



UNIVERSIDAD DE GRANADA

TRABAJO FIN DE GRADO
GRADO EN INGENIERÍA INFORMÁTICA

Herramienta web/móvil para informar sobre eventos de inseguridad ciudadana

Autor

David Carrasco Chicharro

Directores

Juan Manuel Fernández Luna
Sylvia Acid Carrillo



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE
TELECOMUNICACIÓN

Granada, Julio de 2020

Herramienta web/móvil para informar sobre eventos de inseguridad ciudadana

David Carrasco Chicharro

Palabras clave: *desarrollo web, aplicación móvil, incidentes, delitos, mapas de zonas, clustering, notificaciones, botón del pánico*

Resumen

En el presente Trabajo de Fin de Grado se expondrá el proceso de creación de un sitio web y una aplicación móvil cuya finalidad es recoger, por parte de unos usuarios, los incidentes ocurridos en la vía pública e informar de ellos al resto de usuarios de la plataforma, ofreciéndoles, además, diversas herramientas que les permiten establecer una red de contactos a la que alertar en caso de inseguridad o peligro, e incluso la posibilidad de guardar zonas de especial interés. Además, cada nuevo incidente será publicado en redes sociales para ampliar el canal de comunicación más allá del propio entorno del proyecto a fin de lograr un mayor impacto público y ampliar la red de usuarios.

El desarrollo del proyecto contará con distintas fases, comenzando por un estudio del problema a tratar y con un análisis desde el ámbito de la ingeniería del software, continuando por una fase de diseño de la aplicación y finalizando con la implementación de la misma, siguiendo los pasos marcados en las fases anteriores a fin de lograr los objetivos propuestos, que se resumen en proveer a la población una herramienta tecnológica que proporcione, en la medida de lo posible, seguridad e información de lo que ocurre en su entorno o lugares próximos.

La interfaz de usuario, por su componente visual, tiene una gran importancia en la aplicación, donde gran parte del protagonismo se centra en mapas donde se reflejarán los incidentes citados, facilitando la identificación zonas con un mayor número de incidentes o donde estos tienen una mayor gravedad, de modo que esta información sea accesible de manera sencilla a todo tipo de usuarios. Para la identificación de zonas de mayor incidencia se hará uso de algoritmos que permitan, por un lado, asignar un nivel de gravedad a cada incidente de manera automática, y por otro, agruparlos por medio de técnicas de *clustering*.

Web/mobile tool to inform about citizen insecurity events

David Carrasco Chicharro

Keywords: *web development, mobile application, incidents, crimes, area maps, clustering, notifications, panic button*

Abstract

In this Final Degree Project, it will be exposed the process of creating a website and a mobile application. Its purpose is to collect, on the part of some users, the incidents that have occurred on the public road and to inform the rest of users of the platform about the incidents, providing in addition, several tools that allow for establish a network of contacts which can alert in case of insecurity or danger, and even the possibility of store areas of special interest. Besides, each new incident will be published in social networks to extend the communication channel beyond the environment project in order to achieve the greatest public impact and increase the users' network.

The development of the project will have different phases, starting with a study of the problem to be addressed and an analysis from the field of software engineering, continuing with a design phase of the application and ending with the implementation of the same, following the steps outlined in the previous phases in order to achieve the proposed objectives, which can be summarized in supplying the population a technological tool that provides, as far as possible, security and information about what happens in their environment or nearby places.

The user interface, because of its visual component, has a great importance in the application, where much of the attention is focused on maps that reflect the incidents mentioned, facilitating the identification of areas with a greater number of incidents or where these have a greater severity, so that this information is easily accessible to all types of users. In order to identify the areas with the highest incidence, algorithms will be used that allow, on the one hand, to automatically assign a level of severity to each incident, and on the other, to group them by means of clustering techniques.

Yo, **David Carrasco Chicharro**, alumno de la titulación *Grado en Ingeniería Informática* de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación** de la **Universidad de Granada**, con D.N.I. 15520228-N, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: David Carrasco Chicharro

Granada a 7 de julio de 2020

D. Juan Manuel Fernández Luna, Profesor del Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada.

D.^a Sylvia Acid Carrillo, Profesora del Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada.

Informan:

Que el presente trabajo, titulado ***Herramienta web/móvil para informar sobre eventos de inseguridad ciudadana***, ha sido realizado bajo mi supervisión por **David Carrasco Chicharro**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a 7 de Julio de 2020.

Los directores:

Juan Manuel Fernández Luna

Sylvia Acid Carrillo

Agradecimientos

A mi madre, pilar fundamental de mi vida, por su incansable lucha e incommensurable esfuerzo, haciendo posible que llegue hasta aquí. Gracias.

A mi familia, amigas y amigos, que siempre me han respaldado en cada una de las decisiones que he tomado y han hecho más llevadero este duro camino.

A todos los profesores y profesoras que durante toda mi trayectoria académica me han permitido aprender. Y a mis tutores, Juanma y Sylvia, cuya confianza en mí junto a su ayuda han sido cruciales y de gran valor para sacar adelante este trabajo.

Índice

1. Introducción	1
1.1. Motivación	1
1.2. Descripción del problema	1
1.3. Estado del arte	2
1.3.1. Aplicaciones actuales	2
1.3.2. Crítica al estado del arte	2
1.4. Propuesta	3
1.5. Objetivos	4
1.6. Estructura de la memoria	5
2. Planificación	7
2.1. Desarrollo de software	7
2.2. Planificación temporal	9
2.3. Presupuesto	10
3. Análisis	13
3.1. Implicados	13
3.1.1. Entorno de usuario	13
3.1.2. Resumen de implicados	13
3.1.3. Perfiles de los implicados	14
3.2. Especificación de requisitos	15
3.2.1. Requisitos funcionales	15
3.2.2. Requisitos de Información	18

3.2.3. Restricciones Semánticas	19
3.2.4. Requisitos No Funcionales	21
3.3. Modelo de casos de uso	23
3.3.1. Diagramas de casos de uso	23
3.3.2. Descripción de los casos de uso (plantilla básica)	28
3.4. Diagramas de secuencia	52
4. Diseño	68
4.1. Diseño de la base de datos	68
4.1.1. Paso a tablas y fusión	69
4.1.2. Normalización de la base de datos	70
4.2. Diseño de la arquitectura del sistema	71
4.3. Algoritmos	72
4.3.1. Asignación del nivel de gravedad a los incidentes	72
4.3.2. Agrupación de incidentes	73
4.3.3. Asignación de color a zonas de incidentes	74
4.4. Diseño de la interfaz de usuario	75
4.4.1. Diseño de la interfaz web	75
4.4.2. Diseño de la interfaz de la aplicación móvil	83
5. Implementación	87
5.1. Desarrollo web	87
5.1.1. Entorno de desarrollo	87
5.1.2. Diagrama de clases	88
5.1.3. Implementación de la base de datos	91

5.1.4. Desarrollo	95
5.2. Desarrollo de aplicación móvil	104
5.2.1. API	104
5.2.2. Entorno de desarrollo	105
5.2.3. Desarrollo	107
5.3. Despliegue del proyecto	110
6. Conclusiones	112
6.1. Temporización real	112
6.2. Objetivos alcanzados	114
6.3. Lecciones aprendidas	114
6.4. Trabajos futuros	114
7. Bibliografía	116
Referencias	116
8. Anexos	119
A. Delitos extraídos del Código Penal	119
B. Vista del sitio web	122
C. Vista de la aplicación móvil	132
D. Código fuente y sitio web	137

Índice de figuras

Figura 1.	Planificación inicial del proyecto	9
Figura 2.	Diagrama de caso de uso - Gestión de administradores y usuarios	24
Figura 3.	Diagrama de caso de uso - Gestión de incidentes	25
Figura 4.	Diagrama de caso de uso - Gestión de contactos favoritos	26
Figura 5.	Diagrama de caso de uso - Gestión de zonas y ubicación	27
Figura 6.	DS: Alta de un usuario o administrador	53
Figura 7.	DS: Baja de un usuario o administrador	54
Figura 8.	DS: Consultar datos de un usuario o administrador	54
Figura 9.	DS: Modificar datos de un usuario o administrador	55
Figura 10.	DS: Añadir un incidente, publicarlo en RR.SS. y notificarlo a otro usuario	56
Figura 11.	DS: Eliminar un incidente	56
Figura 12.	DS: Ocultar un incidente	57
Figura 13.	DS: Establecer parámetros para caducar un incidente y caducidad del mismo	57
Figura 14.	DS: Establecer nivel de gravedad de un incidente	58
Figura 15.	DS: Analizar incidentes	58
Figura 16.	DS: Consultar incidentes y mostrarlos	59
Figura 17.	DS: Establecer parámetros para añadir un contacto favorito y aceptar o rechazar serlo	60
Figura 18.	DS: Buscar un contacto favorito y eliminarlo	61
Figura 19.	DS: Ordenar contactos favoritos	62
Figura 20.	DS: Consultar de quién se es contacto favorito y eliminarse como tal	63

Figura 21. DS: Establecer parámetros para añadir una zona de interés y añadirla	64
Figura 22. DS: Eliminar una zona de interés	65
Figura 23. DS: Activar/desactivar ubicación, compartirla y dejar de compartir	65
Figura 24. DS: Establecer PIN secreto y rastrear smartphone vía web o desde un contacto favorito	66
Figura 25. DS: Establecer acción de pánico, activar botón del pánico y confirmar buen estado	67
Figura 26. Diagrama E/R	69
Figura 27. Modelo-Vista-Controlador	71
Figura 28. Asignación de color a los centros de zonas de incidentes	75
Figura 29. Interfaz de inicio de sesión	76
Figura 30. Interfaz registro	77
Figura 31. Interfaz de zona personal de usuario	78
Figura 32. Interfaz de mapa de incidentes	79
Figura 33. Interfaz de lista de incidentes	79
Figura 34. Interfaz de alta de incidente	80
Figura 35. Interfaz de contactos favoritos	81
Figura 36. Interfaz de zonas de interés	81
Figura 37. Interfaz de inicio de administración	82
Figura 38. Interfaz configuración en administración	83
Figura 39. Interfaces de login y registro en aplicación móvil	84
Figura 40. Interfaces de zona personal y configuración de acción de pánico y pin secreto en aplicación móvil	84
Figura 41. Interfaces de lista, mapa y filtro de incidentes en aplicación móvil	85

Figura 42. Interfaces de incidente, alta de incidente y notificaciones en aplicación móvil	86
Figura 43. Interfaces de zonas de interés y contactos favoritos en aplicación móvil	86
Figura 44. Diagrama de clases	89
Figura 45. Proceso gráfico del algoritmo de clustering	101
Figura 46. Sistemas operativos en dispositivos móviles en el mercado [41] . .	106
Figura 47. Temporización real del proyecto	112
Figura 48. Web - Login	122
Figura 49. Web - Registro - Paso 1	122
Figura 50. Web - Registro - Paso 2	123
Figura 51. Web - Lista de incidentes	124
Figura 52. Web - Mapa de incidentes	125
Figura 53. Web - Alta de incidentes	126
Figura 54. Web - Contactos favoritos	127
Figura 55. Web - Buscar usuario	127
Figura 56. Web - Añadir contacto	128
Figura 57. Web - Zonas de interés	128
Figura 58. Web - Alta de zona de interés	129
Figura 59. Web - Zona personal	130
Figura 60. Web - Selección de acción de pánico	131
Figura 61. App - Login y registro en dos pasos	132
Figura 62. App - Lista y mapa de incidentes y detalle de incidente	133
Figura 63. App - Alta de un incidente	133

Figura 64. App - Zona personal, modificación de datos personales y establecimiento de acción de pánico	134
Figura 65. App - Zonas de interés y notificación de incidente	135
Figura 66. App - Contactos favoritos y detalle de contacto	135
Figura 67. App - Botón del pánico (vista de usuario)	136
Figura 68. App - Botón del pánico (vista de contacto favorito)	136

Índice de tablas

Tabla 1. Planificación inicial del proyecto	10
Tabla 2. Coste mensual para la empresa	11
Tabla 3. Coste total del proyecto	12
Tabla 4. Resumen de implicados	13
Tabla 5. Perfil de implicado: administrador	14
Tabla 6. Perfil de implicado: usuario	14
Tabla 7. Relaciones en diagramas de casos de uso	23

1. Introducción

1.1. Motivación

A diario, cuando paseamos por la calle o nos desplazamos de un lugar a otro, podemos sentir cierta inseguridad, especialmente en algunas zonas de los pueblos y ciudades donde el contexto y abandono social hacen que estos lugares se perciban como peligrosos. Al margen de este tipo de zonas, en horas nocturnas el miedo a sufrir algún percance es mucho mayor y, actualmente, la preocupación ciudadana por este hecho es creciente, especialmente entre las mujeres, principales víctimas de actos sexistas, acosos, abusos y agresiones sexuales, sin olvidar a otros colectivos vulnerables, propensos a la aversión de seres intolerantes, intransigentes y con ápices de algún fanatismo vacuo que sustancialmente se traduce en delincuencia.

Según el Centro de Investigaciones Sociológicas (CIS), en una encuesta multirrespuesta [1] sobre los tres problemas principales que existen actualmente en España, la inseguridad ciudadana se encuentra como uno de los veinte problemas que la sociedad percibe como relevantes.

Ante este problema, surge la necesidad de desarrollar desde un perfil tecnológico un instrumento que permita de manera sencilla para el usuario conocer qué ocurre en la calle para aportar seguridad a sus trayectos a fin de evitar cualquier tipo de contratiempo ajeno a uno mismo.

1.2. Descripción del problema

Se plantea, a partir de la problemática expuesta, la creación de una página web y una aplicación móvil que permitan la notificación, por parte de la ciudadanía, de aquellos incidentes o actos delictivos de los que sean testigos o incluso que, por desgracia, sufran ellos mismos. Más allá de esto, las personas que hagan uso de la vía pública pueden requerir en un momento de percepción de peligro de una herramienta que les proporcione seguridad. Las nuevas tecnologías pueden ayudar en este cometido proporcionando mecanismos que alerten a personas cercanas de la situación de peligro expuesta, para lo cual la aplicación creada puede ser de gran utilidad.

1.3. Estado del arte

1.3.1. Aplicaciones actuales

Actualmente existen diversas aplicaciones cuya finalidad es alertar de situaciones de peligro a sus contactos. Las de mayor uso son:

- **AlertCops:** aplicación del Ministerio del Interior integrada con los sistemas de la Policía Nacional y la Guardia Civil que permite notificar a las Fuerzas y Cuerpos de Seguridad del Estado de situaciones de riesgo que se estén produciendo. También permite enviarles la ubicación actual en caso de peligro para que estos actúen directamente. Integra un chat que permite establecer una comunicación directa con un agente de los cuerpos mencionados. También dispone de una opción llamada “guardián” que posibilita compartir ubicación con otras personas. [2]
- **Sister:** aplicación orientada principalmente a compartir ubicación con otros usuarios y buscar rutas seguras. Permite activar una alarma disuasoria que emite un fuerte sonido si pasado un tiempo establecido no se desactiva manualmente. [3]
- **Red Panic Button:** botón del pánico que envía una alerta vía correo electrónico y/o SMS a los contactos del usuario. [4]
- **bSafe:** permite compartir una emisión en vídeo en directo a la par que la ubicación del dispositivo y crear una red de contactos para su recepción. [5]
- **Life360:** aplicación orientada principalmente a compartir el trayecto del usuario con sus contactos. Utiliza un mecanismo que detecta posibles accidentes si el trayecto en un vehículo se detiene de manera imprevista. Ofrece una opción de soporte en carretera. Incluye también una opción de alerta y reporte de delitos. [6]

1.3.2. Crítica al estado del arte

Una característica común a todas las aplicaciones mencionadas en el apartado anterior es que sólo están disponibles como aplicaciones para dispositivos móviles, limitando así su uso y su distribución multiplataforma. Todas posibilitan la creación de una red de contactos a quienes alertar en una situación de peligro, siendo este el principal punto en común entre todas. En cuanto a la comunicación pública de incidentes sólo una de las listadas ofrece dicha opción, por lo que no contamos con plataformas que ofrezcan esta información a los usuarios.

De manera particular, se describen a continuación algunos inconvenientes que presentan:

- **AlertCops:** si bien es la más útil por su integración directa con las FFCCSE (Fuerzas y Cuerpos de Seguridad del Estado), su diseño es tosco y su interfaz lenta. El chat con los agentes tarda mucho en establecerse y la respuesta no es lo rápida que debería teniendo en cuenta la importancia de este aspecto dado el contexto. Aunque permite notificar incidentes, estos sólo se comunican a los cuerpos mencionados, de modo que no es transparente para el resto de usuarios.
- **Sister:** destaca por su fluidez, facilidad de uso y gran cantidad de funcionalidades, pero a nivel de marketing se publicita únicamente a un sector de la sociedad, lo cual reduce su uso por parte de los usuarios de aplicaciones móviles.
- **Red Panic Button:** no cuenta con soporte en la actualidad y los usuarios notifican fallos de funcionamiento críticos en las tiendas de aplicaciones donde se encuentra disponible. Además, es de pago si se quiere añadir a más de un contacto a la red.
- **bSafe:** sus funcionalidades son limitadas y está disponible exclusivamente en inglés, lo que hace que su uso en España sea prácticamente nulo. Igualmente, como la aplicación anterior, cuenta como gran cantidad de reseñas negativas en las tiendas de aplicaciones debido a fallos de funcionamiento.
- **Life360:** pese a que es la más completa de todas, está disponible únicamente en Estados Unidos –y por tanto en inglés–, imposibilitando su uso en España. Para las funcionalidades más avanzadas es necesaria una suscripción de pago.

1.4. Propuesta

Ante este breve estudio sobre el estado del arte actual en cuanto a aplicaciones de seguridad ciudadana y con lo planteado en la descripción del problema, propongo la creación de una aplicación multiplataforma, dando protagonismo a una página web que ofrezca el mismo contenido que la aplicación. La característica primordial del proyecto es la notificación y consulta de incidentes en la plataforma a los cuales se les pueda asignar un índice o nivel de peligrosidad, pudiendo así determinar qué zonas concentran una mayor cantidad de incidentes y brindar información detallada sobre lo expuesto, así como con un formato visual, sobrio y sencillo.

Sin dejar atrás una característica tan relevante como es la notificación en tiempo real de situaciones de peligro, la aplicación integrará un botón del pánico que notifique a la red de contactos sobre la situación de peligro del usuario.

El hecho de considerar incidentes con un nivel de gravedad plantea la necesidad de tener una base sólida sobre la cual clasificar dichos incidentes. La propuesta es utilizar un marco legal como es el Código Penal [7] para considerar los delitos adecuados en el contexto del trabajo, puesto que en dicho diario se detallan estos y sus penas. A partir de una revisión en profundidad del documento se seleccionan aquellos delitos tipificados adecuados en el contexto del presente proyecto (Anexo A).

En un primer momento se consideró seleccionar para cada delito sus agravantes asociados con el objetivo de tomarlos como referencia de la gravedad del delito cometido, pero esto planteaba varios problemas:

- Los agravantes a veces no son tenidos en cuenta (sólo en delitos cualificados) y son considerados como algo subjetivo en el ámbito judicial a la hora de ser juzgados.
- No todos los delitos tienen asociados las ocho categorías de agravantes tipificadas, lo que supondría considerar todas las casos posibles y almacenarlos posteriormente, lo cual complicaría innecesariamente el proceso de notificación de incidentes.

Por tanto se incluirá en el formulario de incidentes un apartado denominado “agravantes” que agrupará las cuatro categorías más comunes y tendrán un papel secundario en la aplicación.

1.5. Objetivos

El objetivo principal de este TFG es dar una solución global al problema descrito, cumpliendo la propuesta anteriormente formulada y satisfaciendo las necesidades particulares que surgen de este objetivo global, desagregadas en subobjetivos que han de ser específicos, medibles, asignables, realistas y alcanzables [8]:

OBJ-1. Gestión de usuarios e incidentes. El sistema deberá ser capaz de realizar una correcta gestión y almacenamiento de datos relativos a los administradores y usuarios, así como de los incidentes dados de alta por estos últimos.

OBJ-2. Creación de red de contactos favoritos. El sistema tendrá que permitir a los usuarios crear una red de contactos favoritos y posibilitarle su gestión a cada uno.

OBJ-3. Mantenibilidad de la administración El sistema deberá garantizar que únicamente los administradores tienen la potestad de eliminar incidentes y controlar parámetros relativos las distintas funcionalidades que se ofrecen.

OBJ-4. Garantía de acceso a la información. Cualquier usuario podrá realizar consultas de incidentes publicados.

OBJ-5. Información y alerta sobre incidentes. Ha de proporcionarse un mecanismo eficiente para que los usuarios tengan zonas de interés y sean notificados sobre incidentes que sucedan en el área que abarcan.

OBJ-6. Gestión de ubicación. El sistema tratará la ubicación de los dispositivos de los usuarios cuando estos comparten su localización para que sólo sea compartida con los receptores designados por el usuario.

OBJ-7. Asignación de nivel de gravedad a los incidentes. El sistema deberá ser capaz de estipular el nivel de gravedad de cada incidente en base a sus características propias y ambientales.

OBJ-8. Análisis de incidentes. El sistema tendrá la facultad de realizar un análisis de los incidentes de cada zona y determinará la magnitud e incidencia de cada zona en cuanto al nivel de gravedad de los incidentes.

1.6. Estructura de la memoria

Esta memoria se compone de las siguientes partes:

- **Introducción:** presentación del proyecto donde se trata en distintas secciones qué se pretende realizar.

- Motivación: cuenta brevemente las razones que dan lugar a materializar el proyecto.
- Descripción del problema.
- Estado del arte: investigación y crítica sobre los trabajos actuales similares al que se pretende desarrollar.
- Propuesta: conjunto de ideas orientadas a solucionar el problema tratado.
- Objetivos: metas que se pretenden alcanzar al finalizar el proyecto.

- **Planificación:** plan a seguir para la elaboración del trabajo, detallando en ella:

- Desarrollo de software: metodología utilizada.
- Planificación temporal: programación teórica de los plazos a cumplir durante el desarrollo.
- Presupuesto: cálculo de los costes que suponen llevar a cabo este proyecto.

- **Análisis:** proceso del desarrollo del software llevado a cabo para conseguir un resultado correcto, óptimo y seguro. Cuenta con la descripción de los implicados, la especificación de requisitos, el modelado y descripción de casos de uso y la explicación del sistema mediante diagramas de secuencia.
- **Diseño:** modelado de los componentes del sistema.
 - Base de datos.
 - Arquitectura del sistema.
 - Algoritmos empleados en la fase de implementación.
 - Interfaz de usuario para la web y la aplicación móvil.
- **Implementación:** trata las herramientas, frameworks y lenguajes empleados durante el desarrollo de la aplicación y se explica cómo se ha llevado a cabo la programación tanto para la web como para la aplicación móvil. Se explica también cómo es el proceso de puesta en producción de la aplicación.
- **Conclusiones:** analiza la temporización real del proyecto frente a la planificación inicial, además de los objetivos alcanzados, las lecciones aprendidas durante el desarrollo del trabajo y los trabajos futuros con los que continuar el proyecto.

2. Planificación

2.1. Desarrollo de software

Las metodologías de ingeniería del software surgieron para organizar el desarrollo software que se comenzó a generar intensivamente desde finales de la década de los 60. Las metodologías que surgieron primero pueden resultar demasiado pesadas, caracterizándose por ser rígidas y estar dirigidas por la documentación generada en cada actividad, lo que hizo aparecer enfoques modernos y metodologías ágiles como alternativa a estos procesos tradicionales [9]. Las metodologías iniciales se caracterizaban por no adecuarse a lo que más convenía en el momento, dando lugar esta falta de flexibilidad en muchos casos a no escoger ninguna, lo que daba lugar a un resultado de mala calidad en un proceso caótico.

Las metodologías de desarrollo ágil están pensadas para proyectos donde los equipos de desarrollo son pequeños, con plazos reducidos, requisitos volátiles y basados en nuevas tecnologías. Se orientan a proyectos donde la solución debe ser concisa y simple, sin olvidar la calidad del producto. Estas características encajan a la perfección con el propósito general del proyecto. Esta metodología requiere también una colaboración estrecha con el cliente, que particularmente se tratará de los tutores que dirigen este trabajo.

De entre todas las metodologías de desarrollo ágil cabe destacar Scrum, especialmente popular en proyectos emprendedores. Scrum se desarrolla mediante “sprints”, que son elementos que se corresponden con etapas de trabajo, presentándose en cada una de ellas una versión utilizable del producto [10]. Los “sprints”, también llamados iteraciones, son no lineales y flexibles y en ellos el desarrollador expone qué se ha realizado desde la última reunión y qué se va a hacer hasta la próxima. Puesto que el presente trabajo tiene asociado un importante número de tareas más allá de las propias relacionadas con el desarrollo, se incluyen como iteraciones las primeras reuniones en las que se abordan temas relacionados con la posterior puesta en marcha del software, referidas a continuación:

Sprint 1:

- Tarea: elección de un modelo de memoria, investigación sobre el estado del arte, redacción de objetivos, temporización de tareas y proyección de un modelo para el cálculo de costes.

Sprint 2:

- Revisión de la documentación realizada en el sprint anterior.

- Tarea: análisis de requisitos

Sprint 3:

- Revisión del análisis de requisitos.
- Tarea: análisis de casos de uso y diseño del modelo de base de datos.

Sprint 4:

- Revisión de casos de uso y del diseño de la base de datos.
- Tarea: ampliar casos de uso (plantillas básicas), corregir diseño de base de datos, realizar diagramas de secuencia y comenzar diseño web.

Sprint 5:

- Revisión de casos de uso, diagramas de secuencia y diseño de la base de datos, dando por finalizada la fase de análisis. Se presenta un primer boceto del diseño web.
- Tarea: diseñar aplicación móvil y comenzar desarrollo web con la gestión básica de usuarios y administradores.

Sprint 6:

- Se presenta la primera versión de la página web con toda la gestión de usuarios completa y siendo totalmente funcional el alta, baja y modificación de datos de usuarios en el sistema. Se presenta el boceto de la aplicación móvil.
- Tarea: finalizar desarrollo web

Sprint 7:

- Entrega de la web con la gestión de incidentes, zonas de interés y relación entre contactos concluida.
- Tarea: aplicar cambios de diseño a la web y crear aplicación móvil .

Sprint 8:

- Entrega de la aplicación móvil.
- Tarea: crear algoritmo para calcular la gravedad de los incidentes y mostrar las zonas con concentración de estos.

Sprint 9:

- Finalización del desarrollo de software una vez concluidas todas las tareas necesarias y cumplidos los objetivos planteados.
- Tarea: redacción de la documentación.

2.2. Planificación temporal

En este apartado se expone la planificación inicial realizada para las distintas fases del proyecto. El objetivo no es sólo tener una guía con la que marcar las pautas del proceso a seguir, sino también poder valorar, una vez finalizado, cómo se ha adecuado la planificación inicial con respecto al desarrollo real, qué partes han requerido más tiempo del propuesto y cuáles menos y, finalmente, realizar una evaluación general del trabajo en términos del tiempo empleado con la que obtener conclusiones al respecto.

De manera visual se muestra en la Figura 1 la planificación inicial mediante un diagrama de Gantt, que es una herramienta gráfica con la que exponer previsiones temporales para tareas.

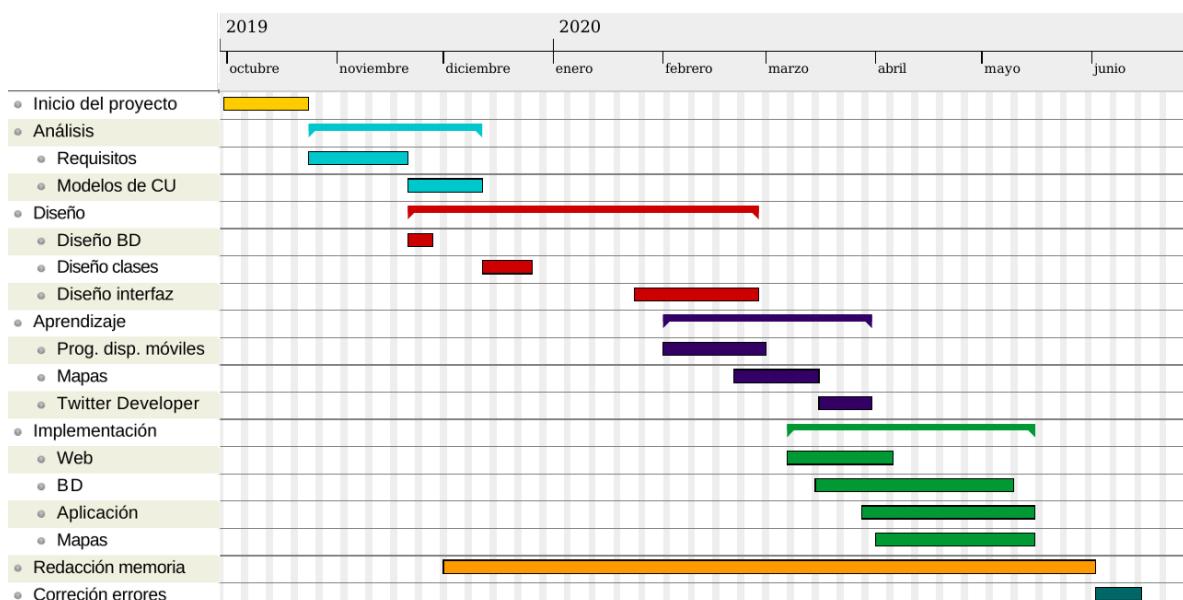


Figura 1: Planificación inicial del proyecto

Nombre	Fecha de inicio	Fecha de fin	Semanas
Inicio del proyecto	30/09/19	23/10/19	3
Análisis	24/10/19	11/12/19	7
Requisitos	24/10/19	20/11/19	4
Modelos de CU	21/11/19	11/12/19	3
Diseño	21/11/19	27/02/20	8
Diseño BD	21/11/19	27/11/19	1
Diseño clases	12/12/19	25/12/19	2
Diseño interfaz	24/01/20	27/02/20	5
Aprendizaje	01/02/20	30/03/20	8
Prog. disp. móviles	01/02/20	29/02/20	4
Mapas	21/02/20	15/03/20	3
Twitter Developer	16/03/20	30/03/20	2
Implementación	07/03/20	15/05/20	10
Web	07/03/20	05/04/20	4
BD	15/03/20	09/05/20	8
Aplicación	28/03/20	15/05/20	7
Mapas	01/04/20	15/05/20	7
Redacción memoria	01/12/19	01/06/20	26
Corrección errores	02/06/20	14/06/20	2

Tabla 1: Planificación inicial del proyecto

2.3. Presupuesto

Todo proyecto tiene, inherentemente, costes asociados que es necesario sufragar. Bajo el supuesto de que este proyecto se desarrollase bajo la contratación de una persona con los conocimientos técnicos correspondientes para tal fin, se detalla a continuación un desglose de los gastos de personal y de ejecución vinculados al desarrollo de este trabajo.

Según las ofertas disponibles en portales profesionales de la web como InfoJobs o Jooble, para programadores junior orientados al desarrollo web y de aplicaciones los costes relativos al salario oscilan en un amplio rango entre 18000€ y 24000€ brutos

anuales. Por tanto, estableciendo un salario de 21600€ anuales para este caso, el coste mensual sería de 1800€. Este importe, sin embargo, no refleja la suma total relativa al gasto de personal a pesar de ser una cantidad en bruto.

Para dicha mensualidad se calcula a continuación el desglose de impuestos por parte del empleado y de la empresa a cargo a fin de conocer cuál es el salario neto que corresponde a quien lleva a cabo la labor del proyecto y los gastos totales mensuales con los que legalmente tiene que correr a cargo la entidad [11].

Coste mensual para la empresa	2.401,20 €	
Seguridad Social a cargo de la empresa		-601,20 €
Contingencias comunes	23,60 %	424,80 €
Prestaciones por desempleo	5,50 %	99,00 €
IT/IMS	3,50 %	63,00 €
Formación profesional	0,60 %	10,80 €
FOGASA	0,20 %	3,60 €
Salario bruto del trabajador	1.800 €	
Impuestos y seguridad social a cargo del trabajador		-418,14 €
I.R.P.F.	16,88 %	303,84 €
Cotización a la Seguridad Social		114,30 €
Contingencias comunes	4,70 %	84,60 €
Desempleo	1,55 %	27,90 €
Formación profesional	0,10 %	1,80 €
Salario neto	1.381,86 €	

Tabla 2: Coste mensual para la empresa

El importe mensual es, para los 1800€ brutos estipulados, de 2401,20€ con la suma de impuestos a cargo de la entidad contratante.

Una vez calculado el coste mensual del personal es necesario conocer los gatos de ejecución, que incluyen el material inventariable, el material fungible y los contratos, facturas y pagos de alquiler. Al material inventariable se le aplica la amortización de su uso, mientras que el resto se especifica por su coste total o su gasto mensual. A todo ello se le aplica el I.V.A. pertinente, que es para todos la casuística del 21 %.

Agregados todos los costes mencionados (detallados a continuación), el importe total se estima en 32766,33 €.

GASTOS DE PERSONAL

Descripción	Uds.	Precio	Meses	Base imponible	I.V.A. (21 %)
Sueldo	1	2.401,20 €	9	21.610,80 €	4.538,27 €

Total gasto de personal **26.149,07 €**

GASTOS DE EJECUCIÓN

Descripción	Uds.	Precio	Amortiz.	Base imponible	I.V.A. (21 %)
Material inventariable					
Disco duro	1	97,90 €	100 %	97,90 €	20,56 €
Smartphone	1	230,00 €	15,00 %	34,50 €	7,25 €
Computador Personal	1	978,11 €	18,75 %	183,40 €	38,51 €

Total material inventariable **382,11 €**

Descripción	Uds.	Precio	Base imponible	I.V.A. (21 %)
-------------	------	--------	----------------	---------------

Material fungible				
Folios (paq. 500 uds.)	1	4,13 €	4,13 €	0,87 €
Grapas, clips, bolígrafos	1	9,53 €	9,53 €	2,00 €

Total material fungible **16,53 €**

Contratos y alquileres

Alquiler del local	9	395,00 €	3.555,00 €	746,55 €
Facturas electricidad	9	63,20 €	568,80 €	119,45 €
Facturas agua	9	23,70 €	213,30 €	44,79 €
Limpieza	9	31,60 €	284,40 €	59,72 €
Internet y línea telefónica	9	39,50 €	355,50 €	74,66 €
Servidor	3	50,00 €	150,00 €	31,50 €
Dominio Internet	1	12,36 €	12,36 €	2,60 €

Total alquiler y facturas **6.218,63 €**

TOTAL 32.766,33 €

Tabla 3: Coste total del proyecto

3. Análisis

3.1. Implicados

Los implicados en un proyecto software son aquellos que tienen un rol determinado en el problema a resolver. En la fase de análisis es crucial centrarse en los actores con implicación en el problema final, es decir, los *stakeholders* (usuarios finales) [12]. Estos se enmarcan en el denominado “entorno de usuario”, descrito a continuación.

3.1.1. Entorno de usuario

Los usuarios directos de la aplicación a desarrollar son dos:

- Administrador. Encargado de gestionar el sistema con todos los privilegios y funciones necesarias para su correcto funcionamiento.
- Usuarios. Podrán añadir y consultar incidentes, así como ser avisados de los que ocurran nuevos y disponer de herramientas que les ayuden en caso de peligro.

3.1.2. Resumen de implicados

Nombre	Descripción	Tipo	Responsabilidad
Administrador	Administrador del sistema	Usuario producto	Gestionar el sistema, establecer parámetros, decidir los análisis a realizar y quitar incidentes.
Usuario	Persona que hace uso del sistema de manera activa.	Usuario producto	Añadir nuevos incidentes, consultarlos, recibir avisos, gestionar contactos y compartir su ubicación.

Tabla 4: Resumen de implicados

3.1.3. Perfiles de los implicados

Administrador

Representante	Nombre Apellidos
Descripción	Administrador
Tipo	Experto en el sistema, con responsabilidades de gestión sobre el mismo.
Responsabilidades	Gestionar el sistema para su correcto funcionamiento: establecer parámetros mínimos y máximos y eliminar u ocultar incidentes.
Criterios de éxito	Que el sistema funcione adecuadamente con los parámetros óptimos y no exista información incorrecta.
Implicación	Total, pues es el encargado principal del sistema.

Tabla 5: Perfil de implicado: administrador

Usuario

Representante	Nombre Apellidos
Descripción	Persona con interés/inquietud en los incidentes de su zona y alrededores.
Tipo	Utiliza el sistema de forma directa, teniendo un conocimiento moderado sobre su uso.
Responsabilidades	Registrarse en el sistema. Añadir y consultar incidentes. Gestionar sus contactos favoritos. Añadir zonas de interés. Hacer uso del botón del pánico.
Criterios de éxito	Que el sistema le permita añadir un incidente o consultarlos de manera sencilla. Recibir de manera oportuna notificaciones en función de sus preferencias o ubicación. Que se contacte con un contacto favorito en caso de situación de peligro.
Implicación	Variable, en función del uso que quiera dar cada usuario; especialmente alta para quienes añadan nuevos incidentes y parcial para quienes solo hagan consultas.

Tabla 6: Perfil de implicado: usuario

3.2. Especificación de requisitos

3.2.1. Requisitos funcionales

Descripción de los requisitos fundamentales del sistema [13] en cuanto a las funcionalidades que debe incluir e implementar, realizando una clasificación en categorías. Cada requisito tiene asignado un código y un nombre con el fin de identificarlo fácilmente.

- RF-1. Gestión de administradores.** Se permitirá el alta/baja con el rol de administrador del sistema, así como consultar y modificar sus datos. También podrá consultar los datos y estadísticas de usuarios y darlos de baja. Se permitirá igualmente eliminar, caducar u ocultar incidentes si se considera oportuno y se podrán establecer parámetros y valores máximos para ciertas funcionalidades del sistema.
- RF-1.1. Alta de administrador.** Se registrará cada administrador con sus datos correspondientes.
- RF-1.2. Baja de administrador.** Se eliminará al administrador y la información relativa a él.
- RF-1.3. Consultar datos de administrador.** Se mostrarán los datos relativos a un determinado administrador.
- RF-1.4. Modificar datos de administrador.** Se podrán cambiar los datos almacenados de un administrador.
- RF-1.5. Baja de usuario.** Eliminará a un usuario y la información relativa a él.
- RF-1.6. Consultar datos y estadísticas por usuario.** Se mostrarán los datos y las estadísticas relativas a un determinado usuario.
- RF-1.7. Eliminar incidente.** Se eliminará un incidente almacenado en el sistema.
- RF-1.8. Ocultar incidentes.** Se ocultará al usuario un incidente o tipo de incidente concreto, pero este no será eliminado del sistema.
- RF-1.9. Establecer radio y tiempo para caducidad de incidentes.** Se determinará, para cada incidente, el radio alrededor de cada uno y el tiempo que ha de pasar sin que se produzcan otros para que pueda caducar un incidente.
- RF-1.10. Caducar incidente.** Se ocultará al usuario un incidente y no se reflejará en ningún lugar, pero no será eliminado del sistema.

- RF-1.11. Establecer máximo número de contactos favoritos.** Se determinará cuántos contactos favoritos puede tener un usuario como máximo.
 - RF-1.12. Establecer radio máximo para zonas de interés.** Se determinará cuál es el radio máximo que puede abarcar una zona de interés.
 - RF-1.13. Establecer número máximo de zonas de interés.** Se determinarán cuántas zonas de interés puede tener como máximo un usuario.
- RF-2. Gestión de usuarios.** Se permitirá el alta/baja con el rol de usuario del sistema, así como consultar y modificar sus datos.
- RF-2.1. Alta de usuario.** Se registrará cada usuario con sus datos correspondientes.
 - RF-2.2. Baja de usuario.** Se eliminará al usuario y la información relativa a él.
 - RF-2.3. Consultar datos y estadísticas por usuario.** Se mostrarán los datos relativos al propio usuario.
 - RF-2.4. Modificar datos de usuario.** Se podrán cambiar los datos almacenados del propio usuario.
- RF-3. Gestión de contactos favoritos.** Se permitirá añadir uno o varios contactos favoritos y eliminarlos, así como consultar cuáles son, ordenarlos, aceptar/rechazar ser contacto favorito de otro usuario, consultar de quién se es contacto favorito y eliminarse como tal.
- RF-3.1. Añadir contacto favorito.** Se añadirá a otro usuario como contacto favorito.
 - RF-3.2. Eliminar contacto favorito.** Se eliminará a otro usuario como contacto favorito.
 - RF-3.3. Consultar mis contactos favoritos.** Se mostrarán los contactos que el usuario tiene añadidos como favoritos.
 - RF-3.4. Ordenar contactos favoritos.** Se ordenarán los contactos favoritos según el criterio del usuario.
 - RF-3.5. Aceptar/rechazar ser contacto favorito.** El usuario pasará a ser contacto favorito de quien le añade como tal o rechazará serlo.
 - RF-3.6. Consultar de quién se es contacto favorito.** Se mostrará de qué usuarios se es contacto favorito.
 - RF-3.7. Eliminarse como contacto favorito.** El usuario será eliminado de la lista de contactos favoritos de un usuario determinado.

RF-4. Gestión de incidencias. Se añadirán incidentes y se podrán consultar por diferentes criterios. También se podrán añadir y eliminar zonas de interés y se notificará de incidentes a usuarios que tengan añadida una zona de interés o se encuentren cerca de donde se ha producido uno.

RF-4.1. Añadir incidente. Se añadirá un nuevo incidente en el sistema con sus datos correspondientes.

RF-4.2. Consultar incidentes. Se podrán consultar los incidentes por diferentes criterios.

RF-4.3. Mostrar incidentes. Se mostrarán los incidentes en base al criterio elegido.

RF-4.3.1. Mostrar lista de incidentes. Los incidentes serán mostrados en una lista.

RF-4.3.2. Mostrar mapa de incidentes. Los incidentes serán mostrados en un mapa.

RF-4.4. Consultar incidentes subidos. Se mostrarán los incidentes que el usuario haya añadido al sistema.

RF-4.5. Consultar detalle de incidente. Se mostrarán los datos relativos a un incidente concreto.

RF-4.6. Añadir zona de interés. Se añadirá una zona concreta al sistema como lugar preferente para el usuario.

RF-4.7. Eliminar zona de interés. Se eliminará una determinada zona de interés.

RF-4.8. Notificar incidente. Se informará a un usuario de un incidente.

RF-4.9. Publicar incidente. Se publicará cada incidente en redes sociales.

RF-5. Gestión de ubicación. Se manejará el uso de la ubicación GPS y de la compartición de esta con los contactos favoritos. Además se podrá establecer un PIN para poder rastrear el smartphone a través de la ubicación.

RF-5.1. Activar ubicación. Utilizará la ubicación GPS del dispositivo.

RF-5.2. Desactivar ubicación. Dejará de utilizar la ubicación GPS del dispositivo.

RF-5.3. Compartir ubicación. Permitirá ver la ubicación actual a un contacto favorito.

RF-5.4. Dejar de compartir ubicación. Cesará el permiso para ver la ubicación actual a los contactos favoritos.

RF-5.5. Establecer PIN secreto. Se guarda un código PIN para que el usuario pueda hacer uso del rastreo de su smartphone.

RF-5.6. Rastrear smartphone. Permitirá ver la ubicación actual del smartphone.

RF-5.6.1. Rastreo web. Se realizará el rastreo desde la versión web.

RF-5.6.2. Rastreo desde contacto favorito. Se realizará el rastreo desde el dispositivo de un contacto favorito.

RF-6. Gestión del botón del pánico. Se establecerá qué acción se realiza si se pulsa el botón del pánico y se definirán las acciones a realizar si el usuario no confirma de manera activa que está bien.

RF-6.1. Establecer acción de pánico. Se determinará qué acción realizará el sistema si se hace uso del botón del pánico.

RF-6.2. Activar botón de pánico. Se requerirá al usuario confirmar si se encuentra bien o en caso contrario hará uso de la acción de pánico preestablecida.

RF-6.2.1. Mandar ubicación. Se enviará la ubicación actual a los contactos favoritos.

RF-6.2.2. Llamar. Se llamará, según el orden establecido por el usuario, al primer contacto favorito en la lista.

RF-6.3. Confirmar buen estado. Se desactivará el botón del pánico al comprobar que el usuario se encuentra bien.

RF-7. Análisis de incidencias. El sistema asociará de forma automática un nivel de gravedad a cada incidente y se realizarán análisis de estos.

RF-7.1. Establecer nivel de gravedad. Se decidirá qué nivel de gravedad presenta cada incidente concreto.

RF-7.2. Analizar incidentes. Se realizarán análisis en base a distintos factores sobre los tipos de incidentes que ocurren y se obtendrá un resumen del factor de peligrosidad de cada zona.

3.2.2. Requisitos de Información

Información que es necesario almacenar en el sistema, clasificada con códigos y relacionada con los requisitos funcionales que hacen uso de la información.

RI-1. Administrador

Contenido: email, nombre, apellidos, contraseña y fecha de alta.

Requisitos asociados: RF-1

RI-2. Usuario

Contenido: email, nombre, apellidos, contraseña, teléfono, teléfono fijo, fecha de nacimiento, D.N.I., fecha de alta, ubicación actual y contactos favoritos.

Requisitos asociados: RF-2, RF-5.1

RI-3. Contacto favorito

Contenido: usuario

Requisitos asociados: RF-3, RF-1.11

RI-4. Incidente

Contenido: tipo, subtipo, lugar, fecha, hora, descripción, agravantes y nivel de gravedad

Requisitos asociados: RF-4.1 a RF-4.4, RF-1.7 a RF-1.10, RF-7.1, RF-7.2

RI-5. Zona de interés

Contenido: lugar, radio

Requisitos asociados: RF-4.6, RF-4.7, RF-1.12, RF-1.13

RI-6. Ubicación

Contenido: lugar

Requisitos asociados: RF-5.1 a RF-5.4

3.2.3. Restricciones Semánticas

Restricciones aplicadas a los requisitos funcionales y a los requisitos de información para su correcto uso y almacenamiento.

RS-1. Alta administrador. En el formulario de alta de administrador debe chequearse previamente para que sea válida la función:

RS-1.1. Email: debe tener un formato válido

RS-1.2. Contraseña: alfanumérica de más de 8 caracteres

RS-2. Alta usuario. En el formulario de alta de usuario debe chequearse previamente para que sea válida la función:

RS-2.1. Email: debe tener un formato válido

RS-2.2. Contraseña: alfanumérica de más de 8 caracteres

RS-2.3. Teléfono: debe ser válido y existir.

RS-2.4. D.N.I.: deber tener 8 números y una letra, la cual debe corresponderse con los números.

RS-2.5. Fecha de nacimiento: debe ser anterior a 12 años y máximo 110 respecto a la fecha actual.

RS-3. Contactos favoritos. Para gestionar contactos favoritos habrá ciertas restricciones:

RS-3.1. Añadir contacto favorito: debe ser un usuario de la aplicación.

RS-3.2. Ordenar contactos favoritos: debe haber más de un contacto favorito.

RS-3.3. Eliminar/rechazar contacto favorito: si se elimina a un contacto favorito o se rechaza serlo más de las veces estipuladas por el administrador se inhabilita la opción de añadir a dicho contacto. Si más de un número de usuarios –también determinado por el administrador– rechazan ser contacto favorito, se inhabilita la opción de añadir uno nuevo.

RS-4. Incidentes. Para añadir un incidente habrá ciertas restricciones, al igual que para ser notificado de uno nuevo.

RS-4.1. Tipo incidente: el tipo y subtipo pertenecen a categorías preestablecidas.

RS-4.2. Lugar incidente: al introducir una dirección se mostrará una sugerencia de dirección existente para confirmarla; si no existe se indicará directamente sobre el mapa.

RS-4.3. Fecha-hora incidente: debe ser anterior al momento actual.

RS-4.4. Notificar incidente: para notificar a un usuario debe producirse un incidente en una de sus zonas de interés o encontrarse con el GPS activo en la zona donde se ha producido.

RS-5. Zona de interés. Habrá un número máximo de zonas de interés, así como un radio máximo para delimitar una zona, ambos parámetros establecidos por el administrador del sistema.

RS-6. Caducidad de incidente. Caducará cuando no se produzca uno nuevo en un radio cercano y pasado cierto tiempo, ambos parámetros determinados por el administrador del sistema.

RS-7. Rastreo. Habrá ciertas restricciones para poder rastrear un dispositivo móvil.

RS-7.1. Web: habrá que iniciar sesión previamente.

RS-7.2. Contacto favorito: será necesario introducir correctamente el PIN secreto para poder rastrear la ubicación desde el dispositivo de un contacto favorito.

RS-8. Botón del pánico. Debe existir al menos un contacto favorito para que se habilite dicha funcionalidad.

3.2.4. Requisitos No Funcionales

Describen aspectos del sistema que no están directamente relacionados con el comportamiento funcional del mismo. Utilizan el modelo FURPS+ [14], proveyendo las siguientes categorías, cada una con requisitos no funcionales identificados con códigos que facilitan su seña y uso.

Usabilidad

RNF-1. Ayuda. Se mostrarán ayudas en todos los formularios para introducir todos los datos correctamente.

RNF-2. Alertas. Se mostrarán alertas cuando el usuario vaya a realizar alguna acción delicada tal como eliminar su cuenta o un contacto favorito.

Fiabilidad

RNF-3. Seguridad de la información. Se realizarán copias de seguridad para prever caídas del sistema o pérdidas de datos.

RNF-4. Autenticación. Cada nuevo usuario registrado deberá confirmar su alta en el sistema mediante verificación por correo electrónico.

RNF-5. Codificación. Se encriptará toda la información sensible relativa a los usuarios.

RNF-6. Administración. Sólo los administradores pueden acceder a funciones de gestión del sistema.

Rendimiento

RNF-7. Concurrencia. Pueden trabajar varios administradores de manera simultánea y hacer uso del sistema varios usuarios al mismo tiempo.

RNF-8. Tiempo de respuesta. El tiempo de respuesta del sistema no es crítico, pero cualquiera de las acciones no tardará en realizarse más de 10 segundos.

RNF-9. Disponibilidad. El sistema se encontrará disponible 24 horas al día.

Soporte

RNF-10. Adaptabilidad. Todas las magnitudes se encontrarán en el sistema métrico decimal.

RNF-11. Mantenimiento. El sistema será adaptable a cambios en función de nuevas tecnologías o necesidad de cambios para introducir mejoras, corregir errores o mejorar el rendimiento.

RNF-12. Portabilidad. El sistema se podrá transferir con facilidad a otras plataformas web o nuevas versiones móviles.

Restricciones de implementación

RNF-13. La implementación se realizará con un patrón arquitectónico MVC, donde el modelo y el controlador en la versión web utilizarán el lenguaje PHP (siendo intermediario de SQL para el modelo) y HTML, CSS y JavaScript para la vista. Se hará uso de TypeScript para el desarrollo de la aplicación, empleando los SDK's Ionic y Android Studio para poder exportar el proyecto a ejecutables en smartphones. Para el análisis de incidentes el lenguaje de programación empleado será Python.

Restricciones legales

RNF-14. Los usuarios registrados deberán leer y aceptar previamente la política de privacidad del software.

RNF-15. Todos los datos personales estarán correctamente protegidos según las leyes *RGPD UE 679/2016* y *LOPDGD 3/2018*. Asimismo los datos se encontrarán almacenados en servidores ubicados en territorio de la Unión Europea.

3.3. Modelo de casos de uso

Mediante el modelo de casos de uso es posible delimitar el sistema, determinar el contexto de uso del mismo y describir el punto de vista de los usuarios que lo utilizan. Los elementos que componen este modelo son los actores –usuarios o sistema–, los casos de uso –requisitos que indican qué hará el sistema– y las relaciones entre todos ellos. Para la descripción y representación de los elementos mencionados se utilizan diagramas de casos de uso (UML) y plantillas estructuradas [15].

3.3.1. Diagramas de casos de uso

Los diagramas UML de casos de uso representan gráficamente la frontera del sistema junto con los elementos que forman parte del modelo [15]. En este caso se presentan los actores con el rol que desempeñan (usuario, administrador o sistema) y los casos de uso, que especifican la secuencia de acciones que realiza el sistema, el cual se ha dividido en cuatro subsistemas. Las relaciones de los casos de uso empleadas son principalmente las de asociación, pero hay otras menos comunes que también cabe explicar para comprender mejor su significado en el diagrama.

Relación	Definición	Notación
Asociación	Relación entre un actor y el caso de uso en el que participa	_____
Generalización	Relación entre un caso de uso general y otros más específicos que heredan y añaden características del caso de uso general	→
Inclusión	Relación entre casos de uso donde la inserción representa un comportamiento adicional dentro del caso de uso base, que debe ser obligatorio y sin el cual el caso de uso base no tiene sentido. Se utilizan de igual modo para representar casos de uso comunes, modularizando funcionalidades	<<Include>> ->
Extensión	Relación entre casos de uso que introducen comportamiento adicional sin que el caso de uso base sepa de los casos de uso de extensión, introduciendo funcionalidad extra, superflua para el caso de uso base, que tiene sentido por sí mismo	<<Extend>> ->

Tabla 7: Relaciones en diagramas de casos de uso

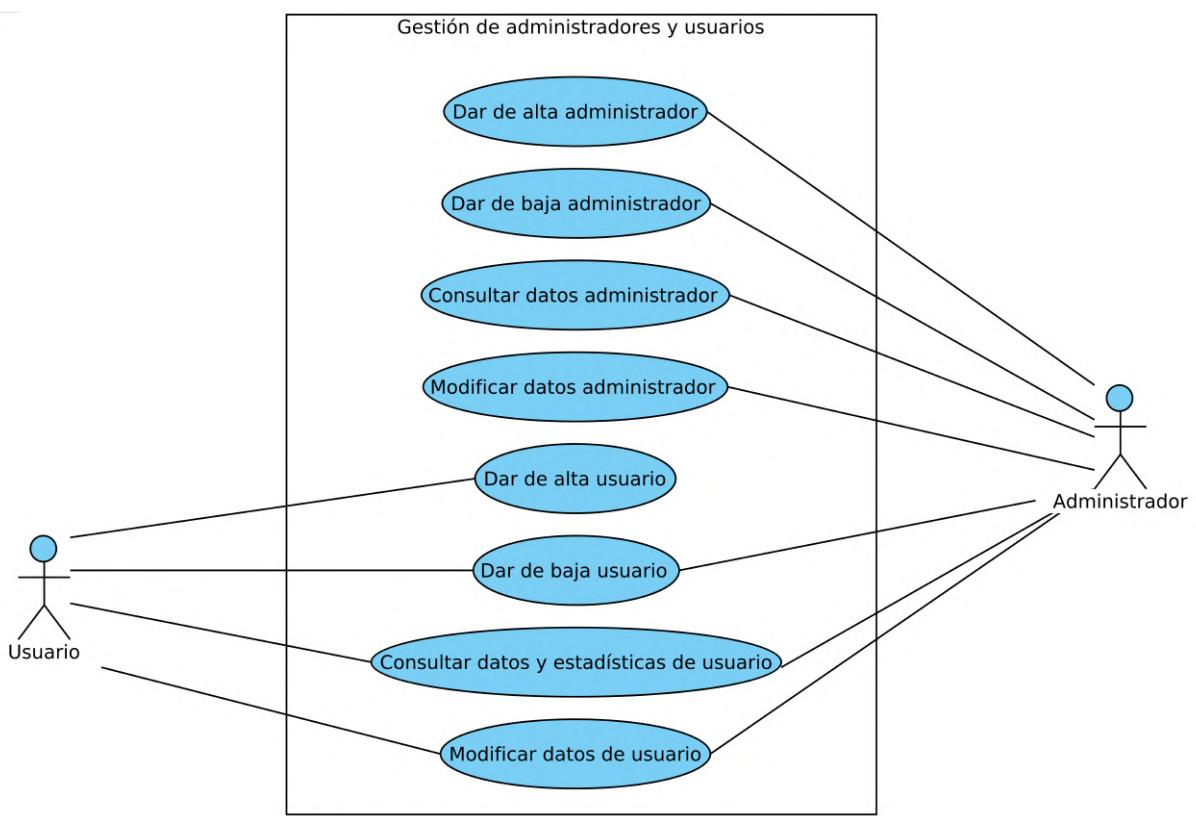


Figura 2: Diagrama de caso de uso - Gestión de administradores y usuarios

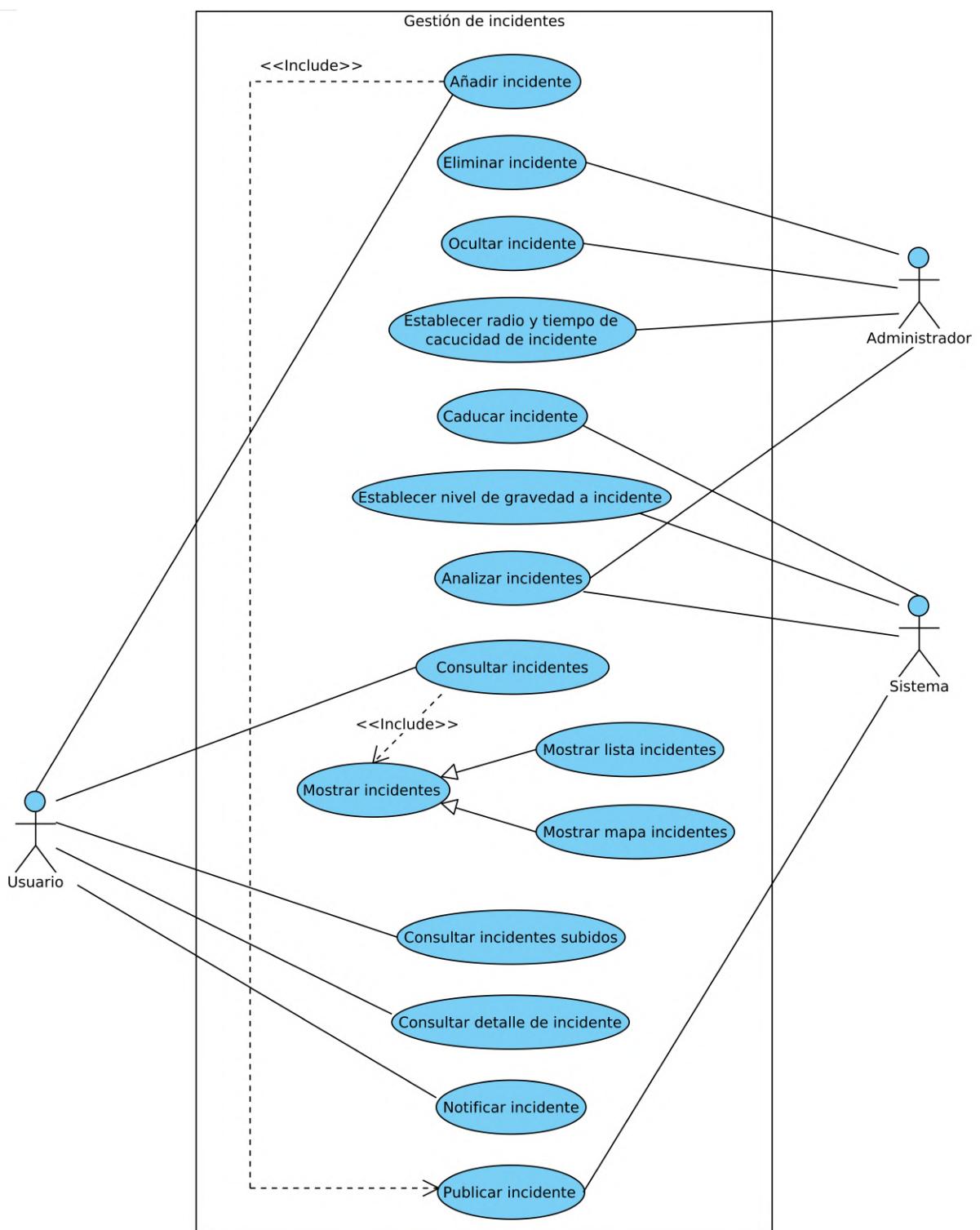


Figura 3: Diagrama de caso de uso - Gestión de incidentes

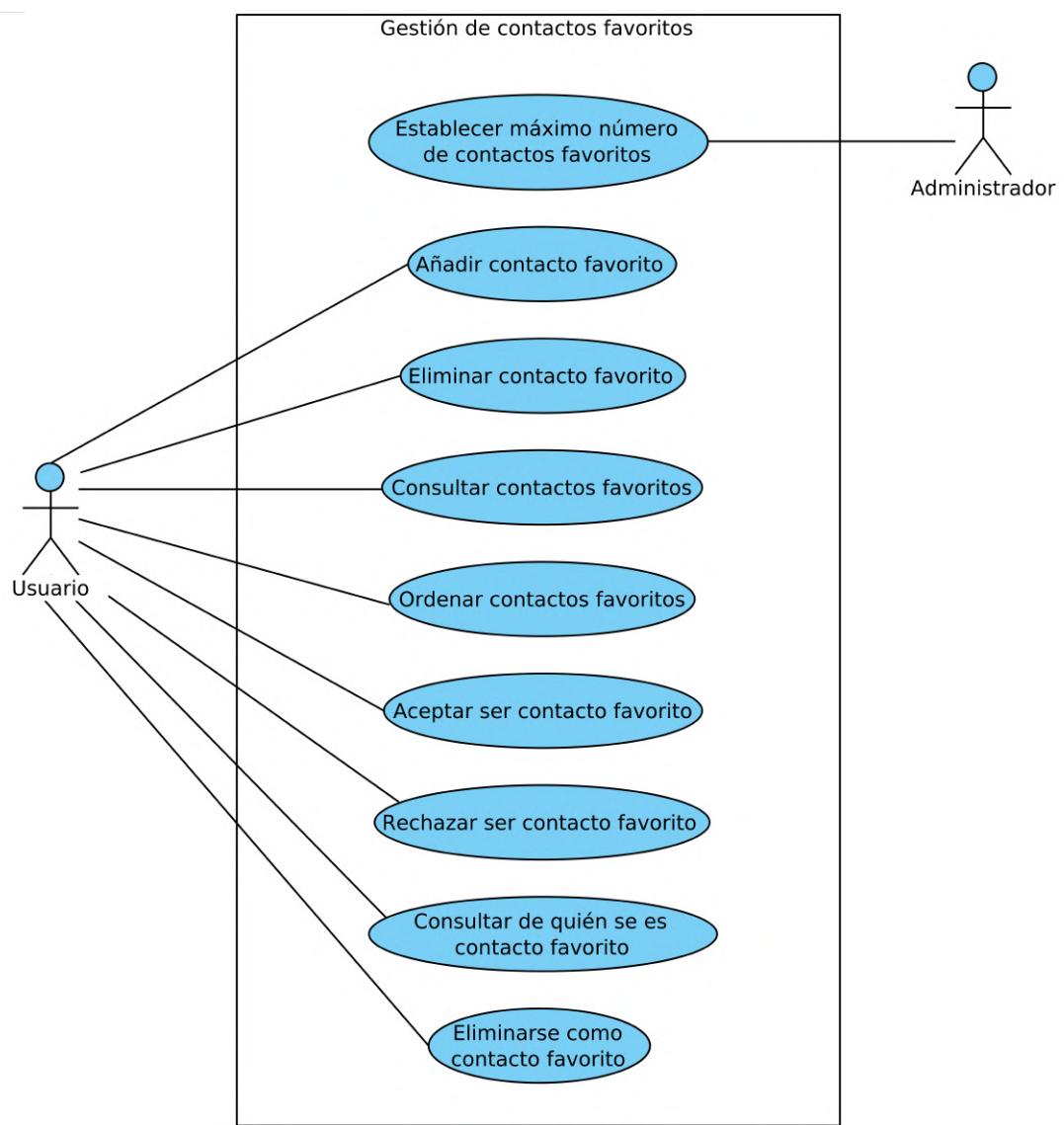


Figura 4: Diagrama de caso de uso - Gestión de contactos favoritos

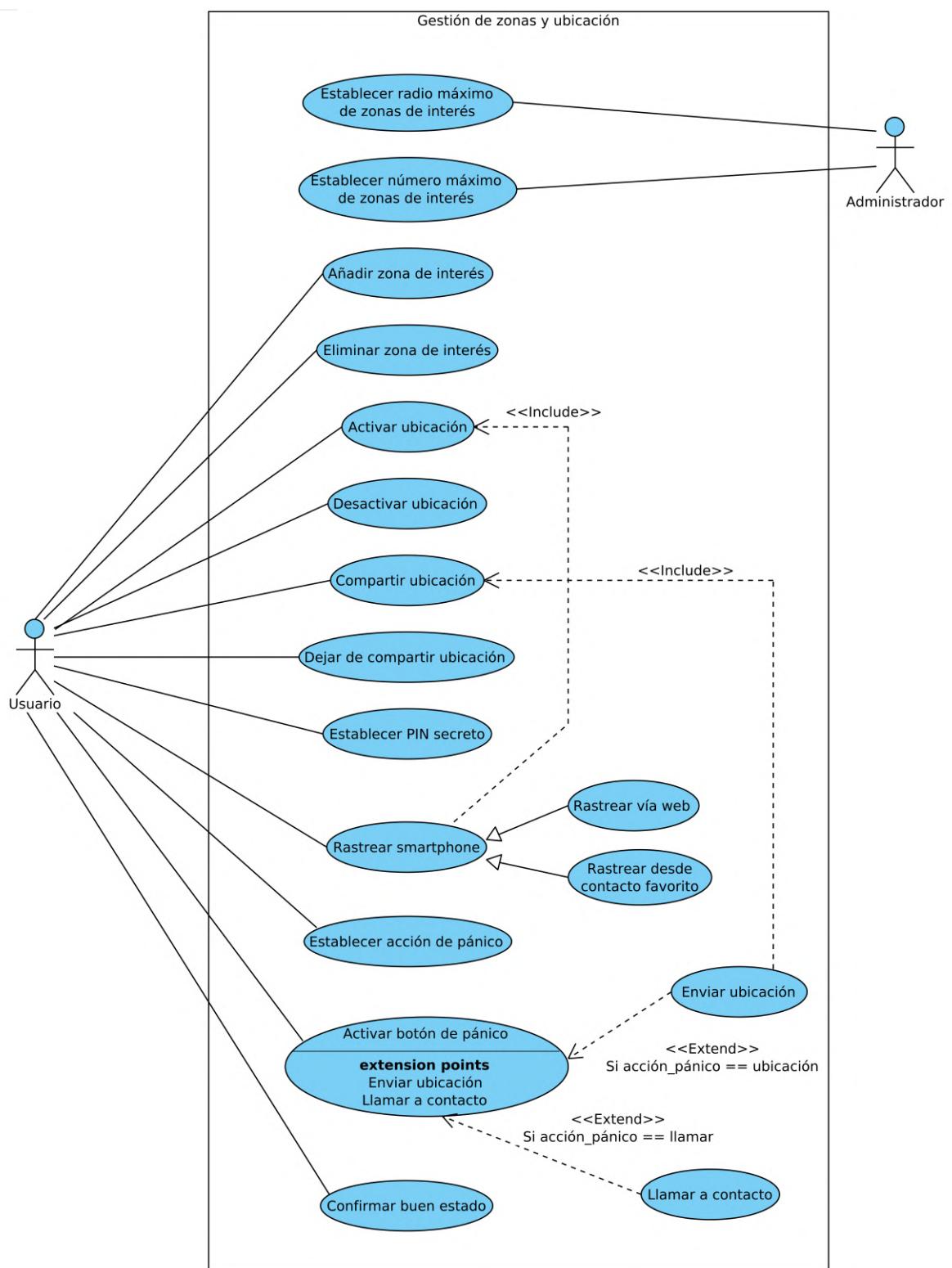


Figura 5: Diagrama de caso de uso - Gestión de zonas y ubicación

3.3.2. Descripción de los casos de uso (plantilla básica)

En las plantillas estructuradas se detallan los actores que interaccionan con el caso de uso, el tipo, los requisitos funcionales incluidos, las precondiciones –condiciones sobre el estado del sistema que tienen que cumplirse para que se pueda realizar– y poscondiciones –efectos que de forma inmediata tiene la realización del caso de uso– relativas a este y el propósito que cumple, además de un breve resumen que describe a alto nivel la secuencia de acciones realizadas.

En cuanto al tipo de caso de uso mencionado anteriormente, se describen a continuación cuáles pueden ser según su:

- Importancia
 - Primarios: Procedimientos comunes más importantes.
 - Secundarios: Procesos de error, poco comunes o que dan soporte para que los primarios puedan trabajar.
 - Opcionales: Puede que no se implementen.
- Procesamiento
 - Básicos: Descripción general del procesamiento.
 - Extendidos: Descripción de la secuencia de acción completa entre actores y sistema.
- Nivel de abstracción
 - Esencial: Expresado de forma abstracta, contiene poca tecnología y pocos detalles de diseño.
 - Real: Expresado en base al diseño actual, en el que aparecen relaciones con la interfaz de usuario.

Gestión de administradores y usuarios

Caso de uso	Dar de alta administrador	CU-1
Actores	Administrador	
Tipo	Primario, básico y esencial	
Referencias	RF-1.1	
Precondición	Debe existir previamente un administrador que pueda llevar a cabo esta función	
Poscondición	El nuevo administrador quedará registrado en el sistema	

Propósito

Dar de alta un nuevo administrador en el sistema

Resumen

El administrador que da el alta proporcionará los datos relativos al nuevo administrador

Caso de uso	Dar de baja administrador	CU-2
Actores	Administrador	
Tipo	Primario, básico y esencial	
Referencias	RF-1.2	
Precondición	Debe estar el administrador dado de alta	
Poscondición	El administrador y toda la información relativa a este serán eliminados del sistema	

Propósito

Dar de baja un administrador del sistema

Resumen

El administrador se dará de baja del sistema por sí mismo o por acción de otro administrador.

Caso de uso	Consultar datos de administrador	CU-3
Actores	Administrador	
Tipo	Primario, básico y esencial	
Referencias	RF-1.3	
Precondición	El administrador debe estar registrado en el sistema	
Poscondición	El administrador podrá consultar sus datos	

Propósito
Consultar los datos relativos a un administrador

Resumen
Se proporciona un mecanismo para que un administrador pueda acceder a sus datos

Caso de uso	Modificar datos de administrador	CU-4
Actores	Administrador	
Tipo	Primario, básico y esencial	
Referencias	RF-1.4	
Precondición	El administrador debe estar registrado en el sistema	
Poscondición	Los datos habrán sido cambiados	

Propósito
Modificar los datos relativos a un administrador

Resumen
Se proporciona un mecanismo para que un administrador pueda modificar sus datos

Caso de uso	Dar de alta usuario	CU-5
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-2.1	
Precondición		
Poscondición	El nuevo usuario quedará registrado en el sistema	

Propósito
Dar de alta un nuevo usuario en el sistema

Resumen
El usuario que se da de alta proporcionará los datos relativos al él mismo

Caso de uso	Dar de baja usuario	CU-6
Actores	Usuario y Administrador	
Tipo	Primario, básico y esencial	
Referencias	RF-1.5 y RF-2.2	
Precondición	El usuario debe estar dado de alta	
Poscondición	El usuario y toda la información relativa a este serán eliminados del sistema	

Propósito

Dar de baja un usuario del sistema

Resumen

El usuario se dará de baja del sistema por sí mismo o por acción de un administrador

Caso de uso	Consultar datos y estadísticas de usuario	CU-7
Actores	Usuario y Administrador	
Tipo	Primario, básico y esencial	
Referencias	RF-1.6 y RF-2.3	
Precondición	El usuario debe estar registrado en el sistema	
Poscondición	Los datos y estadísticas de usuario estarán accesibles para su consulta	

Propósito

Consultar los datos y estadísticas relativas a un usuario

Resumen

Se proporciona un mecanismo para que un administrador o un usuario puedan acceder a los datos de este último y a sus estadísticas generadas en el sistema.

Caso de uso	Modificar datos de usuario	CU-8
Actores	Usuario y Administrador	
Tipo	Primario, básico y esencial	
Referencias	RF-2.4	
Precondición	El usuario debe estar registrado en el sistema	
Poscondición	Los datos habrán sido cambiados	

Propósito
Modificar los datos relativos a un usuario

Resumen
Se proporciona un mecanismo para un usuario pueda modificar sus datos

Gestión de incidentes

Caso de uso	Añadir incidente	CU-9
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-4.1	
Precondición	El usuario debe estar registrado en el sistema	
Poscondición	El incidente estará registrado en el sistema	

Propósito
Añadir un nuevo incidente al sistema

Resumen
El usuario requiere dar de alta un nuevo incidente, rellena los datos relativos al mismo y una vez validados se almacena en el sistema

Caso de uso	Eliminar incidente	CU-10
Actores	Administrador	
Tipo	Secundario, básico y esencial	
Referencias	RF-1.7	
Precondición	El incidente debe estar dado de alta en el sistema	
Poscondición	El incidente y toda la información relativa a este habrán sido eliminados del sistema	

Propósito
Eliminar un incidente del sistema

Resumen
El administrador suprime un incidente almacenado en el sistema.

Caso de uso	Ocultar incidente	CU-11
Actores	Administrador	
Tipo	Primario, básico y esencial	
Referencias	RF-1.8	
Precondición	El incidente debe estar dado de alta en el sistema	
Poscondición	El incidente y no será visible para ningún usuario	

Propósito

Hacer que un incidente o tipo de incidente no sea visible por los usuarios del sistema

Resumen

El administrador oculta un incidente, seleccionando uno concreto o un tipo de estos

Caso de uso	Establecer radio y tiempo de caducidad de incidentes	CU-12
Actores	Administrador	
Tipo	Primario, básico y esencial	
Referencias	RF-1.9	
Precondición		
Poscondición	La caducidad se realizará de manera automática	

Propósito

Hacer que los incidentes caduquen pasado el tiempo establecido y si no se ha producido ningún otro en el radio determinado

Resumen

El administrador establece el radio y el tiempo de caducidad de los incidentes

Caso de uso	Caducar incidente	CU-13
Actores	Sistema	
Tipo	Primario, básico y esencial	
Referencias	RF-1.10	
Precondición	El incidente debe cumplir las restricciones de radio alrededor del que no se han podido producir incidentes dentro del tiempo establecido	
Poscondición	El incidente será caducado y estará oculto a los usuarios	

Propósito
Hacer que los incidentes no sean visibles si, pasado un tiempo, no se ha producido otros incidentes en el radio establecido por el administrador

Resumen
El sistema caduca de automáticamente los incidentes que cumplan las restricciones

Caso de uso	Establecer nivel de gravedad a incidente	CU-14
Actores	Sistema	
Tipo	Primario, básico y esencial	
Referencias	RF-7.1	
Precondición	Tener definido el tipo y subtipo de delito, descripción detallada del mismo y lugar y fecha donde ocurrió	
Poscondición	El incidente tendrá asociado un nivel de gravedad	

Propósito
Establecer un nivel de gravedad a cada incidente en base a lo ocurrido y al contexto donde sucede

Resumen
El sistema analiza los detalles del incidente y, en base a estos, a otros incidentes ocurridos y a factores relacionados, le asigna un nivel de gravedad

Caso de uso	Analizar incidentes CU-15
Actores	Sistema y Administrador
Tipo	Primario, básico y esencial
Referencias	RF-7.2
Precondición	El incidente debe tener definidos todos sus atributos
Poscondición	Se obtendrá un resultado interpretable sobre la clasificación de incidentes acaecidos

Propósito

Examinar qué factores influyen en la ocasión de incidencias

Resumen

El sistema evalúa los detalles de los incidentes y establece criterios de agrupamiento de incidentes en función de las características comunes existentes entre ellos

Caso de uso	Consultar incidentes CU-16
Actores	Usuario
Tipo	Primario, básico y esencial
Referencias	RF-4.2
Precondición	El usuario debe estar registrado en el sistema y elegir un criterio
Poscondición	Se obtendrá un resultado en base a la consulta realizada

Propósito

Permitir al usuario obtener incidentes en base a un criterio elegido por este

Resumen

El usuario selecciona el criterio sobre el que quiere establecer la consulta y el sistema le devuelve el resultado relativo a dicha consulta

Caso de uso	Mostrar incidentes	CU-17
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-4.3	
Precondición	Haber realizado una consulta de incidentes	
Poscondición	Los incidentes estarán disponibles para el usuario	

Propósito
Mostrar los incidentes producidos

Resumen
Tras haber realizado una consulta el sistema mostrará los incidentes que se correspondan con el criterio de búsqueda

Caso de uso	Mostrar lista de incidentes	CU-18
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-4.3.1	
Precondición	Haber realizado una consulta de incidentes	
Poscondición	Los incidentes estarán disponibles para el usuario	

Propósito
Mostrar una lista de incidentes producidos

Resumen
Tras haber realizado una consulta el sistema mostrará en forma de lista los incidentes que se correspondan con el criterio de búsqueda

Caso de uso	Mostrar mapa de incidentes	CU-19
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-4.3.2	
Precondición	Haber realizado una consulta de incidentes	
Poscondición	Los incidentes estarán disponibles para el usuario	

Propósito
Mostrar los incidentes producidos en un mapa

Resumen
Tras haber realizado una consulta el sistema mostrará en un mapa los incidentes que se correspondan con el criterio de búsqueda

Caso de uso	Consultar incidentes subidos	CU-20
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-4.4	
Precondición	El usuario debe estar registrado en el sistema	
Poscondición	Se obtendrán los incidentes subidos por el usuario	

Propósito
Permitir al usuario obtener incidentes subidos por él mismo

Resumen
El usuario consulta los incidentes que ha dado de alta en el sistema y este le devuelve el resultado en caso de haberlos

Caso de uso	Consultar detalle de incidente	CU-21
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-4.5	
Precondición	El usuario debe estar registrado en el sistema y debe existir el incidente	
Poscondición	Se obtendrán los detalles del incidente seleccionado	

Propósito
Permitir al usuario obtener las características de un incidente concreto

Resumen
El usuario selecciona un incidente y el sistema le devuelve los datos relativos a este

Caso de uso	Notificar incidente CU-22
Actores	Usuario
Tipo	Primario, básico y esencial
Referencias	RF-4.8
Precondición	El usuario emisor y los receptores deben estar registrados en el sistema
Poscondición	Los usuarios receptores recibirán una alerta sobre un incidente

Propósito

Comunicar a otros usuarios del sistema sobre un incidente concreto

Resumen

El usuario selecciona un incidente y lo envía como notificación a otros usuarios

Caso de uso	Publicar incidente CU-23
Actores	Sistema
Tipo	Primario, básico y esencial
Referencias	RF-4.9
Precondición	El incidente debe estar dado de alta en el sistema
Poscondición	El incidente será publicado en redes sociales

Propósito

Publicar en redes sociales un incidente con sus detalles

Resumen

El sistema, tras dar de alta un incidente subido por un usuario, lo publica con los datos relativos al mismo en la redes sociales asociadas al sistema

Gestión de contactos favoritos

Caso de uso	Establecer máximo número de contactos favoritos	CU-24
Actores	Administrador	
Tipo	Secundario, básico y esencial	
Referencias	RF-1.11	
Precondición		
Poscondición	Ningún usuario podrá tener más contactos favoritos del número establecido	

Propósito

Establecer para los usuarios del sistema un número de contactos favoritos máximo

Resumen

El administrador introduce el número correspondiente al máximo de contactos favoritos que se pueden tener y el sistema lo guarda en un fichero de parámetros

Caso de uso	Añadir contacto favorito	CU-25
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-3.1	
Precondición	El usuario que añade y el añadido deben estar registrados y no debe haberse sobrepasado el número máximo de contactos favoritos	
Poscondición	El contacto receptor de la petición podrá aceptar o declinar ser contacto favorito	

Propósito

Añadir a otro usuario del sistema como contacto de preferencia para casos de emergencia

Resumen

El usuario elige a otro usuario del sistema para añadirlo como contacto favorito y, si no se ha superado el límite máximo de contactos, se envía una petición al usuario destinatario

Caso de uso	Eliminar contacto favorito	CU-26
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-3.2	
Precondición	El usuario eliminado debe ser contacto favorito	
Poscondición	El usuario ya no será contacto favorito	

Propósito
Dejar de tener como contacto favorito a alguien que lo era

Resumen
El usuario elige a un contacto favorito y lo excluye como tal, desapareciendo la relación de contactos entre ambos en el sistema

Caso de uso	Consultar contactos favoritos	CU-27
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-3.3	
Precondición	El usuario debe estar dado de alta en el sistema	
Poscondición	Los contactos favoritos del usuario serán mostrados	

Propósito
Examinar los usuarios que se tienen como contactos favoritos

Resumen
Se proporciona un mecanismo para que los usuarios pueda consultar quienes son los usuarios del sistema que tiene almacenados como contactos favoritos

Caso de uso	Ordenar contactos favoritos	CU-28
Actores	Usuario	
Tipo	Secundario, básico y esencial	
Referencias	RF-3.4	
Precondición	Deben existir al menos dos contactos favoritos	
Poscondición	Los contactos favoritos estarán ordenados	

Propósito

Establecer un orden propio sobre los contactos favoritos de un usuario

Resumen

El usuario selecciona cada contacto favorito y le asigna un orden respecto al resto de contactos favoritos

Caso de uso	Aceptar ser contacto favorito	CU-29
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-3.5	
Precondición	Ser añadido como contacto favorito de otro usuario	
Poscondición	Ser contacto favorito del usuario que lo solicita	

Propósito

Aceptar que un usuario tenga a otro como contacto favorito

Resumen

El usuario accede a ser contacto favorito y el sistema lo añade como tal en la lista del usuario que solicitó la acción

Caso de uso	Rechazar ser contacto favorito	CU-30
Actores	Usuario	
Tipo	Opcional, básico y esencial	
Referencias	RF-3.5	
Precondición	Ser añadido como contacto favorito de otro usuario	
Poscondición		

Propósito

Rechazar que otro usuario añada a un usuario como contacto favorito

Resumen

El usuario rehúsa a ser contacto favorito

Caso de uso	Consultar de quién se es contacto favorito	CU-31
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-3.6	
Precondición	Debe estar dado de alta en el sistema	
Poscondición	Los usuarios de quienes se es contacto favorito serán mostrados	

Propósito

Examinar los usuarios de los que se es contacto favorito como consecuencia de haber aceptado serlo

Resumen

Mecanismo que proporciona al usuario un método para consultar al sistema quienes son aquellos usuarios que le tienen almacenado como contacto favorito

Caso de uso	Eliminarse como contacto favorito	CU-32
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-3.7	
Precondición	Debe ser contacto favorito de algún otro usuario	
Poscondición	No será contacto favorito del usuario seleccionado	

Propósito

Dejar de ser contacto favorito de otro usuario

Resumen

Mecanismo que proporciona a un usuario la posibilidad de seleccionar otro usuario del que se es contacto favorito para dejar de serlo, eliminando la relación entre ambos en el sistema

Gestión de zonas y ubicación

Caso de uso	Establecer radio máximo de zonas de interés	CU-33
Actores	Administrador	
Tipo	Secundario, básico y esencial	
Referencias	RF-1.12	
Precondición		
Poscondición	Ningún usuario podrá tener una zona de interés cuyo radio abarque más metros del número establecido	

Propósito

Establecer para los usuarios del sistema un radio de zona de interés máximo

Resumen

El administrador introduce la longitud (en metros) correspondiente al radio máximo que se puede establecer para que abarque una zona de interés y el sistema lo guarda en un fichero de parámetros

Caso de uso	Establecer número máximo de zonas de interés	CU-34
Actores	Administrador	
Tipo	Secundario, básico y esencial	
Referencias	RF-1.13	
Precondición		
Poscondición	Ningún usuario podrá tener más zonas de interés del número establecido	

Propósito

Establecer para los usuarios del sistema un número de zonas de interés máximo

Resumen

El administrador introduce el número máximo que se puede tener de zonas de interés y el sistema lo guarda en un fichero de parámetros

Caso de uso	Añadir zona de interés	CU-35
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-4.6	
Precondición	El centro de la zona debe tener unas coordenadas válidas, y el radio y número de zonas de interés del usuario no deben sobrepasar el límite máximo establecido	
Poscondición	La zona de interés será añadida al usuario	

Propósito
Incorporar una nueva zona de interés

Resumen
El usuario introduce un lugar, determinado por una latitud y una longitud, y un radio (en metros) y, si los parámetros son correctos y no se ha superado el número máximo de zonas de interés, se añade la zona a la lista de zonas de interés del usuario

Caso de uso	Eliminar zona de interés	CU-36
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-4.7	
Precondición	Debe existir la zona de interés en la lista de zonas de interés del usuario	
Poscondición	La zona de interés será añadida al usuario	

Propósito
Dejar de tener una zona de interés destacada como tal

Resumen
El usuario elige una de sus zonas de interés y la elimina como tal, desapareciendo la relación entre usuario y zona del sistema

Caso de uso	Activar ubicación	CU-37
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-5.1	
Precondición	El sistema debe tener permisos de ubicación	
Poscondición	El sistema conocerá la ubicación actual del usuario	

Propósito
Utilizar la ubicación GPS del dispositivo del usuario

Resumen
El usuario permite acceder a su ubicación y el sistema la almacena, actualizando periódicamente sus valores de latitud/longitud

Caso de uso	Desactivar ubicación	CU-38
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-5.2	
Precondición	La ubicación debe estar activa	
Poscondición	El sistema no conocerá la ubicación actual del usuario	

Propósito
Dejar de utilizar la ubicación GPS del dispositivo del usuario

Resumen
El usuario cesa el permiso de acceso a su ubicación y el sistema deja como nulos los parámetros que almacenaban el valor de la posición actual

Caso de uso	Compartir ubicación	CU-39
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-5.3	
Precondición	La ubicación debe estar activa y el usuario ha de tener al menos un contacto favorito	
Poscondición	El contacto favorito seleccionado podrá conocer la ubicación actual del usuario	

Propósito
Compartir la ubicación GPS del dispositivo del usuario con un contacto favorito

Resumen
El usuario elige un contacto favorito y permite que este conozca su ubicación actual

Caso de uso	Dejar de compartir ubicación	CU-40
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-5.4	
Precondición	Debe estar siendo compartida con algún contacto favorito	
Poscondición	El contacto favorito con quien se compartía ubicación dejará de conocer la ubicación actual del usuario	

Propósito
Retirar el acceso a la ubicación GPS del dispositivo del usuario a un contacto favorito

Resumen
El usuario selecciona el contacto favorito al que compartía su ubicación y cesa dicha concesión

Caso de uso	Establecer PIN secreto	CU-41
Actores	Usuario	
Tipo	Secundario, básico y esencial	
Referencias	RF-5.5	
Precondición	El sistema debe tener permisos de ubicación	
Poscondición	El usuario podrá hacer uso del rastreo de su dispositivo desde el terminal de un contacto favorito	

Propósito
Acceder a la funcionalidad de rastreo del dispositivo

Resumen
El usuario introduce un PIN y este queda almacenado en el sistema, de modo que queda activa la opción de rastreo

Caso de uso	Rastrear smartphone	CU-42
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-5.6	
Precondición		
Poscondición	El usuario tendrá activa y disponible la ubicación de su dispositivo	

Propósito
Conocer la ubicación del smartphone de manera remota

Resumen
Se proporciona un mecanismo para activar la ubicación del dispositivo y acceder a ella

Caso de uso	Rastrear vía web	CU-43
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-5.6.1	
Precondición		
Poscondición	El usuario tendrá activa y disponible la ubicación de su dispositivo iniciando sesión con su cuenta desde un navegador web	

Propósito
Conocer la ubicación del smartphone y visualizarla en un navegador web

Resumen
Se proporciona un mecanismo para activar la ubicación del dispositivo y acceder a ella desde la propia cuenta de usuario en un navegador web

Caso de uso	Rastrear desde contacto favorito	CU-44
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-5.6.2	
Precondición	Debe haberse establecido el PIN secreto	
Poscondición	El usuario tendrá activa y disponible la ubicación de su dispositivo desde el terminal del contacto favorito	

Propósito
Conocer la ubicación del smartphone y visualizarla desde el dispositivo de un contacto favorito

Resumen
Se proporciona un mecanismo para activar la ubicación del dispositivo y acceder a ella desde el dispositivo de un contacto favorito introduciendo el PIN secreto

Caso de uso	Establecer acción de pánico	CU-45
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-6.1	
Precondición	Tener activos los permisos de ubicación y/o llamada	
Poscondición	El usuario tendrá disponible el botón del pánico y establecida la acción a llevar a cabo	

Propósito

Poder hacer uso del botón del pánico mediante una acción concreta

Resumen

El usuario selecciona la acción que se llevará a cabo en caso de hacer uso del botón del pánico, activándose así la opción de uso de dicha funcionalidad

Caso de uso	Activar botón de pánico	CU-46
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-6.2	
Precondición	Tener establecida la acción de pánico	
Poscondición	El sistema alertará a los contactos favoritos que el usuario ha activado el botón del pánico	

Propósito

Avisar a los contactos favoritos que el usuario está en una posible situación de peligro

Resumen

El sistema proporciona un mecanismo para que el usuario alerte a otros usuarios marcados como contactos favoritos enviando una alerta

Caso de uso	Enviar ubicación	CU-47
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-6.2.1	
Precondición	Tener fijada como acción de pánico el envío de ubicación	
Poscondición	El sistema alertará a los contactos favoritos que el usuario ha activado el botón del pánico y enviará su ubicación actual	

Propósito

Avisar a los contactos favoritos que el usuario está en una posible situación de peligro

Resumen

El sistema, tras alertar a los contactos favoritos, envía la ubicación actual del usuario

Caso de uso	Llamar a contacto	CU-48
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-6.2.2	
Precondición	Tener establecida como acción de pánico la llamada	
Poscondición	El sistema alertará a los contactos favoritos que el usuario ha activado el botón del pánico y llamará a uno de ellos	

Propósito

Avisar a los contactos favoritos que el usuario está en una posible situación de peligro

Resumen

El sistema, tras alertar a los contactos favoritos, llama, según el orden establecido por el usuario de sus contactos favoritos, a cada uno de ellos consecutivamente hasta que uno responda a la llamada

Caso de uso	Confirmar buen estado	CU-49
Actores	Usuario	
Tipo	Primario, básico y esencial	
Referencias	RF-6.3	
Precondición	Haber activado el botón del pánico	
Poscondición	Los contactos favoritos alertados habrán recibido un aviso confirmando el buen estado del usuario y cesarán las acciones llevadas a cabo	

Propósito
Avisar a los contactos favoritos que el usuario se encuentra bien

Resumen
El sistema deja de compartir la ubicación en caso de estar activa e informa a los contactos favoritos alertados que el usuario se encuentra en buen estado

3.4. Diagramas de secuencia

Los diagramas de secuencia son un tipo de diagramas de interacción UML que muestran de manera sencilla la colaboración entre los distintos objetos con funcionalidades en el sistema, establecida a través del paso de mensajes. Tratan de explicar el comportamiento del sistema mediante una descripción de qué hace, sin ahondar en el cómo.

En este tipo de diagramas se puede apreciar la secuencialidad de los mensajes, pues se muestran ordenados temporalmente y de forma visual en la dimensión vertical. Por otro lado, la dimensión horizontal representa a los distintos actores y objetos participantes, sin importar en este caso el orden de aparición de estos.

Los diagramas mostrados a continuación son el resultado de representar, según cada caso, uno o varios casos de uso.

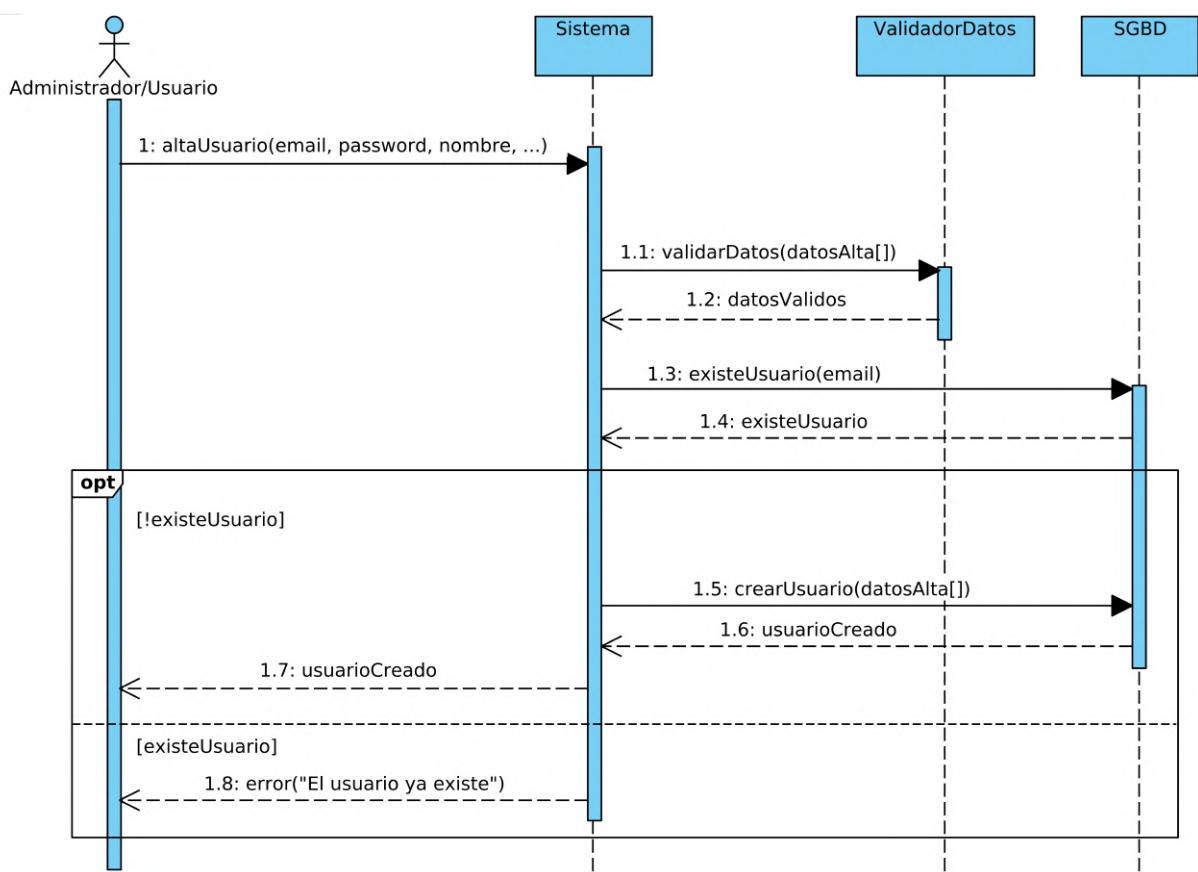


Figura 6: DS: Alta de un usuario o administrador

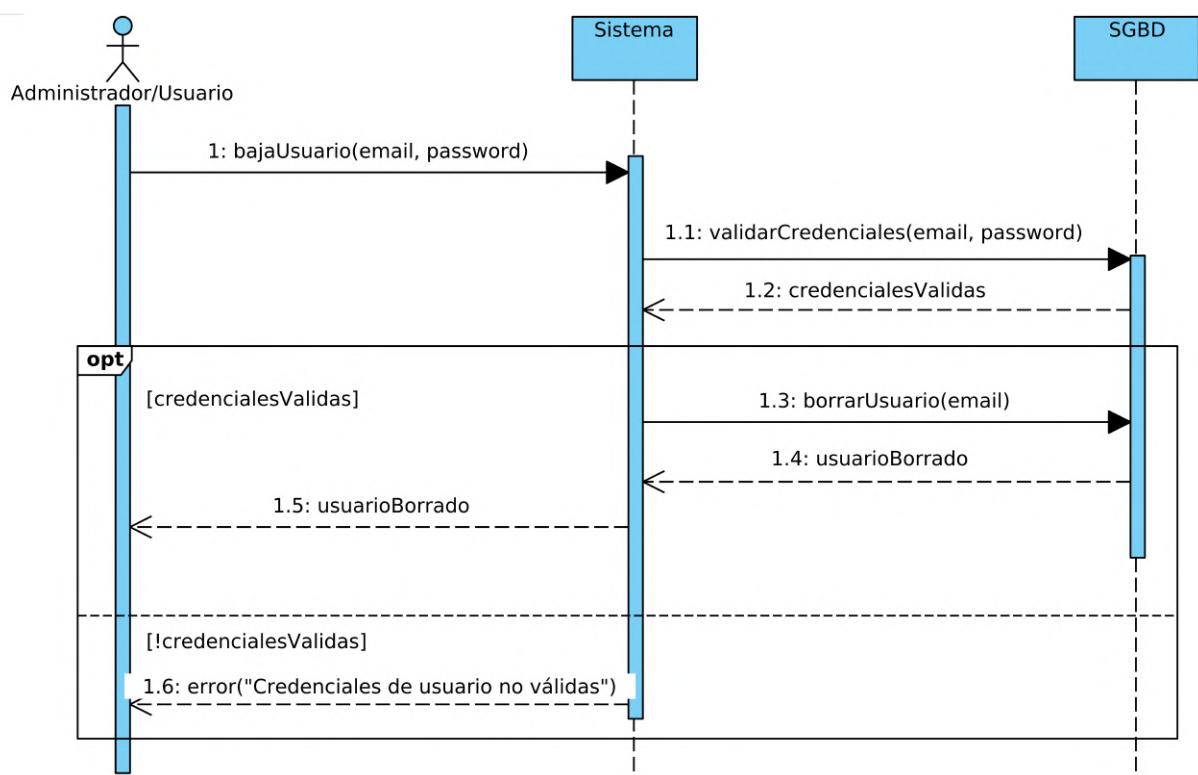


Figura 7: DS: Baja de un usuario o administrador

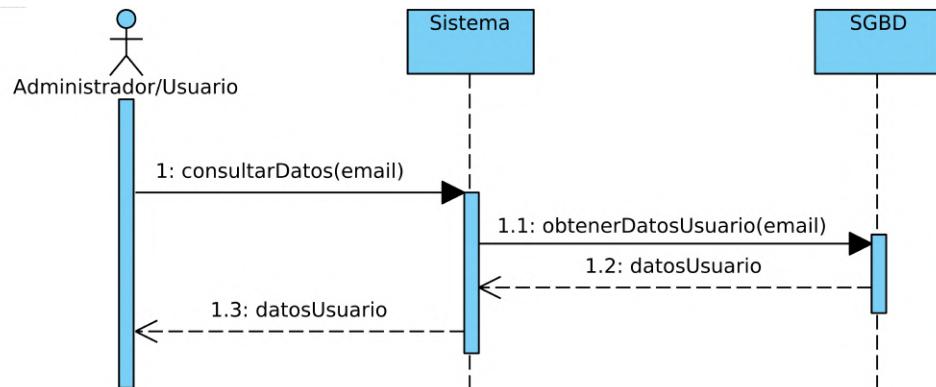


Figura 8: DS: Consultar datos de un usuario o administrador

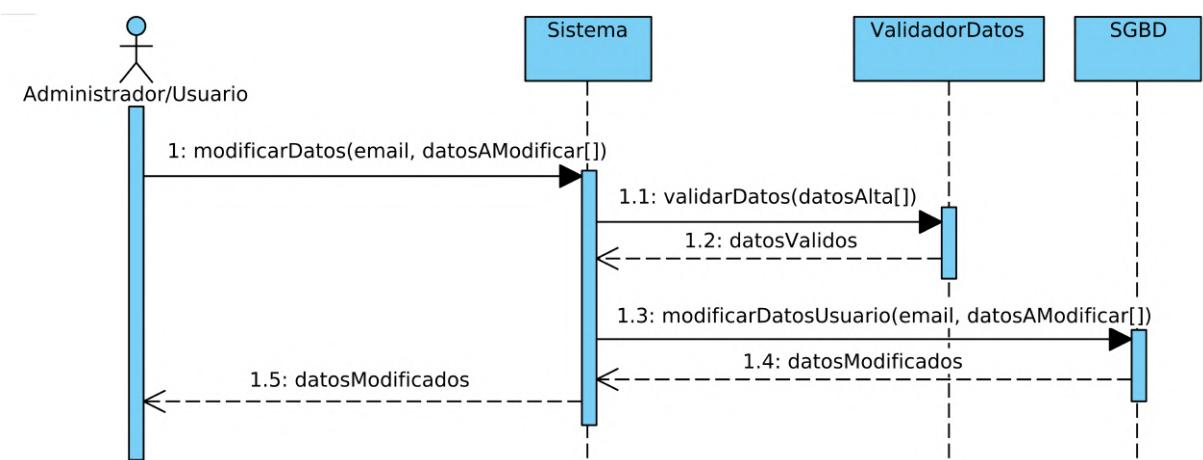


Figura 9: DS: Modificar datos de un usuario o administrador

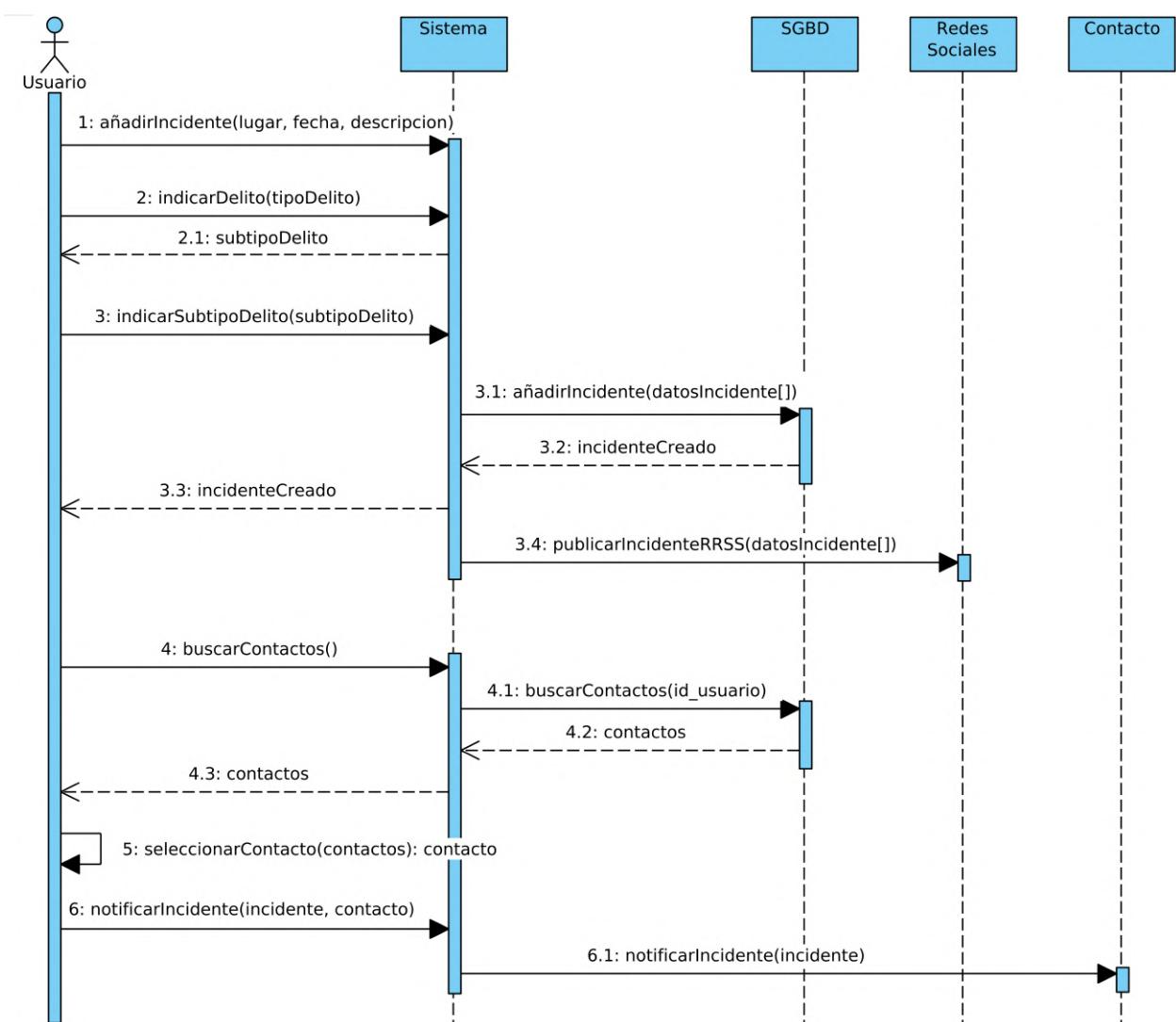


Figura 10: DS: Añadir un incidente, publicarlo en RR.SS. y notificarlo a otro usuario

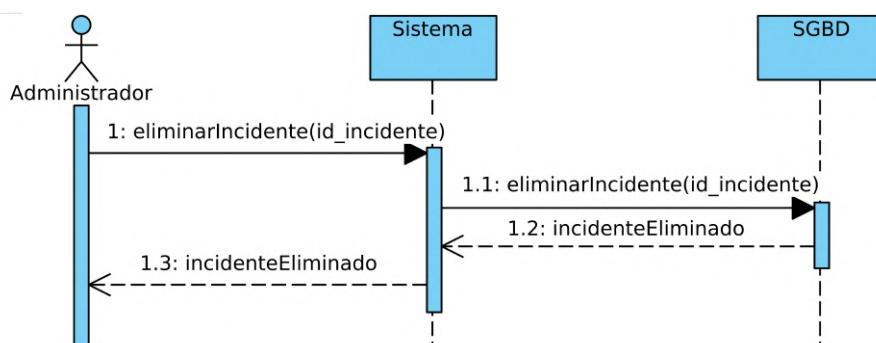


Figura 11: DS: Eliminar un incidente

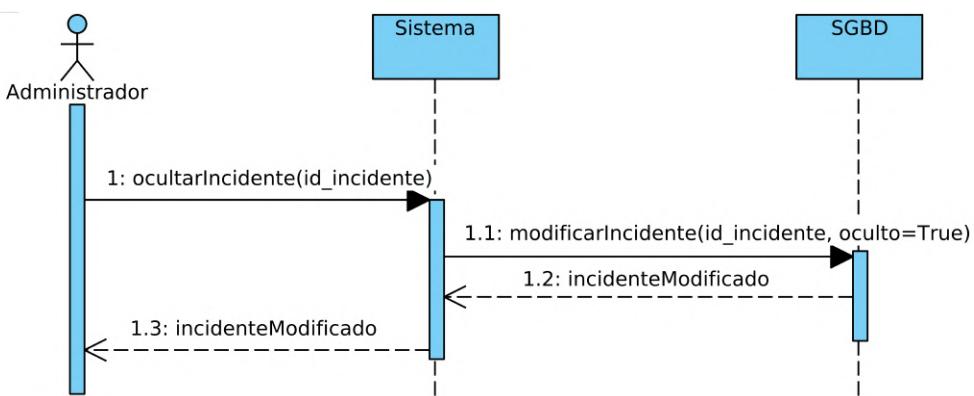


Figura 12: DS: Ocultar un incidente

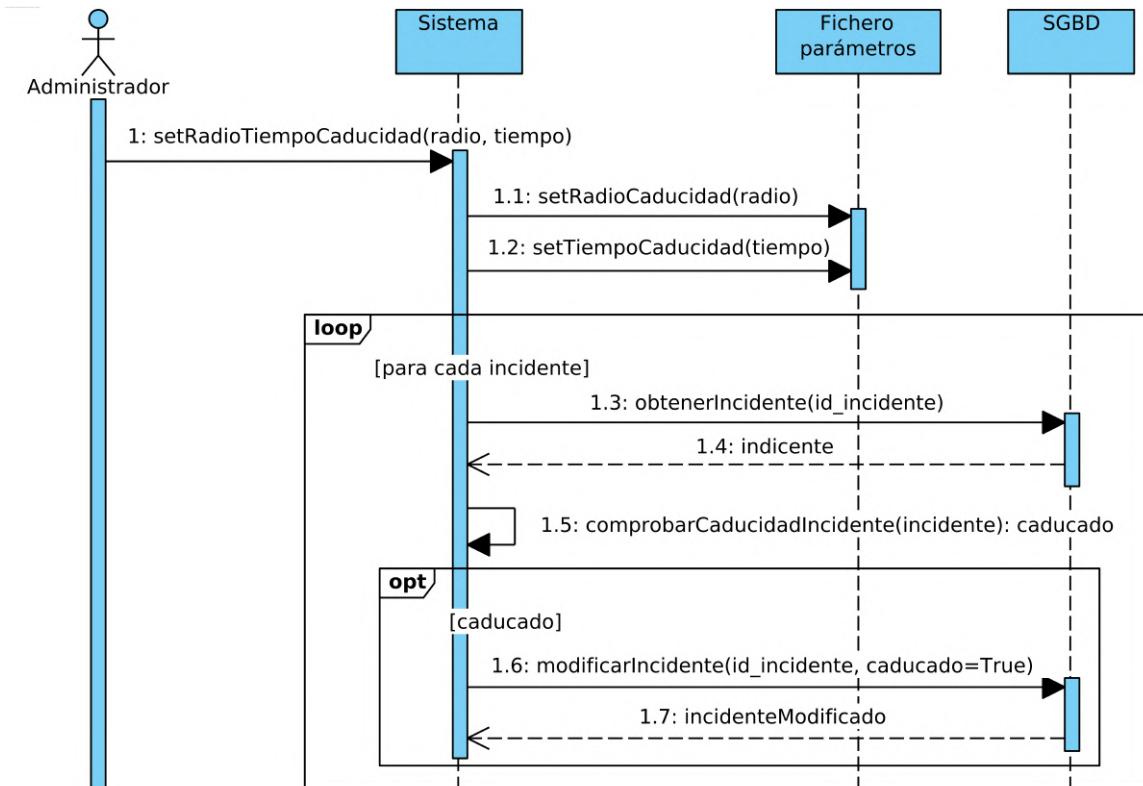


Figura 13: DS: Establecer parámetros para caducar un incidente y caducidad del mismo

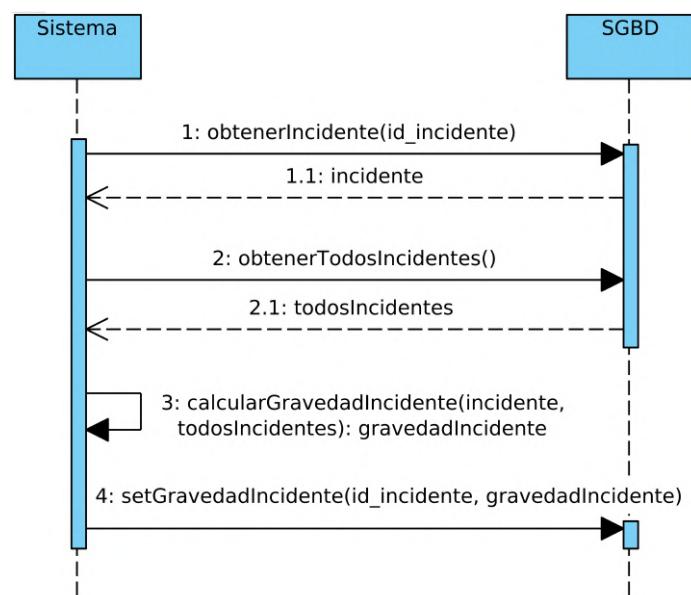


Figura 14: DS: Establecer nivel de gravedad de un incidente

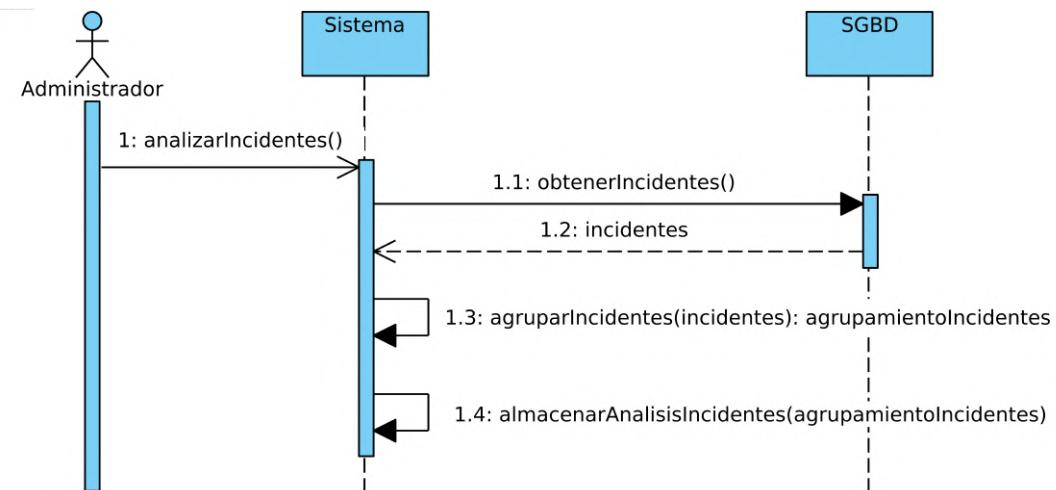


Figura 15: DS: Analizar incidentes

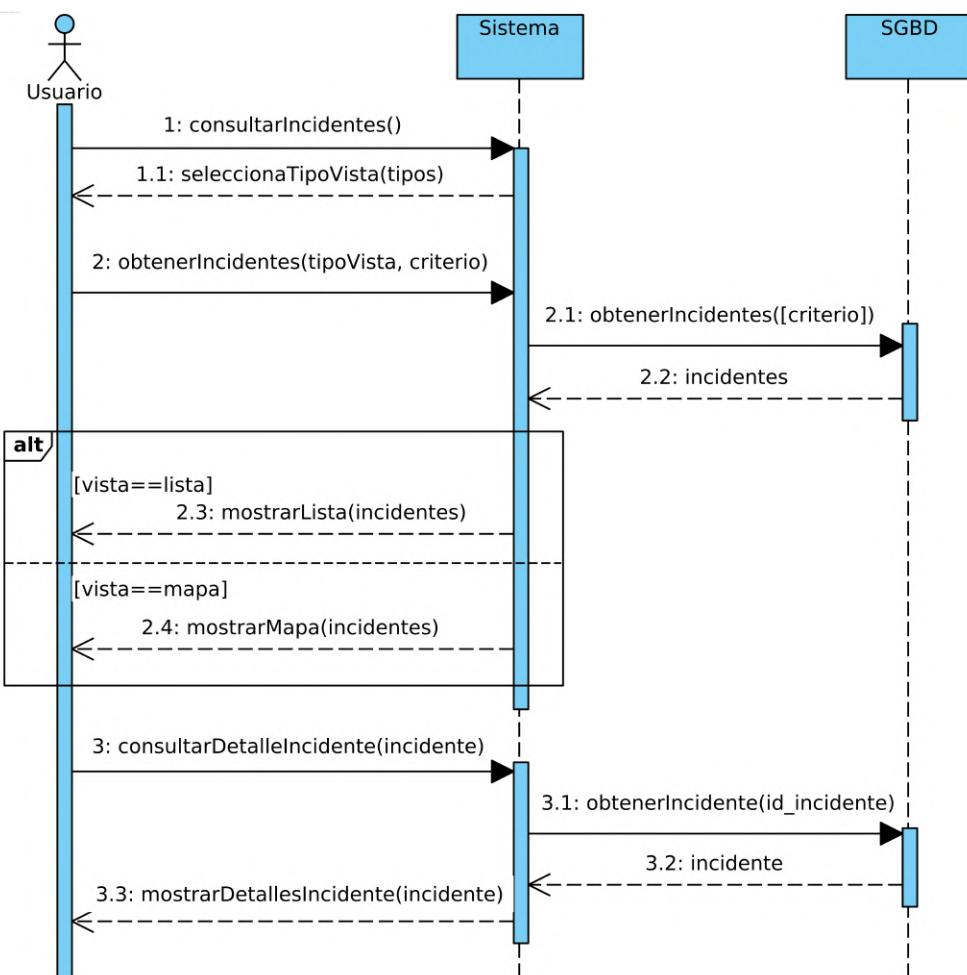


Figura 16: DS: Consultar incidentes y mostrarlos

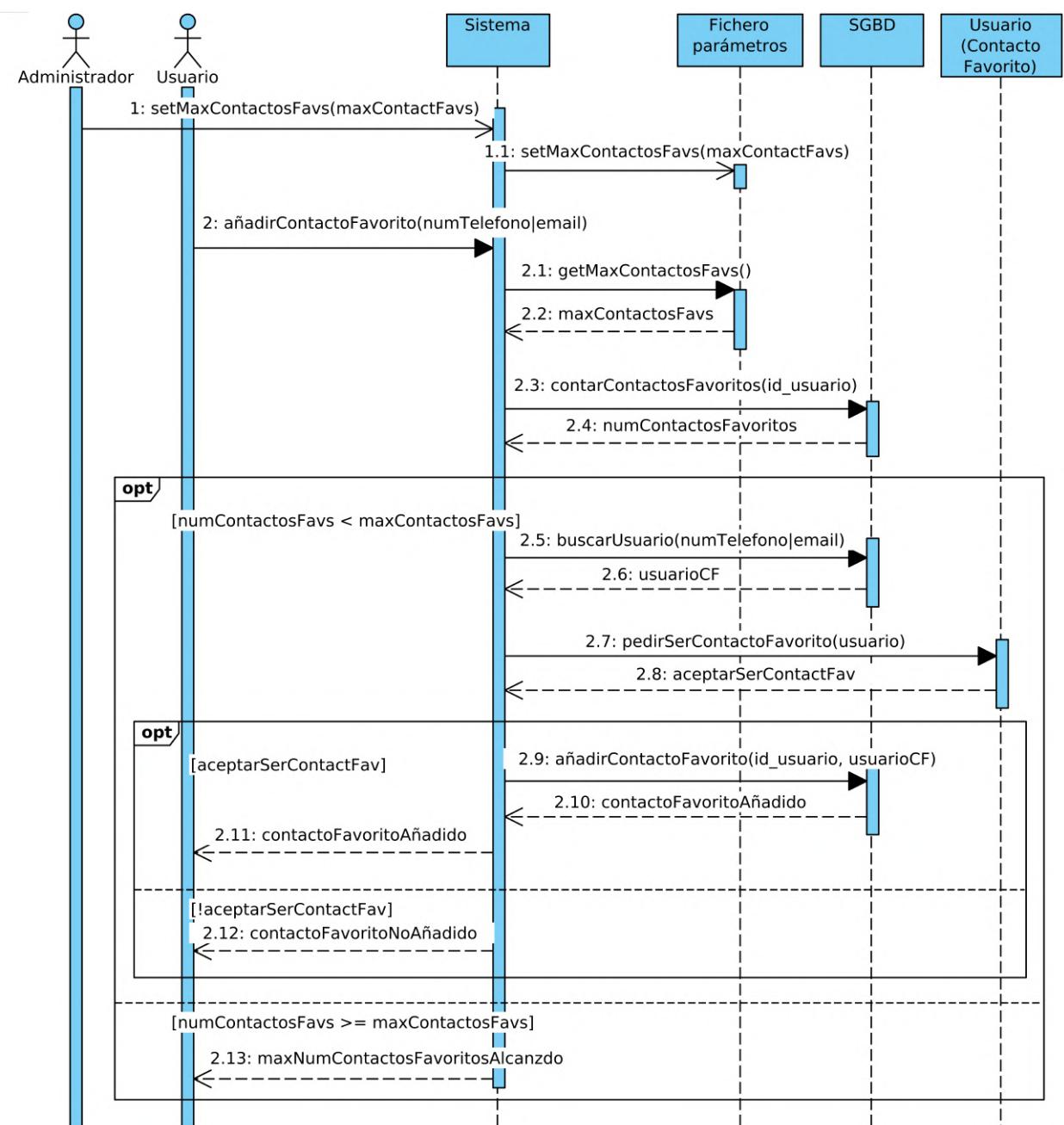


Figura 17: DS: Establecer parámetros para añadir un contacto favorito y aceptar o rechazar serlo

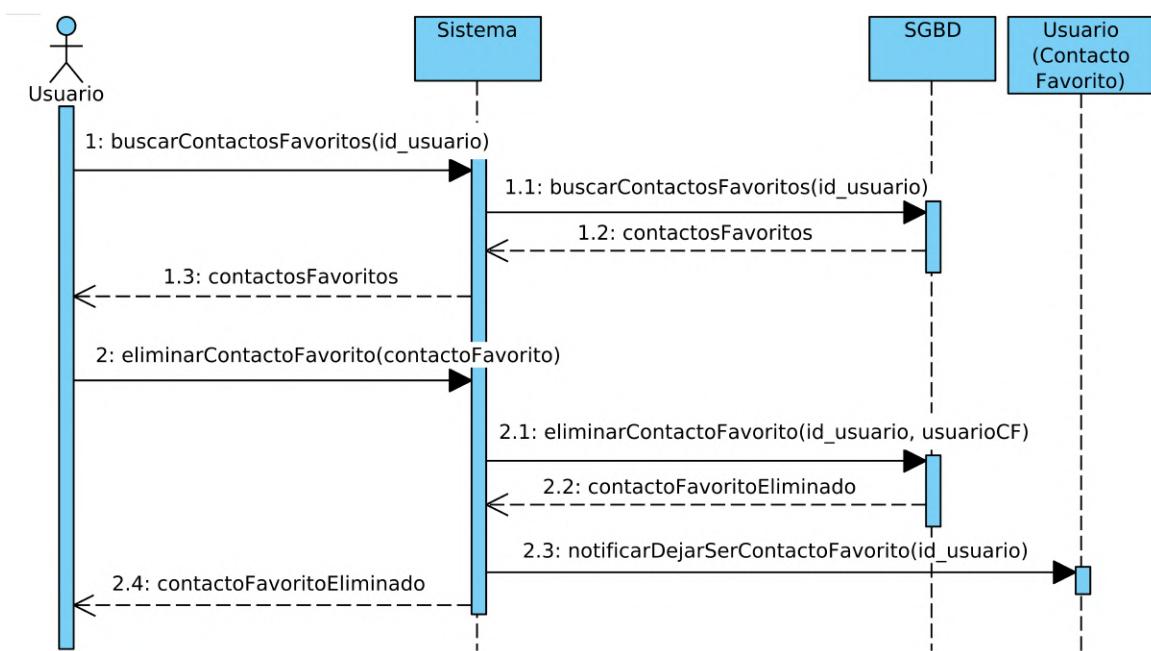


Figura 18: DS: Buscar un contacto favorito y eliminarlo

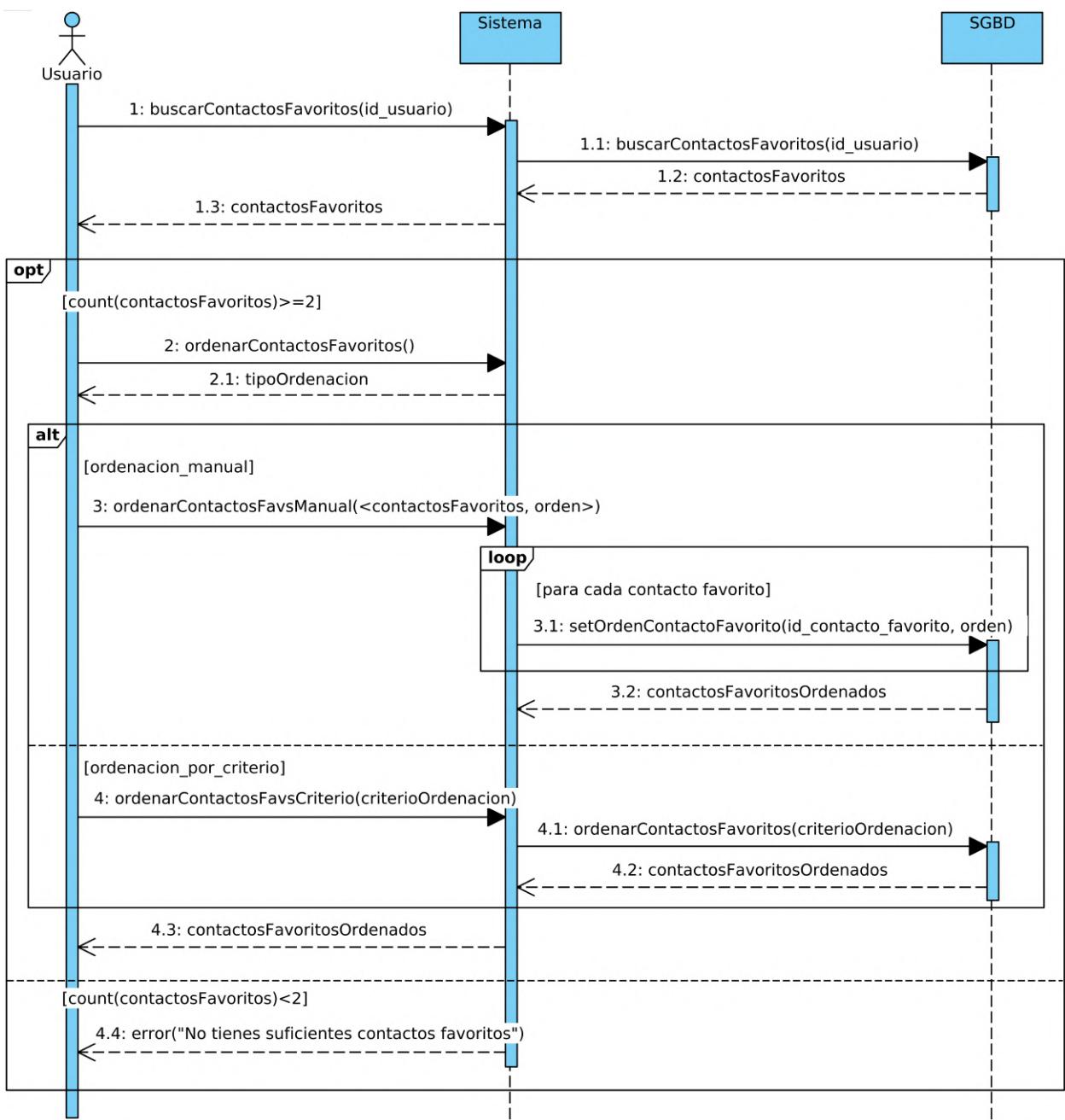


Figura 19: DS: Ordenar contactos favoritos

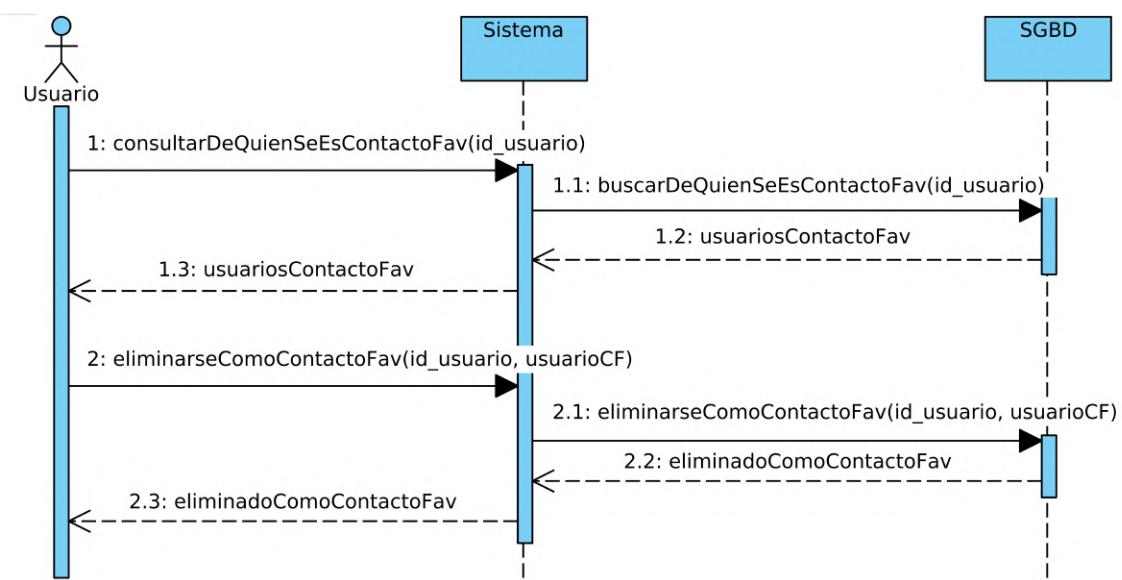


Figura 20: DS: Consultar de quién se es contacto favorito y eliminarse como tal

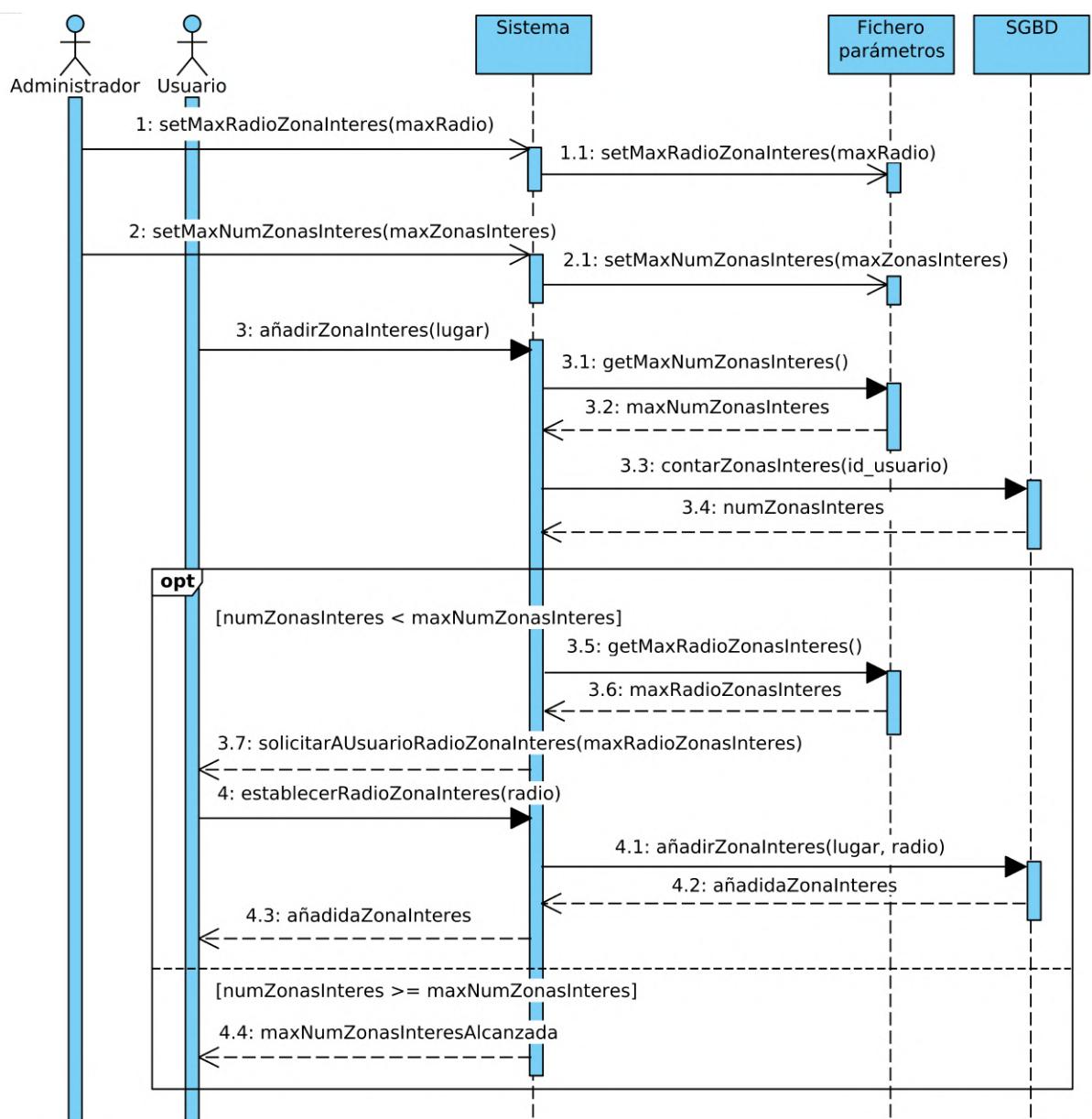


Figura 21: DS: Establecer parámetros para añadir una zona de interés y añadirla

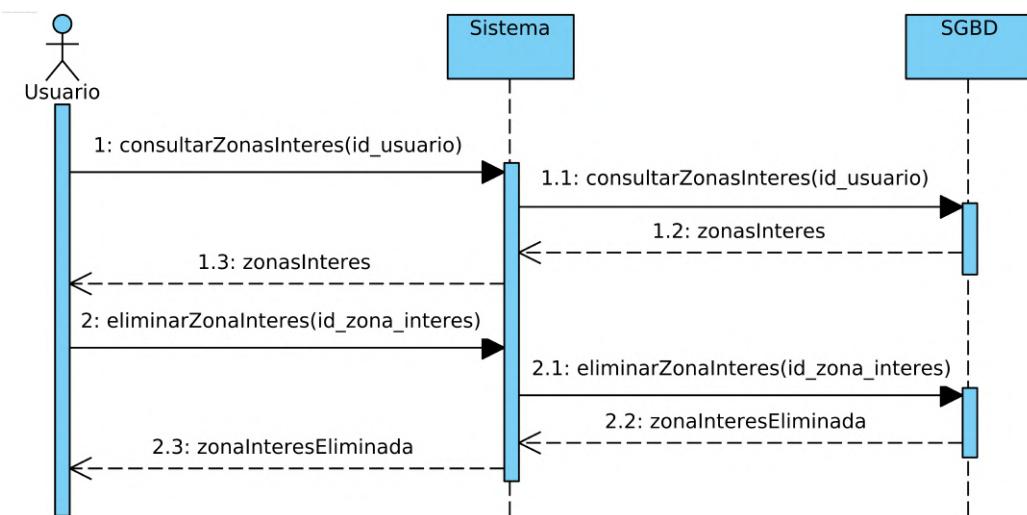


Figura 22: DS: Eliminar una zona de interés

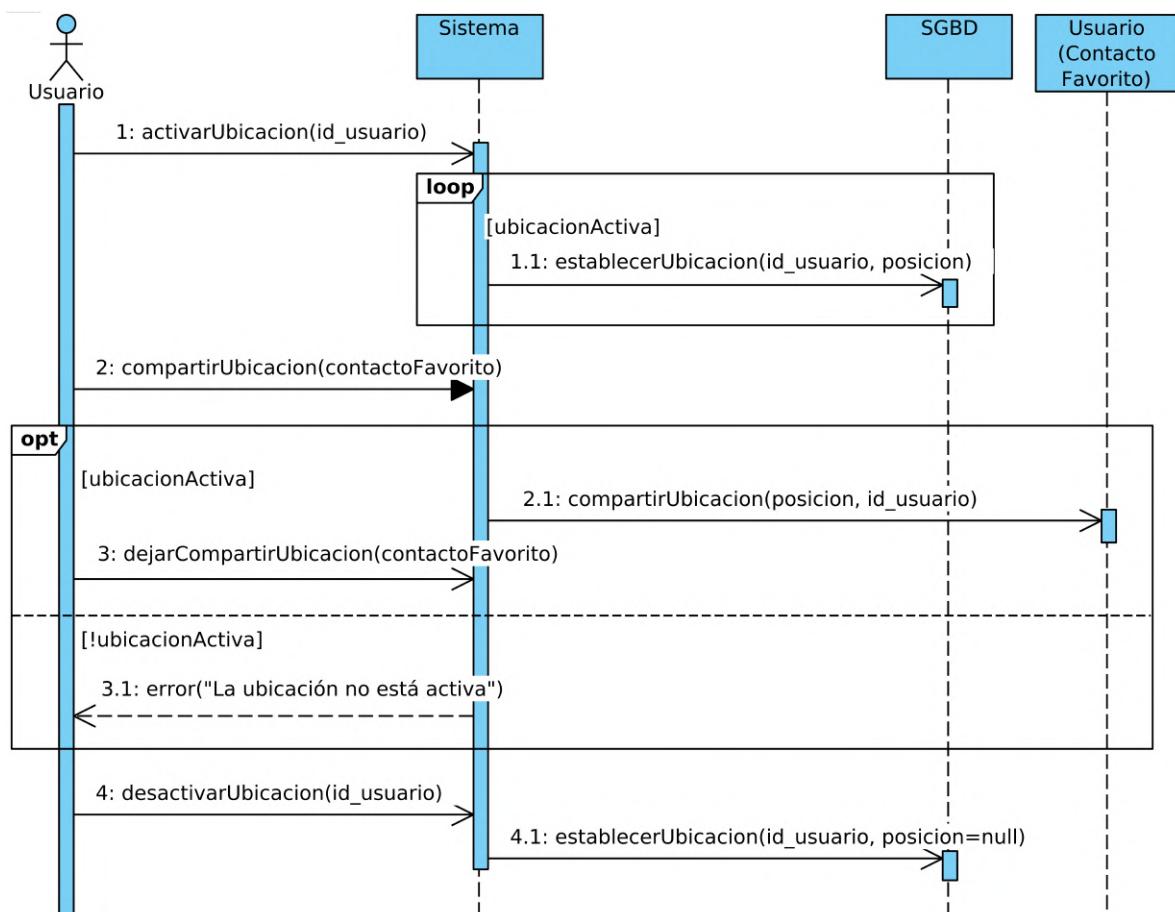


Figura 23: DS: Activar/desactivar ubicación, compartirla y dejar de compartir

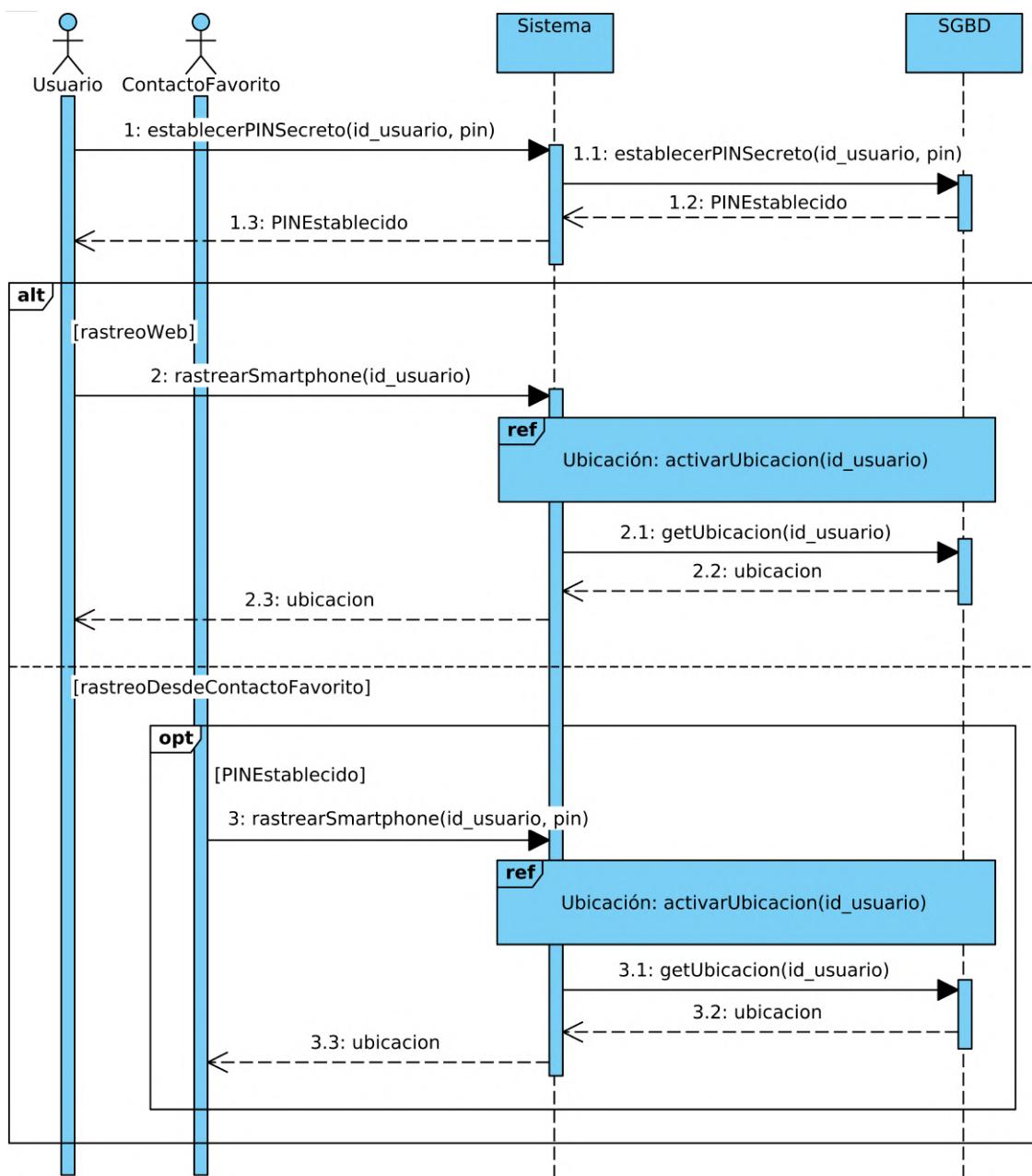


Figura 24: DS: Establecer PIN secreto y rastrear smartphone vía web o desde un contacto favorito

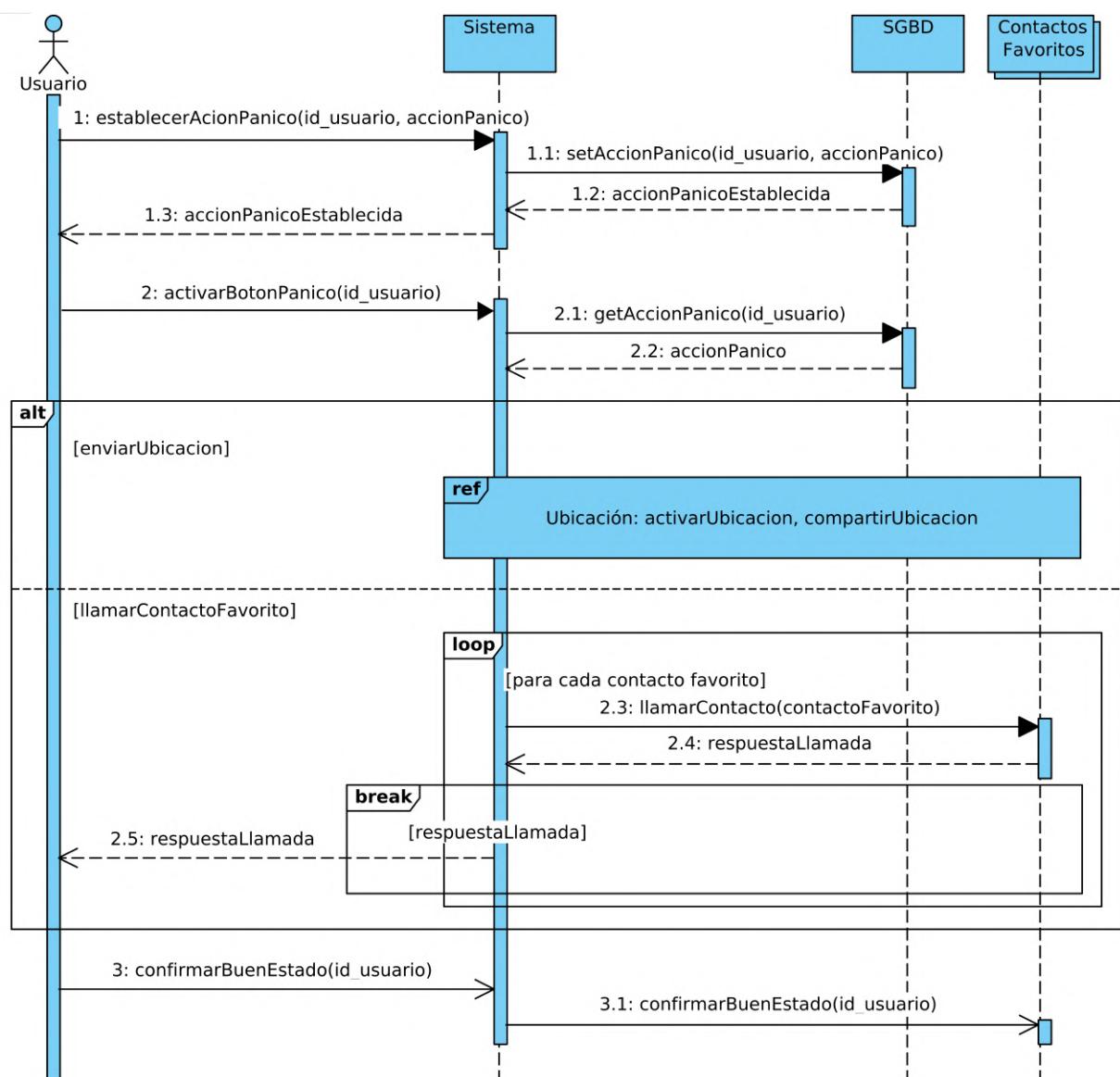


Figura 25: DS: Establecer acción de pánico, activar botón del pánico y confirmar buen estado

4. Diseño

4.1. Diseño de la base de datos

La base de datos del sistema está formada por usuarios y administradores, ambos pertenecientes a la misma entidad “usuarios”, de modo que se simplifica en gran medida su diseño diferenciando a ambos mediante un atributo: es_admin.

La relación entre contactos favoritos se da con la propia entidad usuarios con una cardinalidad de muchos a muchos, y cuenta con tres atributos –orden, contador y son_contactos– para manejar la ordenación y la eliminación y rechazo de peticiones para ser contacto favorito, respectivamente. El hecho de que un usuario solicite a otro ser su contacto favorito ya implica la creación de la relación entre ambos usuarios, pero el atributo “son_contactos” tendrá un valor de 0 por defecto y valdrá 1 cuando se acepte ser el contacto favorito; cuando un usuario elimine a otro, este valor será de 2 para no borrar la relación pero no volver al estado 0, exclusivo para la creación de la relación por primera vez. El atributo “contador” también tendrá un valor de 0 por defecto y se incrementará en uno cada vez que conceptualmente se elimine la relación entre ambos contactos o cada vez que el usuario que recibe la petición para ser contacto favorito rechace serlo, de modo que se pueda controlar el máximo número de peticiones que un usuario puede realizar.

Cada usuario dispone de varias zonas de interés, cada una de las cuales está definida por la latitud, longitud y radio que comprende. Estas zonas pertenecen exclusivamente a cada usuario, es decir, no se comparten entre diferentes usuarios a nivel de almacenamiento de datos, puesto que es más complejo calcular si dos zonas definidas se pueden considerar como iguales en lugar de almacenar directamente varias de ellas, que tan sólo se componen de tres atributos, suponiendo además que ambas abarcaran el mismo radio. Por ello la cardinalidad de la relación usuarios-zonas de interés tendrá una cardinalidad de uno a varios.

Los delitos almacenados en el sistema son únicos y pertenecen a una categoría; se componen de una descripción y de una pena mínima y máxima, tal y como se estipula en el Código Penal, para que el sistema disponga de criterios para el análisis de delitos y para establecer un nivel de gravedad a los incidentes ocurridos, almacenados como una entidad débil de los delitos, y que contienen las características contextuales del suceso así como atributos de utilidad para poder señalarlos como ocultos o caducados.

Los incidentes serán subidos por cada usuario por lo que mantendrán ambas entidades una cardinalidad uno a uno en su relación, dado que el discriminante de fecha y hora en la que se sube el incidente establece que uno sólo puede ser subido por un usuario en un momento dado.

Por último, las zonas que concentran los incidentes se representarán como una entidad inconexa debido a que contendrá resultados procedentes del cálculo automático por parte del sistema, utilizándose para su almacenamiento en la base de datos y su posterior consulta. La entidad contendrá un indicador del nivel de gravedad de cada zona y un color en formato hexadecimal para representarla gráficamente.

Todo lo enunciado queda representado en el diagrama de Entidad-Relación de la Figura 26:

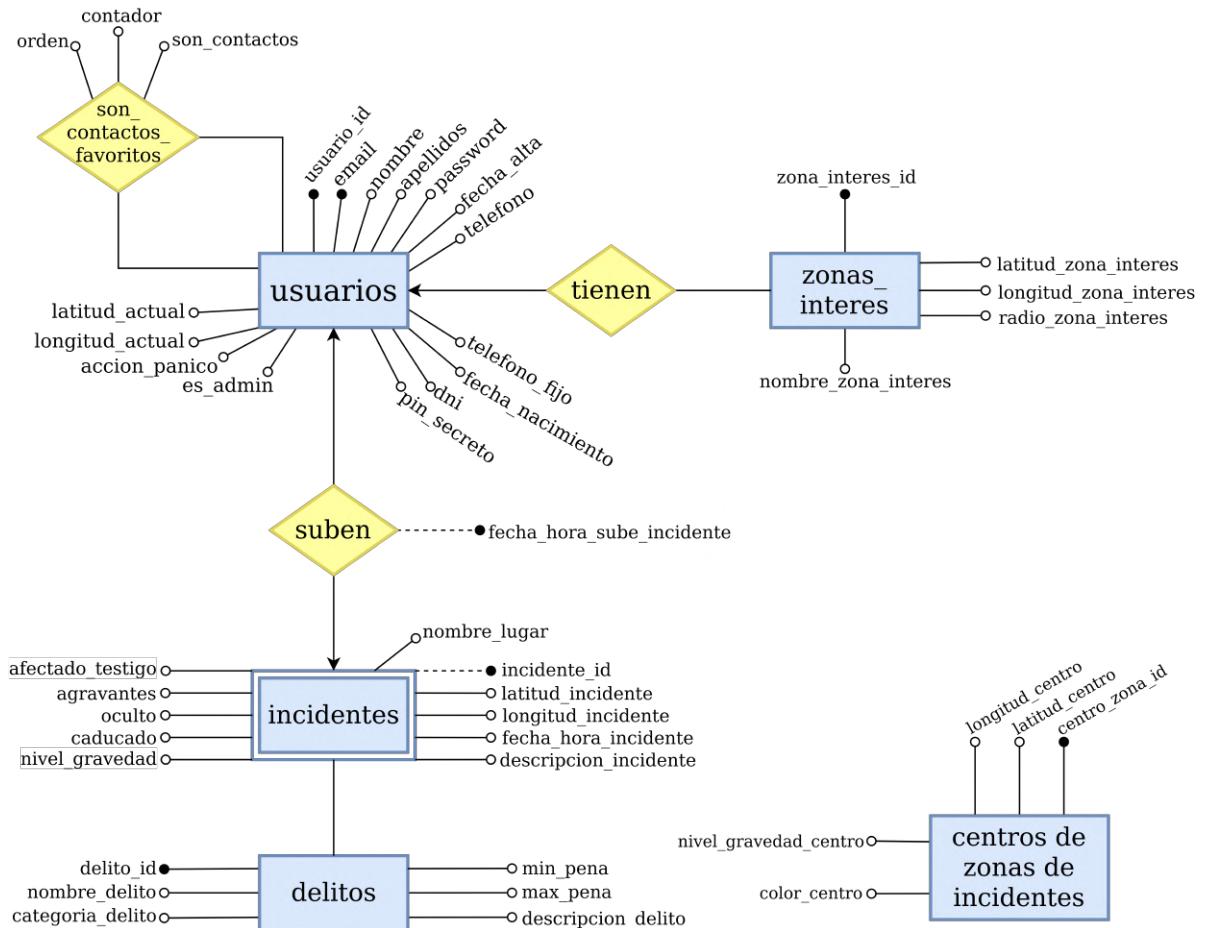


Figura 26: Diagrama E/R

4.1.1. Paso a tablas y fusión

Tras haber definido el modelo relacional es preciso formalizar el paso a tablas. En primer lugar se disponen cada una de las entidades y relaciones del modelo, cada una con sus claves primarias y candidatas (1 a 7). A continuación se realiza la fusión de tablas con el fin de reducir el número existente siempre y cuando no se produzca pérdida.

de información y a su vez se mejore el almacenamiento y el rendimiento del sistema. Para ello es necesario que las tablas fusionadas tengan la misma clave primaria y que estas no procedan de herencia. En este caso sólo se cumple para las tablas “zonas_interes” (2) y “tienen” (6), dando lugar a la tabla fusionada “usuarios_tienen_zonas_interes” (2-6).

- 1: usuarios (usuario_id, email)

$$\frac{\text{CP}}{\text{CC}}$$
- 2: ~~zonas_interes (zona_interes_id)~~

$$\frac{\text{CP}}{\text{CP}}$$
- 3: delitos (delito_id, nombre_delito)

$$\frac{\text{CP}}{\text{CC}}$$
- 4: incidentes (delito_id, incidente_id)

$$\frac{\text{CP}}{\text{CE(3)}}$$
- 5: son_contactos_favoritos (usuario_id, usuario_id)

$$\frac{\text{CE(1)}}{\text{CP}} \quad \frac{\text{CE(1)}}{\text{CP}}$$
- 6: ~~tienen (usuario_id, zona_interes_id)~~

$$\frac{\text{CE(1)}}{\text{CP}} \quad \frac{\text{CE(2)}}{\text{CP}}$$
- 7: suben (usuario_id, fecha hora sube incidente, delito_id, incidente_id)

$$\frac{\text{CE(1)}}{\text{CP}} \quad \frac{\text{CE(4)}}{\text{CE(1)}}$$
- 2-6: usuarios_tienen_zonas_interes (usuario_id, zona_interes_id)

$$\frac{\text{CE(1)}}{\text{CP}}$$

4.1.2. Normalización de la base de datos

La normalización de las bases de datos se concibe como una forma de asegurar que el diseño lógico de una base de datos relacional se encuentre libre de anomalías a la hora de realizar inserciones, actualizaciones y eliminación de datos [16].

La base de datos ya se encuentra en primera forma normal (1FN), ya que en ninguna tabla existe repetición de valores ni hay grupos de repetición (los datos son atómicos en cada campo). También está en segunda forma normal (2FN) ya que está en primera forma normal y todos sus atributos no primos dependen de forma completa de las claves candidatas, es decir, no hay ninguna parte que sea dependiente sólo de una parte de la clave compuesta.

Por último, también se puede afirmar que se encuentra en tercera forma normal (3FN) puesto que está en segunda forma normal y no hay tablas que presenten ningún campo no clave dependiente de ningún otro campo no clave, esto es, no presenta dependencias transitivas problemáticas. Esto se verifica comprobando que en las tres relaciones existentes (“son_contactos_favoritos”, “suben” y “usuarios_tienen_zonas_interes”) no existen atributos no primarios dependientes transitivamente de sus claves primarias.

4.2. Diseño de la arquitectura del sistema

El patrón de diseño ideal para este proyecto es sin duda la arquitectura Modelo-Vista-Controlador (MVC), excelente en proyectos software donde el componente visual cuenta con un importante protagonismo además de proporcionar separación entre la lógica y la información con la que interactúa el usuario [17], tal y como se muestra en la Figura 27. Este patrón es completamente válido tanto para la organización y desarrollo del software tanto en la web como en la aplicación móvil.

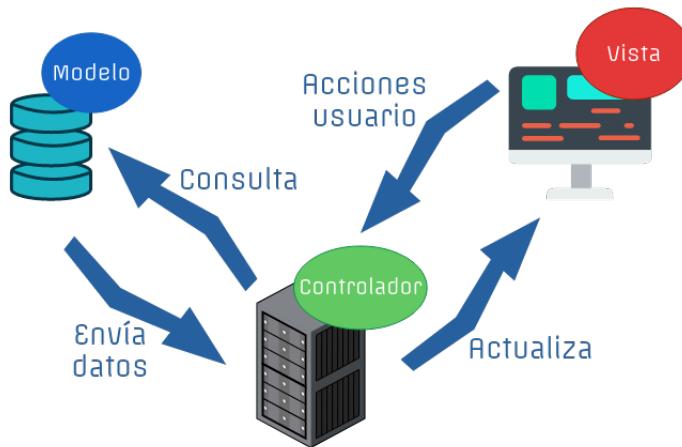


Figura 27: Modelo-Vista-Controlador

Las ventajas [18] que aporta el MVC son:

- Alta cohesión: agrupación lógica de acciones relacionadas en un controlador.
- Bajo acoplamiento: poca unión entre vistas, modelos y controladores.
- Facilidad de modificación: la separación de responsabilidades favorece la modificación a posteriori.
- Depuración y testeo: de nuevo, la separación ayuda a la solución de posibles errores.

Las tareas de cada componente del diseño [19] son las siguientes:

- Modelo: define los componentes del mundo real en colecciones, representando así al conjunto de datos.
- Vista: interactúa con el usuario de forma visual mostrando la información del modelo.

- Controlador: actúa como enlace entre el modelo y la vista, aplicando la lógica necesaria para tal fin. Actúa como el cerebro de todo el entramado de la arquitectura.

4.3. Algoritmos

4.3.1. Asignación del nivel de gravedad a los incidentes

Considerados los distintos tipos incidentes hay que calcular el nivel de gravedad de cada uno de ellos, para el cual se tienen en cuenta dos factores:

1. Las características del propio incidente.
2. Los incidentes de su alrededor.

De esta forma se estima que un mismo tipo incidente –en cuanto a características propias se refiere– no tiene el mismo nivel de gravedad según su entorno. El algoritmo para el cálculo del nivel de gravedad consta de los siguientes pasos:

1. Se obtienen, por un lado, los incidentes previamente existentes con un nivel de gravedad asignado, que no estén ocultos ni caducados y, por otro, los nuevos. A la lista de nuevos incidentes se le asignará el nivel de gravedad correspondiente según sus propias características.
2. Se recorre la lista de nuevos incidentes; cada uno recibe la pena mínima según el delito pertinente como nivel base.
3. Si tiene agravantes asociados, estos suman un valor a una variable numérica que tendrá un valor de 0 en caso de no existir ningún agravante y 1 si los tiene todos, pudiendo tener valores intermedios según los seleccionados.
4. Se suma a la pena mínima el valor resultante de la diferencia entre el máximo y mínimo de las penas del delito multiplicado por el valor obtenido de los agravantes más la pena mínima establecida en el paso anterior, de modo que como máximo el resultado podrá ser el de la pena máxima, obteniendo así una primera aproximación al nivel de gravedad.

Siendo: m = pena mínima; M = pena máxima; agr = suma de agravantes [0, 1]; p = nivel de gravedad de partida

$$p = m + (M - m) \cdot agr$$

5. En siguiente lugar se normaliza el valor obtenido según un intervalo acotado correspondiente a la pena mínima y máxima de todos los delitos existentes y se almacena en una variable para su posterior uso, siendo este el nivel de gravedad propio del incidente, es decir, el nivel relativo a las características inherentes al incidente.
6. A continuación se procede al cálculo del nivel de gravedad de los incidentes vecinos, considerados como tal todos aquellos que tienen un nivel de gravedad asignado previamente y que se encuentren en un radio inferior a 550m.

Si el incidente tiene “vecinos”, el nivel de gravedad asociado (gv) se calcula como el producto de la sumatoria del nivel de gravedad de cada incidente cercano (i) por el inverso de la raíz cuadrada del número total de incidentes cercanos (N):

$$gv = \frac{1}{\sqrt{N}} \cdot \sum_{i=0}^N i$$

De esta forma se logra tener en consideración la presencia de incidentes próximos y a su vez suavizar el crecimiento en caso de haber demasiados.

7. Por último se asigna un peso del 60% al nivel de gravedad propio (gp) y un 40% al de los incidentes cercanos, obteniendo así el nivel de gravedad final del incidente:

$$gravedad_{incidente} = 0,6 \cdot gp + 0,4 \cdot gv$$

4.3.2. Agrupación de incidentes

El agrupamiento de incidentes para determinar cuáles son las zonas donde estos se acumulan se llevará a cabo mediante un algoritmo de *clustering*. Esta elección se debe a que se trata de una técnica de clasificación no supervisada de datos con clases no predefinidas que puede realizar un agrupamiento de los incidentes, lo cual es interesante para este problema.

Probablemente el algoritmo de *K-Means* sea el más popular para este tipo de casos por su eficiencia, pero en este supuesto presenta varias desventajas como son el hecho de tener que fijar el número de clusters de manera anticipada y la gran probabilidad de encontrar datos ruidosos y/o *outliers*. Por el contrario, el algoritmo *Mean Shift* no requiere fijar el número de clusters con antelación, lo cual es un punto importante en este problema. La idea subyacente de este algoritmo es que posee una función de

densidad de probabilidad sobre los datos existentes y para las regiones más densas fija un centroide que va desplazando hasta encontrar el centro.

En el problema que nos ocupa el propósito es identificar zonas con cierta concentración de incidentes, encontrar su centro y establecer la zona como lugar de alta incidencia, para lo cual *Mean Shift* es la opción escogida. El proceso a seguir es:

1. Se obtienen todos los incidentes existentes y se realiza una primera agrupación por ciudades/distritos y se aplica el algoritmo de *clustering* a cada zona.
2. A cada elemento de los subconjuntos de incidentes obtenidos se le aplica una normalización a la posición en dicha zona en un intervalo $[0, 10]$.
3. Una vez que los incidentes de cada zona se agrupan en un *dataset* normalizado se aplica el algoritmo *Mean Shift* para que obtenga los centros del conjunto, que serán los centros de las zonas de incidentes en este caso.
4. Se desnormalizan los centros para tener su posición en coordenadas geográficas y para todos los incidentes cercanos a dicho centro se suma el nivel de gravedad de cada uno, obteniendo así el nivel de gravedad de la zona.

4.3.3. Asignación de color a zonas de incidentes

Definida una escala de n colores, se establece el color de cada zona de incidentes normalizando el conjunto de niveles de gravedad de todas las zonas en un intervalo $[0, 1]$. De este modo, como hay $n-1$ franjas de colores, según el nivel de gravedad $\alpha \in [0, 1]$ le corresponderá un lugar concreto dentro de dicha franja de color. Al haber n colores, a un incidente con nivel de gravedad α le corresponderá la franja x si su valor normalizado está en el intervalo $[\frac{x-1}{n-1}, \frac{x}{n-1}]$.

Como cada tono se puede expresar con un valor hexadecimal según el modelo RGB, la equivalencia exacta entre dos valores hexadecimales se calcula como la diferencia de entre cada una de las ternas de colores que componen a cada color en el modelo mencionado, tal y como se expone a continuación:

$$\begin{aligned} nivel_gravedad' &= normalizacion(nivel_gravedad) \\ diferencia_color &= color1_H - color2_H \\ color_zona &= color1_H + (diferencia_color \cdot nivel_gravedad') \end{aligned}$$

El resultado gráfico se ejemplifica en la Figura 28:

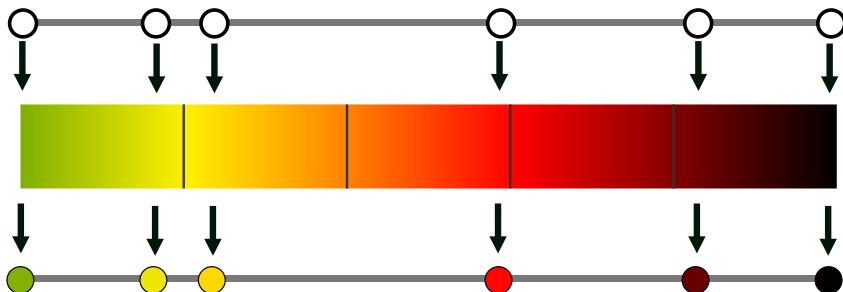


Figura 28: Asignación de color a los centros de zonas de incidentes

4.4. Diseño de la interfaz de usuario

Antes de comenzar con la implementación del proyecto es importante realizar bocetos de cómo será la interfaz de usuario, más aún teniendo en cuenta que tanto en una página web como en una aplicación móvil la disposición de los elementos es muy importante a fin de que quien haga uso de ellas se encuentre con elementos visualmente agradables y que faciliten la navegación entre distintas partes de la plataforma [20].

El uso de *Mock Ups* o bocetos de la interfaz permite realizar diseños y aproximaciones de cómo será el resultado final. El tiempo invertido en esta parte permite tener una idea clara de qué se quiere crear, ahorrando a posteriori gran cantidad de esfuerzo en pensar, diseñar y rediseñar la interfaz hasta dar con un resultado aceptable pero que muy probablemente no será tan bueno como uno que previamente haya pasado por una fase de diseño conceptual.

4.4.1. Diseño de la interfaz web

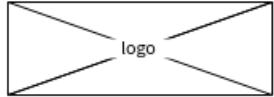
En la página web del proyecto lo primero que encuentra el usuario es una página de bienvenida con el logo, una o varias imágenes relacionadas con el proyecto y un formulario de inicio de sesión (Figura 29).



Figura 29: Interfaz de inicio de sesión

El formulario de registro se divide en dos pasos: el primero en el que el usuario introduce el email, la contraseña y la confirmación de esta y el segundo en el que, si los datos del primer paso son correctos, se termina de completar la información requerida para el registro (Figura 30).

Registro



Registro

E-mail

Contraseña

Confirmar contraseña

Siguiente

1 ————— 2

¿Ya tienes cuenta?
[Inicia sesión](#)

Registro

Nombre !

Apellidos !

D.N.I.

Fecha de nacimiento !

Número de teléfono móvil

Número de teléfono fijo

Registrarse

1 ————— 2

¿Ya tienes cuenta?
[Inicia sesión](#)

Figura 30: Interfaz registro

En las vistas de la web con sesión iniciada aparecen siempre dos elementos fijos: un menú de navegación lateral a la izquierda y una sección con las notificaciones y un acceso a la zona personal del usuario en la parte superior-derecha. Esta zona cuenta con dos secciones principales, la de configuración y la de datos personales y modificación de contraseña. Las opciones de configuración abren cada una un *modal* con las acciones que llevan a cabo (Figura 31).

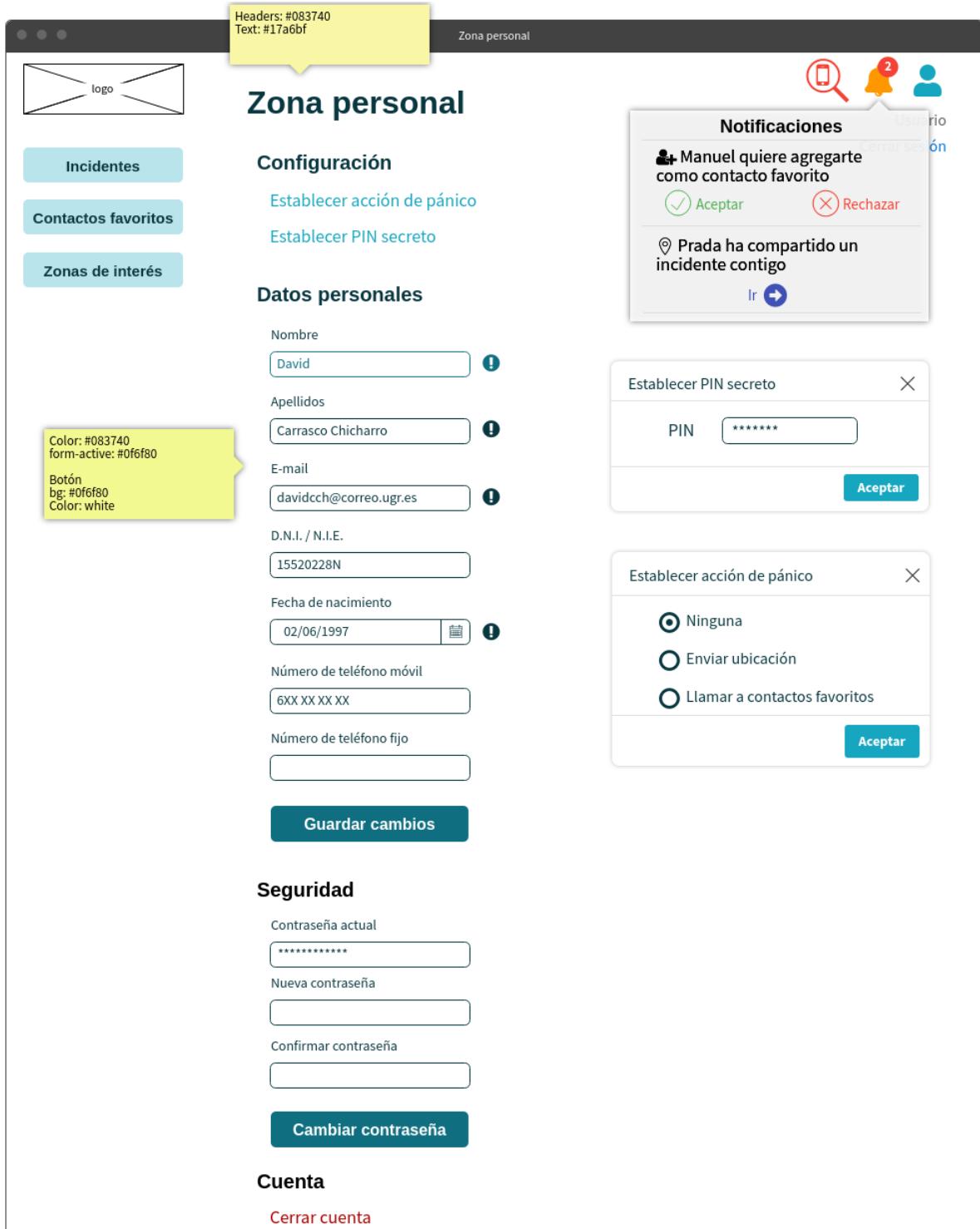


Figura 31: Interfaz de zona personal de usuario

Las vistas de los incidentes se muestran tanto con un mapa como en una lista,

ambas con filtros que permiten consultarlos según su tipo y rango de fecha (Figura 32 y Figura 33, respectivamente).

Figura 32: Interfaz de mapa de incidentes

Incidente 1	dd/mm/yyy - hh:mm
Lugar	Ver más
Incidente 2	dd/mm/yyy - hh:mm
Lugar	Ver más
Incidente 3	dd/mm/yyy - hh:mm
Lugar	Ver más
Descripción del incidente	
Compartir incidente	
Ver menos	

Figura 33: Interfaz de lista de incidentes

Para dar de alta un nuevo incidente en el sistema se brinda al usuario un formulario donde completar los distintos apartados del mismo (Figura 34).

The screenshot shows a web-based application for reporting incidents. At the top right are icons for search, notifications, and user profile, along with links for 'Usuario' and 'Cerrar sesión'. On the left, there's a sidebar with a logo and three buttons: 'Incidentes' (highlighted in blue), 'Contactos favoritos', and 'Zonas de interés'. The main content area has a title 'Dar de alta incidente'. It includes fields for 'Tipo de delito' (Category of crime) and 'Delito' (Crime), both with dropdown menus. Below these are fields for 'Fecha' (Date) and 'Hora' (Time), each with a date picker and time picker respectively. A 'Lugar' (Place) field contains a search bar with the placeholder 'Calle' and a map view showing a city street layout with zoom controls. An 'Descripción' (Description) text area has a character limit of 1000 characters indicated by '0/1000'. Under 'Agravantes' (Aggravating factors), there are four checkboxes: 'Disfraz' (Costume), 'Abuso de superioridad' (Abuse of superiority), 'Sufrimiento inhumano' (Inhuman suffering), and 'Racismo, discriminación, homofobia, ma' (Racism, discrimination, homophobia, etc.). Below this is a section 'He sido' (I was) with radio buttons for 'Afectado' (Affected) and 'Testigo' (Witness). At the bottom is a large blue button labeled 'Añadir incidente' (Add incident).

Figura 34: Interfaz de alta de incidente

Los contactos favoritos del usuario se muestran en forma de lista, similar a la de los incidentes, con un desplegable para cada uno donde consultar los datos en detalle y realizar acciones sobre él. En una sección lateral a la lista (derecha) se presentan las opciones para añadir a un nuevo contacto, ordenarlo y consultar de quién se es contacto favorito (Figura 35).

1	Daniel Montoro	+
2	Lucía López	+
3	Mario Garrido	+
4	Josefa González	+
5	Ana Gómez	-
Teléfono: XXX XX XX XX		
Eliminar como contacto favorito		

Figura 35: Interfaz de contactos favoritos

Las zonas de interés del usuario se muestran de forma totalmente visual en un único mapa que abarca todas las zonas existentes. En el lateral se presenta la opción de añadir nuevas o eliminar una existente (Figura 36).

Figura 36: Interfaz de zonas de interés

Si quien inicia sesión tiene el rol de administrador, este posee una pantalla de inicio donde aparecen diversas opciones de administración (Figura 37), todas ellas –

excepto el alta de un administrador nuevo— dispuestas en un *modal* interactivo al ser seleccionadas (Figura 38).



Figura 37: Interfaz de inicio de administración

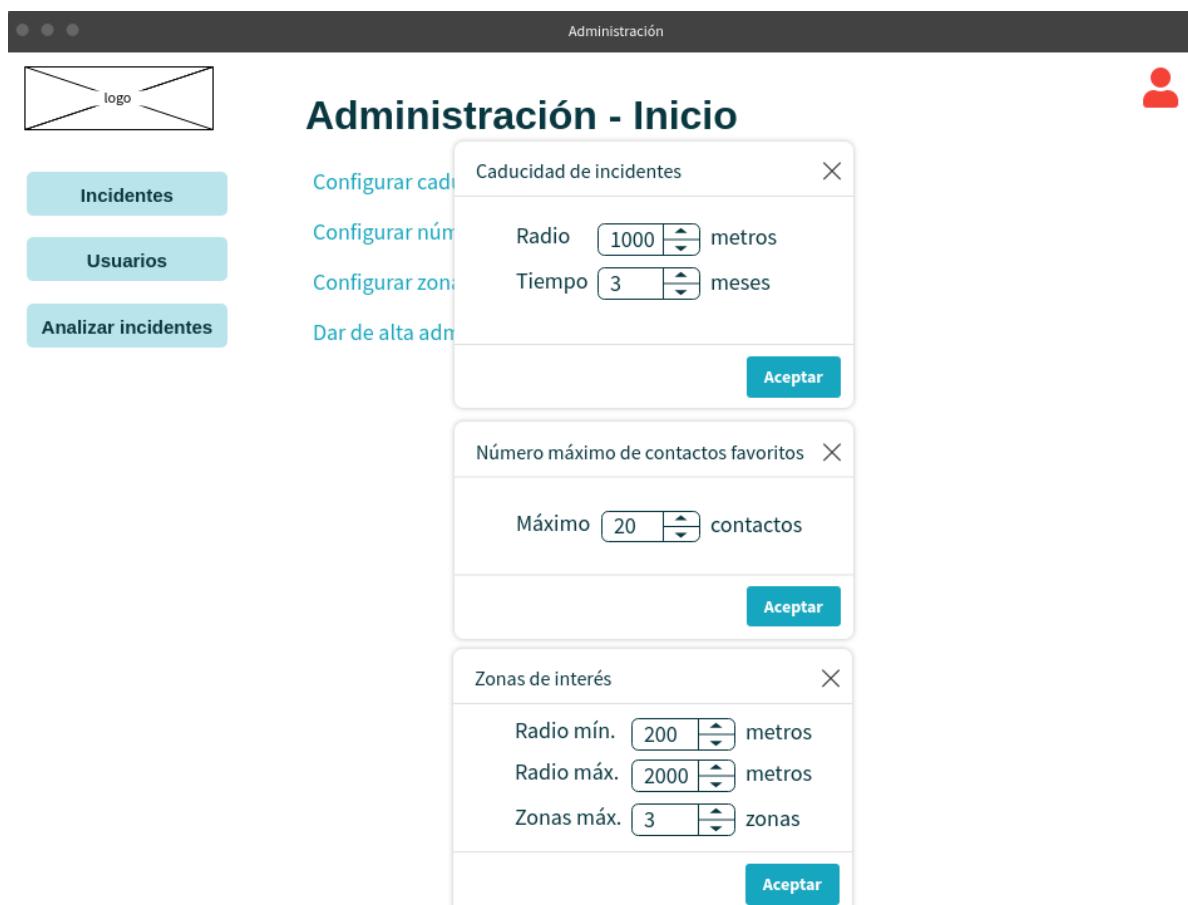


Figura 38: Interfaz configuración en administración

4.4.2. Diseño de la interfaz de la aplicación móvil

El diseño de las interfaces que se muestran en esta sección es extensible para la vista adaptable de la web. De igual modo, conforme a lo visto en la sección anterior, sucede en la interfaz móvil cuando el usuario abre la aplicación sin sesión previa iniciada, encontrando una vista de inicio de sesión o el formulario de registro en su caso, también en dos pasos (Figura 39).

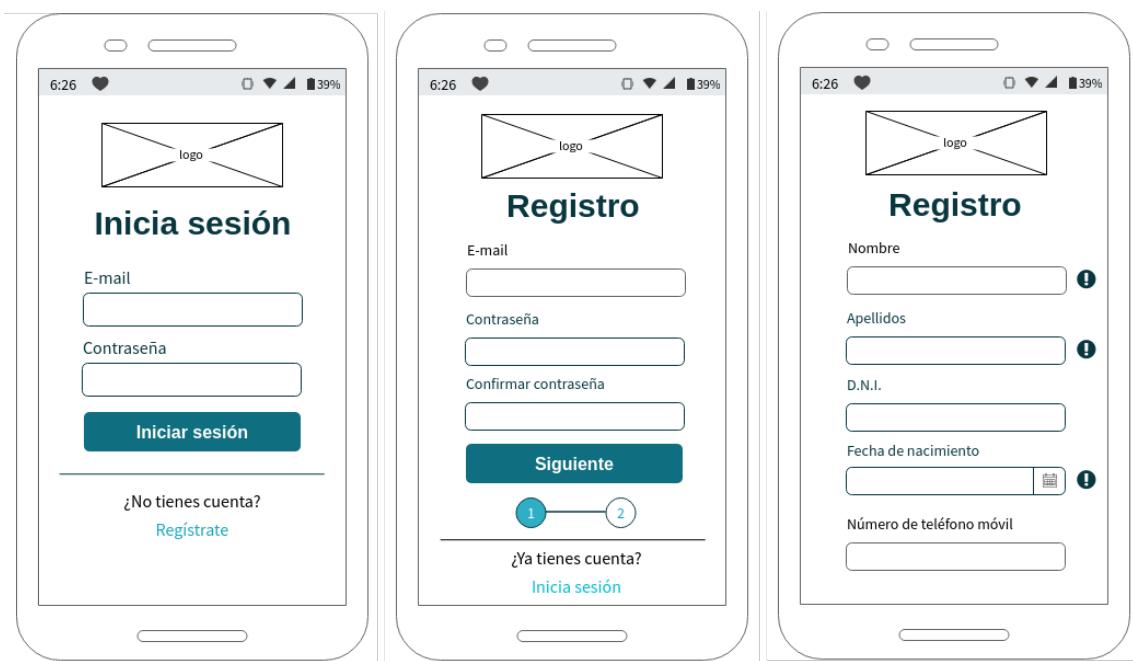


Figura 39: Interfaces de login y registro en aplicación móvil

La aplicación se organiza con un menú inferior dividido en pestañas (*tabs*). En la zona personal del usuario se encuentran las mismas opciones de la interfaz web, también con *modals* para las acciones de pánico y el PIN (Figura 40).



Figura 40: Interfaces de zona personal y configuración de acción de pánico y pin secreto en aplicación móvil

La lista y el mapa de incidentes se encuentran en la misma pestaña, donde se puede intercalar la vista por medio de un botón. El filtro de incidentes se dispone en un *modal* (Figura 41).

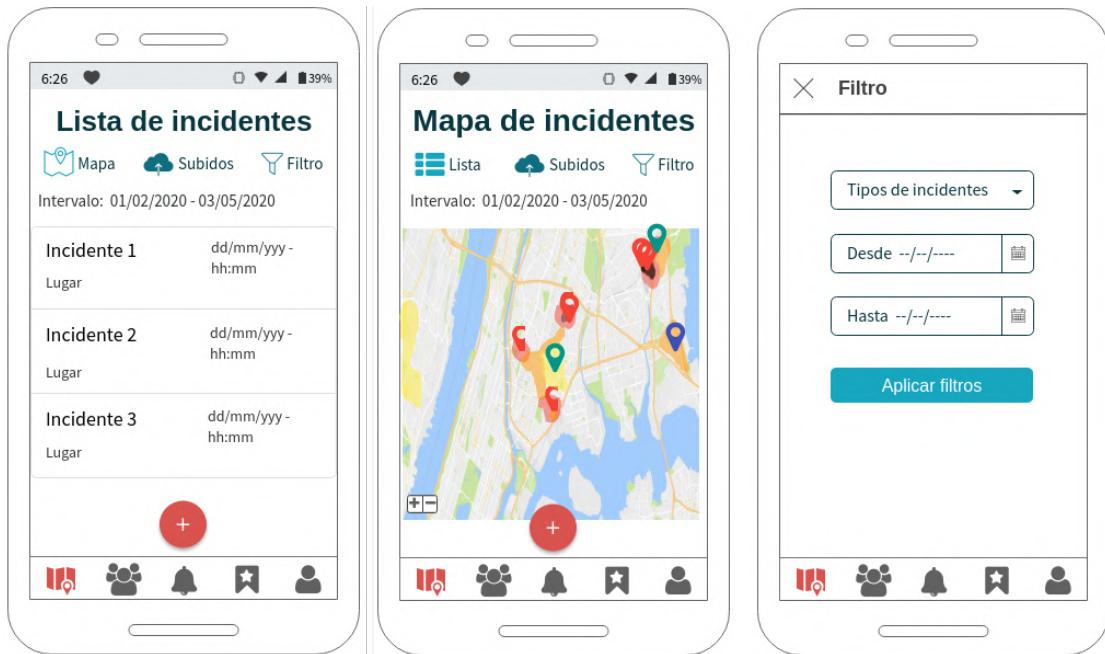


Figura 41: Interfaces de lista, mapa y filtro de incidentes en aplicación móvil

Desde la lista se puede acceder al detalle de cada incidente, además de navegar hasta el alta de uno mediante el botón central. La vista de las notificaciones dispone de una pestaña propia (Figura 42).

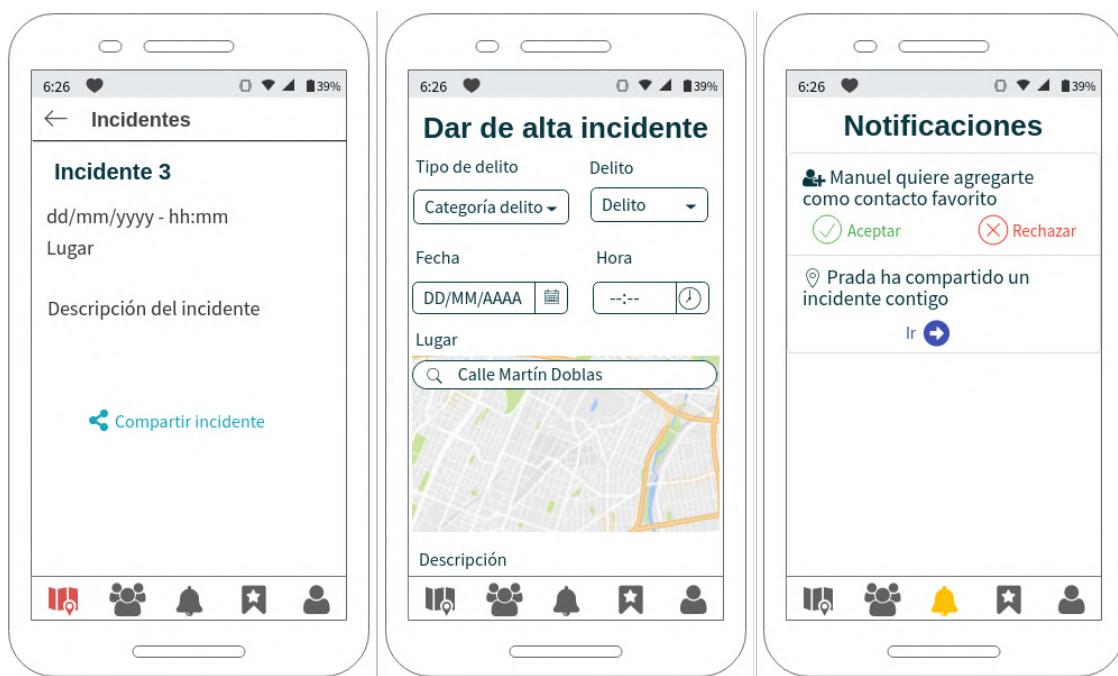


Figura 42: Interfaces de incidente, alta de incidente y notificaciones en aplicación móvil

Por último, las vistas de zonas de interés y contactos favoritos tienen cada una pestañas propias en el menú de navegación, con la misma disposición que la web y con una pantalla individual para los detalles de contacto (Figura 43).

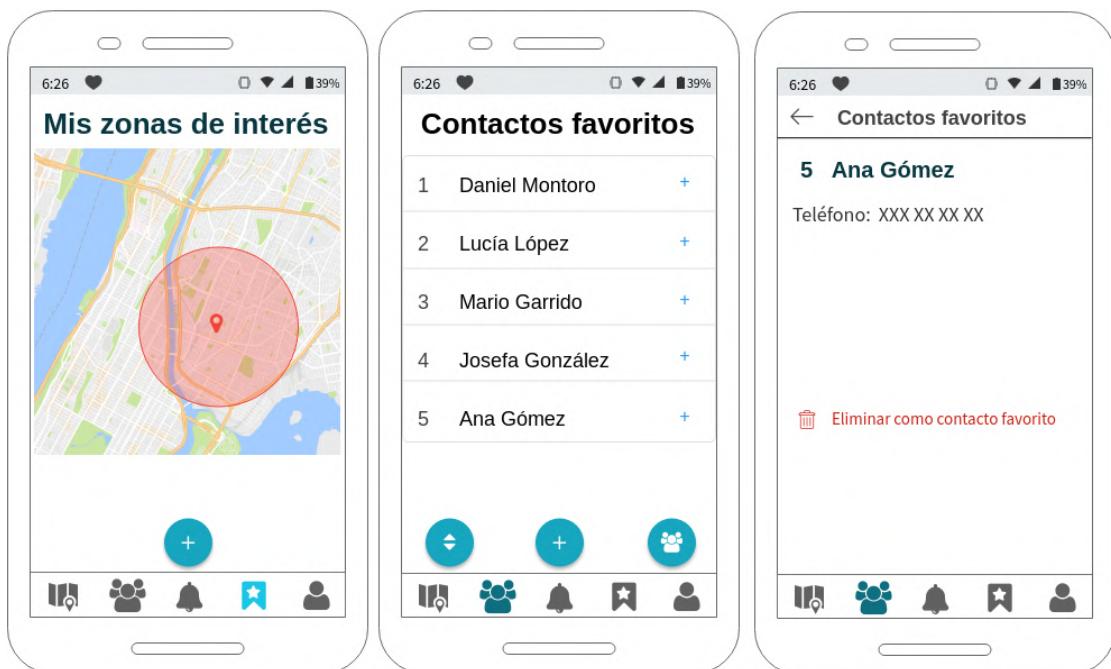


Figura 43: Interfaces de zonas de interés y contactos favoritos en aplicación móvil

5. Implementación

5.1. Desarrollo web

5.1.1. Entorno de desarrollo

Para el desarrollo web en el lado del servidor el lenguaje elegido es PHP [21], un lenguaje de código abierto orientado al desarrollo web. La elección de este lenguaje se fundamenta en su soporte a la orientación a objetos, sencilla sintaxis y débil tipado, además de por mi propia experiencia con el lenguaje. Otra opción interesante hubiera sido Python, un lenguaje en auge y muy “limpio”, pero con mayor orientación hacia proyectos científicos y cuyos *frameworks* no tienen tanto soporte comunitario. Aún así, se hará uso del *scripting* de Python para aplicar un algoritmo de clustering a los incidentes del proyecto que sea capaz de agrupar estos.

En el lado del cliente los lenguajes utilizados son HTML, CSS y JavaScript ya que son los más habituales para la programación *front end*.

HTML (*HyperText Markup Language*) [22] es un lenguaje de marcado que fue creado para la elaboración de páginas web. Es un estándar a cargo del *World Wide Web Consortium (W3C)*. Su última versión es HTML5 [23], que como novedad frente a anteriores versiones introduce por un lado nuevos elementos que permiten clarificar la semántica de la estructura del documento, y por otro, un conjunto más amplio de tecnologías que permite hacer la web más diversa.

CSS (*Cascading Style Sheets*) [24] un lenguaje de diseño para documentos estructurados (como HTML) que dota a las páginas web de estilo, proporcionando a la web flexibilidad y elegancia. Contará con el soporte de Bootstrap 4 [25], una biblioteca de código abierto para el diseño de sitios web.

JavaScript es un lenguaje interpretado, orientado a objetos, que otorga a las páginas web dinamismo para una mejor experiencia de usuario en el *front end*. Con el uso de jQuery [26], otra biblioteca de código abierto, se aportarán funcionalidades para el manejo del árbol DOM del HTML, facilitará la comunicación con el servidor mediante AJAX (*Asynchronous JavaScript And XML*) [27] y, como punto más importante, dotará de mayor potencia las acciones interactivas con el sitio web. En cuanto a AJAX, que no es un lenguaje de programación sino una técnica de desarrollo, provee de interactividad a la aplicación, permitiendo que la comunicación con el servidor se realice de manera asíncrona sin necesidad de recargar las páginas web para cada intercambio de información entre el cliente y el servidor.

Una vez seleccionado el lenguaje de programación es importante elegir un *framework* adecuado para desarrollar el proyecto, pues facilita el mantenimiento del código, hace el trabajo más fluido y permite centrarse en el desarrollo, haciendo que la aplicación sea mucho más robusta que si se realizara un desarrollo desde cero sin ningún soporte.

Para PHP los *frameworks* más conocidos y empleados son *Symfony*, *CodeIgniter*, *CakePHP* y *Laravel*, entre otros [28]. Todos ellos utilizan un modelo-vista-controlador (MVC), lo cual facilita el desarrollo y el mantenimiento, ambos dos claves según lo planteado previamente.

Symfony está muy documentado, es de código abierto, muy rápido y fácil de usar; por el contrario sus mecanismos de seguridad son difíciles de implementar y está orientado a proyectos a gran escala. *CodeIgniter* es ligero, permite una fácil migración al servidor, es altamente customizable y ayuda a reducir la cantidad de código, pero requiere utilizar muchas bibliotecas y está impulsado por una empresa y no por una comunidad. *CakePHP* es de código abierto, también es fácil de usar, permite la reutilización de código, pero su enrutamiento es unidireccional y la documentación no es tan completa comparada con el resto. Por último, *Laravel* es fácil de utilizar, está muy documentado, el enrutamiento permite tener URLs claras y expresivas, tiene una línea de comandos que facilita la creación de controladores y modelos, las migraciones que ofrece permiten un buen mantenimiento de los esquemas de la base de datos mediante el uso de objetos y el acceso a dicha base de datos es muy sencillo. Frente a lo expuesto, *Laravel* [29] es la opción elegida para desarrollar este proyecto.

5.1.2. Diagrama de clases

Los diagramas de clases UML son la forma con la que se denomina al modelo del dominio, que es la representación conceptual del mundo real en un dominio de interés. En ingeniería del software las clases UML se consideran un componente muy importante del proceso de creación de software debido a que aborda mediante un lenguaje sencillo y expresivo, y que conserva suficiente semántica para mantener su comprensión, una manera estándar de poder reflejar las ideas de lo que se va a llevar a cabo [15]. El diagrama de clases del diseño muestra por tanto, de manera gráfica, las especificaciones de las clases e interfaces software y las relaciones entre éstas en una aplicación.

Puesto que el proyecto se va a desarrollar con una arquitectura de tipo MVC (según lo referido en la sección 4.2) se distinguen dos tipos de clases: los modelos y los controladores. Los controladores son las clases encargadas de operar con toda la lógica de la aplicación y quienes asumen la tarea de trasladar a la vista la información que el usuario necesita. Dicha información, almacenada en la base de datos, es gestionada

por los modelos, que a su vez se relacionan con los controladores y que también llevan a cabo esta función intermediadora. En el diagrama de la Figura 44 se aprecian las clases a utilizar con sus correspondientes funciones –métodos de clase– y las relaciones entre ellas.

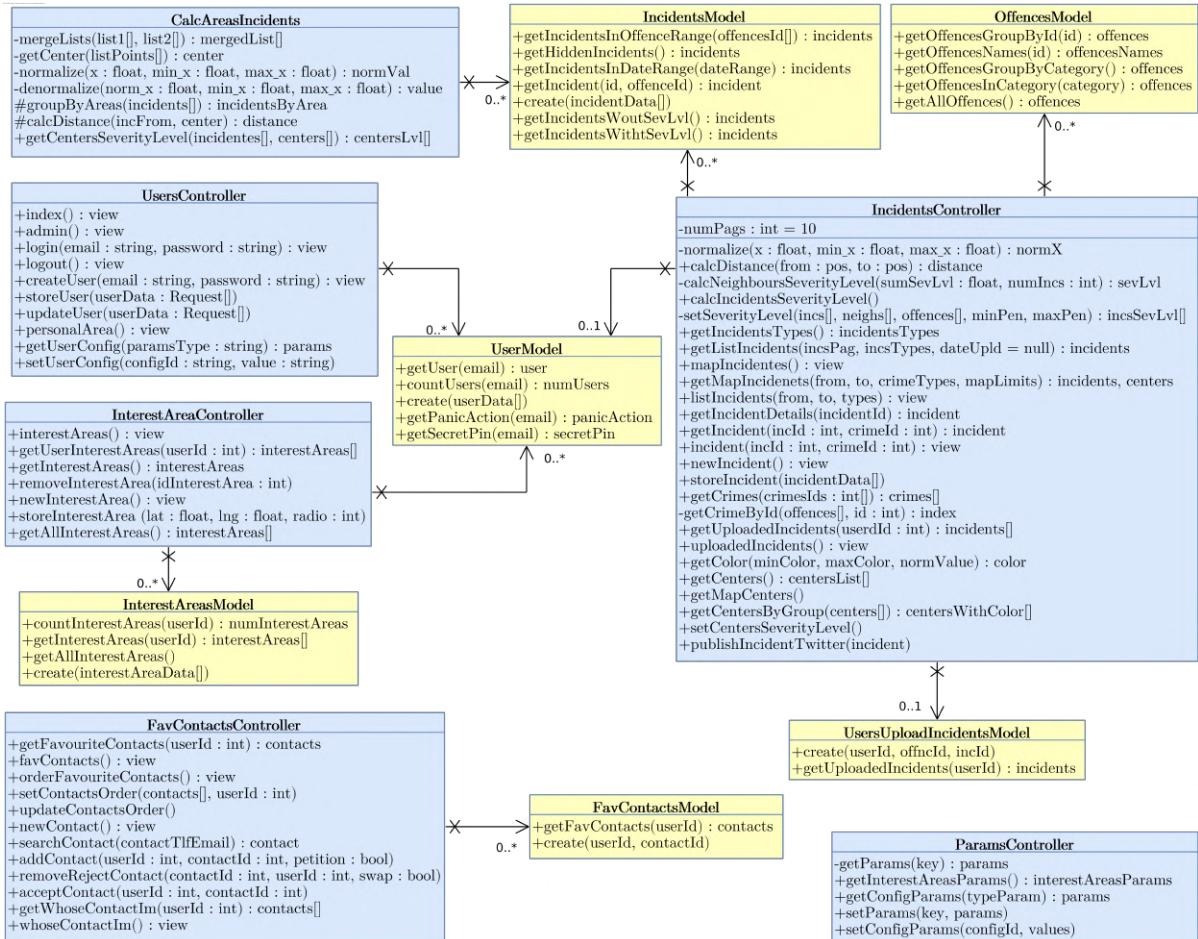


Figura 44: Diagrama de clases

A continuación se explica brevemente qué funciones lleva a cabo cada clase:

Controladores:

- UsersController:** Se encarga de almacenar los datos de nuevos usuarios y actualizar los de los existentes, además de obtener y establecer los parámetros de configuración de estos. Interacciona con las vistas de login, registro, zona personal y página principal de administración.
- IncidentsController:** Se ocupa del alta de incidentes y la obtención de estos según su necesidad según diversos criterios. También, gestiona los distintos delitos. Asimismo se encarga de establecer el nivel de gravedad de los incidentes,

establecer el nivel de las zonas con concentración de estos, asignarles un color para su visualización y publicarlos en RR.SS. En cuanto a su interacción con las vistas, proporciona los datos necesarios a la lista y mapa de incidentes y al formulario de alta de los mismos.

- **FavContactsController:** Es el responsable de obtener los contactos favoritos de cada usuario y de tramitar las peticiones para serlo o eliminarse entre ellos. Igualmente, establece el orden de los contactos para cada usuario cuando estos lo requieren. Se comunica con las vistas de contactos favoritos, ordenación, consulta sobre de quién se es contacto y agregación de uno nuevo.
- **InterestAreaController:** Se hace cargo de la obtención de las zonas de interés de cada usuario y la gestión de estas (añadir y eliminar). Interacciona con las vistas de las zonas de interés y el alta de una nueva.
- **ParamsController:** Obtiene y establece parámetros de configuración globales que son necesarios para el cumplimiento de requisitos de información y por ende para el correcto funcionamiento de la plataforma. Únicamente interactúa con la vista del panel principal del administrador.
- **CalcAreasIncidents:** Aunque no es un controlador como tal, pues se trata de un script independiente, sí que realiza la misma función que otro cualquiera, teniendo como labor la agrupación de las zonas de acumulación de incidentes, contando para ello con métodos que realizan operaciones sobre la estructura de datos.

Modelos:

- **UserModel:** Devuelve un usuario, cuenta cuántos existen en el sistema, crea uno nuevo y obtiene/establece sus parámetros de configuración. El intercambio de datos mencionado se establece con los controladores *UsersController*, *InterestAreaController* e *IncidentsController*.
- **IncidentsModel:** Obtiene incidentes en base a diversos criterios establecidos por las necesidades del usuario y crea nuevos a partir de los datos que lo forman. Se comunica con los controladores *IncidentsController* y *CalcAreasIncidents*.
- **OffencesModel:** Devuelve los delitos del sistema de manera individual o agrupados al controlador *IncidentsController*.
- **UsersUploadIncidentsModel:** Crea una entrada para cada incidente dado de alta por un usuario y devuelve esta misma información a *IncidentsController*.
- **FavContactsModel:** Similar al modelo anterior, crea una relación entre usuarios y devuelve a todos los contactos de un usuario concreto al controlador *FavContactsController*.

- **InterestAreasModel**: Obtiene las zonas de interés de cada usuario, el número de estas y crea una nueva a partir de los datos proporcionados por el usuario al controlador *InterestAreaController*.

Las vistas no son ni pertenecen a clases, por lo que quedan fuera del diagrama anterior; tampoco tienen funciones que poder reflejar en un diagrama de este tipo. Son archivos renderizados por un motor de plantillas capaces de proyectar en la vista los datos del modelo preprocesados por el controlador.

5.1.3. Implementación de la base de datos

La implementación de la base de datos se ha realizado mediante un esquema de migraciones proporcionado por *Laravel* [30]. Las migraciones permiten realizar un control de versiones, y lo más importante, implementar la base de datos al completo sin utilizar el lenguaje SQL directamente, de modo que se trabaja con una “fachada” (*facade*) encargada de la manipulación de tablas en el SGBD. El código de la migración es el siguiente:

```

1 public function up() {
2     Schema::create('users', function (Blueprint $table) {
3         $table->bigIncrements('id');
4         $table->string('email')->unique();
5         $table->string('password');
6         $table->string('nombre');
7         $table->string('apellidos')->nullable();
8         $table->string('dni',10)->unique();
9         $table->dateTime('fecha_nacimiento',0)->nullable();
10        $table->dateTime('fecha_alta',0)->useCurrent()->nullable();
11        $table->tinyInteger('es_admin')->default(0);
12        $table->string('telefono',9)->unique();
13        $table->string('telefono_fijo',9)->nullable();
14        $table->string('pin_secreto',8)->nullable();
15        $table->string('accion_panico')->nullable();
16        $table->decimal('latitud_actual',8,4)->nullable();
17        $table->decimal('longitud_actual',8,4)->nullable();
18        $table->rememberToken();
19        $table->string('api_token')->unique();
20    });
21
22    Schema::create('delitos', function (Blueprint $table) {
23        $table->bigIncrements('id');
24        $table->string('nombre_delito')->unique();
25        $table->string('categoria_delito');
```

```

26     $table->string('descripcion_delito')->nullable();
27     $table->smallInteger('pena_min')->nullable();
28     $table->smallInteger('pena_max')->nullable();
29 });
30
31 Schema::create('incidentes', function (Blueprint $table) {
32     $table->unsignedBigInteger('id');
33     $table->unsignedBigInteger('delito_id');
34     $table->decimal('latitud_incidente',8,4);
35     $table->decimal('longitud_incidente',8,4);
36     $table->string('nombre_lugar')->nullable();
37     $table->datetime('fecha_hora_incidente');
38     $table->text('descripcion_incidente');
39     $table->tinyInteger('afectado_testigo');
40     $table->string('aggravantes')->nullable();
41     $table->decimal('nivel_gravedad', 10, 8)->nullable();
42     $table->tinyInteger('oculto')->default(0);
43     $table->tinyInteger('caducado')->default(0);
44     $table->foreign('delito_id')->references('id')->on('delitos');
45     $table->primary(['delito_id', 'id']);
46 });
47
48 Schema::create('suben', function (Blueprint $table) {
49     $table->unsignedBigInteger('usuario_id');
50     $table->datetime('fecha_hora_sube_incidente')->useCurrent();
51     $table->unsignedBigInteger('delito_id');
52     $table->unsignedBigInteger('incidente_id');
53     $table->foreign('usuario_id')->references('id')->on('users');
54     $table->foreign(['delito_id', 'incidente_id'])
55         ->references(['delito_id', 'id'])->on('incidentes');
56     $table->primary(['usuario_id', 'fecha_hora_sube_incidente']);
57 });
58
59 Schema::create('son_contactos_favoritos', function (Blueprint $table) {
60     $table->unsignedBigInteger('usuario_id');
61     $table->unsignedBigInteger('contacto_favorito_id');
62     $table->tinyInteger('son_contactos')->default(0);
63     $table->tinyInteger('contador')->default(0);
64     $table->smallInteger('orden')->default(99);
65     $table->foreign('usuario_id')->references('id')->on('users');
66     $table->foreign('contacto_favorito_id')
67         ->references('id')->on('users');
68     $table->primary(['usuario_id', 'contacto_favorito_id']);
69 });
70
71 Schema::create('usuarios_zonas_interes', function (Blueprint $table) {

```

```

72     $table->bigIncrements('id');
73     $table->unsignedBigInteger('usuario_id');
74     $table->decimal('latitud_zona_interes',8,4);
75     $table->decimal('longitud_zona_interes',8,4);
76     $table->string('nombre_zona_interes')->nullable();
77     $table->integer('radio_zona_interes');
78     $table->foreign('usuario_id')->references('id')->on('users');
79 });
80
81 Schema::create('notifications', function (Blueprint $table) {
82     $table->uuid('id')->primary();
83     $table->string('type');
84     $table->morphs('notifiable');
85     $table->text('data');
86     $table->timestamp('read_at')->nullable();
87     $table->timestamps();
88 });
89
90 Schema::create('incidents_areas_centers', function (Blueprint $table) {
91     $table->bigIncrements('id');
92     $table->decimal('lat_center',9,6);
93     $table->decimal('lng_center',9,6);
94     $table->decimal('severity_level',9,6);
95     $table->string('color',6)->nullable();
96 });
97 }

```

Después de que la migración sea ejecutada se genera internamente el código SQL necesario para formar la base de datos, cuyo modelo relacional se detalló en la sección 4.1, que es el siguiente:

```

1 CREATE TABLE usuarios(
2     usuario_id bigint AUTO_INCREMENT PRIMARY KEY,
3     email varchar(255) NOT NULL UNIQUE,
4     password varchar(255) NOT NULL,
5     nombre varchar(255) NOT NULL,
6     apellidos varchar(255),
7     dni varchar(10) UNIQUE,
8     fecha_nacimiento DATETIME,
9     fecha_alta DATETIME DEFAULT CURRENT_TIMESTAMP,
10    es_admin tinyint NOT NULL DEFAULT 0,
11    telefono varchar(9) UNIQUE,
12    telefono_fijo varchar(9),
13    pin_secreto varchar(8),
14    accion_panico varchar(255),

```

```

15     latitud_actual decimal(8,4),
16     longitud_actual decimal(8,4),
17     api_token varchar(255) UNIQUE
18 );
19
20 CREATE TABLE usuarios_tienen_zonas_interes(
21     zona_interes_id bigint AUTO_INCREMENT PRIMARY KEY,
22     usuario_id bigint NOT NULL UNIQUE,
23     latitud_zona_interes decimal(8,4) NOT NULL,
24     longitud_zona_interes decimal(8,4) NOT NULL,
25     radio_zona_interes int NOT NULL,
26     nombre_zona_interes varchar(255),
27     FOREIGN KEY (usuario_id) REFERENCES usuarios(usuario_id)
28 );
29
30 CREATE TABLE delitos(
31     delito_id bigint AUTO_INCREMENT PRIMARY KEY,
32     nombre_delito varchar(255) NOT NULL UNIQUE,
33     categoria_delito varchar(255) NOT NULL,
34     descripcion_delito varchar(255),
35     pena_min smallint,
36     pena_max smallint
37 );
38
39 CREATE TABLE incidentes(
40     delito_id bigint NOT NULL,
41     incidente_id bigint NOT NULL,
42     latitud_incidente decimal(8,4) NOT NULL,
43     longitud_incidente decimal(8,4) NOT NULL,
44     fecha_hora_incidente DATETIME NOT NULL,
45     descripcion_incidente text NOT NULL,
46     afectado_testigo tinyint NOT NULL,
47     agravantes varchar(255),
48     nivel_gravedad decimal(10,8),
49     oculto tinyint NOT NULL DEFAULT 0,
50     caducado tinyint NOT NULL DEFAULT 0,
51     FOREIGN KEY (delito_id) REFERENCES delitos(delito_id),
52     PRIMARY KEY (delito_id, incidente_id)
53 );
54
55 CREATE TABLE son_contactos_favoritos(
56     usuario_id bigint,
57     contacto_favorito_id bigint,
58     contador tinyint NOT NULL DEFAULT 0,
59     son_contactos tinyint NOT NULL DEFAULT 0,
60     orden smallint NOT NULL DEFAULT 99,

```

```

61      FOREIGN KEY (usuario_id) REFERENCES usuarios(usuario_id),
62      FOREIGN KEY (contacto_favorito_id) REFERENCES usuarios(usuario_id),
63      PRIMARY KEY (usuario_id, contacto_favorito_id)
64 );
65
66 CREATE TABLE suben(
67     usuario_id bigint,
68     fecha_hora_sube_incidente DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
69     delito_id bigint NOT NULL,
70     incidente_id bigint NOT NULL,
71     FOREIGN KEY (usuario_id) REFERENCES usuarios(usuario_id),
72     FOREIGN KEY (delito_id, incidente_id) REFERENCES incidentes(delito_id,
73     ↳ incidente_id),
74     PRIMARY KEY (usuario_id, fecha_hora_sube_incidente)
75 );
76
76 CREATE TABLE centros_zonas_incidentes(
77     centro_zona_id bigint NOT NULL PRIMARY KEY,
78     latitud_centro decimal(9,6) NOT NULL,
79     longitud_centro decimal(9,6) NOT NULL,
80     nivel_gravedad_centro decimal(9,6) NOT NULL,
81     color_centro varchar(6)
82 );

```

5.1.4. Desarrollo

Para el servidor de desarrollo he utilizado un servicio web LAMP (Linux, Apache, MySQL y PHP) [31], para el cual el software elegido ha sido XAMPP [32], un paquete de software libre que permite un desarrollo local del proyecto y testeo del mismo antes de subirlo a un servidor en producción.

Gestión de usuarios

Tal y como se presentó en el diagrama de clases (sección 5.1.2), hay un controlador específico para manejar toda la lógica relativa a los usuarios. De este controlador cabe destacar las funciones encargadas del registro de nuevos usuarios, actualización de sus datos y el *login* en el sistema.

Registro

Para proceder al almacenamiento de un nuevo usuario en el sistema lo primero que hay que hacer es validar los datos obtenidos del formulario de registro. Para ello *Laravel*

proporciona una función de validación donde basta con incluirle como parámetro un array asociativo ciertas reglas sobre validaciones a cumplir:

```
1 $datos = $request->validate([
2     'email' => 'bail|required|min:7|max:255',
3     'password' => 'bail|required|min:8',
4     'nombre' => 'bail|required|min:2|max:255',
5     'apellidos' => 'bail|required|min:2|max:255',
6     'dni' => 'bail|required|min:9|max:10',
7     'fecha_nacimiento' => 'bail|required|date|before:-12years',
8     'telefono' => 'bail|required|regex:/([6-7])[0-9]{8}/'
9 ]);
```

A continuación se procede al hashing de la contraseña, para lo cual *Laravel* proporciona una faceta que hace uso de *bcrypt* [33], una función de hashing basada en el cifrado de *Blowfish* [34]. Este método lleva incorporado un valor denominado *salt*, que se utiliza para crear un fragmento aleatorio en el hash de modo que evite crear para dos contraseñas iguales el mismo hash final, previniendo así ataques como los basados en diccionarios.

Efectivamente, el hashing se realiza en el servidor, lo cual supone enviar la contraseña “plana” al mismo. El hashing en el cliente sólo tiene sentido si no se confía en el servidor por ser este un lugar hostil y objeto de los atacantes y por no querer “enseñarle” la contraseña (la que el humano recuerda). La desconfianza proviene de que los usuarios no quieren que nadie conozca su contraseña para una aplicación concreta, y esto se debe a que generalmente usan la misma para todo. Esto se podría subsanar si cada usuario tomara alguna medida como:

1. Utilizar una contraseña distinta para cada sitio.
2. Hacer un hash previo de su contraseña maestra a partir de un parámetro –que puede ser cualquier palabra sencilla que el usuario recuerde fácilmente– con el cual generar una nueva contraseña hash segura.

Volviendo a por qué carece de sentido enviar la contraseña con un hash realizado en el cliente, se parte de la base de querer realizar hashing a la contraseña por desconfianza hacia el servidor, que es el mismo que envía al usuario el código JavaScript para aplicar el algoritmo de hashing a la contraseña, por lo que se incurriría en la misma hostilidad del servidor atacado en su caso.

Si el atacante consigue acceder al servidor y obtener una copia de la base de datos es muy probable que utilice un diccionario de contraseñas para obtener aquellas que son más débiles –por lo que es importante obligar al usuario a que introduzca una contraseña segura–, y para lo cual se aplica *bcrypt* como ya se ha explicado.

Otro motivo destacable para evitar el hashing en el lado del cliente es la lentitud de éste. Javascript es mucho más lento aplicando algoritmos de hashing que otros lenguajes como C o incluso Java. Además, aplicar hashing en el lado del cliente transforma la contraseña hash en la contraseña “actual”, siendo el algoritmo aplicado un medio mnemotécnico para la contraseña, aunque este punto es el menos relevante teniendo en cuenta que ni la velocidad ni la transformación de la contraseña afectan en gran medida al sistema. El punto crítico se encuentra en que un atacante intervenga la comunicación entre el cliente y el servidor, y ante esto la manera más efectiva de tomar una medida de precaución efectiva es enviar la contraseña al servidor a través de un túnel HTTPS y aplicarle el algoritmo de hashing en el servidor [35].

Tras realizar el hashing de la contraseña, se crea un token para la API con una cadena de texto aleatoria de 255 caracteres que servirá para identificar al usuario que utilice la aplicación móvil.

Actualización de datos

Al igual que en el registro, el primer paso es validar los datos del usuario. Después hay que identificar si se ha producido una modificación del email, que es la clave de la sesión en la web, para proceder a actualizarlo. Al igual que con el email, el teléfono móvil requiere comprobar que ambos, en caso de ser modificados, no existen en otro campo de la base de datos puesto que se son valores únicos. Realizadas estas comprobaciones, si los datos son válidos, se procede a actualizárselos al usuario.

Gestión de incidentes

La gestión de incidentes es sin duda la parte más relevante del proyecto en cuanto a implementación se refiere.

Un apartado importante en el alta de incidentes es el mecanismo utilizado para obtener de manera automática el nombre del lugar donde se producen. Para ello se hace uso de *Nominatim* [36], un servicio que proporciona OpenStreetMap para obtener una lista estructurada con datos relativos a una posición geográfica concreta. Esta función permitirá dar información legible sobre dónde suceden los incidentes y también su posterior agrupación por zonas. Dado que *Nominatim* provee los datos con direcciones demasiado extensas o con campos que aparecen o no según el lugar, es conveniente aplicar un método previo a su almacenamiento para que el resultado sea útil y claro. El resultado proporcionado será el nombre del municipio en el caso de localidades pequeñas o del nombre del distrito junto al del municipio para ciudades de mayor tamaño, acompañado de la provincia donde se encuentra. A continuación se muestra el código en JavaScript utilizado para tal fin:

```

1 placeName = ((typeof(data.address.locality)!==
2   ↵  'undefined')?data.address.locality +", ":"")+
2 ((typeof(data.address.city_district)!==
2   ↵  'undefined')?data.address.city_district +", ":"")+
3 ((typeof(data.address.village)!== 'undefined')?data.address.village:"")+
4 ((typeof(data.address.town)!== 'undefined')?data.address.town:"")+
5 ((typeof(data.address.city)!== 'undefined')?data.address.city:"")+
6 ((typeof(data.address.county)!== 'undefined')?" ("+data.address.county
7   ↵  +") ":"");
8 $('#nombre_lugar').val((placeName == "")?placeName:data.address.country);

```

Tras dar de alta un incidente, este se publica automáticamente en la red social Twitter en forma de *tweet*. Esta tarea se realiza mediante la API proporcionada para desarrolladores [37], que proporciona claves de acceso para enlazar la aplicación con una cuenta de usuario en la red social y automatizar la publicación de contenido.

Desde el controlador de incidentes se invoca al método encargado para crear el *tweet* tras crear el incidente en la base de datos. A partir de los detalles del incidente se elabora un mensaje comprensible desde el punto de vista semántico y se envía a la API.

```

1 function publishIncidentTwitter($incident) {
2     // Preparación de la conexión con las claves de la API
3     $connection = new TwitterOAuth(
4         getenv('CONSUMER_KEY'),
5         getenv('CONSUMER_SECRET'),
6         getenv('ACCESS_TOKEN'),
7         getenv('ACCESS_TOKEN_SECRET')
8     );
9
10    // Construcción del mensaje a partir de los datos del incidente
11    $status = ...
12
13    $post_tweets = $connection->post("statuses/update", [
14        "status" => $status,
15        "lat" => $incident['latitud_incidente'],
16        "long" => $incident['longitud_incidente']
17    ]);
18 }

```

El siguiente apartado de relevancia es la asignación del nivel de gravedad a cada

uno de los incidentes, llevado a cabo de manera periódica por parte de un *cron* que ejecuta la tarea implementada para tal efecto según el algoritmo expuesto en la sección 4.3.1 y cuya implementación es la siguiente:

```

1 foreach ($incidents as $incident) {
2     $minPenalty = $basePenalty = $offence['pena_min'];
3     $maxPenalty = $offence['pena_max'];
4     $diffPenalty = $maxPenalty - $minPenalty;
5
6     $aggravs_accum = 0.0;
7     if (!empty($aggravs)) {
8         if (in_array('1', $aggravs))      // disfraz
9             $aggravs_accum += 0.175;
10        if (in_array('2', $aggravs))    // abuso de superioridad
11            $aggravs_accum += 0.225;
12        if (in_array('3', $aggravs))    // sufrimiento inhumano
13            $aggravs_accum += 0.3;
14        if (in_array('4', $aggravs))    // Discriminación
15            $aggravs_accum += 0.3;
16    }
17
18    if ($aggravs_accum > 0.0)
19        $basePenalty += $diffPenalty * $aggravs_accum;
20
21    $ownSevLvl = $this->normalize($basePenalty, $minGlobalPenalty,
22    ↳ $maxGlobalPenalty);
23
24    $sumNeighSevLvl = 0.0;
25    $numNeighIncs = 0;
26    foreach($neighIncidents as $neighInc) {
27        if ($incident['id'] != $neighInc['id']) {
28            $latFrom = $incident['lat'];
29            $lngFrom = $incident['long'];
30            $latTo= $neighInc['lat'];
31            $lngTo = $neighInc['long'];
32            if ($this->calcDistance($latFrom, $lngFrom, $latTo, $lngTo) <
33                ↳ 550) {
34                $sumNeighSevLvl += $neighInc['nivel_gravedad'];
35                $numNeighIncs++;
36            }
37        }
38        $neighSevLvl = $numNeighIncs > 0 ?
39            $this->calcNeighboursSeverityLevel($sumNeighSevLvl, $numNeighIncs) :
40            ↳ 0.0;

```

```

40 $incidentSeverityLevel = 0.6 * $ownSevLvl + 0.4 * $neighSevLvl;
41
42 $incident['nivel_gravedad'] = $incidentSeverityLevel;
43 array_push($incidentsWithSeverityLevel, $incident);
44 }

```

Con todos los incidentes con un nivel de gravedad asignado se realiza una segunda iteración igual a la anterior con todos los incidentes para aplicar la influencia de los incidentes aledaños a todo el conjunto por igual, debido a que en la primera iteración puede haber incidentes que tengan nuevos vecinos y que no hayan sido considerados en la primera vuelta por ser todos ellos nuevos. Además se recalculará el nivel de gravedad de los previamente existentes si estos se encuentran en el radio de influencia de alguno nuevo.

Para finalizar este tarea se actualiza la tabla de incidentes de la base de datos añadiendo el nivel de gravedad a los incidentes nuevos y actualizando aquellos que han modificado el que tenían asignado.

Tras haber calculado el nivel de gravedad, el *cron* lanza otro proceso cuya labor es ejecutar la tarea encargada de agrupar los incidentes y asignar un nivel de gravedad a cada zona. El procedimiento se lleva a cabo con un *script* de Python que hace uso del algoritmo de *clustering* [38] planteado en la sección 4.3.2. Para su implementación hay que destacar ciertos detalles que no han sido mencionados en el diseño:

- La primera agrupación no automática de incidentes según el nombre de zona se realiza según el campo “nombre de lugar” de la tabla de incidentes de la base de datos. Esto se hace porque en caso de aplicar el algoritmo sobre el conjunto de datos en bruto el resultado es nefasto, debido a que no es capaz de determinar y abstraer correctamente lugares diferenciados.
- La normalización en un intervalo [0, 10] se debe a que en unas primeras pruebas con las propias coordenadas el resultado era de un único cluster, probablemente porque la variación del valor de las coordenadas es ínfimo dentro de una misma zona; con una normalización en un intervalo [0, 1] el resultado no era mucho más esperanzador. Finalmente con un intervalo [0, 10] sí se obtienen varios grupos diferenciados.
- El parámetro *bandwidth* se estima de manera automática con una función de estimación de *scikit-learn* de Python y se le proporciona al algoritmo para proceder a su ejecución.

El código en Python relativo a los pasos más relevantes del algoritmo (*clustering* y desnormalización) es el siguiente, mostrado también gráficamente en la Figura 45:

```

1 norm_dataset = merge(norm_lat, norm_lng)
2 X = np.array(norm_dataset)
3
4 bandwidth = estimate_bandwidth(X, random_state=82202551, n_jobs=-1)
5 area_centers = []
6
7 if (bandwidth != 0.0):
8     ms = MeanShift(bandwidth=bandwidth, bin_seeding=True).fit(X)
9     area_centers = ms.cluster_centers_
10
11 for c in area_centers:
12     de_n_center = [denormalize(c[0], min_lat, max_lat), denormalize(c[1],
13         ↪ min_lng, max_lng)]
14     center_sev_lvl = get_centers_sev_lvl(incidents_by_area[key],
15         ↪ de_n_center)
16     if (len(center_sev_lvl) > 0):
17         centers.append( center_sev_lvl )

```

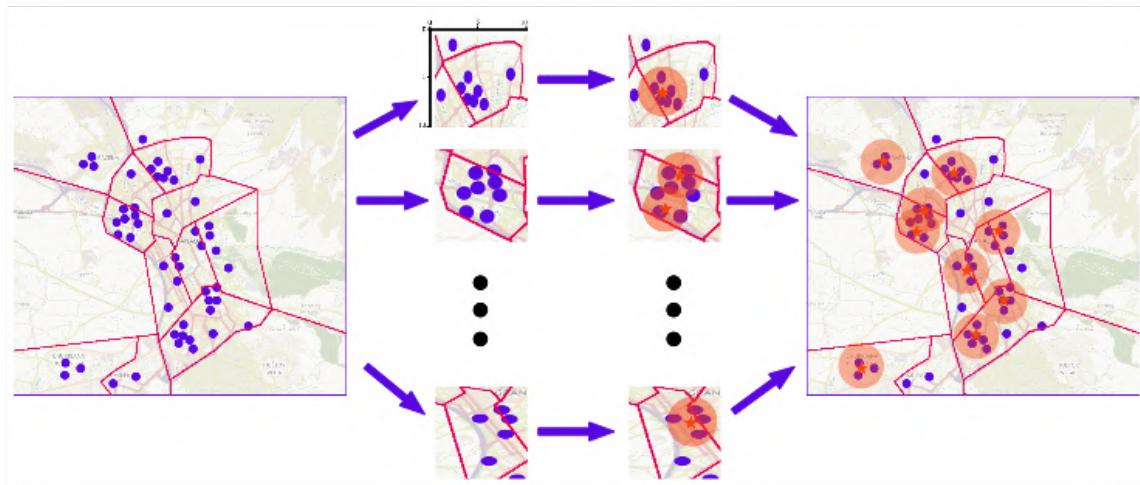


Figura 45: Proceso gráfico del algoritmo de clustering

Una vez obtenidos los centros de las zonas con concentración de incidentes es oportuno mostrar al usuario de manera gráfica en el mapa de incidentes cuáles son estas zonas. Además de esto, se le aplica a cada zona un color en función del nivel de gravedad asociado aplicando el algoritmo visto en la sección 4.3.3.

Se define una escala de seis colores: verde, amarillo, naranja, rojo, granate y negro. Una zona con menor nivel de gravedad tendrá asignado un color próximo al verde y, cuanto mayor sea, más próximo al negro lo obtendrá.

Al haber seis colores, a un incidente le corresponderá la primera franja si su valor normalizado está en el intervalo [0, 0.2), la segunda si está en el [0.2, 0.4), y así sucesivamente. El código en PHP relativo a su implementación es:

```

1  /**
2   * Obtiene el color asociado según su valor en un intervalo
3  */
4  function getColor($minColor, $maxColor, $normValue) {
5      $difColor = $maxColor - $minColor;
6      $valColor = $minColor + ($difColor * $normValue);
7      return ($valColor < 16) ? '0'.dechex($valColor) : dechex($valColor);
8 }
9
10 /**
11  * Calcula a partir de los centros de las zonas con incidentes,
12  * según su nivel de gravedad asociado, el color que le corresponde
13  * en una escala que refleja el nivel de gravedad de forma visual
14 */
15 function getCentersByGroup($centers) {
16     $centersWithColor = array();
17     $minLevel = floatval(min(array_column($centers, '2')));
18     $maxLevel = floatval(max(array_column($centers, '2')));
19
20     foreach ($centers as $center) {
21         $normCenter = $this->normalize($center[2], $minLevel, $maxLevel);
22         if ($normCenter >= 0.0 && $normCenter < 0.2) {
23             $normInterval = $this->normalize($normCenter, 0.0, 0.2);
24             $hexColor = $this->getColor(hexdec('80'), hexdec('FF'),
25             $normInterval);
26             $hexColor2 = $this->getColor(hexdec('B0'), hexdec('F0'),
27             $normInterval);
28             $defColor = $hexColor.$hexColor2."00";
29         } elseif ($normCenter >= 0.2 && $normCenter < 0.4) {
30             $normInterval = $this->normalize($normCenter, 0.2, 0.4);
31             $hexColor = $this->getColor(hexdec('F0'), hexdec('80'),
32             $normInterval);
33             $defColor = "FF".$hexColor."00";
34         } else {
35             $normInterval = $this->normalize($normCenter, 0.8, 1.0);
36             $hexColor = $this->getColor(hexdec('80'), hexdec('00'),
37             $normInterval);
38             $defColor = $hexColor."0000";
39         }
40     }
41 }

```

```

39     $center['color'] = $defColor;
40     array_push($centersWithColor, $center);
41 }
42
43 return $centersWithColor;
44 }

```

Cuando todos los centros tienen un color asignado, se almacenan en la base de datos, que será actualizada cada día una vez actualizado el nivel de gravedad de los incidentes.

Gestión de contactos favoritos

La función principal de la gestión de contactos es mantener las relaciones entre distintos usuarios del sistema y manteniendo un control sobre cuántas veces se solicita ser contacto de otros usuarios para prevenir cierto hostigamiento por parte de alguien si fuera su intención. Adicionalmente sólo se permite la búsqueda de otros usuarios por email o número de teléfono con la intención de limitar la red de contactos a un ámbito cercano de cada uno.

Visualización de mapas

La representación de incidentes, zonas de acumulación de estos y zonas de interés se realizado mediante el uso de Leaflet [39], que es una librería de JavaScript de código abierto que permite la visualización de mapas de OpenStreetMap tanto para web como para aplicaciones móviles.

Esta herramienta permite situar los incidentes o zonas de interés en el mapa a partir de sus coordenadas. En el caso de obtener varios puntos previamente almacenados en la base de datos, estos se obtienen mediante AJAX y posteriormente se itera sobre el resultado para su representación:

```

1 $.ajax({
2   url: resourceUrl,
3   data: data
4   type: 'get',
5   success: function (response) {
6     let jsonResponse = JSON.parse(response);
7     let result = jsonResponse.incidents;
8
9     $.each(result, function(index, value) {

```

```
10         L.marker([value.latitud, value.longitud])
11             .addTo(incidentsLayerGroup);
12     });
13 }
14 );
```

5.2. Desarrollo de aplicación móvil

5.2.1. API

El desarrollo de la aplicación móvil no necesita de gran desarrollo extra en la parte del servidor, puesto que con lo construido para la parte de la web se puede extender un servicio para API REST que permita integrar en el proyecto el software de la aplicación móvil.

Una API (*Application Programming Interface*) es un conjunto de funciones, protocolos y métodos que proporcionan una capa de abstracción para la programación de aplicaciones, utilizando el software del proveedor como base para su funcionamiento.[40] El acrónimo REST significa “transferencia de estado representacional” (del inglés *representational state transfer*), esto es, una interfaz que utilice HTTP (*Hyper-text Transfer Protocol*). Una API REST es por tanto una interfaz entre sistemas que utiliza HTTP como protocolo en el canal de comunicación de las operaciones.

En este caso la comunicación la entablan el servidor y la aplicación final mediante el uso de JSON (*JavaScript Object Notation*), un formato de texto sencillo y estructurado ideal para comunicaciones alternativo a XML.

La realización de la API ha requerido de controladores propios que realizan llamadas a los ya existentes (con el código modularizado para su reutilización) con el fin de obtener la información necesaria o realizar acciones sobre el modelo de datos. Existen dos diferencias fundamentales con respecto a lo realizado para la web:

- La verificación de usuarios para las peticiones se realiza mediante un *middleware* que comprueba que el token del usuario sea válido.
- Adaptar las respuestas a formato JSON tal y como se ha descrito en cuanto al modo de comunicación.

La primera parte de la implementación pasa por el sistema de enrutamiento (*routing*) de *Laravel*. Un ejemplo es:

```
1 Route::get('/api/get_fav_contacts', 'API\FavContactsController@getFavContacts')
2     ->middleware('auth:api');
```

De esta manera, cuando se instancie el método definido en la ruta tan sólo será necesario invocar al método de autenticación proporcionado por el *framework*, tal y como se puede ver en el siguiente fragmento de código correspondiente al enrutamiento anterior:

```
1 function getFavContacts(Request $request) {
2     $user = Auth::user();
3
4     if (!is_null($user)) {
5         $favContactsCtrl = new FavContactsCtrl();
6         $contacts = $favContactsCtrl->getFavouriteContacts($user['id']);
7
8         return response()
9             ->json([
10                 'status' => 'success',
11                 'favContacts' => $contacts
12             ], 200);
13     }
14     return response()
15         ->json([
16             'status' => 'error',
17             'message' => 'El usuario no existe!'
18         ], 401);
19 }
```

5.2.2. Entorno de desarrollo

Actualmente más del 99 % de las aplicaciones móviles utilizan como sistema operativo Android o iOS (Figura 46). Decantarse por desarrollar una aplicación nativa para uno o ambos sistemas operativos puede resultar costoso, especialmente en tiempo, pero existen tecnologías híbridas que permiten desarrollar para sendas plataformas simplificando de este modo el proceso. Las aplicaciones híbridas pierden el principal beneficio de las nativas, que es el rendimiento, pero en contraposición es mucho más lo que se gana debido al ahorro de desarrollo y posterior mantenimiento.

Las tecnologías híbridas se fundamentan en el uso de HTML5, CSS y JavaScript, que son tecnologías ya descritas en la sección 5.1.1 de lenguajes de desarrollo web. Una vez programada la aplicación el resultado no se renderiza sobre un *WebView* del

dispositivo, sino que se compila al entorno nativo con un intermediario y finalmente se renderiza con los componentes nativos de Android/iOS. En cuanto al resto de componentes nativos de los dispositivos móviles es posible la conexión con estos, de modo que se puede hacer uso de los sensores, GPS, cámara, etc. a través del uso de plugins.

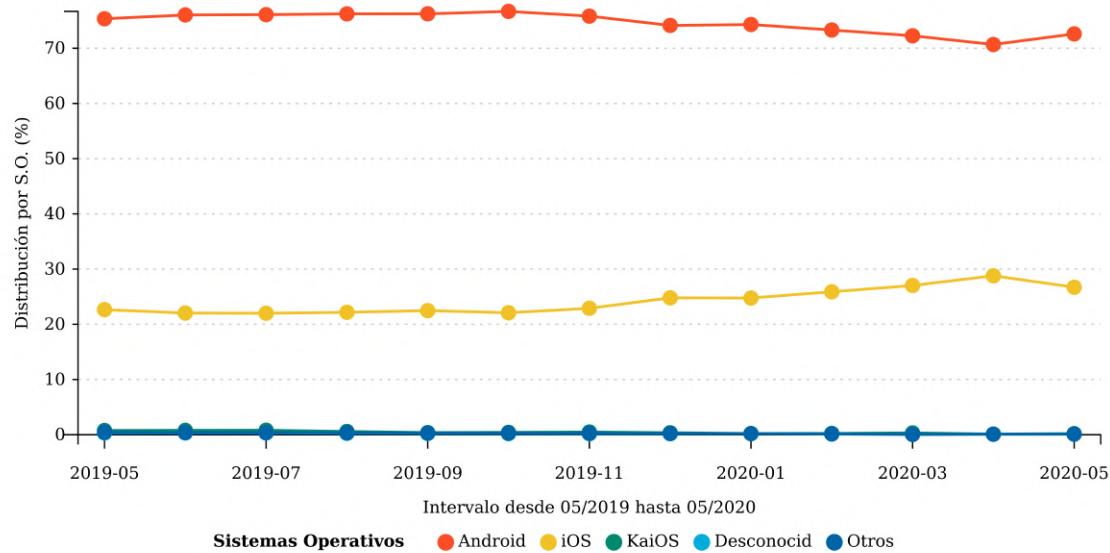


Figura 46: Sistemas operativos en dispositivos móviles en el mercado [41]

Los dos principales frameworks para desarrollo de aplicaciones híbridas son React Native [42] e Ionic [43]. El primero es un entorno de código abierto impulsado por Facebook en 2015, siendo dicha empresa su mayor baza competitiva. Ionic es muy similar, pero tiene una mayor comunidad detrás que da soporte, a la par que gana en sencillez, accesibilidad y eficiencia, siendo estos puntos los que me decantan por dicho framework tras probar ambos.

Una vez escogido Ionic no cabe mucha elección en cuanto a los lenguajes a usar. Si bien es posible escoger entre Angular y React, las diferencias entre ambas bibliotecas apenas ofrecen ventajas o inconvenientes de una sobre la otra. La principal diferencia es que Angular es un framework para TypeScript y React para JavaScript.

La mayor parte de usuarios en los foros de Ionic utilizan Angular, así que opto por este, para lo cual es necesario usar, como ya se ha especificado, TypeScript. Este es un lenguaje nuevo, lanzado en 2018 por Microsoft, que esencialmente es una evolución de JavaScript a la que añade tipos estáticos y objetos basados en clases, por lo que su aprendizaje no entraña gran complejidad una vez conocido JS.

Mientras que TypeScript se utiliza para programar la lógica de la aplicación, la interfaz de usuario utiliza HTML y CSS. Ionic proporciona multitud de etiquetas

propias, atributos, clases y componentes adicionales con el fin de obtener una interfaz elegante sin requerir amplios conocimientos de diseño.

5.2.3. Desarrollo

Estructura

El desarrollo de la aplicación con Ionic se basa en implementar, para cada pantalla, lo que en el lenguaje interno del *framework* se denomina página (*page*). Las páginas se componen de varios archivos cada una, de los que cabe destacar tres:

- **.page.ts**: comportamiento de la página (funciones y eventos), definición de variables y declaración de componentes externos
- **.page.html**: parte visual
- **.page.scss**: estilos de la página

Las páginas se sustentan en servicios (*services*) que, como su propio nombre indica, se encargan de proveer servicios a distintas páginas, de modo que suelen implementar funcionalidades tales como la comunicación con el servidor, manejo de funciones comunes a varias páginas, utilidades del desarrollador, etc.

Comunicación con la API

Una de las principales características del desarrollo de la aplicación móvil es el modo en el que se realiza la comunicación con el servidor, a través de los “observables”, una herramienta de Angular basada en la programación reactiva, que se diferencia de la programación tradicional en la utilización de flujos de datos (*streams*) asíncronos.

El patrón observable trata de producir los eventos de los flujos de datos y consumirlos después mediante eventos que cambian (lo observado). En el caso de la aplicación son utilizados para realizar llamadas HTTP al servidor (eventos) y recibir una respuesta del mismo, de modo que se “observa”/espera hasta recibir una contestación. Para ello las respuestas JSON de la API están implementadas con códigos HTTP y mensajes de éxito/error, de modo que la aplicación conozca en cada llamada si la operación se ha ejecutado correctamente.

Continuando con las llamadas al servidor, es conveniente desarrollar el método de autenticación en la aplicación móvil. Como ya se ha comentado anteriormente, se hace uso de un *token* que se asigna al usuario cuando crea su cuenta. En el uso de la app,

tras iniciar sesión correctamente con su usuario y su contraseña, el usuario recibe su *token* y queda guardado en el almacenamiento interno del dispositivo para ser utilizado en las subsiguientes peticiones. Un ejemplo estándar del proceso de petición sería:

```
1 /**
2  * Obtiene el token del almacenamiento interno (storage)
3  * y se suscribe al evento lanzado, esperando una respuesta
4  * asíncrona del servidor
5 */
6 func() {
7     this.storage.get('api_token').then(user => {
8         if(user !== null) {
9             this.service.getResource(user).subscribe(response => {
10                 if (response.status === 'success') {
11                     // Action;
12                 }
13             });
14         }
15     });
16 }
17 /**
18  * Solicita un determinado recurso al servidor y devuelve
19  * la respuesta como un tipo de dato Observable
20 */
21 getResource(user): Observable<any> {
22     let params = new HttpParams();
23     params = params.append('api_token', user);
24
25     return this._http.get<any>(
26         SERVER_URL + '/api/get_resource',
27         {params: params}
28     );
29 }
```

Botón del pánico

El botón del pánico es el mecanismo que dispone el usuario para notificar a sus contactos favoritos de una posible situación de peligro. Para ello, previamente, habrá que haber establecido una acción de pánico para poder llevarla a cabo. Las opciones disponibles son enviar la ubicación en tiempo real o realizar una llamada telefónica al primer contacto favorito.

Una vez accionado el botón del pánico aparecerá una cuenta atrás de cinco segundos donde el usuario puede cancelar la acción antes de que el contador llegue a cero. El proceso que se realiza es por tanto distinto según el caso.

Envío de ubicación:

Se hace uso del componente GPS nativo del dispositivo mediante un plugin de Ionic llamado *Geolocation*, que hace uso de un método llamado `watchPosition()` que es llamado cada vez que detecta un cambio en la posición del dispositivo móvil. Debido a que este método es llamado constantemente, se restringe en código la actualización de la posición para no saturar la comunicación con la base de datos, de modo que se realiza cuando esta varía en latitud o longitud en más de 0.0002º (22 metros aproximadamente). Asimismo se notifica a todos los contactos favoritos de la activación del botón del pánico y se les permite consultar la ubicación del usuario en tiempo real.

```
1 shareLocation() {
2     this.watch = this.geolocation.watchPosition({
3         enableHighAccuracy: true
4     }).subscribe((data) => {
5         this.updateLocation(data.coords);
6     });
7
8     // Notifica a los usuarios de la situación de peligro
9     // y les habilita poder consultar la ubicación
10    this.shareLocationContacts();
11 }
12
13 updateLocation(coords: Coordinates) {
14     let difLat = Math.abs(this.latitude - coords.latitude);
15     let difLng = Math.abs(this.longitude - coords.longitude);
16
17     if(difLat > 0.0002 || difLng > 0.002) {
18         this.storage.get('api_token').then(user => {
19             if(user !== null) {
20                 this.updateLocationApi(user, coords)
21                     .subscribe(response => {
22                         if(response.status === 'success') {
23                             this.latitude = coords.latitude;
24                             this.longitude = coords.longitude;
25                         }
26                     });
27             }
28         });
29     }
30 }
```

Llamada telefónica:

Haciendo uso de la llamada telefónica, como es lógico, no se puede contactar por esta vía con todos los contactos favoritos de manera simultánea, por lo que se selecciona al primero de todos según el orden establecido. Al igual que con la ubicación, se hace uso de un componente nativo del dispositivo mediante un plugin, *CallNumber*, que por medio del método `callNumber(numTlf)`, cuyo argumento es el número de teléfono del contacto favorito, se llama automáticamente a dicho contacto. Como el número de teléfono es un campo obligatorio en el registro de los usuarios, este se obtiene del contacto en cuestión sin necesidad de que el usuario lo solicite en ningún momento.

5.3. Despliegue del proyecto

Todo proyecto web requiere ser desplegado en un servidor tras terminar la fase de desarrollo, pasando así a un ambiente de producción para que los usuarios puedan hacer uso de la aplicación. En esta sección se detalla el proceso a seguir para su puesta en producción. Para ello se ha hecho uso de un *Droplet* de DigitalOcean [44], que es una máquina virtual escalable proporcionada por este proveedor de servidores.

La máquina virtual creada cuenta con una distribución Ubuntu 18.04 (LTS) x64, donde será instalado el servidor con la misma infraestructura que en el ambiente de desarrollo (LAMP). El primer paso es instalar:

- PHP (versión 7.2) y varias dependencias de este.
- Composer: sistema de gestión de paquetes de PHP.
- MySQL Server: servidor de bases de datos SQL.
- phpMyAdmin: herramienta PHP para administración de MySQL.

```
$ sudo apt-get update
$ sudo apt-get dist-upgrade
$ sudo apt-get install unzip composer git
$ sudo apt-get install php7.2-xml php7.2-curl php7.2-mbstring
$ sudo apt-get install mysql-server
$ sudo apt-get install phpmyadmin
```

A continuación se crea un usuario en la base de datos y se le proporcionan permisos:

```
$ mysql -u root -p
> create database laravel;
> GRANT ALL ON laravel.* TO 'david'@'localhost' IDENTIFIED BY
→   '<password>';

```

El siguiente paso es descargar el proyecto en el directorio del servidor web (`/var/www`) e instalar todas las dependencias del proyecto utilizando `composer`. También se le conceden permisos de lectura y escritura al directorio `storage` del proyecto:

```
$ git clone https://github.com/DavidChicharro/Trabajo-Fin-Grado.git  
$ composer install  
$ sudo chown -R www-data: storage  
$ sudo chmod -R 755 storage
```

Se activa el módulo `rewrite` de *Apache* para poder modificar el directorio raíz de la carpeta `public`, que almacenará los recursos estáticos de la web, y se establece la ruta donde se encontrará:

```
$ a2enmod rewrite  
$ vim /etc/apache2/sites-enabled/000-default.conf  
$ systemctl restart apache2
```

En este punto el servidor ya es funcional en producción, pero falta migrar la base de datos.

```
$ php artisan migrate --seed
```

Por último se instala un certificado SSL que permita utilizar el protocolo HTTPS, tal y como se indicó en la sección 5.1.4 para poder garantizar la seguridad en las comunicaciones entre el cliente y el servidor. Para esto se hace uso de Certbot [45], una herramienta de software libre que permite la instalación manual de certificados SSL utilizando *Let's Encrypt*, que es una autoridad de certificación.

```
$ sudo add-apt-repository ppa:certbot/certbot  
$ sudo apt-get update  
$ sudo apt-get install python-certbot-apache  
$ sudo certbot --apache -d kifungo.live -d www.kifungo.live
```

6. Conclusiones

6.1. Temporización real

En esta sección se analiza la temporización real de la ejecución de este trabajo en comparación con la planificación expuesta en la sección 2.2. Además del análisis se muestra el diagrama de Gantt asociado al desarrollo real en la Figura 47.

La fase de análisis fue llevada a cabo en un tiempo menor al inicialmente planificado, si bien la realización de los diagramas de secuencia no estuvo contemplada, no prolongó la duración de esta etapa, dado que el análisis de requisitos duró menos de lo programado.

El paro previsto por el período de evaluación ordinario se extendió más de lo previsto –más de dos meses en lugar de uno–, pero la corta duración final de la fase de diseño, al igual que ocurrió con el plazo proyectado para el aprendizaje de nuevas herramientas –todo esto añadido a la mayor brevedad de la fase de análisis–, permitió comenzar con la implementación del proyecto en la fecha estipulada en un principio.

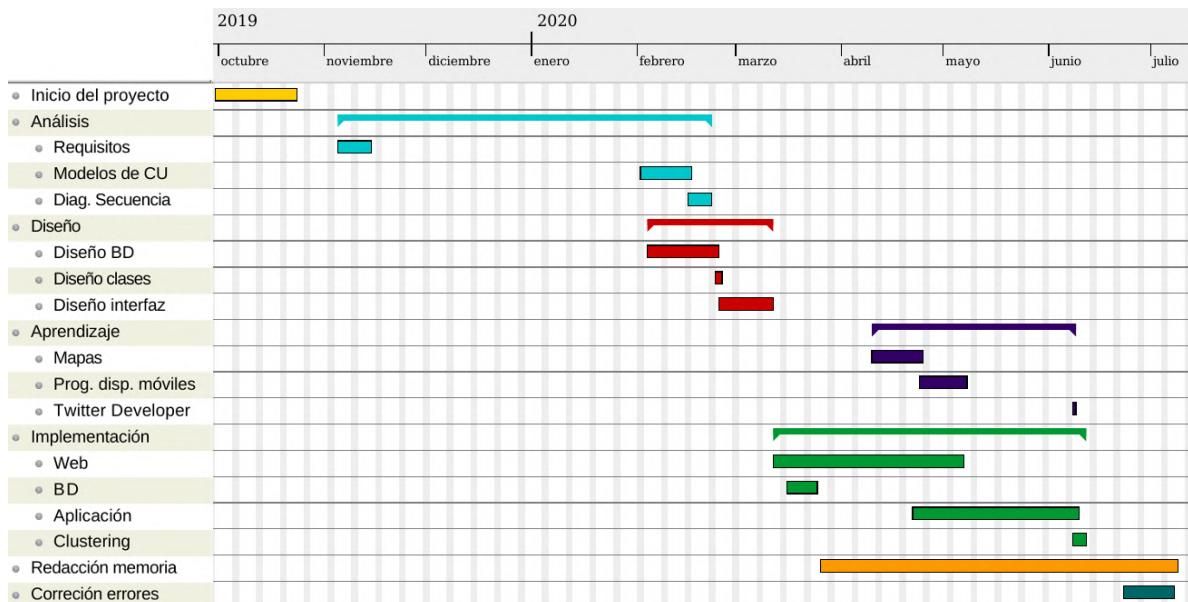


Figura 47: Temporización real del proyecto

El desarrollo web duró algo más de lo previsto. A pesar de ser la parte con mayor dominancia, llevar a cabo un trabajo tan complejo requiere de más tiempo. En cuanto a los mapas que se requerían para la visualización de incidentes, preví más complicación por el desconocimiento en el campo, pero su uso está muy bien integrado con el

desarrollo web y no hubo problema alguno con ellos, por lo que la duración de esta parte terminó siendo menor.

No se terminó solapando tanto como esperaba el desarrollo web con la aplicación, habida cuenta de que esta última comenzó cuando estaban finalizados prácticamente todos los servicios web. No obstante esta fase se extendió más en el tiempo debido a su novedad en cuanto a conocimientos y consecuentemente a su complejidad.

En líneas generales ante lo expuesto, no han surgido problemas de planificación dado que la fecha final de presentación es ligeramente posterior a lo que con cautela consideré, permitiendo así la finalización en tiempo.

Nombre	Fecha de inicio	Fecha de fin	Semanas
Inicio del proyecto	30/09/19	23/10/19	3
Análisis	05/11/19	22/02/20	4
Requisitos	05/11/19	14/11/19	1
Modelos de CU	02/02/20	16/02/20	2
Diag. Secuencia	16/02/20	22/02/20	1
Diseño	04/02/20	11/03/20	7,3
Diseño BD	04/02/20	24/02/20	3
Diseño clases	24/02/20	25/02/20	2
Diseño interfaz	25/02/20	11/03/20	2,3
Aprendizaje	10/04/20	08/06/20	4,14
Mapas	10/04/20	24/04/20	2
Prog. disp. móviles	24/04/20	07/05/20	2
Twitter Developer	08/06/20	08/06/20	0,14
Implementación	12/03/20	11/06/20	16,6
Web	12/03/20	06/05/20	8
BD	16/03/20	24/03/20	1
Aplicación	22/04/20	09/06/20	7
Clustering	08/06/20	11/06/20	0,6
Redacción memoria	26/03/20	08/07/20	15
Corrección errores	23/06/20	07/07/20	2

6.2. Objetivos alcanzados

De forma global puedo concluir haber alcanzado los objetivos propuestos al inicio de este proyecto tanto a nivel técnico, considerando el planteamiento inicial y las soluciones a llevar a cabo, como a nivel personal, sintiéndome satisfecho por el trabajo realizado y los resultados obtenidos, habiendo superado en todo momento las vicisitudes propias de un Trabajo de Fin de Grado.

6.3. Lecciones aprendidas

Más allá de la satisfacción por lograr los objetivos propuestos se encuentra la gratitud por haber aprendido multitud de cuestiones nuevas. Entre tantas, hay que señalar el haber aprendido a desarrollar una aplicación móvil completamente funcional partiendo de un conocimiento nulo en la materia. Otro aspecto a destacar es la puesta en producción del proyecto sin conocimientos previos al respecto. Si bien no es una tarea excesivamente complicada, se han presentado diversos contratiempos durante el proceso que sin duda alguna, de haber cursado alguna asignatura relacionada al respecto, hubieran entrañado menor dificultad. También considero haber mejorado en otros aspectos como en el desarrollo web, donde he podido profundizar en multitud de conocimientos y, especialmente, en el uso de un framework con el que llevar a cabo un trabajo mucho más cuidado y de mayor calidad.

Por último quiero poner en valor la importancia de la ingeniería del software y de la fase de diseño, sendas de vital importancia para que el desarrollo se presente de manera clara, sin obstáculos y permitiendo alcanzar la meta a la que se aspira llegar al inicio del proyecto. Estas dos fases, que en principio parecen tediosas e incluso infructuosas, son elementales para tener un soporte y una guía de trabajo sólida durante la implementación.

6.4. Trabajos futuros

Es innegable que, aún habiendo logrado los objetivos propuestos, en todo proyecto siempre se puede perfeccionar cualquier aspecto del mismo. Adicionalmente, sería interesante añadir nuevas funcionalidades que amplíen y mejoren las que ya existen. Algunas de ellas son:

- Utilizar un sistema de recuperación de información para seleccionar automáticamente el tipo de delito de un incidente a partir de su descripción.
- Recopilar nuevos incidentes de manera automática mediante un “crawler” en

redes sociales y otros sitios web.

- Añadir un “widget” o acceso rápido para la activación del botón del pánico
- Planificar rutas seguras.
- Analizar e informar más allá de las zonas con concentración de incidentes, concretando los tipos de delitos que se producen, en qué momentos del día, etc.

En definitiva, la intención es hacer de este proyecto un servicio público con un amplio uso entre la ciudadanía con el fin de resolver el problema planteado desde el inicio, que no es más que conseguir que cualquier persona se sienta y esté segura en la calle. Para hacer posible su explotación, una vía a seguir es presentar el proyecto ante entidades como pueden ser ayuntamientos o la policía, los cuales, tras una revisión de los delitos y su aplicación, puedan lanzar de manera pública la aplicación.

7. Bibliografía

Referencias

- [1] Centro de Investigaciones Sociológicas (CIS). *Tres problemas principales que existen actualmente en España (Multirrespuesta %)*. (Accedido el 14/06/2020). Mar. de 2020. URL: http://www.cis.es/cis/export/sites/default/-Archivos/Indicadores/documentos_html/TresProblemas.html.
- [2] *AlertCops*. URL: <https://alertcops.ses.mir.es/mialertcops/>.
- [3] *Sister*. URL: <https://joinsister.com/es/>.
- [4] *Red Panic Button*. URL: <http://redpanicbutton.com/>.
- [5] *bSafe*. URL: <https://getbsafe.com/>.
- [6] *Life360*. URL: <https://www.life360.com/>.
- [7] Ley Orgánica 10/1995, de 23 de noviembre, del Código Penal («B.O.E.» núm. 281, de 24 de noviembre de 1995). Libro II. URL: <https://www.boe.es/buscar/pdf/1995/BOE-A-1995-25444-consolidado.pdf>.
- [8] G. T. Doran. “There’s a S.M.A.R.T. Way to Write Management’s Goals and Objectives”. En: *Management Review* 70.11 (1987), págs. 35-36.
- [9] Ailin Orjuela Duarte y Mauricio Rojas C. “Las Metodologías de Desarrollo Ágil como una Oportunidad para la Ingeniería del Software Educativo”. En: *Revista Avances en Sistemas e Informática* (2008). ISSN: 1657-7663. URL: <https://www.redalyc.org/articulo.oa?id=133115027022>.
- [10] Bryan Molina M., Harry Vite C. y Jefferson Dávila C. “Metodologías ágiles frente a las tradicionales en el proceso de desarrollo software”. En: *Espirales: revista multidisciplinaria de investigación* (2018). ISSN: 2550-6862. URL: <http://revistaespirales.com/index.php/es/article/view/269/225>.
- [11] Iberley. *Bases máximas y mínimas por contingencias comunes y tipos de cotización al Régimen General Seguridad Social para el año 2020*. (Accedido el 21/04/2020). URL: <https://www.iberley.es/temas/bases-tipos-cotizacion-regimen-general-seguridad-social-ano-2020-11041>.
- [12] Athanasia Pouloudi. “Stakeholder Analysis as a Front-End to Knowledge Elicitation”. En: *AI Soc.* 11.1–2 (1997), págs. 122-137. ISSN: 0951-5666. DOI: 10.1007/BF02812443. URL: <https://doi.org/10.1007/BF02812443>.
- [13] Bernd Bruegge y Allen H. Dutoit. *Object-Oriented Software Engineering - Using UML, patterns and JAVA*. 3.^a ed. Pearson, 2009. ISBN: 9789332518681.
- [14] Robert B. Grady. *Practical software metrics for project management and process improvement*. 1.^a ed. Prentice Hall, 1992. ISBN: 0137203845.

- [15] Craig Larman. *UML y patrones: Una introducción al análisis y diseño orientado a objetos y al proceso unificado*. 2.^a ed. Pearson Educación, 2003. ISBN: 8420534382.
- [16] Edgar F. Codd. *The relational model for database management, Version 2*. 2.^a ed. Addison-Wesley, 1990. ISBN: 9780201141924.
- [17] Trygve Reenskaug y James O. Coplien. *The DCI Architecture: A New Vision of Object-Oriented Programming*. Mar. de 2009. URL: https://www.artima.com/articles/dci_vision.html.
- [18] Akshi Kumar. *Web Technology*. 1.^a ed. CRC Press, 2019. ISBN: 9781138550438. DOI: 10.1201/9781351029902. URL: <https://www.taylorfrancis.com/books/9781351029902>.
- [19] Codecademy. *MVC: Model, View, Controller*. (Accedido el 20/06/2020). URL: <https://www.codecademy.com/articles/mvc>.
- [20] Brian D. Miller. *Above the Fold: Understanding the Principles of Successful Web Site Design*. 1.^a ed. Adams Media Corporation, 2011. ISBN: 9781440308420.
- [21] *PHP*. URL: <https://www.php.net/>.
- [22] *HTML*. URL: <https://www.w3.org/html/>.
- [23] *HTML 5*. URL: <https://www.w3.org/TR/html52/>.
- [24] *CSS*. URL: <https://www.w3.org/Style/CSS/>.
- [25] *Bootstrap*. URL: <https://getbootstrap.com/>.
- [26] *jQuery*. URL: <https://jquery.com/>.
- [27] *AJAX*. URL: <https://api.jquery.com/jquery.ajax/>.
- [28] *PHP frameworks*. URL: <https://www.hostinger.es/tutoriales/mejores-frameworks-php/>.
- [29] *Laravel*. URL: <https://laravel.com/>.
- [30] *Laravel - Database: Migrations*. URL: <https://laravel.com/docs/7.x/migrations/>.
- [31] *LAMP: Linux, Apache, MySQL, PHP*. URL: <https://whatis.techtarget.com/definition/LAMP-Linux-Apache-MySQL-PHP>.
- [32] *XAMPP*. URL: <https://www.apachefriends.org/>.
- [33] *Bcrypt*. URL: <https://solidgeargroup.com/en/hashing-passwords-nodejs-mongodb-bcrypt/>.
- [34] *Blowfish*. URL: <https://www.schneier.com/academic/blowfish/>.
- [35] *Should password be hashed server-side or client-side?* URL: <https://security.stackexchange.com/questions/8596/https-security-should-password-be-hashed-server-side-or-client-side>.

- [36] *OpenStreetMap Nominatim*. URL: <https://nominatim.openstreetmap.org/>.
- [37] *Twitter Developer*. URL: <https://developer.twitter.com/>.
- [38] *scikit learn: Mean Shift algorithm for clustering*. URL: <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.MeanShift.html>.
- [39] *Leaflet*. URL: <https://leafletjs.com/>.
- [40] Red Hat. *Qué son las API y para qué sirven*. URL: <https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces>.
- [41] StatCounter. *Mobile Operating System Market Share Worldwide*. (Accedido el 15/06/2020). Mayo de 2020. URL: <https://gs.statcounter.com/os-market-share/mobile/worldwide>.
- [42] *React Native*. URL: <https://reactnative.dev/>.
- [43] *Ionic*. URL: <https://ionicframework.com/>.
- [44] *Digital Ocean Droplets*. URL: <https://www.digitalocean.com/products/droplets/>.
- [45] *Certbot*. URL: <https://certbot.eff.org>.

8. Anexos

A. Delitos extraídos del Código Penal

1. Contra la vida y la integridad

- 1.1. **Homicidio.** Causar la muerte a una o varias personas.
- 1.2. **Lesiones.** Causar a otro una lesión que menoscabe su integridad corporal o su salud física o mental, otro tipo de lesiones, golpear o maltratar.
- 1.3. **Lesiones a una embarazada.** Lesionar a una mujer embarazada de modo que pueda causar en el feto una lesión o enfermedad, pudiéndose producir además el aborto.

2. Contra la libertad

- 2.1. **Detención ilegal y secuestro.** Encerrar o detener a otro privándole de su libertad
- 2.2. **Amenazas.** Amenazar con causar a una persona, su familia u otras personas de su ámbito cercano un mal que constituya delitos de homicidio, lesiones, aborto, contra la libertad, etc.
- 2.3. **Coacciones.** Forzar a alguien a realizar una acción en contra de su voluntad, ejercer intimidación, vigilancia o persecución, así como cualquier otro tipo de acoso.
- 2.4. **Humillación y acoso.** Infligir a otra persona un trato degradante, menoscabando gravemente su integridad moral. Realizar actos hostiles o humillantes que supongan grave acoso a la víctima.
- 2.5. **Utilización de menores o personas con discapacidad.** Utilizar o presolar a menores de edad o personas con discapacidad necesitadas de especial atención para la práctica de la mendicidad

3. Contra los derechos fundamentales y libertades públicas

- 3.1. **Delitos de odio.** Fomentar o incitar al odio, discriminación o violencia o llevar a cabo estos actos contra una persona o grupo por motivos racistas, ideológicos, religión, creencias, etnia, raza, nacionalidad, sexo, orientación o identidad sexual, género, enfermedad, etc.
- 3.2. **Manifestaciones ilícitas.** Asistir a una reunión o manifestación portando armas, artefactos explosivos, objetos contundentes u otros medios igualmente peligrosos.
- 3.3. **Desórdenes públicos.** Alterar la paz pública ejecutando actos de violencia sobre las personas o sobre las cosas, o amenazando a otros con llevarlos a cabo.

4. Contra la libertad sexual

- 4.1. Agresión sexual.** Atentar contra la libertad sexual de otra persona, utilizando violencia o intimidación.
- 4.2. Abuso sexual.** Realizar actos que atenten contra la libertad o indemnidad sexual sin violencia o intimidación.
- 4.3. Abuso o agresión sexual a menores de 16 años.** Realizar actos de carácter sexual con un menor de dieciséis años u obligarle a presenciar o participar en un comportamiento de naturaleza sexual.
- 4.4. Acoso sexual.** Solicitar favores de naturaleza sexual, para sí o para un tercero, de manera continuada o habitual, de manera que provoque a la víctima una situación intimidatoria, hostil o humillante.
- 4.5. Exhibicionismo y provocación.** Ejecutar o hacer ejecutar a otra persona actos de exhibición obscena ante menores de edad o personas con discapacidad. Vender, difundir o exhibir material pornográfico entre personas del colectivo anteriormente mencionado.
- 4.6. Prostitución y explotación sexual.** Determinar a una persona mayor de edad a ejercer o mantenerse en la prostitución, empleando violencia, intimidación o engaño.
- 4.7. Corrupción de menores.** Inducir, promover, favorecer o facilitar la prostitución de un menor de edad o una persona con discapacidad necesitada de especial protección.

5. Contra la intimidad, la inviolabilidad del domicilio, el patrimonio y el orden socioeconómico

- 5.1. Allanamiento.** Entrar en morada ajena, domicilio de una persona jurídica u otro establecimiento, o mantenerse en el mismo contra la voluntad de su morador o propietario.
- 5.2. Hurto.** Tomar las cosas muebles ajenas sin la voluntad de su dueño con ánimo de lucro.
- 5.3. Robo.** Apoderarse de las cosas muebles ajenas empleando la fuerza para acceder o abandonar el lugar donde estas se encuentran, o violencia o intimidación en la víctima o quienes le auxiliaran.
- 5.4. Estafa.** Engañar a una persona o realizar otros actos cuyo fin sea el ánimo de lucro en perjuicio de su titular o un tercero.
- 5.5. Daños a la propiedad.** Causar daños en propiedad ajena.
- 5.6. Sustracción de propiedad con utilidad social o cultural.** Destruir, inutilizar o dañar una cosa propia de utilidad social o cultural.

6. Contra el urbanismo, el patrimonio histórico y el medio ambiente

- 6.1. Delitos sobre el patrimonio histórico.** Derribar o alterar edificios, yacimientos protegidos por su interés histórico, científico, artístico, cultural o monumental. Causar daños en archivos, museos, bibliotecas, centros docentes o instituciones análogas con alto valor.
- 6.2. Delitos contra la flora, la fauna y los animales domésticos.** Cortar, recolectar, poseer, traficar o destruir especies protegidas de flora silvestre. Introducir o liberar especies de flora o fauna no autóctona. Cazar o pescar y traficar con fauna silvestre. Maltratar a animales lesionándolos gravemente.

7. Contra la seguridad colectiva

- 7.1. Explosiones.** Provocar explosiones o similares causando la destrucción de infraestructuras, depósitos de materiales inflamables o explosivos, vías de comunicación o cualquier perturbación cuyos estragos comporten un peligro para la vida o la seguridad de las personas.
- 7.2. Incendios.** Provocar un incendio que comporte un gran peligro.
- 7.3. Drogas y estupefacientes.** Traficar o promover, facilitar o favorecer el consumo ilegal de drogas tóxicas, estupefacientes o sustancias psicóticas, o las posean con aquellos fines.
- 7.4. Seguridad vial.** Conducir un vehículo de motor o ciclomotor a velocidad superior a la permitida reglamentariamente o de forma manifiestamente temeraria, poniendo en peligro la vida o integridad de las persona.

B. Vista del sitio web

Al acceder al sitio web, la primera página que se encuentra el usuario es la de inicio de sesión (Figura 48), desde donde puede navegar hasta la página de registro.

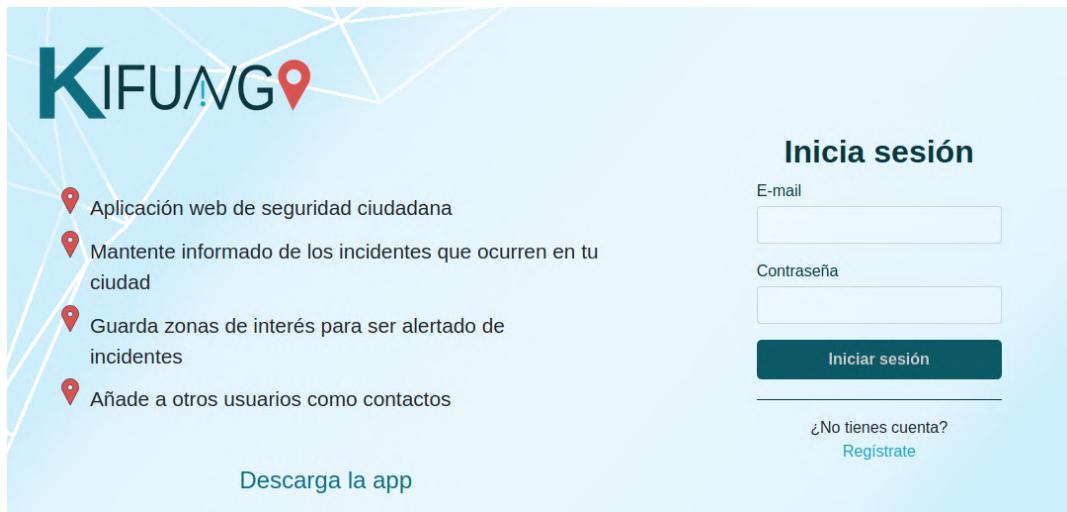


Figura 48: Web - Login

La primera vista del registro en dos pasos muestra un formulario para introducir el correo electrónico y la contraseña con su confirmación (Figura 49).



Figura 49: Web - Registro - Paso 1

La segunda vista contiene el resto de campos del formulario necesarios para completar el registro, todos de ellos obligatorios y, por tanto, marcados con un asterisco rojo para reflejarlo (Figura 50).

Figura 50: Web - Registro - Paso 2

Una vez iniciada la sesión, la página web mostrada es la que contiene la lista de incidentes (Figura 51), cada uno con el tipo de delito, el lugar y la fecha y hora donde ha ocurrido. También incluye una opción para ver la descripción del mismo.

Sobre la lista se muestran enlaces que llevan a los incidentes publicados por el usuario y al mapa de incidentes, así como información sobre el filtro aplicado en caso de haberlo utilizado.

A la izquierda de la página web se sitúa el menú de navegación con las secciones principales del sitio. A la derecha hay un filtro que permite obtener los incidentes según un tipo y en un rango de fechas concreto. Debajo de los filtros se encuentra un botón con un enlace al formulario de alta de incidentes. En la parte superior derecha se ubica la zona de notificaciones y un acceso al perfil del usuario. Tanto esta última sección como el menú de navegación son comunes a todas las páginas.

Mis publicaciones de incidentes

Intervalo de fecha: 01/03/2020 - 31/05/2020

Sustracción de propiedad con utilidad social o cultural 30/05/2020 17:50
Beiro, Granada (Granada) [Ver más](#)

Seguridad vial 30/05/2020 17:20
Beiro, Granada (Granada)
Un coche negro ha atropellado a una mujer de mediana edad
[Compartir incidente](#) [Ver menos](#)

Drogas y estupefacientes 27/05/2020 18:38
Chana, Granada (Granada) [Ver más](#)

Tipos incidentes ▾

Desde dd / mm / aaaa

Hasta dd / mm / aaaa

Filtrar

Añadir incidente

Figura 51: Web - Lista de incidentes

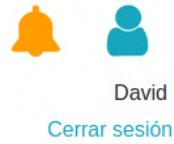
El mapa presenta los incidentes ocurridos en forma de marcador, el cual ofrece un detalle de este al clicar en él tal y como se ve en la Figura 52. También se aprecian las zonas de concentración de incidentes con circunferencias que indican el nivel de gravedad de la zona según su color.

La vista ofrece una estructura similar al de la lista de incidentes, con los filtros aplicados encima del mapa, enlaces a los incidentes publicados por el usuario y a la lista en la parte inferior y los filtros a la derecha.

Mapa de incidentes

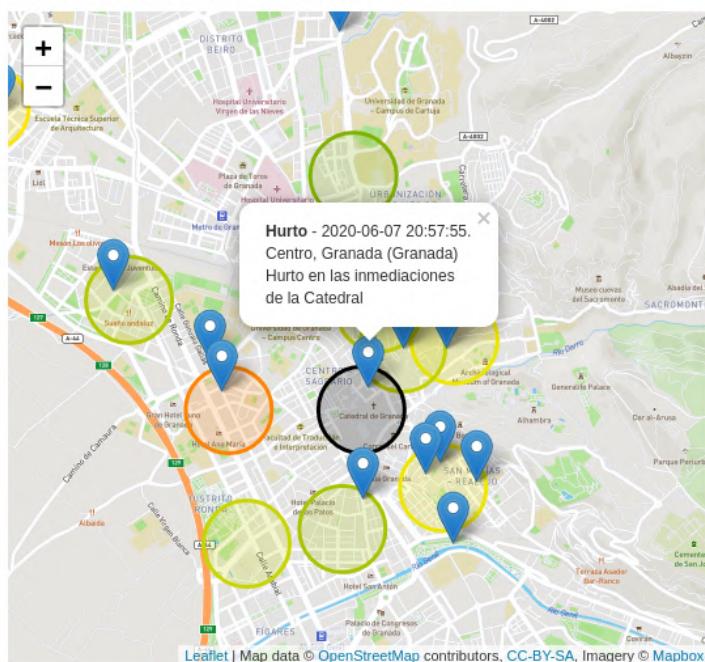
Intervalo de fecha: 01/06/2020 - 26/06/2020

Tipos de incidentes: homicidio, lesiones, hurto



David

Cerrar sesión



Homicidio, Lesiones, Hurto ▾

Desde

01 / 06 / 2020



Hasta

26 / 06 / 2020



Filtrar

Añadir incidente

Mis publicaciones de incidentes

Ver lista

Figura 52: Web - Mapa de incidentes

El formulario de alta de incidentes (Figura 53) contiene un primer campo para seleccionar uno o varios tipos de delitos y un segundo para escoger uno de cualquiera de las categorías seleccionadas. A continuación se encuentran los campos de fecha y hora y el mapa para situar el incidente, el cual dispone de una opción de búsqueda de un lugar por texto. Seguidamente está el campo de descripción, la lista de agravantes y, por último, una opción para seleccionar si el incidente dado de alta ha sido sufrido por el usuario u observado por este.

Dar de alta incidente

Tipo de delito

Categoría delito >> Delito

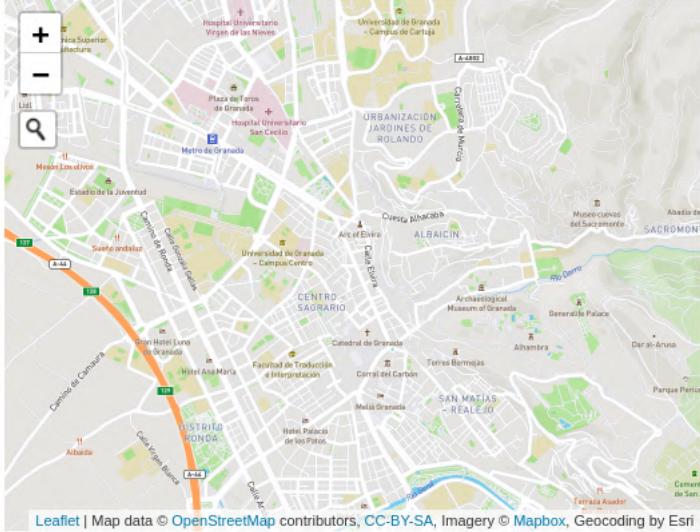
Fecha

dd / mm / aaaa

Hora

-- : --

Lugar



Leaflet | Map data © OpenStreetMap contributors, CC-BY-SA, Imagery © Mapbox, Geocoding by Esri

Descripción

Agravantes

Disfraz Abuso de superioridad Sufrimiento inhumano
 Racismo, discriminación, homofobia, machismo, ...

He sido

Testigo Afectado

Añadir incidente

Figura 53: Web - Alta de incidentes

La sección de contactos favoritos contiene una lista con los contactos del usuario, cada uno con un desplegable para acceder a los detalles de este y la opción de poder eliminarlo. A la derecha de la lista hay tres enlaces que llevan a las páginas para buscar y añadir un nuevo contacto, consultar de quién se es contacto favorito y ordenarlos, respectivamente (Figura 54).

The screenshot shows a user profile at the top right with a yellow bell icon, a blue person icon, the name 'David', and a 'Cerrar sesión' (Logout) link. Below the header is the title 'Contactos favoritos'. The main content area displays two contacts in a list:

- Contacto 1**: Includes the number 'Teléfono: 7XXXXXX' and email 'Email: contacto1@mail.com'. It features a red 'Eliminar contacto favorito' (Delete favorite contact) button with a trash bin icon.
- Contacto 2**: This entry is partially visible and ends with a blue '+' sign.

On the right side of the list are three buttons: 'Añadir contacto favorito' (Add favorite contact), 'De quién soy contacto favorito' (Who am I favorite contact of), and 'Ordenar contactos' (Sort contacts).

Figura 54: Web - Contactos favoritos

La página para agregar un nuevo contacto contiene un campo de búsqueda para introducir el email o número de teléfono del usuario, el cual aparece como en la Figura 55 si existe. Al agregarlo se muestra un campo indicando que la solicitud ha sido enviada, como se puede apreciar en la Figura 56.

The screenshot shows the 'Contactos favoritos' page with a search bar containing the email 'david@mail.com'. To the right of the search bar is a teal 'Buscar' (Search) button. Below the search results, the contact 'David Carrasco Chicharro' is listed with a teal person icon and a plus sign.

Figura 55: Web - Buscar usuario

Contactos favoritos

david@mail.com

Buscar

David Carrasco Chicharro

Solicitud enviada

Figura 56: Web - Añadir contacto

La página de zonas de interés ofrece un mapa con las zonas del usuario con un marcador por zona y una circunferencia que abarca el radio de esta (Figura 57). Al clicar en el marcador de la zona aparece un “popup” con el nombre de la zona y el radio que comprende.

A la derecha del mapa se muestra un botón para añadir una nueva zona y, debajo de este, un desplegable con las zonas de interés del usuario; al seleccionar una aparece un ícono con una papelera que permite eliminarla al clicar en ella.

Zonas de interés

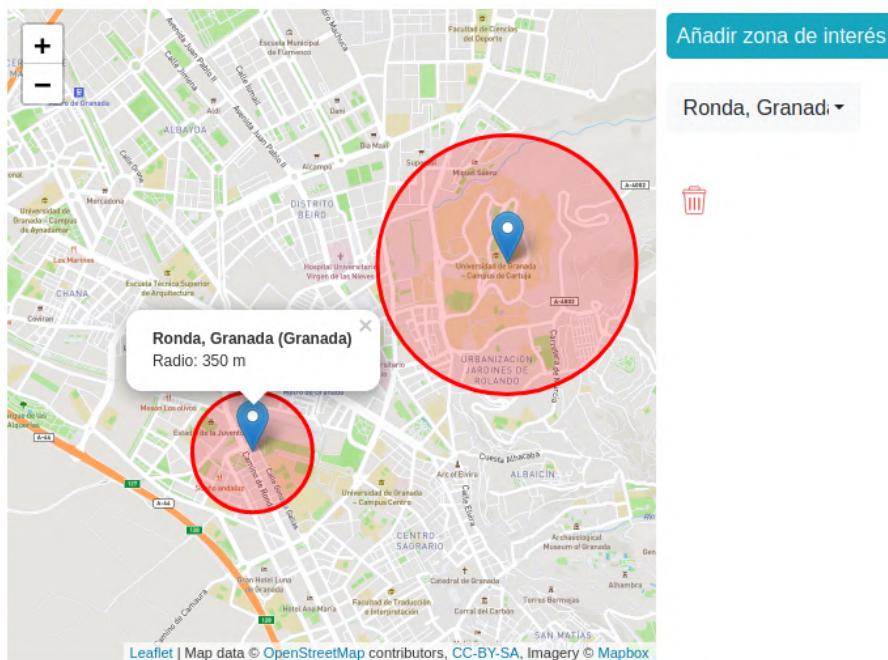


Figura 57: Web - Zonas de interés

Para añadir una nueva zona de interés hay que seleccionar esta en el mapa e indicar el radio que ocupará con el *slider* de la parte inferior, como se puede apreciar en la Figura 58.

Añadir zona de interés

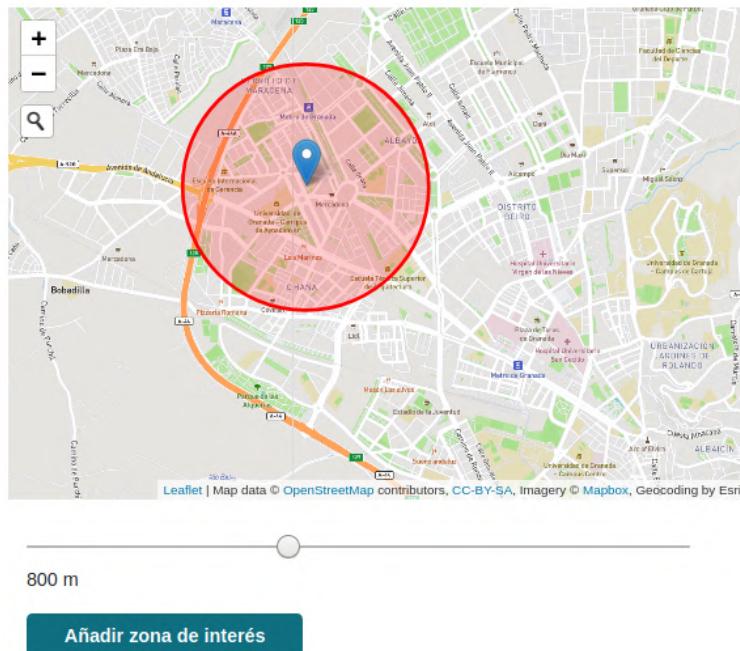


Figura 58: Web - Alta de zona de interés

La zona personal del usuario se divide en cuatro secciones principales:

- Configuración: establecer tanto la acción de pánico del usuario como su PIN secreto.
- Datos personales: formulario para modificar sus datos personales.
- Seguridad: formulario para modificar la contraseña.
- Cuenta: opción para cerrar la cuenta del usuario.

En la Figura 59, además de los apartados descritos, se puede apreciar un ejemplo de notificación de petición para ser contacto favorito de un usuario.



Zona personal

- Incidentes
- Contactos favoritos
- Zonas de interés

Configuración

- Establecer acción de pánico
- Establecer PIN secreto



Notificaciones

+ Usuario_1 quiere agregarle como contacto favorito

Aceptar

Rechazar

Datos personales

Nombre

!

Apellidos

!

E-mail

!

D.N.I. / N.I.E.

Fecha de nacimiento

!

Número de teléfono móvil

!

Número de teléfono fijo

Guardar cambios

Seguridad

Contraseña actual

Nueva contraseña

Confirmar contraseña

Cambiar contraseña

Cuenta

[Cerrar cuenta](#)

Figura 59: Web - Zona personal

Un ejemplo del *modal* que aparece al seleccionar la opción para establecer la acción de pánico es el que se muestra en la Figura 60. De forma similar se muestra el *modal* para establecer el PIN.



Figura 60: Web - Selección de acción de pánico

C. Vista de la aplicación móvil

Al iniciar la aplicación móvil se encuentra la pantalla de inicio de sesión, desde donde se puede navegar hasta la pantalla de registro (Figura 61), dispuesta en dos pasos para verificar tras la primera que el email introducido no existe registrado en el sistema. Los botones para continuar y de registro se activan cuando los campos introducidos son correctos.

The figure consists of three side-by-side screenshots of the KIFUANG mobile application interface. The first screenshot shows the 'Inicia sesión' (Login) screen with fields for 'Email *' and 'Contraseña *', a large blue 'INICIAR SESIÓN' button, and links for '¿No tienes cuenta?' and 'Regístrate'. The second screenshot shows the first step of the 'Registro' (Registration) process, requiring 'Email *', 'Contraseña *', and 'Confirmar contraseña *', with a 'SIGUIENTE' (Next) button and a link for '¿Ya tienes cuenta?'. The third screenshot shows the second step of the 'Registro' process, requiring 'Nombre *', 'Apellidos *', 'D.N.I. *', 'Fecha de nacimiento * DD/MM/AAAA', and 'Número de teléfono móvil *', with a 'REGISTRARSE' (Register) button and a link for 'Inicia sesión'.

Figura 61: App - Login y registro en dos pasos

Con la sesión iniciada se muestra la pestaña de incidentes en formato de lista, la cual lleva al incidente en detalle al pulsar en uno de ellos (tercera pantalla de la Figura 62). En la parte superior se encuentran los botones que llevan a las pantallas del mapa de incidentes, a los que el usuario ha dado de alta y a una pantalla de tipo *modal* con los filtros que se pueden aplicar.

El botón de la parte inferior derecha lleva a la pantalla de alta de un nuevo incidente, mostrada en la Figura 63, con la misma estructura que en la página web, exceptuando el desplegable de selección de delito, que ofrece una lista con todos ellos por simplicidad.

Lista de incidentes

Mapa Subidos Filtro

Intervalo: 01/03/2020 - 31/05/2020

Sustracción de propiedad con utilidad social o cultural

Beiro, Granada (Granada) 30/05/2020 17:50

Seguridad vial

Beiro, Granada (Granada) 30/05/2020 17:20

Drogas y estupefacientes

Chana, Granada (Granada) 27/05/2020 18:38

Detención ilegal y secuestro

Centro, Granada (Granada) 08/05/2020

Mapa de incidentes

Lista Subidos Filtro

Intervalo: 01/06/2020 - 25/06/2020

Tipos de incidentes: Homicidio, Lesiones, Abuso sexual

Incidentes

Seguridad vial

Beiro, Granada (Granada)
30/05/2020 - 17:20
Un coche negro ha atropellado a una mujer de mediana edad

Compartir incidente

Figura 62: App - Lista y mapa de incidentes y detalle de incidente

Dar de alta incidente

Delito
Delito

Fecha DD/MM/AAAA Hora HH:MM

Lugar

Descripción
Descripción del incidente...

Agravantes

Dar de alta incidente

Descripción
Descripción del incidente...

0 / 1000 (mín. 10)

Agravantes

Disfraz
 Abuso de superioridad
 Sufrimiento inhumano
 Racismo, discriminación, homofobia, machi...

He sido

Testigo Afectado

AÑADIR INCIDENTE

Figura 63: App - Alta de un incidente

La zona personal del usuario se encuentra en la última pestaña de la aplicación, la cual cuenta con las mismas secciones que su equivalente en la página web, pero cada una con una pantalla independiente de tipo *modal* como en el segundo y tercer ejemplo de la Figura 64.

The screenshot shows the 'Personal Zone' (Zona personal) interface on the left, which includes sections for Configuration (Configuración), Personal Data and Security (Datos personales y seguridad), and Account (Cuenta). Below these are icons for notifications, favorites, and profile. Three modals are overlaid on the screen:

- Modificar datos personales**: A modal for editing personal data. It contains fields for Name (Nombre) with value 'David', Surname (Apellidos) with value 'Carrasco Chicharro', Email (E-mail) with value 'david@mail.com', and DNI (D.N.I.) with value '15520228-N'. It also has fields for Mobile phone number (Número de teléfono móvil) with value '6XXXXXX' and Fixed phone number (Número de teléfono fijo). A 'GUARDAR CAMBIOS' (Save Changes) button is at the bottom.
- Establecer acción de pánico**: A modal for setting a panic action. It lists three options: Ninguna (None), Ubicación (Location), and Llamada (Call). The 'Ubicación' option is selected. A 'ACEPTAR' (Accept) button is at the bottom.
- Another Establecer acción de pánico**: A second instance of the panic action settings modal, identical to the first one shown above it.

Figura 64: App - Zona personal, modificación de datos personales y establecimiento de acción de pánico

La pestaña de zonas de interés muestra la pantalla con dichas zonas y un botón (*fab button*) en la parte inferior derecha, que contiene, de arriba a abajo como se muestra en la primera pantalla de la Figura 65, botones para recargar las zonas, eliminarlas y añadir una nueva. La pantalla de esta última se muestra también en la Figura, con el mismo mecanismo que en la web.

Por último se puede apreciar también la pestaña de notificaciones, donde se muestran dos ejemplos de una notificación de petición de contacto favorito y de un nuevo incidente en una de las zonas de interés.

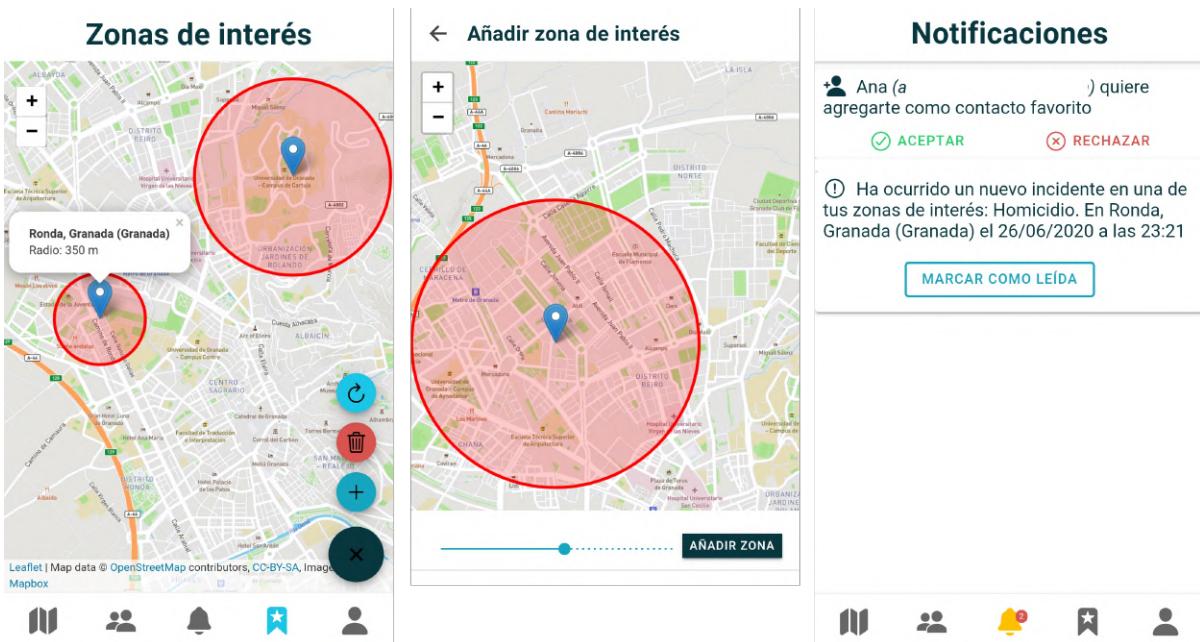


Figura 65: App - Zonas de interés y notificación de incidente

La pestaña de contactos favoritos ofrece una lista con estos y el detalle de cada uno al pulsarlo en una pantalla aparte (Figura 66). En la primera aparece un botón de tipo *fab button* con las siguientes opciones, de arriba a abajo: botón del pánico, usuarios de quien se es contacto favorito, añadir nuevo contacto favorito y ordenar contactos.

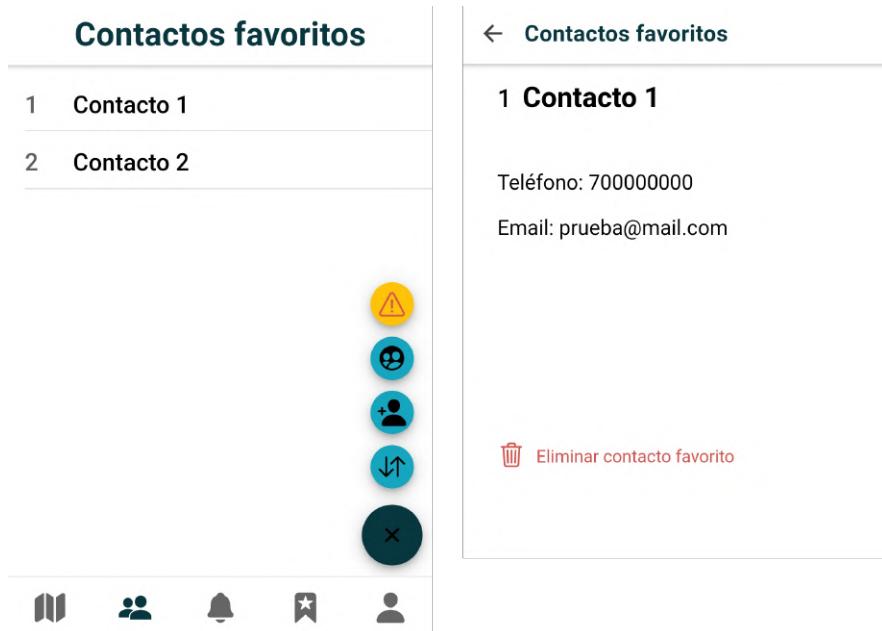


Figura 66: App - Contactos favoritos y detalle de contacto

A continuación se muestra en la Figura 67 un ejemplo del uso del botón del pánico, concretamente del envío de ubicación, que presenta en primer lugar una cuenta atrás con la opción de cancelar antes de llevar a cabo la acción.

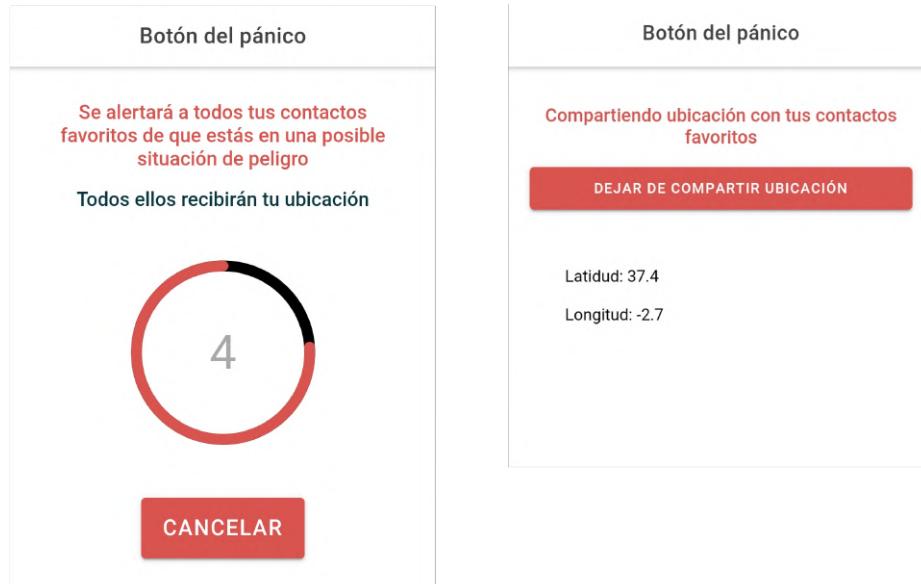


Figura 67: App - Botón del pánico (vista de usuario)

En la Figura 68 se expone la secuencia que ve el contacto favorito de quien hace uso de la acción de pánico. Tras recibir la notificación y acceder a ella, carga la ubicación y se muestra la posición del usuario en tiempo real.

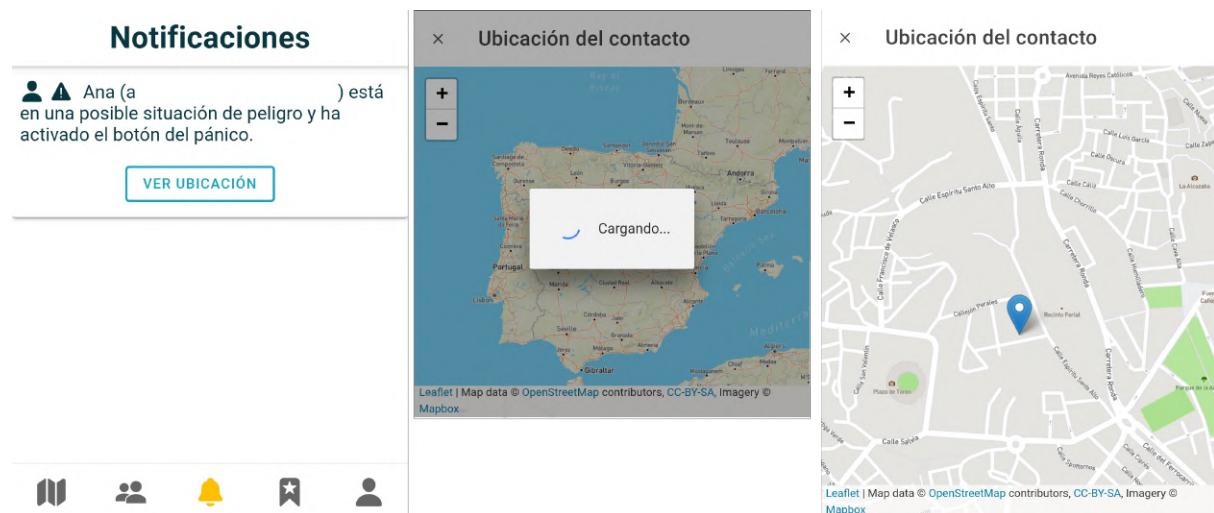


Figura 68: App - Botón del pánico (vista de contacto favorito)

D. Código fuente y sitio web

El código fuente del proyecto se encuentra ubicado en el siguiente repositorio de GitHub: <https://github.com/DavidChicharro/Trabajo-Fin-Grado>.

El sitio web está accesible en la siguiente dirección: <https://kifungo.live/>.