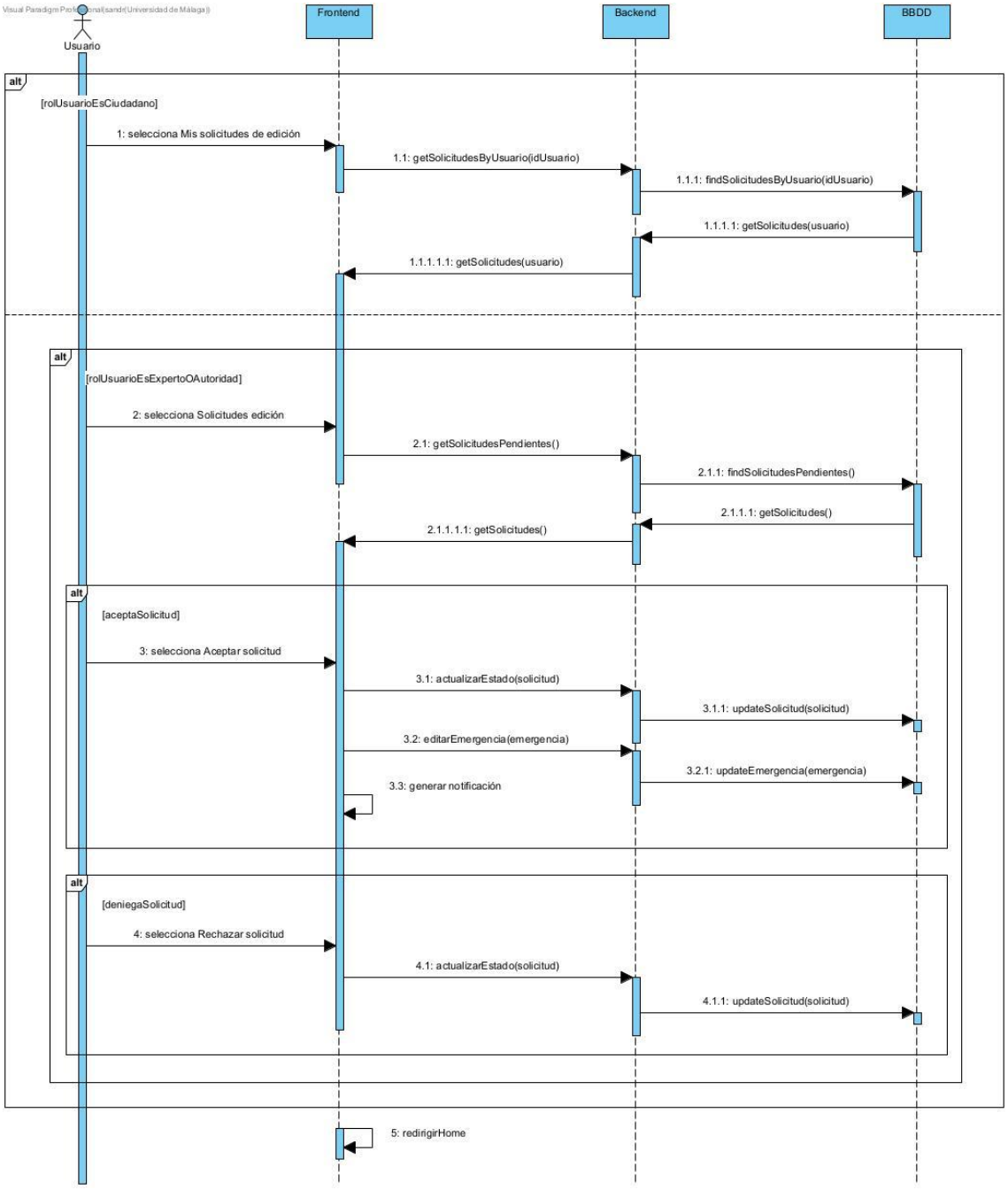

Figura 58: Diagrama de secuencia de Gestión solicitudes edición de emergencias

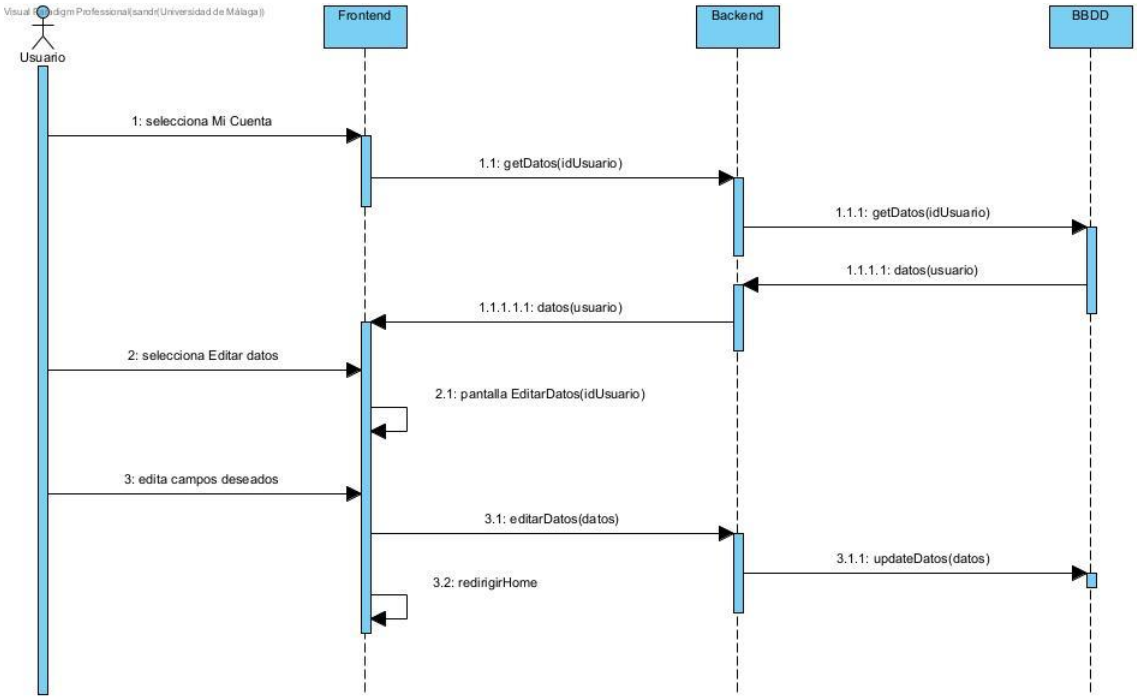

Figura 59: Diagrama de secuencia de Gestión de datos personales

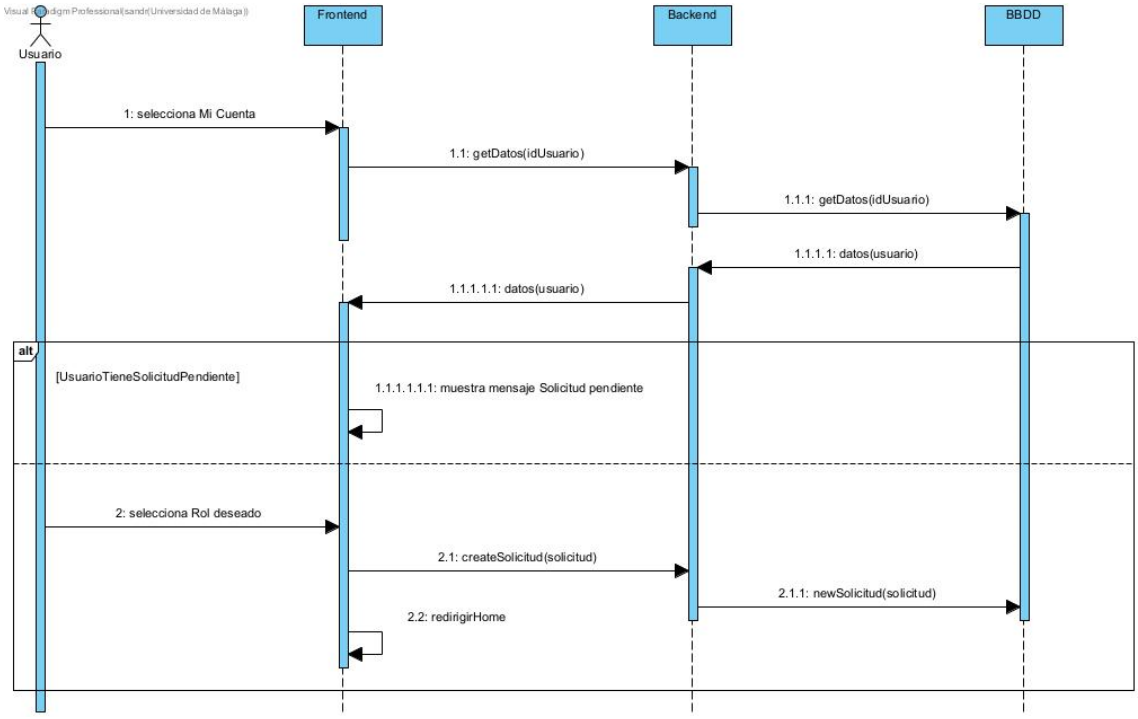

Figura 60: Diagrama de secuencia de Solicitud de cambio de rol

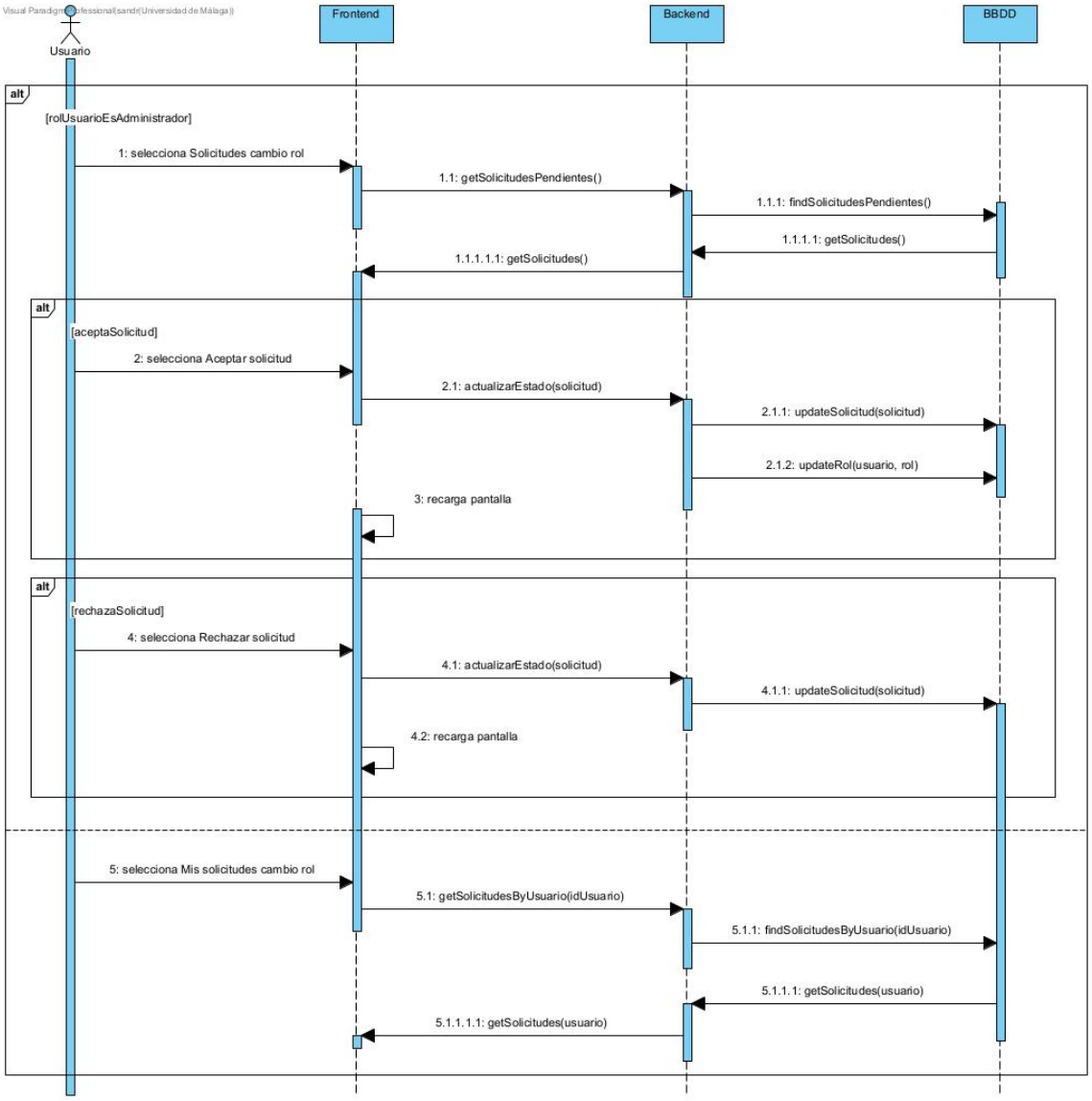

Figura 61: Diagrama de secuencia de Gestión de solicitudes de cambio de rol

4

Modelado y Diseño del sistema

En este capítulo se aborda el proceso de modelado y diseño del sistema, centrado en tres áreas clave para el desarrollo de la aplicación: la estructura de la base de datos, el diseño de la interfaz de usuario (UI) y la arquitectura general del sistema.

4.1 Base de Datos

Se ha utilizado una base de datos relacional, en concreto MySQL, la cual organiza la información en tablas y permite relaciones entre ellas, pudiendo realizar consultas estructuradas mediante SQL. Con este tipo de base de datos se ha conseguido la integridad y consistencia de los datos necesaria y además se asegura que las operaciones sean ACID, es decir, atómicas, consistentes, aisladas y durables.

4.1.1 Modelo E/R

En la Figura 62 se muestra el diagrama E/R diseñado para la aplicación. Este diagrama tiene como objetivo cumplir con todos los requisitos de la aplicación, añadiendo todos los campos necesarios en cada caso para poder hacer un correcto seguimiento.

En este diagrama se pueden observar 2 entidades principales: La entidad Emergencia, la cual representa una emergencia con todos sus campos correspondientes. Además, se puede ver cómo tiene claves foráneas a otras entidades como el tipo, nivel, localización y usuario que creó la emergencia para poder hacer referencia a esos datos también.

La otra entidad principal sería la de Usuario, la cual representa los distintos usuarios que se encuentran en la aplicación, donde se crea una nueva entrada cada vez que un usuario se registra. En ella se puede observar que tiene dos claves foráneas, la clave de su rol, la cual indica el rol que desempeña dentro de la aplicación y la clave foránea de la localización, la cual es muy importante debido a que la localización del usuario se está actualizando cada cierto tiempo para permitir que si un usuario se mueve pueda recibir las notificaciones de las emergencias que se encuentran en su radio con respecto a su actual localización.

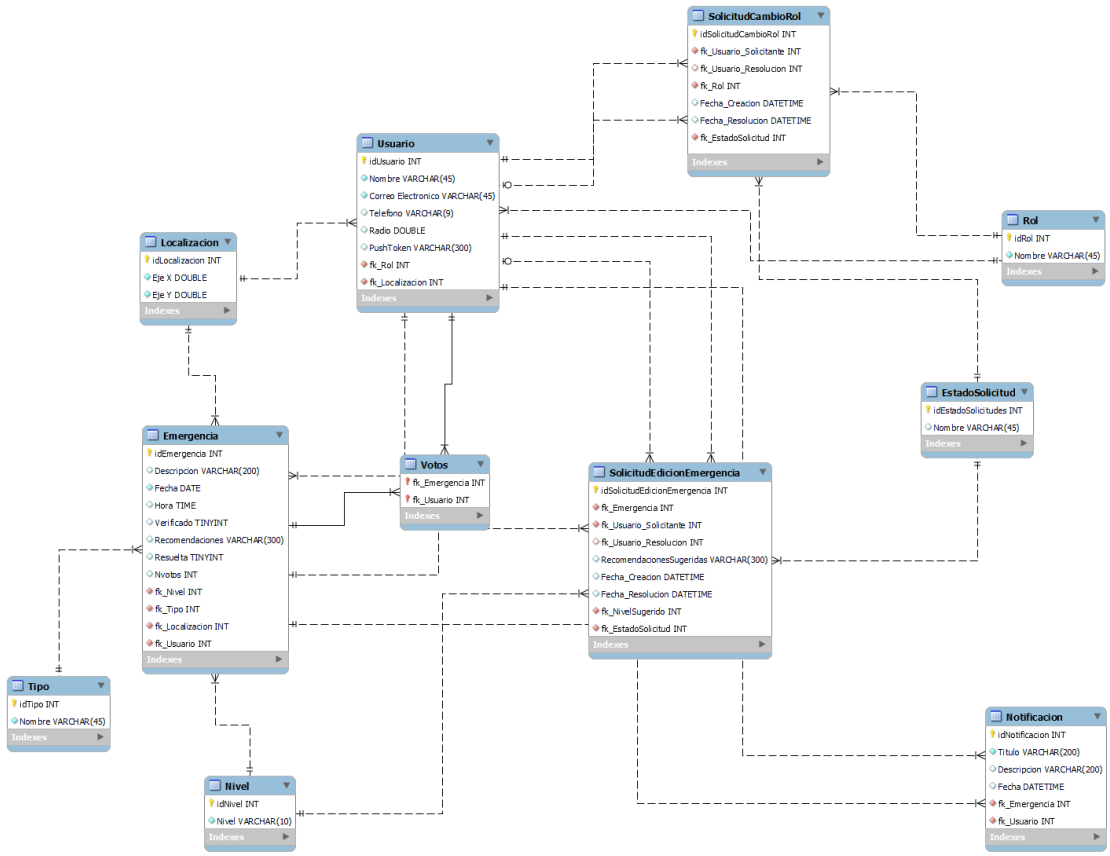

Figura 62: Modelo Entidad Relación de la aplicación

Otra tabla que se puede destacar es la de Notificación. Dicha tabla contiene la información necesaria para informar a los usuarios cuando cualquier cambio en las emergencias se produzca, e incluso contiene el usuario que hizo que se genere dicha notificación.

Además, se pueden ver dos entidades de solicitudes, una sobre las solicitudes de edición de emergencias, las cuales muestra los nuevos campos que se pretenden cambiar y el autor que la solicita. Además, en el caso en que el estado de la solicitud esté resuelto (aceptada o rechazada) también saldrá quién la resolvió y el momento en que se resolvió. La otra entidad sobre solicitudes es la de cambio de rol, la cual presenta también el rol que un usuario solicita, el usuario solicitante, y en caso de estar resuelta la solicitud, el usuario y fecha de cuando se resolvió.

También se puede observar la tabla intermedia Votos, la cual representa cuando un ciudadano reporta una emergencia, para evitar que ese mismo ciudadano pueda volver a reportar la misma emergencia.

4.2 Diseño de la interfaz de usuario

Se ha decidido realizar una interfaz de usuario para móviles, en concreto Android, ya que, al ser una aplicación sobre emergencias, lo más lógico sería poder acceder con el móvil porque es lo que se tiene en cualquier lugar y momento. Para ello, se decidió elegir React Native para desarrollar el Frontend ya que permite un desarrollo eficiente y la reutilización de código mediante la utilización de componentes. Aunque en este caso esté diseñada solo para Android, la reutilización de código facilita futuras expansiones a iOS. Además, React Native utiliza componentes nativos que mejoran significativamente el rendimiento, y la integración con Expo ha facilitado el desarrollo debido a que permite desplegar y probar la aplicación rápidamente y acceder a funcionalidades nativas como la geolocalización y notificaciones sin necesidad de escribir código nativo. También permite crear interfaces dinámicas y adaptables utilizando estilos similares a CSS.

Durante el diseño se ha intentado crear una aplicación lo más intuitiva posible y sin cargar de información las pantallas debido a que como es una aplicación sobre emergencias, deberías poder encontrar lo necesario lo más rápido posible.

4.3 Diagrama de arquitectura y tecnologías de la aplicación

En la Figura 63 se muestra el diseño de la arquitectura de la aplicación y las tecnologías empleadas. La arquitectura del sistema se ha desarrollado de forma modular y eficiente. El backend ha sido implementado utilizando Spring Boot, con control de versiones gestionado a través de un repositorio en GitHub, lo que ha facilitado su integración y posterior despliegue en la plataforma Railway. Asimismo, la base de datos MySQL ha sido alojada en Railway, permitiendo una conexión directa y segura con el backend. Por otro lado, el frontend se ha

desarrollado con React Native, y gracias a Expo se ha generado el archivo APK para Android, posibilitando su instalación en dispositivos móviles y su acceso por parte del usuario final.

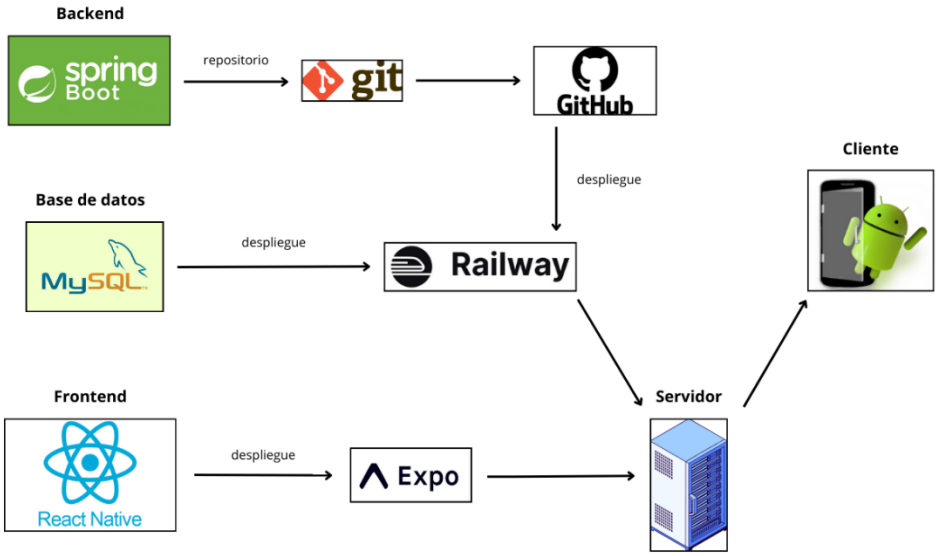

Figura 63: Diagrama de arquitectura de la aplicación

5

Implementación y pruebas

El desarrollo de la aplicación se ha llevado a cabo siguiendo un enfoque ágil, lo que ha permitido una evolución progresiva del proyecto mediante iteraciones incrementales, en el que se han implementado incrementalmente los casos de uso en el backend y frontend, asegurando que las funcionalidades añadidas fueran probadas y validadas constantemente para garantizar su correcto funcionamiento.

El orden de desarrollo seguido ha sido el siguiente:

- Diseño del modelo relacional
- Creación del proyecto backend
- Creación del proyecto frontend
- Implementación del mapa interactivo
- Implementación de la visualización de los símbolos de las emergencias
- Implementación de los detalles de las emergencias
- Implementación de la creación de emergencias
- Implementación de la edición de emergencias
- Implementación del inicio de sesión y registro
- Implementación de los detalles de la cuenta
- Implementación de las solicitudes

Este enfoque ha permitido desarrollar la aplicación de manera estructurada y flexible, asegurando que cada nueva funcionalidad se integre de forma coherente con las anteriores, manteniendo un proceso iterativo de mejora continua.

5.1 Diseño del modelo relacional y módulo de carga de datos

Para el diseño y creación de la base de datos, se ha utilizado MySQL Workbench, una herramienta que facilita la generación de modelos relacionales de manera visual e intuitiva. Gracias a su interfaz gráfica, ha sido posible definir las tablas y establecer sus relaciones de forma clara y estructurada.

El proceso de desarrollo de la base de datos se ha llevado a cabo en las siguientes etapas:

1. ***Definición de las tablas:*** Se crearon todas las tablas necesarias, especificando su nombre, atributos y tipos de datos correspondientes.

2. ***Establecimiento de relaciones:*** Se definieron las relaciones entre las tablas, asegurando la coherencia del modelo mediante la inclusión de claves foráneas.

3. ***Inserción de datos iniciales:*** Se añadieron datos de prueba mediante consultas SQL (INSERT) para su posterior utilización durante el desarrollo y las pruebas de la aplicación.

Este enfoque permitió garantizar una base de datos bien estructurada, facilitando su integración con el backend y asegurando un correcto manejo de la información.

5.2 Desarrollo del backend

5.2.1 Estructura del proyecto

El desarrollo del backend se ha realizado en Spring Boot, el cual facilita la creación de servicios REST. Su principal función es procesar las solicitudes provenientes del frontend y gestionar la comunicación con la base de datos.
Para organizar el código y facilitar su mantenimiento, se ha seguido una arquitectura en capas, dividiendo la aplicación en módulos bien definidos. A continuación en la Figura 64 se describe cada una de estas capas.

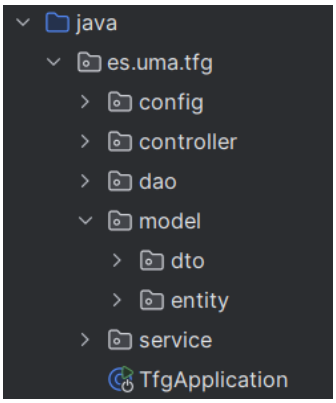

Figura 64: Estructura Backend

- ***Controllers:*** En esta capa se encuentran los controladores, los cuales contienen los endpoints REST que permiten la comunicación con el frontend. Su función principal es recibir las solicitudes HTTP, procesarlas y delegarlas a los servicios correspondientes. La Figura 65 muestra un ejemplo de un controlador que obtiene una emergencia por su ID.

```
@GetMapping("/{id}")
public EmergenciaDTO getEmergenciaById(@PathVariable int id){
    return emergenciaService.getEmergenciaById(id);
}
```

Figura 65: Endpoint carpeta controllers

En este caso, el controlador recibe una solicitud GET, extrae el ID de la emergencia desde la URL y delega la lógica al servicio correspondiente.

Además, se han realizado pruebas a los endpoints de los controladores utilizando la herramienta Postman, con el objetivo de verificar su correcto funcionamiento y asegurar que las respuestas del backend sean las esperadas.

- ***Servicios:*** Esta capa contiene la lógica de negocio de la aplicación. Los servicios se encargan de procesar los datos y comunicarse con la capa de repositorios para obtener información de la base de datos.

Ejemplo en la Figura 66 de un método de servicio que obtiene una emergencia por su ID.

```
public EmergenciaDTO getEmergenciaById(int id) {  1 usage
    Emergencia emergencia = emergenciaRepository.findById(id).orElse( other: null);
    return emergencia.toDTO();
}
```

Figura 66: Método de la capa de Servicios

En este caso, el servicio consulta el repositorio, obtiene una entidad Emergencia y la convierte a un DTO antes de devolverla al controlador.

- ***Repositorios:*** La capa de repositorios es la encargada de realizar las consultas a la base de datos utilizando Spring Data JPA. Se puede visualizar un ejemplo en la Figura 67.

```
@Repository  5 usages
public interface EmergenciaRepository extends JpaRepository<Emergencia, Integer> {
}
```

Figura 67: Capa Repository

Gracias a JpaRepository, se pueden realizar operaciones CRUD sin necesidad de escribir consultas SQL manualmente.

- ***Capa de modelos:*** Esta capa contiene las entidades que representan las tablas de la base de datos. Cada entidad es una clase con atributos que reflejan las columnas de la tabla correspondiente.

En la Figura 68 se puede observar el ejemplo de una entidad.

```
@Entity
@Table(name = "Notificacion")
public class Notificacion implements DTO<NotificacionDTO> {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "idNotificacion", nullable = false)
    private Integer id;

    @Column(name = "Titulo", nullable = false, length = 200)
    private String titulo;

    @Column(name = "Descripcion", length = 200)
    private String descripcion;

    @Column(name = "Fecha")
    private Instant fecha;

    @ManyToOne(fetch = FetchType.LAZY, optional = false)
    @JoinColumn(name = "fk_Emergencia", nullable = false)
    private Emergencia fkEmergencia;

    @ManyToOne(fetch = FetchType.LAZY, optional = false)
    @JoinColumn(name = "fk_Usuario", nullable = false)
    private Usuario fkUsuario;
```

Figura 68: Entidad base de datos en backend

En este caso se puede observar que esta clase pertenece a la entidad Notificación y sus atributos representan el nombre de la columna en la base de datos. En el caso de ser una clave foránea, además, incluye la relación que posee con dicha entidad como es el caso de fkEmergencia.

-Capa DTO (Data Transfer Object): Para evitar exponer directamente las entidades de la base de datos, se utilizan objetos DTO (Data Transfer Object), los cuales contienen solo la información necesaria para transferir datos entre capas. En la Figura 69 se puede observar un ejemplo de una clase DTO.

```
@Getter  35 usages
@Setter
public class NotificacionDTO implements Serializable {

    private Integer id;
    private String titulo;
    private String descripcion;
    private Instant fecha;
    private Integer fkEmergencia;
    private Integer fkUsuario;
}
```

Figura 69: Ejemplo clase DTO

Este enfoque mejora la seguridad y el rendimiento de la aplicación.

- ***Configuración del backend:*** También se ha incluido una capa de configuración, donde se definen aspectos importantes como la gestión del CORS (Cross-Origin Resource Sharing), evitando bloqueos en las solicitudes entre el frontend y el backend.

5.2.2 Base de datos en el Backend

La conexión entre el backend y la base de datos se configura en el archivo application.properties, donde se establecen los parámetros de conexión utilizando Spring DataSource.
En este archivo se especifican:

- La URL de la base de datos.
- El usuario y la contraseña de acceso.

Las tablas de la base de datos están representadas en el código mediante entidades de Spring Data JPA, utilizando la anotación @Entity. Cada entidad define tanto sus atributos como sus relaciones con otras entidades.

Respecto a las consultas, Spring Boot gestiona las consultas a la base de datos mediante Spring Data JPA, lo que permite realizar operaciones CRUD sin necesidad de escribir consultas SQL manualmente.
Gracias a esta integración, se optimiza el acceso a la base de datos y se reduce la cantidad de código necesario para gestionar las consultas.

5.2.3 Endpoints destacados

1. *Emergencias en el mapa interactivo*
   Para mostrar las emergencias en el mapa interactivo es necesario mostrar las que no se han resuelto aún para mantenerlo actualizado. Por lo tanto, es necesario el endpoint de la Figura 70.

```
@GetMapping("/noResueltas")
public List<EmergenciaDTO> getEmergenciasNoResueltas(){
    List<EmergenciaDTO> emergencias = emergenciaService.getEmergenciasNoResueltas();
    return emergencias;
}
```

Figura 70: Endpoint Emergencias No resueltas

En la Figura 71 se puede visualizar el método en la capa Service.

```
public List<EmergenciaDTO> getEmergenciasNoResueltas() {  1 usage
    List<Emergencia> emergencias = emergenciaRepository.findByNoResueltas();
    return entidadesADTO(emergencias);
}
```

Figura 71: Método en la capa servicio Emergencias no resueltas

Y en la Figura 72 se puede visualizar el método en la capa Repository.

```
@Repository  5 usages
public interface EmergenciaRepository extends JpaRepository<Emergencia, Integer> {

    @Query("select a from Emergencia a where a.resuelta=0 and a.nvotos<5")  1 usage
    List<Emergencia> findByNoResueltas();
}
```

Figura 72: Consulta Base de datos Emergencias no resueltas

Es decir, para mostrar las emergencias no resueltas es necesario hacer una consulta a la base de datos desde el repositorio, y serán las emergencias que no hayan sido resueltas por un experto o autoridad (a.resuelta=0) y las que tengan menos de cinco reportes de ciudadanos (a.nvotos<5).

2. *Creación de emergencias*

Para guardar las emergencias se utiliza el método POST, como se puede ver en la Figura 73.

```
@PostMapping("/")
public EmergenciaDTO createEmergencia(@RequestBody EmergenciaDTO emergencia){
    return emergenciaService.createEmergencia(emergencia);
}
```

Figura 73: Endpoint para crear una emergencia

En la Figura 74 se puede visualizar el método en la capa Service.

```
public EmergenciaDTO createEmergencia(EmergenciaDTO emergencia) {  1 usage
    Emergencia nueva = new Emergencia();
    nueva.setId(nueva.getId());
    nueva.setDescripcion(emergencia.getDescripcion());
    nueva.setFecha(emergencia.getFecha());
    nueva.setHora(emergencia.getHora());
    nueva.setVerificado(emergencia.getVerificado());
    nueva.setRecomendaciones(emergencia.getRecomendaciones());
    nueva.setResuelta(emergencia.getResuelta());
    nueva.setNvotos(emergencia.getNVotos());

    Nivel nivel = nivelRepository.findById(emergencia.getFkNivel()).orElse( other: null);
    nueva.setFkNivel(nivel);

    Tipo tipo = tipoRepository.findById(emergencia.getFkTipo()).orElse( other: null);
    nueva.setFkTipo(tipo);

    Localizacion localizacion = localizacionRepository.findById(emergencia.getFkLocalizacion()).orElse( other: null);
    nueva.setFkLocalizacion(localizacion);

    Usuario usuario = usuarioRepository.findById(emergencia.getFkUsuario()).orElse( other: null);
    nueva.setFkUsuario(usuario);

    nueva = emergenciaRepository.save(nueva);
    return nueva.toDTO();
}
```

Figura 74: Método en la capa servicio para crear una emergencia

El controlador recibe el objeto de emergencia como parámetro, y el servicio se encarga de crear el objeto correspondiente y guardarlo en la base de datos.

3. *Editar una emergencia*

Para editar las emergencias se utiliza el método PUT, como se puede visualizar en la Figura 75.

```
@PutMapping("/")
public EmergenciaDTO editarEmergencia(@RequestBody EmergenciaDTO emergenciaDTO){
    return emergenciaService.editarEmergencia(emergenciaDTO);
}
```

Figura 75: Endpoint para editar una emergencia

En la Figura 76 se puede visualizar el método en la capa Service.

```java
public EmergenciaDTO editarEmergencia(EmergenciaDTO emergenciaDTO) {  1 usage
    Emergencia emergencia = emergenciaRepository.findById(emergenciaDTO.getId()).orElse( other: null);
    emergencia.setRecomendaciones(emergenciaDTO.getRecomendaciones());

    emergencia.setVerificado(emergenciaDTO.getVerificado());
    emergencia.setResuelta(emergenciaDTO.getResuelta());
    emergencia.setNvotos(emergenciaDTO.getNVotos());

    Nivel nivel = nivelRepository.findById(emergenciaDTO.getFkNivel()).orElse( other: null);
    emergencia.setFkNivel(nivel);

    emergenciaRepository.save(emergencia);
    return emergencia.toDTO();
}
```

Figura 76: Método en la capa servicio para editar una emergencia

En este caso recibe el objeto emergencia a editar con los campos necesarios y lo que se hace es encontrar el objeto por el id y modificar los campos correspondientes, para posteriormente actualizarlo en la base de datos.

4. *Nueva localización*

El endpoint de la Figura 77 se encarga de crear una nueva localización en caso de que no exista. Esto puede ocurrir cuando un usuario se registra, inicia sesión o se desplaza. Asimismo, si al crear una emergencia la localización asociada no está previamente registrada, se procederá a guardarla.

```java
@PostMapping("/")
public LocalizacionDTO createLocalizacion(@RequestBody LocalizacionDTO localizacion){
    LocalizacionDTO nueva = localizacionService.createLocalizacion(localizacion);
    return nueva;
}
```

Figura 77: Endpoint para crear una nueva localización

En la Figura 78 se puede visualizar el método en la capa Service.

En definitiva, si la localización ya existe, se devuelve; de lo contrario, se crea una nueva.

```java
public LocalizacionDTO createLocalizacion(LocalizacionDTO localizacion) {  1 usage
    Localizacion obtener = localizacionRepository.findByEjeXEjeY(localizacion.getEjeX(), localizacion.getEjeY());
    if(obtener != null) {
        return obtener.toDTO();
    }else{
        Localizacion nueva = new Localizacion();
        nueva.setEjeX(localizacion.getEjeX());
        nueva.setEjeY(localizacion.getEjeY());

        localizacionRepository.save(nueva);
        return nueva.toDTO();
    }

}
```

Figura 78: Método en la capa servicio para crear una nueva localización

5. *Notificación generada al crear una Emergencia*
   Cuando se crea una emergencia, este endpoint registra automáticamente la notificación correspondiente, incluyendo todos los campos necesarios para su posterior tratamiento y envío a los usuarios afectados. Esto se puede visualizar en la Figura 79.

```
@PostMapping("/")
public NotificacionDTO createNotificacion(@RequestBody NotificacionDTO notificacionDTO){
    NotificacionDTO res = notificacionService.createNotificacion(notificacionDTO);
    return res;
}
```

Figura 79: Endpoint para crear una nueva notificación

En la Figura 80 se puede visualizar el método en la capa Service. Es decir, se crea la nueva notificación y se completan todos los campos necesarios.

```
public NotificacionDTO createNotificacion(NotificacionDTO notificacionDTO) {  1 usage
    Notificacion notificacion = new Notificacion();

    Emergencia emergencia = emergenciaRepository.findById(notificacionDTO.getFkEmergencia()).orElse( other: null);

    Nivel nivel = nivelRepository.findById(emergencia.getFkNivel().getId()).orElse( other: null);
    notificacion.setTitulo("Emergencia de nivel " + nivel.getNivel() + " cerca de ti");

    Tipo tipo = tipoRepository.findById(emergencia.getFkTipo().getId()).orElse( other: null);
    notificacion.setDescripcion(tipo.getNombre() + ": " + notificacionDTO.getDescripcion());

    notificacion.setFecha(Instant.now());
    notificacion.setFkEmergencia(emergencia);

    Usuario usuario = usuarioRepository.findById(notificacionDTO.getFkUsuario()).orElse( other: null);
    notificacion.setFkUsuario(usuario);

    notificacionRepository.save(notificacion);
    return notificacion.toDTO();
}
```

Figura 80: Método en la capa de servicio para crear una nueva notificación

Para notificar a los usuarios en caso de una emergencia, es necesario determinar si se encuentran dentro del radio de alcance definido para dicha alerta, el cual se mide en kilómetros. Dado que las ubicaciones de los usuarios y de las emergencias están almacenadas como coordenadas geográficas (latitud y longitud), se requiere convertir la distancia entre estos puntos en una medida real. Para ello, se ha utilizado la fórmula de Haversine, que permite calcular la distancia entre dos puntos sobre la superficie terrestre considerando la curvatura del planeta.
La fórmula de Haversine, mostrada a continuación, calcula la distancia entre dos coordenadas geográficas utilizando funciones trigonométricas, y resulta especialmente útil para estimaciones de distancia a escala global.

$$a = \sin\left(\frac{\Delta\varphi}{2}\right)^2 + \cos(\varphi_1) \cdot \cos(\varphi_2) \cdot \sin\left(\frac{\Delta\lambda}{2}\right)^2$$

$$c = 2 \cdot atan2\left(\sqrt{a}, \sqrt{1-a}\right)$$

$$d = R \cdot C$$

Donde:

- φ_1 y φ_2 son las latitudes de los dos puntos en radianes.
- $\Delta\varphi$ es la diferencia entre las latitudes de los dos puntos.
- $\Delta\lambda$ es la diferencia entre las longitudes de los dos puntos.
- R es el radio de la Tierra (aproximadamente 6371 km).
- d es la distancia resultante en kilómetros.

Esta fórmula ha sido implementada en el método mostrado en la Figura 81, que se encarga de obtener la distancia en kilómetros entre un usuario y una emergencia.

```
private static double calcularDistancia(double lat1, double lon1, double lat2, double lon2) {  1 usage
    double dLat = Math.toRadians(lat2 - lat1);
    double dLon = Math.toRadians(lon2 - lon1);

    double a = Math.sin(dLat / 2) * Math.sin(dLat / 2) +
            Math.cos(Math.toRadians(lat1)) * Math.cos(Math.toRadians(lat2)) *
                    Math.sin(dLon / 2) * Math.sin(dLon / 2);

    double c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));

    double RADIO_TIERRA = 6371.0;

    return RADIO_TIERRA * c;
}
```

Figura 81: Método que utiliza fórmula de Haversine

Dicho método se utiliza como parte del proceso para filtrar y obtener la lista de usuarios que deben ser notificados en función de su cercanía geográfica con la emergencia. Este proceso puede observarse en la Figura 82.

```java
public List<UsuarioDTO> listaUsuariosNotificar(int idEmergencia) {  1 usage
    Emergencia emergencia = emergenciaRepository.findById(idEmergencia).orElse( other: null);

    Localizacion localizacionEmergencia = emergencia.getFkLocalizacion();
    List<Usuario> usuarios = usuarioRepository.findAll();
    List<UsuarioDTO> res = new ArrayList<>();

    for(Usuario usuario : usuarios) {
        Localizacion localizacionUsuario = usuario.getFkLocalizacion();

        double distancia = calcularDistancia(
                localizacionUsuario.getEjeY(),
                localizacionUsuario.getEjeX(),
                localizacionEmergencia.getEjeY(),
                localizacionEmergencia.getEjeX()
        );

        if (distancia <= usuario.getRadio()) {
            res.add(usuario.toDTO());
        }
    }

    return res;
}
```

Figura 82: Lista de usuarios a notificar

6. Autenticación de usuarios

El endpoint de la Figura 83 gestiona tanto el registro como el inicio de sesión de los usuarios.

```java
@PostMapping("/inicio-sesion")
public UsuarioDTO inicioORegistro(@RequestBody UsuarioDTO usuarioDTO){
    return usuarioService.inicioORegistro(usuarioDTO);
}
```

Figura 83: Endpoint para que un usuario se registre o inicie sesión

En la Figura 84 se puede visualizar el método en la capa Service.

En caso de que el usuario no exista, se crea un nuevo registro en la base de datos. Si ya está registrado, se actualiza su localización y su token de notificaciones push, permitiendo así el correcto envío de alertas dentro del sistema.

```java
public UsuarioDTO inicioORegistro(UsuarioDTO usuarioDTO){  1 usage
    Usuario usuario = usuarioRepository.findUsuarioByEmailIgnoreCase(usuarioDTO.getCorreoElectronico());

    if (usuario==null) {
        usuario = new Usuario();
        usuario.setNombre(usuarioDTO.getNombre());
        usuario.setCorreoElectronico(usuarioDTO.getCorreoElectronico());
        usuario.setTelefono(usuarioDTO.getTelefono());
        usuario.setRadio(3.0);
        usuario.setPushToken(usuarioDTO.getPushToken());

        Rol rol = rolRepository.findById(usuarioDTO.getFkRol()).orElse( other: null);
        usuario.setFkRol(rol);
        Localizacion localizacion = localizacionRepository.findById(usuarioDTO.getFkLocalizacion()).orElse( other: null);
        usuario.setFkLocalizacion(localizacion);

        usuarioRepository.save(usuario);
    }else{
        if(usuario.getFkLocalizacion().getId()!=usuarioDTO.getFkLocalizacion()){
            Localizacion localizacion = localizacionRepository.findById(usuarioDTO.getFkLocalizacion()).orElse( other: null);
            usuario.setFkLocalizacion(localizacion);
        }

        if(usuario.getPushToken()!=usuarioDTO.getPushToken()){
            usuario.setPushToken(usuarioDTO.getPushToken());
        }
        usuarioRepository.save(usuario);
    }
    return usuario.toDTO();
}
```

Figura 84: Método en la capa de servicio para que un usuario se registre o inicie sesión

5.3 Desarrollo del frontend

5.3.1 Estructura

A continuación, en la Figura 85, se describe la estructura seguida para el desarrollo del frontend de la aplicación. Esta organización tiene como objetivo mejorar la legibilidad, modularidad y mantenibilidad del código.

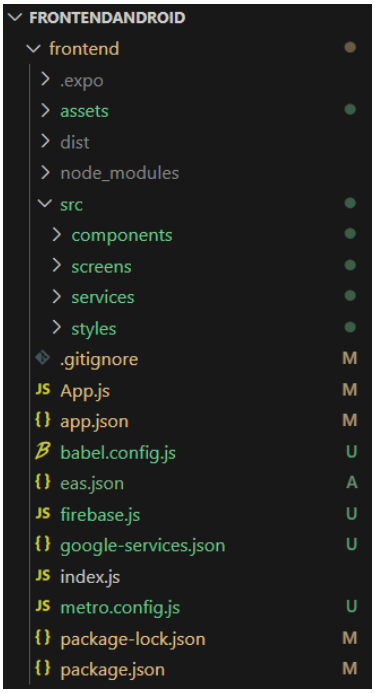

Figura 85: Estructura del frontend

Se ha dividido en las siguientes capas, estructuradas en carpetas:

- ***Components:*** Contiene todos los componentes reutilizables que se integran en las diferentes pantallas. Facilita la organización y reutilización del código visual y funcional.

- ***Screens:*** Incluye las distintas pantallas de la aplicación. Cada archivo representa una vista completa, compuesta por diversos componentes y elementos visuales.

- ***Services:*** En esta carpeta se centralizan todas las peticiones al backend, actuando como punto de comunicación entre el frontend y el backend. Aquí se definen las funciones encargadas de realizar operaciones como el inicio de sesión, la creación de emergencias, la actualización de localizaciones, etc.

- ***Styles:*** Agrupa todos los estilos utilizados en la aplicación, tanto para componentes individuales como para pantallas completas. Esto permite mantener un diseño coherente y separar la lógica de estilos del resto del código.

Además de estas carpetas principales, existen dos archivos clave en la raíz del proyecto:

- ***firebase.js:*** Gestiona la configuración e integración con Firebase, específicamente para el sistema de autenticación de usuarios.

- ***App.js:*** Es el punto de entrada de la aplicación. Desde aquí se inicializa la navegación entre pantallas y se definen las rutas principales de la interfaz.

5.3.2 Partes de componentes destacadas

1. Autenticación de usuario

Se va a describir algunas partes importantes del componente para la autenticación de usuarios.

1.1 Petición servicio

Una vez que el usuario se ha autenticado, la solicitud correspondiente es enviada al backend a través del servicio encargado de gestionar la autenticación, como se puede observar en la Figura 86.

```
const inicioSesion = async (nuevoUsuario) => {
  try {
    const response = await axios.post(
      API_URL + "/usuario/inicio-sesion",
      nuevoUsuario
    );
    return response.data;
  } catch (error) {
    console.error("Error al crear el usuario en service", error);
  }
};
```

Figura 86: Petición de servicio inicio de sesión

1.2 Obtención ubicación actual

A continuación, en la Figura 87, se procede a obtener la ubicación actual del dispositivo de forma asíncrona, en el cual, la aplicación accede al GPS del dispositivo y devuelve un objeto con la ubicación actual del usuario. Esta información se utiliza para asociar al usuario con una ubicación geográfica concreta, la cual será almacenada en la base de datos para su posterior uso como, por ejemplo, en la generación o recepción de alertas.

```
let location = await Location.getCurrentPositionAsync({
  accuracy: Location.Accuracy.Balanced,
});
```

Figura 87: Obtención localización usuarios

Es importante destacar que, previamente, es necesario solicitar al sistema los permisos necesarios, como se puede observar en la Figura 88, para acceder a la ubicación del dispositivo, respetando así las políticas de privacidad y seguridad del usuario.

```
let { status } = await Location.requestForegroundPermissionsAsync();
```

Figura 88: Petición permisos obtención ubicación

1.3 Notificaciones

En cuanto a las notificaciones push, la función registerForPushNotificationsAsync() mostrada en la Figura 89, tiene como objetivo registrar el dispositivo del usuario para que pueda recibir notificaciones a través del sistema de Expo Notifications. Este proceso es fundamental para permitir el envío de alertas personalizadas relacionadas con emergencias u otros eventos relevantes dentro de la aplicación.

Dentro de esta función, se utiliza getExpoPushTokenAsync(), que genera un token único asociado al dispositivo actual. Este token permite identificar de forma individual a cada usuario y se almacena en la base de datos, asociado al usuario

autenticado, con el fin de poder enviarle notificaciones específicas desde el backend en el futuro.

```
async function registerForPushNotificationsAsync() {
  if (Platform.OS === "android") {
    await Notifications.setNotificationChannelAsync("default", {
      name: "default",
      importance: Notifications.AndroidImportance.MAX,
      vibrationPattern: [0, 250, 250, 250],
      lightColor: "#FF231F7C",
    });
  }

  const { data } = await Notifications.getExpoPushTokenAsync();
  return data;
}
```

Figura 89: Obtención token de notificación usuario

Adicionalmente, se emplea AsyncStorage como se puede observar en la Figura 90. Una API de React Native que permite gestionar el almacenamiento local en formato clave-valor. Gracias a esta herramienta, se guardan de forma persistente los datos del usuario en el dispositivo, lo que permite mantener la sesión iniciada.

```
await AsyncStorage.setItem("user", JSON.stringify(responseUsuario));
```

Figura 90: Utilización de almacenamiento persistente

1.4 Recuperación de contraseña

En la aplicación se ha implementado la funcionalidad para que los usuarios puedan recuperar su contraseña en caso de haberla olvidado. Para ello, se utiliza el método sendPasswordResetEmail proporcionado por Firebase Authentication. Este método, mostrado en la Figura 91, permite enviar un correo electrónico de recuperación al usuario de forma automática.

```
const handlePasswordReset = async () => {
  if (!email) {
    Alert.alert("Error", "Por favor, ingresa un correo electrónico.");
    return;
  }

  try {
    await sendPasswordResetEmail(auth, email);

    props.navigation.push("LoginCorreo");
  } catch (error) {
    console.error("Error al enviar correo de recuperación:", error);
    Alert.alert(
      "Error",
      "No se pudo enviar el correo de recuperación. Verifica que el correo esté registrado."
    );
  }
};
```

Figura 91: Método para la recuperación de contraseñas

2. Gestión de autenticación mediante Context API

Para gestionar la autenticación de los usuarios de manera global en la aplicación, se ha utilizado Context API de React. Este enfoque permite compartir el estado del usuario autenticado entre los diferentes componentes sin necesidad de pasar propiedades manualmente a través de múltiples niveles de la jerarquía de componentes.

Se define un contexto llamado AuthContext mediante createContext() como se puede observar en la Figura 92, que se encarga de almacenar y proporcionar el estado del usuario autenticado, así como funciones para iniciar y cerrar sesión.

```
const AuthContext = createContext();
```

Figura 92: Creación de un contexto

El proveedor de autenticación (AuthProvider) es un componente que envuelve toda la aplicación y proporciona acceso al estado de autenticación a cualquier componente que lo necesite.

Cuando la aplicación inicia, el useEffect configura un listener con onAuthStateChanged(), que se suscribe a los cambios en el estado de autenticación de Firebase. Esto permite que, si un usuario ya estaba autenticado, su información se cargue automáticamente sin necesidad de iniciar sesión nuevamente. Esto se puede visualizar en la Figura 93.

Para facilitar el acceso a la información de autenticación en cualquier parte de la aplicación, se define el hook useAuth() de la Figura 94, que permite obtener el contexto de autenticación de forma sencilla.

```
useEffect(() => {
    const auth = getAuth();
    const unsubscribe = onAuthStateChanged(auth, (currentUser) => {
        setUser(currentUser);
        setLoading(false);
    });

    return () => unsubscribe();
}, []);

const loginUser = (userData) => {
    setUser(userData);
};
```

Figura 93: Listener para notificar sobre cambios en el estado de autenticación

```
export const useAuth = () => useContext(AuthContext);
```

Figura 94: Obtención contexto de autenticación

3. Mapa interactivo

Para mostrar la ubicación del usuario y las emergencias en un mapa interactivo, se ha utilizado el componente MapView mostrado en la Figura 95 de la biblioteca React Native Maps, que permite integrar mapas interactivos en aplicaciones móviles. Esta biblioteca facilita la visualización de mapas con funcionalidades como el seguimiento de la ubicación del usuario, marcadores personalizados y la navegación entre diferentes ubicaciones. El mapa se centra en la ubicación del usuario, utilizando las coordenadas de latitud y longitud almacenadas en el estado de la aplicación. Además, se utiliza el componente Marker de React Native Maps para colocar marcadores en las ubicaciones correspondientes a las emergencias. Cada marcador está asociado a una coordenada específica. En lugar de utilizar los marcadores predeterminados de Google Maps, se ha creado un componente personalizado llamado SimboloMarker que se utiliza dentro de cada marcador para representar de forma visual las emergencias, en función de su tipo y nivel de riesgo. La ubicación del usuario se actualiza cada 30 segundos para poder tener un seguimiento real y poder notificar sobre las emergencias cercanas a él, según su radio de notificaciones.

```
<MapView
  style={stylesSafeArea.map}
  region={{
    latitude: location.latitude,
    longitude: location.longitude,
    latitudeDelta: 0.0922,
    longitudeDelta: 0.0421,
  }}
  showsUserLocation={true}
>
  {ubicacionesEmergencia.map((ubicacion, index) => {
    if (!ubicacion || !emergencias[index]) return null;
    return (
      <Marker
        key={`${ubicacion.id}-${index}`}
        coordinate={{
          longitude: parseFloat(ubicacion.ejeX),
          latitude: parseFloat(ubicacion.ejeY),
        }}
        onPress={() =>
          navigation.navigate("Emergencia", {
            id: emergencias[index].id,
          })
        }
      >
        <SimboloMarker emergencia={emergencias[index]} />
      </Marker>
    );
  })}
</MapView>
```

Figura 95: Componente MapView

4. Visualización de emergencias

Para visualizar con precisión la ubicación de una emergencia en el mapa, representado en la Figura 96, se hace necesario convertir las coordenadas geográficas (longitud y latitud) almacenadas en la base de datos en una

representación más comprensible, como una dirección o nombre de lugar. Para lograr esta conversión, se realiza una llamada a la API de OpenStreetMap representada en la Figura 97, que permite obtener la ubicación exacta a partir de las coordenadas geográficas. Esta API utiliza un proceso conocido como geocodificación inversa, que consiste en transformar un conjunto de coordenadas (longitud y latitud) en una descripción legible por el usuario.

```
useEffect(() => {
  if (localizacion != null) {
    const getNombreLocalizacionByLocalizacion = async () => {
      const response =
        await localizacionServices.getDireccionFromLocalizacion(localizacion);
      setNombreLocalizacion(response);
    };

    getNombreLocalizacionByLocalizacion();
  }
}, [localizacion]);
```

Figura 96: Llamada al servicio de geocodificación inversa

```
const getDireccionFromLocalizacion = async (localizacion) => {
  try {
    const response = await fetch(
      `https://nominatim.openstreetmap.org/reverse?format=json&lat=${localizacion.ejeY}&lon=${localizacion.ejeX}`,
      {
        headers: {
          "User-Agent": "TuApp/1.0 (sandravazquezp08@gmail.com)",
        },
      }
    );
    const text = await response.text();
    try {
      const data = JSON.parse(text);
      console.log("Dirección obtenida:", data);
      return data.display_name;
    } catch (e) {
      console.error("Error al parsear JSON:", e);
      console.log("Respuesta recibida:", text);
      return null;
    }
  } catch (error) {
    console.error("Error al obtener la dirección:", error);
    return null;
  }
};
```

Figura 97: Obtención dirección mediante geocodificación inversa

5. Crear emergencia

Al momento de crear una emergencia, los usuarios tienen dos opciones para definir la ubicación del incidente: pueden seleccionar su ubicación actual o ingresar la ubicación manualmente.

Si el usuario elige ingresar la ubicación manualmente, como se puede observar en la Figura 98, la aplicación proporciona una funcionalidad de autocompletado que sugiere posibles ubicaciones a medida que se escribe. Esta funcionalidad se obtiene

a través de una llamada a la API de OpenStreetMap, mostrada en la Figura 99, la cual realiza una búsqueda de direcciones basada en la entrada del usuario.

Una vez que el usuario selecciona una de las sugerencias proporcionadas, la aplicación obtiene automáticamente las coordenadas geográficas (latitud y longitud) de la ubicación elegida. Estas coordenadas se almacenan en la base de datos para su posterior uso y visualización en el mapa.

```
const fetchSuggestions = async () => {
    setIsSearching(true);
    const suggestions = await localizacionServices.searchAddress(query);
    if (active) {
        setResults(suggestions);
        setIsSearching(false);
    }
};
```

Figura 98: Método para obtener las sugerencias de localizaciones

```
const searchAddress = async (query) => {
  try {
    const response = await fetch(
      `https://nominatim.openstreetmap.org/search?format=json&q=${encodeURIComponent(
        query
      )}`,
```

Figura 99: Obtención de sugerencias de localizaciones

6. Gestión de Mi Cuenta

Desde la sección "Mi cuenta", los usuarios tienen la opción de cerrar sesión. Este método se puede ver en la Figura 100.

```
const handleLogout = () => {
  logout();
  navigation.reset({
    index: 0,
    routes: [{ name: "LoginCorreo" }],
  });
};
```

Figura 100: Método para cerrar sesión

Este proceso de cierre de sesión es gestionado mediante el Contexto de Autenticación, que se detalló previamente en la aplicación.

El cierre de sesión se lleva a cabo a través de la función logout(), mostrada en la Figura 101, que interactúa con Firebase Authentication. Esta función utiliza el método signOut proporcionado por Firebase para cerrar la sesión del usuario en la aplicación. Este método asegura que, una vez cerrada la sesión, el usuario ya no tenga acceso a las funcionalidades restringidas y que el estado de autenticación se actualice correctamente en la aplicación.

```
const logout = () => {
    const auth = getAuth();
    signOut(auth)
        .then(() => {
            console.log('Sesión cerrada');
            setUser(null);
        })
        .catch((error) => {
            console.error('Error al cerrar sesión:', error);
        });
};
```

Figura 101: Cierre de sesión Firebase

7. Manejo de la navegación de las pantallas

Se está utilizando React Navigation para gestionar la navegación entre las distintas pantallas de la aplicación, mostrado en la Figura 102. Esto es clave, ya que permite que los usuarios naveguen entre pantallas con un sistema de navegación basado en pila (stack navigation).

```
return (
  <AuthProvider>
    <NavigationContainer>
      <Stack.Navigator
        initialRouteName={user ? "Home" : "LoginCorreo"}
        screenOptions={{ headerShown: false }}
      >
        <Stack.Screen name="Login" component={Login} />
```

Figura 102: Gestión navegación entre pantallas

5.4 Pruebas de la aplicación

A continuación, se explicará el proceso de pruebas llevado a cabo para asegurar el correcto funcionamiento de la aplicación. En este caso, se ha optado por la utilización de casos de prueba, los cuales son situaciones específicas diseñadas para verificar que el sistema cumple con los requisitos establecidos y se comporta según lo esperado. Cada caso de prueba incluye un identificador único, una descripción clara del escenario de prueba, las condiciones iniciales necesarias, los datos a utilizar, los pasos a seguir, el resultado esperado, el resultado obtenido y el caso de uso asociado.

La implementación de estos casos de prueba es crucial para detectar posibles errores o fallos en el sistema, garantizando así que el producto final cumpla con los estándares de calidad establecidos. A continuación, se detallan algunos ejemplos representativos de casos de prueba para evaluar distintas funcionalidades de la aplicación. No obstante, no se incluyen todos los casos de prueba de manera exhaustiva, ya que esto podría hacer que la memoria fuera demasiado extensa y difícil de seguir.

- ***CP-1: Registro de un usuario***
  - *Descripción:* Comprobar que el usuario puede registrarse correctamente.
  - *Condiciones previas:* El usuario ha accedido a la aplicación y no ha iniciado sesión.
  - *Datos de prueba:* El correo electrónico del usuario y la contraseña que desea poner.
  - *Pasos a ejecutar:* Entra en la aplicación, selecciona Crear una cuenta, introduce su correo electrónico y contraseña, confirma la contraseña y selecciona Registrarse.
  - *Resultados esperados:* El usuario es redirigido al mapa interactivo de la aplicación.
  - *Resultados reales:* Coinciden con los esperados.
  - *Caso de uso relacionado:* Se relaciona con el caso de uso 1.

- ***CP-2: Iniciar sesión***
  - *Descripción:* Comprobar que el usuario puede iniciar sesión en la aplicación.
  - *Condiciones previas:* El usuario ya se ha registrado previamente.
  - *Datos de prueba:* El correo electrónico y contraseña con la que se registró el usuario.
  - *Pasos a ejecutar:* Entra en la aplicación, introduce el correo electrónico y la contraseña y selecciona Iniciar sesión.
  - *Resultados esperados:* El usuario es redirigido al mapa interactivo de la aplicación.
  - *Resultados reales:* Coinciden con los esperados.
  - *Caso de uso relacionado:* Se relaciona con el caso de uso 2.

- ***CP-3: Cerrar sesión***
  - *Descripción:* Comprobar que el usuario puede cerrar su sesión correctamente.
  - *Condiciones previas:* El usuario ha iniciado sesión y se encuentra en el mapa interactivo de la aplicación.
  - *Datos de prueba:* La cuenta iniciada del usuario.
  - *Pasos a ejecutar:* Seleccionar el menú desplegable, seleccionar Mi cuenta y darle a cerrar sesión.
  - *Resultados esperados:* El usuario será redirigido a la página de inicio de sesión.
  - *Resultados reales:* Coinciden con los esperados.
  - *Caso de uso relacionado:* Se relaciona con el caso de uso 4.

- ***CP-4: Visualización detalles de una emergencia***

- *Descripción:* Comprobar que el usuario puede visualizar los detalles de una emergencia.
- *Condiciones previas:* El usuario ha iniciado sesión y se encuentra en el mapa interactivo de la aplicación.
- *Datos de prueba:* La cuenta del usuario y una emergencia creada.
- *Pasos a ejecutar:* El usuario visualiza la emergencia y selecciona el símbolo de la emergencia en el mapa interactivo.
- *Resultados esperados:* Se muestran todos los detalles de la emergencia.
- *Resultados reales:* Coinciden con los esperados.
- *Caso de uso relacionado:* Se relaciona con el caso de uso 7.

- ***CP-5: Crear una emergencia***
- *Descripción:* Comprobar que el usuario puede crear una emergencia.
- *Condiciones previas:* El usuario ha iniciado sesión y se encuentra en el mapa interactivo de la aplicación.
- *Datos de prueba:* La cuenta del usuario.
- *Pasos a ejecutar:* El usuario selecciona el botón inferior del mapa interactivo, rellena los campos necesarios y selecciona Crear emergencia.
- *Resultados esperados:* El usuario es redirigido al mapa interactivo y la emergencia registrada correctamente.
- *Resultados reales:* Coinciden con los esperados.
- *Caso de uso relacionado:* Se relaciona con el caso de uso 10.

- ***CP-6: Verificar emergencias***
- *Descripción:* Comprobar que los expertos, autoridades y administradores pueden verificar las emergencias ya existentes.
- *Condiciones previas:* El usuario ha iniciado sesión y se encuentra en el mapa interactivo de la aplicación.
- *Datos de prueba:* El usuario que ha iniciado sesión posee uno de los roles mencionados en la descripción.
- *Pasos a ejecutar:* El usuario selecciona la emergencia deseada, pulsa sobre Editar Emergencia, selecciona el switch de verificar emergencia y pulsa Guardar.
- *Resultados esperados:* El usuario es redirigido al mapa interactivo y visualmente la emergencia ahora se ve con un pulso animado para diferenciarla de las no verificadas.
- *Resultados reales:* Coinciden con los esperados.
- *Caso de uso relacionado:* Se relaciona con el caso de uso 16.

- ***CP-7: Reportar emergencias***
- *Descripción:* Comprobar que los ciudadanos pueden reportar emergencias.

- *Condiciones previas:* El usuario ha iniciado sesión y se encuentra en el mapa interactivo de la aplicación.
- *Datos de prueba:* El usuario que ha iniciado sesión posee el rol de ciudadano.
- *Pasos a ejecutar:* El usuario selecciona la emergencia deseada y selecciona Reportar Emergencia.
- *Resultados esperados:* El usuario es redirigido al mapa interactivo y el reporte es guardado correctamente.
- *Resultados reales:* Coinciden con los esperados.
- *Caso de uso relacionado:* Se relaciona con el caso de uso 18.

- ***CP-8: Resolver emergencias***
- *Descripción:* Comprobar que los expertos, autoridades y administradores pueden resolver emergencias directamente.
- *Condiciones previas:* El usuario ha iniciado sesión y se encuentra en el mapa interactivo de la aplicación.
- *Datos de prueba:* El usuario que ha iniciado sesión posee uno de los roles mencionados en la descripción.
- *Pasos a ejecutar:* El usuario selecciona la emergencia deseada, selecciona Resolver Emergencia y le da a confirmar.
- *Resultados esperados:* El usuario es redirigido al mapa interactivo y la emergencia ha desaparecido del mapa.
- *Resultados reales:* Coinciden con los esperados.
- *Caso de uso relacionado:* Se relaciona con el caso de uso 19.

- ***CP-9: Aceptar solicitudes edición emergencias***
- *Descripción:* Comprobar que los expertos y autoridades pueden aceptar solicitudes de edición de emergencias pendientes de aprobación.
- *Condiciones previas:* El usuario ha iniciado sesión y se encuentra en el mapa interactivo de la aplicación.
- *Datos de prueba:* El usuario que ha iniciado sesión posee el rol de experto o autoridad.
- *Pasos a ejecutar:* El usuario selecciona el menú desplegable, selecciona Solicitudes Edición Emergencias, visualiza la solicitud deseada y pulsa Aceptar.
- *Resultados esperados:* La solicitud se acepta y se recargan las pendientes.
- *Resultados reales:* Coinciden con los esperados.
- *Caso de uso relacionado:* Se relaciona con el caso de uso 25.

- ***CP-10: Solicitar cambio de rol***

- *Descripción:* Comprobar que los usuarios pueden solicitar un cambio de rol.
- *Condiciones previas:* El usuario ha iniciado sesión y se encuentra en el mapa interactivo de la aplicación.
- *Datos de prueba:* La cuenta iniciada del usuario.
- *Pasos a ejecutar:* El usuario selecciona el menú desplegable, pulsa sobre Mi cuenta, selecciona el desplegable de Cambiar Rol, selecciona el rol deseado y pulsa sobre Solicitar cambio de rol.
- *Resultados esperados:* La solicitud se registra correctamente y el usuario es redirigido al mapa interactivo.
- *Resultados reales:* Coinciden con los esperados.
- *Caso de uso relacionado:* Se relaciona con el caso de uso 31.

- ***CP-11: Denegar solicitudes cambio de rol***
- *Descripción:* Comprobar que los administradores pueden rechazar solicitudes de cambio de rol.
- *Condiciones previas:* El usuario ha iniciado sesión y se encuentra en el mapa interactivo de la aplicación.
- *Datos de prueba:* El usuario que ha iniciado sesión posee el rol de administrador.
- *Pasos a ejecutar:* El usuario selecciona el menú desplegable, pulsa sobre Solicitudes pendientes Cambio de Rol, visualiza la solicitud deseada y selecciona la opción de Rechazar.
- *Resultados esperados:* La solicitud se rechaza y se recargan las pendientes.
- *Resultados reales:* Coinciden con los esperados.
- *Caso de uso relacionado:* Se relaciona con el caso de uso 33.

- ***CP-12: Recibir notificaciones creación de Emergencias***
- *Descripción:* Comprobar que los usuarios reciben notificaciones cuando se crean emergencias que se encuentran dentro de su radio de notificaciones respecto a su ubicación actual.
- *Condiciones previas:* El usuario ha iniciado sesión.
- *Datos de prueba:* La cuenta del usuario.
- *Pasos a ejecutar:* Un usuario crea una emergencia cercana al otro usuario.
- *Resultados esperados:* Ambos usuarios reciben una notificación push sobre la aparición de una nueva emergencia cerca de ellos.
- *Resultados reales:* Coinciden con los esperados.
- *Caso de uso relacionado:* Se relaciona con el caso de uso 35.

6

Despliegue de la aplicación

En este apartado se describe detalladamente el proceso de despliegue de las distintas partes que componen la aplicación, prestando especial atención a la elección de entornos que ofrecieran un equilibrio entre bajo coste, simplicidad de uso y fiabilidad. Dado que se trata de un Trabajo de Fin de Grado, se ha optado por soluciones que, sin incurrir en gastos elevados, permitieran poner en funcionamiento la aplicación de forma funcional y profesional, facilitando tanto las pruebas como la posible publicación futura.

6.1 Frontend

El frontend de la aplicación ha sido desarrollado utilizando React Native junto con el framework Expo, el cual proporciona una serie de herramientas y servicios que simplifican significativamente el proceso de desarrollo y despliegue de aplicaciones móviles. Al tratarse de una aplicación dirigida únicamente a dispositivos Android, el objetivo final era generar un archivo APK (Android Package) que pudiera instalarse en cualquier dispositivo compatible.

Para ello, se ha utilizado la plataforma Expo Application Services (EAS), que permite construir aplicaciones de forma remota sin necesidad de configurar entornos de compilación locales como Android Studio. Esta funcionalidad resulta

especialmente útil en proyectos académicos o personales, ya que evita las complejidades técnicas que normalmente conlleva el empaquetado y firmado de aplicaciones móviles.

Proceso de Generación del APK

1. ***Preparación del proyecto:***
   Una vez completado el desarrollo de la aplicación, se realizó la subida del proyecto a los servidores de Expo, asegurándose de que todas las dependencias estuvieran correctamente configuradas y que la aplicación funcionara sin errores en el entorno de prueba de Expo Go.

2. ***Compilación mediante EAS Build:***
   Utilizando el comando eas build -p android --profile production, se inició el proceso de generación del archivo APK. Expo ofrece una cola gratuita para la construcción de aplicaciones, en la que los proyectos se procesan por orden de entrada. Si bien esta cola puede tener cierta espera en momentos de alta demanda, por lo que en general el tiempo de compilación ha sido razonable.

3. ***Descarga del APK:***
   Una vez finalizada la compilación, Expo proporciona un enlace directo para descargar el archivo APK resultante, el cual ya está firmado y listo para ser instalado en cualquier dispositivo Android. Este enlace puede compartirse fácilmente con otros usuarios para pruebas o distribución manual de la aplicación

6.2 Backend

El proceso de despliegue del backend supuso uno de los mayores retos del proyecto, principalmente debido a la escasez de servicios gratuitos que ofrecieran entornos de producción estables y compatibles con las necesidades específicas de la aplicación.

Inicialmente, se optó por utilizar Amazon Web Services (AWS), concretamente el servicio Elastic Beanstalk, debido a su robustez y la posibilidad de desplegar aplicaciones de manera sencilla. Durante las pruebas locales con Expo Go, el backend funcionaba correctamente y respondía como se esperaba. No obstante, al generar la APK de la aplicación para Android y probarla en un dispositivo físico, el backend dejaba de estar accesible.

Tras investigar el problema, se descubrió que el fallo se debía a que el entorno desplegado en Elastic Beanstalk funcionaba bajo una URL no segura (HTTP en lugar de HTTPS). Android, por razones de seguridad, bloquea por defecto las peticiones a direcciones no seguras en aplicaciones compiladas. En cambio, Expo Go sí permite estas conexiones porque internamente aplica configuraciones de red menos restrictivas para facilitar el desarrollo.

Para solucionar este inconveniente, la opción más adecuada habría sido configurar un balanceador de carga (load balancer) en Elastic Beanstalk para habilitar HTTPS, lo que implicaba la gestión de certificados SSL y ajustes avanzados en la infraestructura. Sin embargo, esta configuración resultaba demasiado compleja y poco práctica para los recursos y el tiempo disponibles.

Tras analizar otras alternativas, se decidió cambiar la estrategia de despliegue y opté por utilizar Railway, una plataforma más sencilla e intuitiva que permite el despliegue de aplicaciones de manera rápida y con menos configuraciones técnicas. Railway ofrecía un crédito gratuito de 5 dólares, lo cual permitía poner en marcha el backend sin coste inicial, y en caso de agotarse, el coste del servicio seguía siendo asumible, mucho más económico en comparación con AWS.

Además, Railway cuenta con una funcionalidad muy útil: la integración directa con GitHub. Esto permite que, cada vez que se realiza un cambio en el repositorio, el entorno se despliegue automáticamente con la última versión del código, agilizando enormemente el proceso de actualización y mantenimiento del backend.

Gracias a este cambio, fue posible disponer de una URL segura (HTTPS) que cumplía con los requisitos de Android y garantiza la disponibilidad del backend desde la aplicación móvil.

6.3 Base de datos

En un primer momento, la base de datos del sistema se desplegó utilizando el servicio Amazon RDS (Relational Database Service) de Amazon Web Services (AWS). Esta decisión inicial se tomó por la integración nativa que RDS ofrece con otros servicios de AWS, como Elastic Beanstalk, lo cual facilitaba la configuración y la comunicación directa entre el backend y la base de datos.

No obstante, al tratarse de un entorno de prueba gratuito con una duración y recursos limitados, y considerando que el backend también se encontraba desplegado en AWS, la prueba gratuita se agotó rápidamente. Esto supuso un problema, ya que los costes derivados del uso continuado de los servicios de AWS

resultaban elevados. Por esta razón, se optó por cerrar la cuenta de AWS y buscar alternativas más económicas y sostenibles.

La solución elegida fue trasladar el despliegue de la base de datos a Railway, la misma plataforma utilizada para desplegar el backend. Railway ofrece soporte para bases de datos MySQL y presenta una interfaz sencilla que facilita la creación, configuración y administración de la base de datos.

Una de las principales ventajas de Railway es que, al crear una base de datos, se proporciona automáticamente un entorno de conexión accesible públicamente. Esta URL de conexión se puede integrar fácilmente en herramientas como MySQL Workbench, permitiendo así gestionar la base de datos de manera remota y visual. Esta facilidad de integración también permite que el backend se conecte sin necesidad de configuraciones complejas, lo que reduce significativamente el tiempo de implementación y los posibles errores de conexión.

Gracias a este cambio ha sido posible mantener una base de datos operativa sin incurrir en costes elevados.

7

Conclusiones y trabajos futuros

Para finalizar, se va a hablar sobre los objetivos cumplidos, las dificultades que se han encontrado y las posibles líneas futuras.

7.1 Evaluación del cumplimiento de los objetivos

Los objetivos propuestos al inicio del desarrollo se han cumplido satisfactoriamente. La aplicación desarrollada permite gestionar emergencias de manera más eficaz, respondiendo así a una necesidad creciente debido al incremento de situaciones de emergencia en la actualidad y en el que la implementación de diferentes roles dentro del sistema ha fomentado la participación ciudadana. Permitiendo así una colaboración activa con expertos y autoridades, y aprovechando en todo momento la localización precisa de los usuarios. Esta cooperación contribuye a mejorar la respuesta ante emergencias, ayudando a salvar vidas y minimizar daños, especialmente considerando que el tiempo de reacción es un factor crítico en este tipo de situaciones.

Desde el punto de vista del diseño, se ha priorizado la simplicidad y la usabilidad. Se ha evitado sobrecargar la interfaz con botones o información innecesaria, ya que en contextos de emergencia resulta fundamental que la aplicación sea rápida, clara e intuitiva.

En relación con los requisitos y casos de uso definidos, todos han sido implementados correctamente, lo que ha resultado en una aplicación completa en cuanto a funcionalidades e integración de roles.

Finalmente, desde una perspectiva académica y tecnológica, el desarrollo de este proyecto ha supuesto una inmersión total en las distintas fases y desafíos del ciclo de vida de un proyecto software. Además, ha permitido adquirir conocimientos relevantes y actuales sobre tecnologías ampliamente utilizadas como Spring Boot y React Native, contribuyendo de forma significativa a la formación como ingeniera del software.

7.2 Dificultades encontradas y soluciones adoptadas

A lo largo del desarrollo del proyecto se han presentado diversas dificultades, muchas de ellas relacionadas con la integración de tecnologías y el despliegue de la aplicación, las cuales han sido superadas con éxito gracias a la investigación, la documentación y la experimentación constante.

Una de las principales complicaciones surgió durante el uso de React Native, concretamente en la gestión de dependencias. Al añadir nuevas librerías, se producían frecuentes incompatibilidades entre ellas, lo que provocaba fallos en la ejecución del proyecto. Identificar con precisión qué librería causaba el conflicto resultaba complejo, ya que los errores no siempre proporcionaban información clara. Esta situación exigió un análisis detallado de las versiones utilizadas y una cuidadosa gestión del entorno de desarrollo.

Otra dificultad significativa fue el despliegue de la aplicación. La falta de servicios gratuitos o con periodos de prueba suficientemente amplios complicó la publicación tanto del frontend, backend y base de datos. Inicialmente, el backend fue desplegado en Amazon Web Services (AWS) utilizando Elastic Beanstalk. Sin embargo, la URL generada por defecto era mediante el protocolo HTTP, y no HTTPS. Esto no supuso un problema al utilizar Expo Go, ya que este entorno no impone restricciones sobre las URLs inseguras. Sin embargo, al generar la APK de la aplicación, dejaba de funcionar sin mostrar ningún mensaje de error. Tras una extensa investigación, se identificó que Android bloqueaba automáticamente las peticiones a URLs no seguras (HTTP), lo que impedía la comunicación con el backend. La solución más sencilla habría sido configurar un balanceador de carga en AWS para habilitar HTTPS, pero debido a la complejidad y coste adicional, se optó por migrar el backend a Railway, una plataforma que permite generar URLs seguras de forma más sencilla y rápida.

Tras lograr el despliegue y el funcionamiento completo de la aplicación, en el que inicialmente se utilizaban Amazon S3 para almacenar los símbolos de emergencia (servicio de almacenamiento de objetos ofrecido por AWS) y Amazon RDS como base de datos relacional gestionada, se agotó el período de prueba gratuito de AWS. Debido a que el coste de sus servicios resultaba elevado para el desarrollo de este proyecto académico, hubo que cerrar la cuenta y buscar alternativas más accesibles.

En lo referente a los símbolos de emergencia, se optó por Cloudinary, una plataforma en la nube especializada en la gestión de recursos multimedia, que ofrece un plan gratuito con funcionalidades básicas. Esta alternativa permitió mantener la gestión centralizada de los símbolos y acceder a ellos mediante URLs públicas sin incrementar los costes del proyecto.

Finalmente, otro problema relevante se presentó en la generación de la APK respecto a la visualización del mapa implementado con React Native Maps. Aunque el mapa funcionaba correctamente en Expo Go, no se mostraba correctamente en la generación de la APK. La causa se encontraba en que Expo Go posee configuraciones internas, las cuáles no están presentes en la APK y para resolverlo fue necesario configurar una API Key de Google Maps desde Google Cloud Console, vinculada específicamente al proyecto. Gracias a esta configuración, el mapa se mostró correctamente en la aplicación final.

7.3 Líneas futuras de mejora

Aunque el desarrollo actual de la aplicación cumple con los objetivos propuestos y proporciona una base sólida para la gestión colaborativa de emergencias, en todo proyecto existen diversas líneas de mejora que podrían explorarse en el futuro para ampliar sus funcionalidades y optimizar la experiencia del usuario. En el caso de este TFG se identifican las siguientes:

- ***Visualización mediante mapas de calor:*** Una posible mejora sería la incorporación de mapas de calor para representar gráficamente las zonas con mayor concentración de emergencias. Esta funcionalidad permitiría a los usuarios identificar de forma rápida y visual las áreas más afectadas, facilitando la toma de decisiones.

- ***Mejora en la autenticación de roles:*** Para garantizar una mayor fiabilidad en la verificación de roles específicos (como experto o autoridad), se podría permitir la subida de documentación acreditativa al solicitar dichos roles.

Esto permitiría a los administradores validar de manera más rigurosa las solicitudes, fortaleciendo la seguridad y la credibilidad del sistema.

- ***Inicio de Sesión con Google:*** Con el objetivo de agilizar el proceso de acceso y mejorar la usabilidad, se podría implementar la opción de registro e inicio de sesión mediante cuentas de Google. Esta funcionalidad evitaría la necesidad de introducir manualmente el correo electrónico y la contraseña, simplificando el acceso a la aplicación.

- ***Sistema de puntos y recompensas:*** Para fomentar la participación activa de los ciudadanos, se propone la incorporación de un sistema de puntos que recompensen acciones como la creación de emergencias, la solicitud de cambios o reportes de emergencias. Este sistema serviría como incentivo para promover el uso continuado y el compromiso con la aplicación.

- ***Difusión en redes sociales:*** En situaciones de alta urgencia, sería útil permitir la generación de publicaciones automáticas para redes sociales (como historias o enlaces directos), facilitando la difusión de emergencias críticas y alertando a un mayor número de personas en menor tiempo.

Referencias

[1] React Native: https://reactnative.dev/
[2] Spring Boot: https://spring.io/projects/spring-boot
[3] MySQL: https://www.mysql.com/
[4] Expo: https://docs.expo.dev/
[5] Expo Go: https://expo.dev/go
[6] Axios: https://axios-http.com/es/docs/intro
[7] Firebase Authentication: https://firebase.google.com/docs/auth?hl=es-419
[8] Google Cloud Console: https://cloud.google.com/docs
[9] Amazon Web Services: https://docs.aws.amazon.com/
[10] Railway: https://docs.railway.com/
[11] Visual Paradigm: https://www.visual-paradigm.com/support/documents/
[12] Postman: https://learning.postman.com/docs/introduction/overview/

Apéndice A

Manual de Instalación

En este apéndice se explica cómo desplegar la aplicación en un entorno local. Para ello, es necesario desplegar tanto la base de datos, como el backend y el frontend.

A.1 Requisitos previos

Antes de poder ejecutar o desplegar la aplicación completa, es necesario tener instaladas las siguientes herramientas:

- ***Java 17:*** Necesario para compilar y ejecutar el backend desarrollado con Spring Boot.
- ***Apache maven:*** Herramienta de construcción y gestión de dependencias del proyecto backend.
- ***Node js:*** Plataforma necesaria para ejecutar el entorno de desarrollo del frontend en React Native.
- ***Expo CLI:*** Interfaz de línea de comandos utilizada para gestionar y ejecutar el proyecto de React Native con Expo.
- ***Android Studio:*** Requerido para configurar y ejecutar un emulador Android, útil para probar la aplicación en un entorno local.
- ***MySQL Server:*** Motor de base de datos utilizado para almacenar la información del sistema.
- **MySQL Workbench:** Herramienta visual que permite gestionar la base de datos y configurar conexiones de manera gráfica.

A.2 Ejecución de la base de datos en local

Para ejecutar la base de datos en local se requiere tener MySQL instalado en el sistema. A continuación, se detallan los pasos para configurar la conexión y ejecutar el script de creación de la base de datos utilizando MySQL Workbench:

1. Abrir MySQL Workbench.
2. En la pantalla de inicio, seleccionar la opción "+" para crear una nueva conexión.
3. En el campo "Connection Name", introducir un nombre identificativo para la conexión.
4. En los campos correspondientes, introducir el usuario y la contraseña configurados durante la instalación de MySQL.
5. Pulsar en "Test Connection" para comprobar que la conexión es correcta. Si aparece el mensaje indicando que la conexión ha sido exitosa, pulsar OK.
6. Una vez creada la conexión, seleccionarla en la lista de conexiones disponibles.
7. Con la conexión abierta, importar y ejecutar el archivo script.sql

Este proceso dejará la base de datos configurada y lista para ser utilizada por la aplicación en el entorno local.

A.3 Instalación y ejecución del backend

Para ejecutar el backend se deben seguir los siguientes pasos:

1. Abrir el proyecto con algún entorno de desarrollo y abrir el archivo application.properties el cual se encuentra en ./src/main/resources
2. Editar las líneas de la Figura 103.

```
spring.datasource.url=jdbc:mysql://localhost:3306/{nombreBD}
spring.datasource.username={usuario}
spring.datasource.password={contraseña}
```

Figura 103: Edición parámetros base de datos

Es decir, añadir en **{nombreBD}** el nombre que se puso en la base de datos en A.2, y poner el usuario y contraseña con la que se guardó la conexión ({usuario} y {contraseña}).

Una vez configurado esto, ir a una terminal con ruta en la raíz del proyecto y ejecutar el siguiente comando: **mvn spring-boot:run**

En este punto el backend debería estar iniciado en la terminal.

A.4 Instalación y ejecución del frontend

Para iniciar el proyecto frontend, se deben seguir los siguientes pasos:

- Abrir una terminal en la raíz del proyecto.
- Ejecutar el siguiente comando para instalar todas las dependencias necesarias: **npm install**
- Una vez finalizada la instalación, iniciar el servidor de desarrollo de Expo con: **npm start**

Tras esto, el frontend se estará ejecutando correctamente.

A.5 Ejecución del emulador

Para poder ejecutar la aplicación en un emulador Android desde el entorno local, se deben seguir los siguientes pasos:

1. Abrir Android Studio.
2. Dirigirse al menú "Virtual Device Manager".
3. Seleccionar la opción "Create Virtual Device".
4. Elegir un dispositivo, como por ejemplo Pixel 4a, y pulsar en Next.
5. Seleccionar una imagen del sistema, como API 33 (Android 13.0 "Tiramisu") – x86_64, y continuar con la configuración hasta finalizar la creación del emulador.
6. Una vez creado, pulsar el icono de play para iniciar el emulador y esperar a que cargue completamente.
7. En la terminal donde se está ejecutando el frontend (con Expo), pulsar la tecla a para lanzar la aplicación en el emulador Android.
8. Automáticamente, se instalará Expo Go en el emulador si no estaba instalado previamente.
9. Una vez se cargue la aplicación, ya estará disponible para interactuar con ella desde el emulador.

APÉNDICE B

Manual de usuario

En este apéndice se detallan todas las funcionalidades de la aplicación, con el objetivo de facilitar su correcto uso por parte del usuario.
Primero se describen las funcionalidades comunes a todos los roles, y posteriormente se especifican aquellas exclusivas de cada uno de ellos.

Cabe señalar que, tras el apartado B.1, todas las funcionalidades se muestran a partir de la pantalla principal tras el inicio de sesión o registro, la cual corresponde al mapa interactivo, con el fin de evitar repeticiones.

B.1. Registro de un usuario

Cuando el usuario accede a la aplicación, si no ha iniciado sesión ni se ha registrado previamente, se muestra la pantalla de la Figura 104.

En este caso, como se desea registrarse, se selecciona la opción Crear una cuenta, mostrada en la Figura 105.

A continuación, se introduce una dirección de correo electrónico y una contraseña, y se pulsa el botón Registrarse.
Una vez completado el registro, la aplicación redirige automáticamente al mapa interactivo.

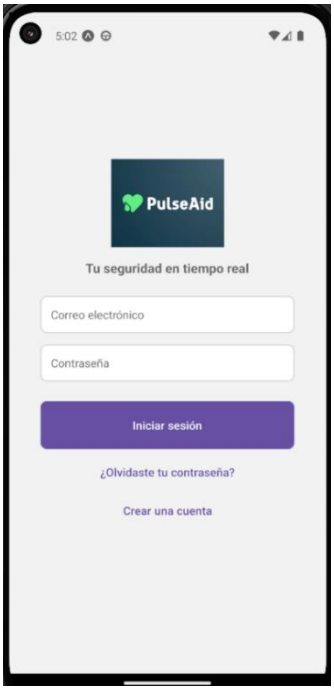

Figura 104: Pantalla de inicio de sesión

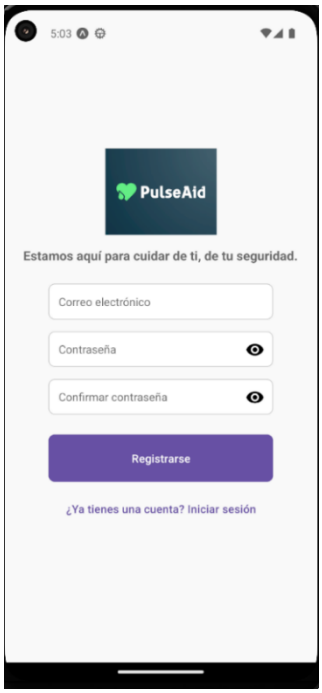

Figura 105: Pantalla de registro

Funcionalidades comunes:

B.2. Creación de una emergencia

Una vez registrado o habiendo iniciado sesión, el usuario es redirigido al mapa interactivo, como se puede ver en la Figura 106.

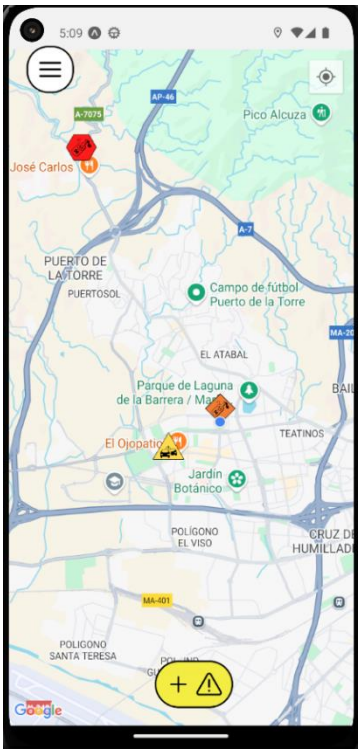

Figura 106: Pantalla principal

Para crear una nueva emergencia, se debe pulsar el botón “+” situado en la parte inferior de la pantalla. Al hacerlo, se mostrará la pantalla de la Figura 107.

En este punto, el usuario debe rellenar la descripción, las recomendaciones, seleccionar el nivel de riesgo, así como la categoría y subcategoría de la emergencia. A continuación, se desliza hacia abajo y se muestra la pantalla de la Figura 108.

En esta sección es posible seleccionar “Usar mi ubicación actual”, para que la emergencia se cree automáticamente en base a la geolocalización del dispositivo, o bien “Ingresar ubicación manual”, opción que permite introducir una dirección manualmente, con sugerencias a medida que se escribe como se puede observar en la Figura 109.

Una vez completados todos los campos, se selecciona Crear emergencia.

Como resultado, se genera una notificación en tiempo real sobre la creación de una emergencia próxima (si se encuentra dentro del radio de notificaciones). Además, en el mapa se puede observar la nueva emergencia añadida (por ejemplo, un incendio de Riesgo Bajo en el Polígono Industrial El Viso), como se puede visualizar en la Figura 110.

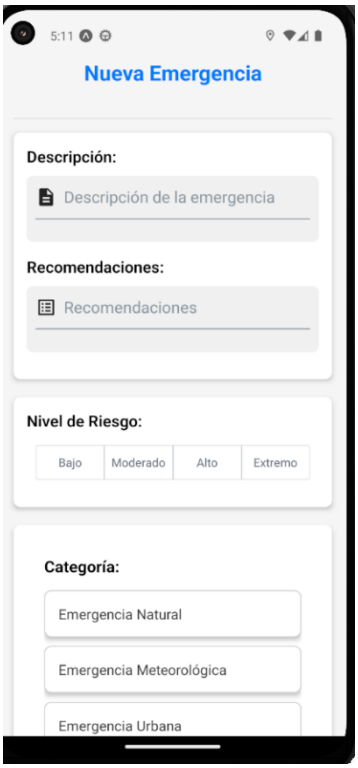

Figura 107: Pantalla para crear una emergencia

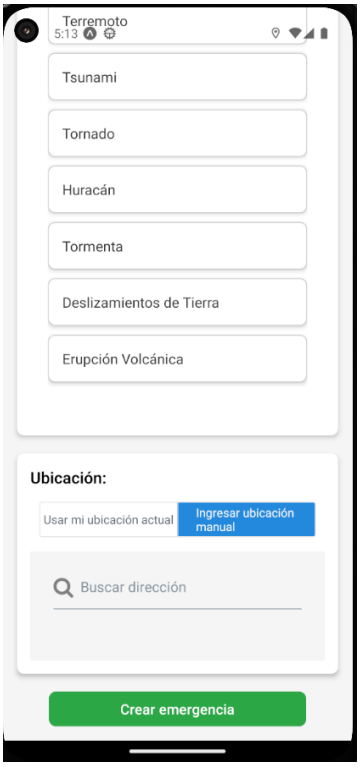

Figura 108: Pantalla para crear una emergencia 2

La notificación recibida en el dispositivo contiene los detalles en la Figura 111.

Figura 109: Pantalla de sugerencias al introducir una dirección

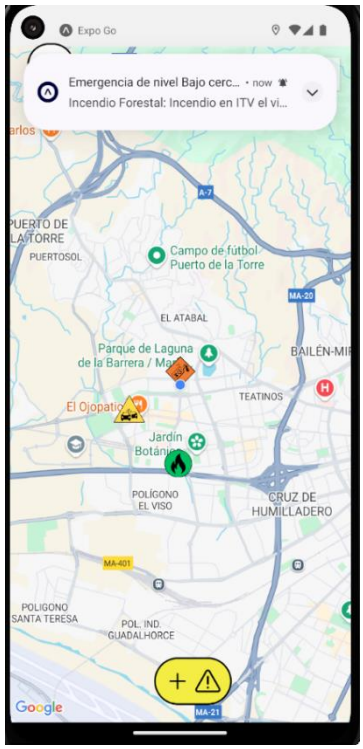

Figura 110: Pantalla tras crear una emergencia

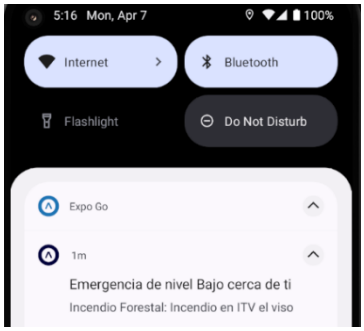

Figura 111: Pantalla de notificación de creación de una emergencia

B.3. Edición de datos Mi cuenta

Para editar los datos del usuario, se debe acceder al menú desplegable situado en la esquina superior izquierda, el cuál mostrará la Figura 112.

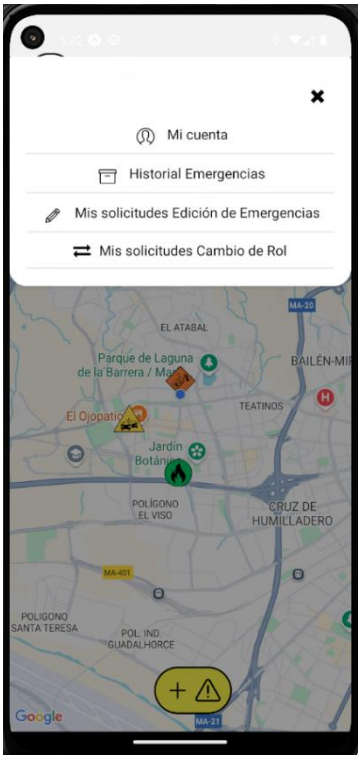

Figura 112: Pantalla con menú desplegable

Seleccionando la opción Mi cuenta, se accede a la pantalla de la Figura 113.

Al pulsar en Editar datos, se pueden modificar los datos personales del usuario como se puede observar en la Figura 114.

Tras realizar los cambios y pulsar Guardar cambios, el sistema redirige de nuevo a la sección Mi cuenta, donde se pueden ver los datos actualizados, como en la Figura 115.

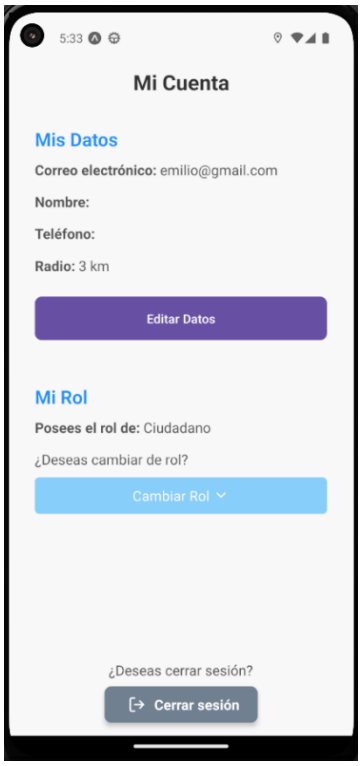

Figura 113: Pantalla de Mi cuenta

Figura 114: Pantalla de edición de datos personales

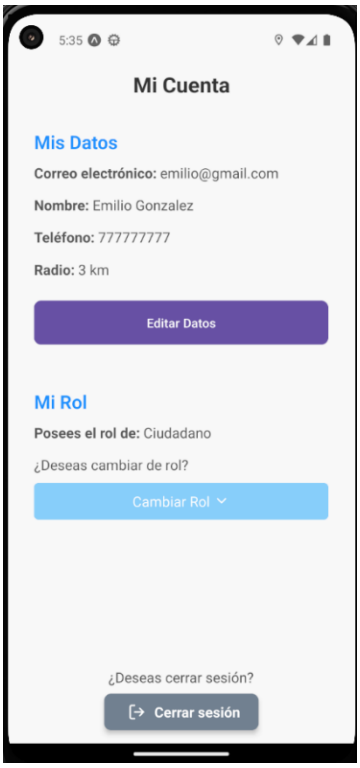

Figura 115: Pantalla Mi cuenta con datos actualizados

B.4. Solicitud nuevo rol

Desde la misma sección Mi cuenta, al seleccionar la opción Cambiar rol, se abre el desplegable de la Figura 116.

Aquí, por ejemplo, se puede seleccionar el rol Autoridad y pulsar en Solicitar cambio de rol. Y se mostrará la Figura 117.

Una vez realizada la solicitud, el usuario no podrá solicitar otro cambio hasta que el administrador la acepte o rechace.

En caso de ser aceptada, se otorgarán automáticamente las funcionalidades del nuevo rol.

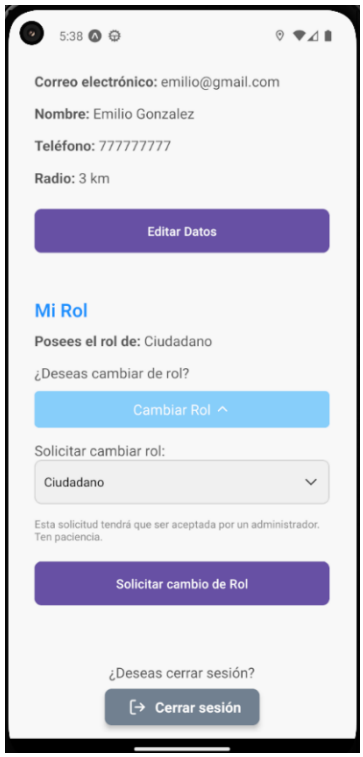

Figura 116: Pantalla Mi cuenta con desplegable solicitud de rol

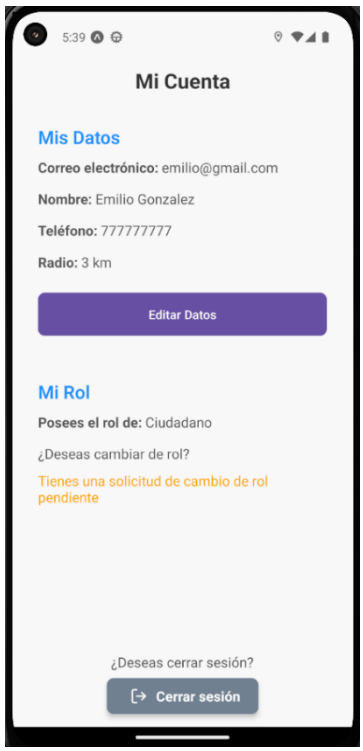

Figura 117: Pantalla tras solicitar el cambio de rol

B.5. Visualización historial de emergencias

Desde el menú desplegable, al seleccionar la opción Historial de

emergencias, se mostrará la pantalla de la Figura 118.

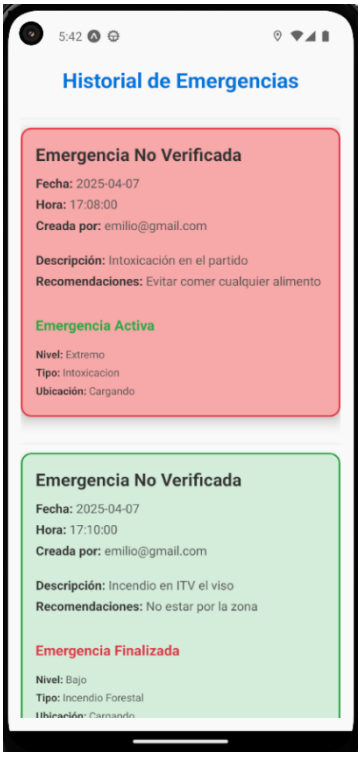

Figura 118: Pantalla Historial de emergencias

En ella se visualizan todas las emergencias registradas, tanto activas como finalizadas, junto con sus respectivos niveles de riesgo, descripciones y demás detalles relevantes para su seguimiento.

Funcionalidades como ciudadano

B.6. Solicitud edición de emergencia

Un usuario con rol de ciudadano puede solicitar la edición de una emergencia. Para ello, desde el mapa interactivo se selecciona la emergencia deseada, mostrando la Figura 119.

Luego se pulsa Solicitar editar emergencia y se visualiza la pantalla de la Figura 120.

Tras modificar los campos deseados, se selecciona Solicitar cambios.
Después de enviar la solicitud, el usuario es redirigido al mapa y la solicitud queda pendiente de revisión por parte de una autoridad o experto.

Figura 119: Pantalla visualización detalles de una emergencia

Figura 120: Pantalla solicitud edición de emergencia

B.7. Reporte de emergencia

El rol de ciudadano también puede reportar una emergencia si considera que ya no está activa.
Para ello, se accede a la emergencia deseada desde el mapa y se desliza hacia abajo, como se puede visualizar en la Figura 121.

El sistema permite reportar la emergencia y una vez que se acumulen cinco reportes distintos, la emergencia desaparece automáticamente del mapa.
En caso de haber reportado ya, el sistema lo indica y no permite un segundo reporte por el mismo usuario como se puede ver en la Figura 122.

Figura 121: Pantalla visualización detalles de una emergencia deslizada hacia abajo

En este caso se puede ver como el número de reportes ha pasado a 1 y ahora al usuario no le permite reportar la misma emergencia porque ya se reportó.

Figura 122: Pantalla visualización detalles de una emergencia cuando ha reportado ya

B.8. Visualización mis solicitudes de Edición de Emergencias

Desde el menú desplegable se accede a la opción Mis solicitudes edición de emergencias y se accede a la Figura 123.

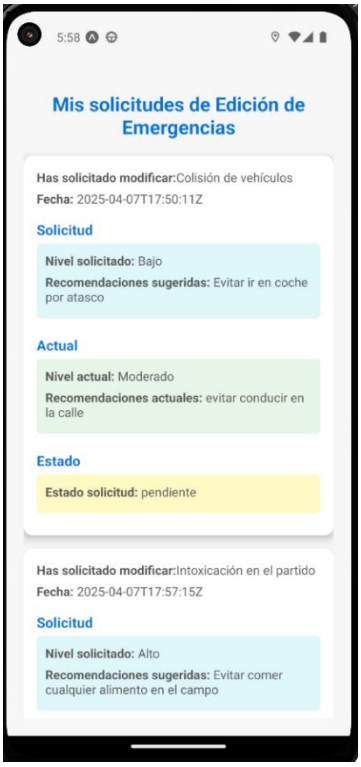

Figura 123: Pantalla Mis solicitudes de edición de emergencias

En este caso se pueden visualizar todos los detalles de las solicitudes de edición de emergencias que ha realizado el usuario. Además, si se desliza se ve que hay una solicitud aceptada en la Figura 124.

Y en este caso se puede observar que una vez que las solicitudes son resueltas incluye quién la resolvió y el momento en que lo realizó, lo cual permite un seguimiento claro y transparente.

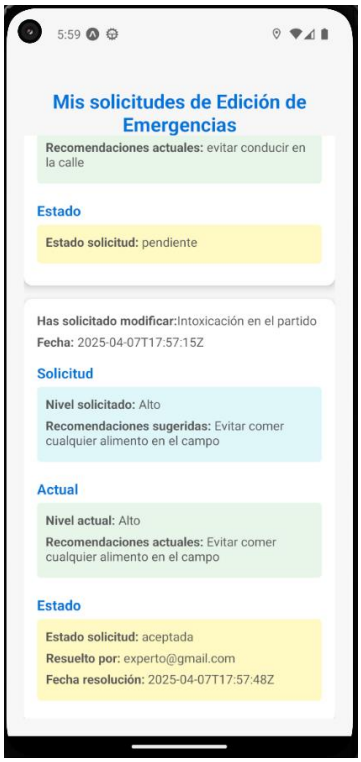

Figura 124: Pantalla Mis solicitudes de edición de emergencias con solicitud resuelta

B.9. Visualización mis solicitudes de cambio de rol

Seleccionando en el menú Mis solicitudes cambio de rol, se accede a la pantalla de la Figura 125.

En ella se listan todas las solicitudes de cambio de rol realizadas por el usuario. Las solicitudes resueltas incluyen también la información necesaria para su seguimiento.

Figura 125: Pantalla Mis solicitudes de cambio de rol

Funcionalidades como experto y autoridad:

B.10. Edición y verificación de una emergencia

Si se inicia sesión con uno de los roles mencionados y se accede a una emergencia en concreta, se accede a la Figura 126.

Se puede visualizar como en este caso en vez de solicitar aparece Editar Emergencia, por lo que si se selecciona dicha opción se pasa a la Figura 127.

Se pueden editar directamente los campos deseados e incluso verificar la emergencia seleccionando el switch de verificación.
Si se verifica la emergencia y se guardan los cambios, en el mapa interactivo el símbolo aparecerá con un efecto de pulso, diferenciándose visualmente de las demás.

Figura 126: Pantalla visualización detalles de una emergencia como experto o autoridad

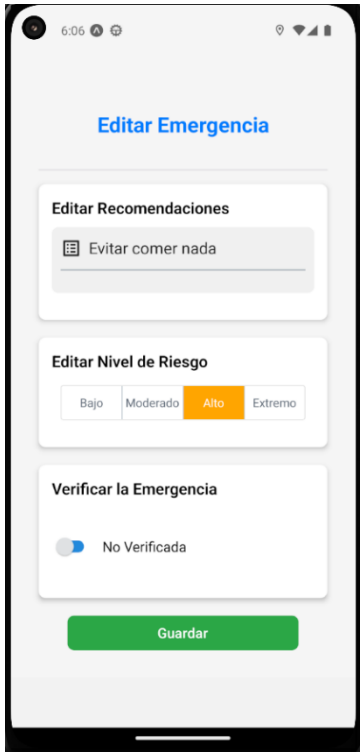

Figura 127: Pantalla edición de emergencia como experto o autoridad

B.11. Resolución de una emergencia

En el caso de la resolución de emergencias, dichos roles pueden hacer que se resuelvan directamente. Por lo que si se accede a una emergencia concreta y se desliza hacia abajo se acede a la Figura 128.

Figura 128: Pantalla resolución de emergencia como experto o autoridad

Se ofrece la opción de resolver emergencia. Al seleccionarla y confirmar se visualiza la Figura 129.

Figura 129: Pantalla confirmación resolución de emergencia

La emergencia desaparece del mapa y pasa a estar disponible únicamente desde el historial de emergencias como Emergencia Finalizada.

B.12. Gestión solicitudes de Edición de Emergencias

Desde el menú se accede a Solicitudes edición emergencias y se puede visualizar la Figura 130.

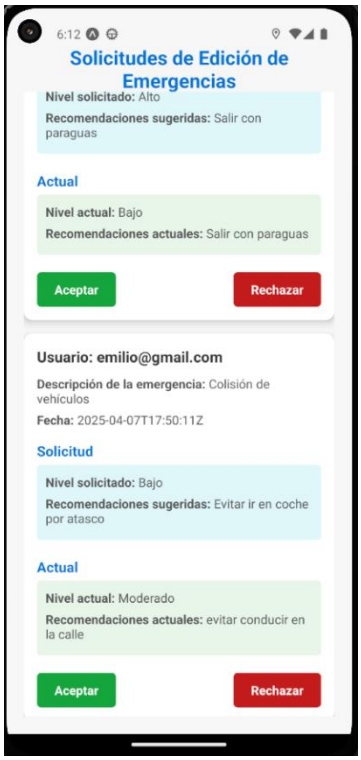

Figura 130: Pantalla Solicitudes de edición de emergencias pendientes

Se puede aceptar o rechazar las solicitudes, en las cuáles pone toda la información necesaria, tanto la actual como la solicitud, la emergencia en concreto que se quiere editar y la fecha en que se realizó la solicitud.

En caso de aceptarla, se aplican los cambios y la lista se actualiza

automáticamente mostrando solo las pendientes.
En caso de rechazar, simplemente se recarga la pantalla ocultando la solicitud resuelta.
Por lo que si se selecciona Rechazar se pasa a la Figura 131.

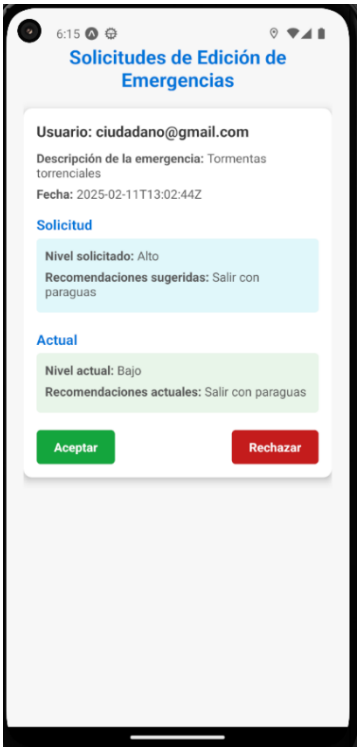

Figura 131: Pantalla Solicitudes de edición de emergencias tras rechazar solicitud

Aquí se puede observar que la otra solicitud como se ha resuelto ya no aparece aquí.

Funcionalidades como administrador:

B.13. Gestión solicitudes de Cambio de Rol

Desde el menú, el administrador accede a Solicitudes pendientes cambio de rol, como se puede visualizar en la Figura 132.

Aquí se muestran todas las solicitudes activas. El administrador puede Aceptar o Rechazar cada una de ellas, completando así la gestión de roles en el sistema.

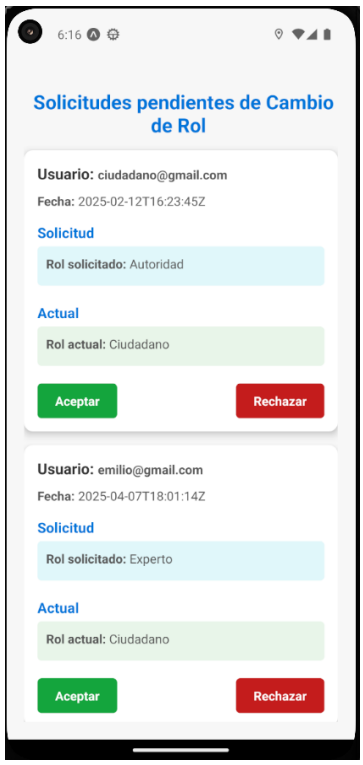

Figura 132: Pantalla Solicitudes pendientes de cambio de rol

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