



universidad
de león



Escuela de Ingenierías Industrial, Informática y Aeroespacial

GRADO EN INGENIERÍA INFORMÁTICA

Trabajo de Fin de Grado

TecnoMaps: Aplicación web para la gestión y
visualización de mapas.

TecnoMaps: Web application for managing and
displaying maps.

Autor: Raquel Chamorro Giganto

Tutor: Jose Alberto Benítez Andrades

(Julio, 2020)



UNIVERSIDAD DE LEÓN
Escuela de Ingenierías Industrial, Informática y
Aeroespacial

GRADO EN INGENIERÍA INFORMÁTICA
Trabajo de Fin de Grado

ALUMNO: Raquel Chamorro Giganto

TUTOR: Jose Alberto Benítez Andrades

TÍTULO: TecnoMaps: Aplicación web para la gestión y visualización de mapas.

TITLE: TecnoMaps: Web application for managing and displaying maps.

CONVOCATORIA: Julio, 2020

RESUMEN:

TecnoMaps es una aplicación web realizada en Vue por Andrea Fernández y por mí centrada en la gestión y visualización de mapas y capas, junto con registro y control de usuarios. En ella, los usuarios pueden visualizar mapas, añadir o eliminar capas, editarlas, gestionar su galería de capas, y gestionar a los propios usuarios si se es administrador. La parte de la aplicación que me correspondió a mí fue el Frontend. Es decir, la parte visual de la interfaz y su funcionalidad, y parte de su comunicación con el Backend. Y es por eso que mi trabajo se centrará en toda esta parte del Frontend, pudiendo llegar a mencionar algún aspecto relacionado con la parte realizada por mi compañera que realizó el Backend. En este trabajo realizaré un estudio sobre el problema a resolver explicando los objetivos propuestos para la aplicación, herramientas o tecnologías usadas, una explicación de la planificación del trabajo, cómo ha sido gestionado el producto software, una parte centrándome en la solución final que consta de la estructura de la aplicación y su diseño con todas sus utilidades y por último cómo ha sido el proceso de evaluación de dicho proyecto con sus conclusiones finales.

ABSTRACT:

TecnoMaps is a web application made in Vue by Andrea Fernández and myself focused on the management and display of maps and layers, along with user registration and control. In it, users can view maps, add or remove layers, edit them, manage their gallery of layers, and manage users themselves if they are administrators. The part of the application that corresponded to me was the Frontend, that is, the visual part of the interface and its functionality, and part of its communication with the Backend. Hence why my work will focus on this whole part of the Frontend, and I may mention some aspects related to the part done by my colleague who programmed the Backend. In this work I will carry out a study on the problem to be solved explaining the proposed objectives for the application, tools and technologies used, an explanation of the work planning, how the software product has been managed, a part focusing on the final solution consisting of the structure of the application and its design with all its utilities and finally how the evaluation process of said project has been with its final conclusions.

Palabras clave: Aplicación, Web, Mapas, Capas, Frontend, Vue, Interfaz.

Firma del alumno:



VºBº Tutor/es:

Índice de contenidos

Índice de contenidos.....	4
Índice de figuras.....	6
Índice de tablas.....	8
Introducción.....	10
Planteamiento del problema	10
Objetivos	10
Metodología	11
Estructura del trabajo	14
1. Estudio del problema	15
1.1 El contexto del problema	15
1.2 El estado de la cuestión.....	16
1.2.1 Tecnología existente	16
1.2.2 ArcGIS.....	17
1.3 La definición del problema.....	18
2. Gestión de proyecto software.....	20
2.1 Alcance del proyecto.....	20
2.1.1 Definición del proyecto.....	20
2.1.2 Estimación de tareas y recursos	21
2.2 Plan de trabajo	22
2.2.1 Identificación de tareas	22
2.2.2 Estimación de tareas.....	22
2.2.3 Planificación de tareas.....	25
2.3 Gestión de recursos	27
2.3.1 Especificación de recursos	27
2.3.2 Asignación de recursos	28
2.4 Gestión de riesgos.....	29
2.4.1 Identificación de riesgos	29
2.4.2 Análisis de riesgos	29
3. Solución	32
3.1 El proceso de desarrollo.....	32
3.1.1 Análisis	32
3.1.2 Diseño	36

3.1.3	Implementación	40
3.2	El producto del desarrollo	44
3.2.1	Flujo de la aplicación	44
3.2.2	Análisis de cada componente	45
4.	Evaluación	62
4.1	Proceso de evaluación	62
4.1.1	Forma de evaluación	62
4.1.2	Casos de prueba	63
4.1.3	Cuestionarios de evaluación	70
4.1.4	Test de caja blanca	70
4.1.5	Test de caja negra	74
4.2	Análisis de resultados	76
4.2.1	Casos de prueba	76
4.2.2	Cuestionarios de evaluación	76
4.2.3	Caja negra y caja blanca	77
	Conclusión	79
	Objetivos conseguidos	79
	Trabajos futuros	80
	Problemas encontrados	81
	Opiniones personales	82
	Lista de referencias	84
	ANEXO A: Control de Versiones	86
	ANEXO B: Seguimiento de Proyecto fin de Carrera	88
	Seguimiento de la primera parte	88
	Seguimiento de la segunda parte	89
	ANEXO C: Cuestionarios de evaluación	101

Índice de figuras

Figura 0.1.1 Esquemas de las fases del modelo en cascada. (Fuente: [1]).....	12
Figura 0.1.2 Esquema de la metodología Scrum. (Fuente: [2])	13
Figura 1.1 Logo de ArcGIS (Fuente: [7])	17
Figura 3.1 "Arquitectura con Vue.js + Nodejs RestAPIs – Sequelize ORM + MySQL" (Fuente: [12])	37
Figura 3.2 Directorio con todas las carpetas y archivos de la implementación.	38
Figura 3.3 color destacado de la aplicación.....	40
Figura 3.4 - Logos de (Vue, Vuetify, Electron y Visual Code Studio).....	41
Figura 3.5 Configuración de la conexión ssh en Putty.....	43
Figura 3.6 Flujo de la aplicación TecnoMaps	44
Figura 3.7 Vista del login.....	46
Figura 3.8 Detalle del formulario de Sign Up y Login In.	47
Figura 3.9 Función de control de registro de usuarios.	48
Figura 3.10 Detalle del menú principal.....	49
Figura 3.11 Detalle del menú de capas.....	50
Figura 3.12 Array con los enlaces del Router a cada componente.	50
Figura 3.13 Vista del mapa.....	51
Figura 3.14 Código del mapa	52
Figura 3.15 Vista de añadir capa.....	53
Figura 3.16 Métodos para crear y guardar una capa.	53
Figura 3.17 Vista de gestión de capas.....	54
Figura 3.18 Cuadro de diálogo del borrado de capa.	55
Figura 3.19 Notificaciones del estado de guardado de capas.	55
Figura 3.20 Vista de Gestión de usuarios.	56
Figura 3.21 Ejemplo de búsqueda de usuario.	56
Figura 3.22 Cuadro de diálogo de confirmación de borrado de usuario.....	57
Figura 3.23 Función para la eliminación de usuarios.	57
Figura 3.24 Ejemplo de ventana emergente con la información del perfil de usuario.....	58

Figura 3.25 Vista del perfil de usuario.	59
Figura 3.26 Detalle de la sección de cambio de contraseña.	60
Figura 3.27 Ventana de confirmación de eliminación del propio usuario.	61
Figura 3.28 Extracto de código de Users.vue	61

Índice de tablas

Tabla 2.1 Media nacional y cálculo por hora del salario de las especialidades requeridas.....	28
Tabla 2.2 Periodo de tiempo trabajado y cálculo de las horas trabajadas.....	28
Tabla 2.3 Especificación de riesgos de la aplicación.....	29
Tabla 2.4 Riesgo 1 - Fallo técnico del equipo.....	29
Tabla 2.5 Riesgo 2 - Fallo de la base de datos.	30
Tabla 2.6 Riesgo 3 - Modificación de interfaz.....	30
Tabla 2.7 Riesgo 4 - Cambio de requerimientos.....	30
Tabla 2.8 Riesgo 5 - Retraso de entregables.....	30
Tabla 2.9 Riesgo 6 - Falta de experiencia.....	31
Tabla 2.10 Riesgo 7 - Aumento del presupuesto.....	31
Tabla 3.1 Requisito funcional 1 - Pantalla de registro de usuarios.	32
Tabla 3.2 Requisito funcional 2 - Registro de usuarios.....	33
Tabla 3.3 Requisito funcional 3 - Log in de usuarios.	33
Tabla 3.4 Requisito funcional 4 - Log out de usuarios.....	33
Tabla 3.5 Requisito funcional 5 - Visualización de capas.....	33
Tabla 3.6 Requisito funcional 6 - Creación de capas.	33
Tabla 3.7 Requisito funcional 7- Edición de capas.....	34
Tabla 3.8 Requisito funcional 8 - Eliminar capas.	34
Tabla 3.9 Requisito funcional 9 - Edición de la información de usuario.	34
Tabla 3.10 Requisito funcional 10 - Gestión de usuarios.	34
Tabla 3.11 Requisito funcional 11 - Roles de usuario.....	34
Tabla 3.12 Requisito funcional 12 -Eliminar cuenta de usuario.	35
Tabla 3.13 Requisito no funcional 1 - Diseño web adaptable.	35
Tabla 3.14 Requisito no funcional 2 - Accesibilidad.	35
Tabla 3.15 Requisito no funcional 3 - Compatibilidad con navegadores.	35
Tabla 3.16 Requisito no funcional 4 - Acceso a la aplicación.	36
Tabla 3.17 Requisito no funcional 5 - Empleo de Vue.....	36

Tabla 4.1 Caso de prueba 1 – Login.	63
Tabla 4.2 Caso de prueba 2 - Sign in.	63
Tabla 4.3 Caso de prueba 3 - Añadir capa.	64
Tabla 4.4 Caso de prueba 4 - Editar capa.	64
Tabla 4.5 Caso de prueba 5 - Eliminar capa.	65
Tabla 4.6 Caso de prueba 6 - Editar perfil del propio usuario.	65
Tabla 4.7 Caso de prueba 7 - Eliminar perfil del propio usuario.	66
Tabla 4.8 Caso de prueba 8 - Visualizar información de otros usuarios.	66
Tabla 4.9 Caso de prueba 9 - Eliminar otros usuarios.	67
Tabla 4.10 Caso de prueba 10 - Cambiar de rol a otros usuarios.	67
Tabla 4.11 Caso de prueba 11 - Visualizar las capas en un mapa.	68
Tabla 4.12 Caso de prueba 12 - Log out.	68
Tabla 4.13 Caso de prueba 13 - Editar privacidad del usuario.	69
Tabla 4.14 Caso de prueba 14 - Añadir foto de perfil.	69
Tabla 4.15 Caso de prueba 15 - Cambiar contraseña.	70
Tabla 4.16 - Especificación de clases de equivalencia.	74
Tabla 4.17 - Pruebas de clases de equivalencia.	75

Introducción

El proyecto que hemos realizado mi compañera Andrea Fernández Sánchez y yo, TecnoMaps, consiste en una aplicación web centrada en la gestión y visualización de mapas territoriales, un tema que pretende asemejarse a lo que realizan en la empresa donde hemos hecho las prácticas, Tecnosylva.

Como nota aclaratoria, señalar que este proyecto ha sido llevado a cabo en un equipo de dos personas: Andrea Fernández Sánchez, que realizó el Backend y yo. Durante todo este trabajo me centraré en el Frontend, la parte visual de la aplicación, de la cual me he encargado yo.

PLANTEAMIENTO DEL PROBLEMA

La empresa, Tecnosylva, que trabaja en la detección de incendios y una de sus partes clave son los mapas, nos propuso como proyecto a realizar durante las prácticas en empresa a Andrea y a mí elaborar una aplicación web en una versión mucho más reducida de lo que realizan en la empresa, pero con tecnologías similares.

Nuestra aplicación también consiste, en menor medida y complejidad, en usar mapas. En este caso, un mapa de base y poder gestionar y administrar una serie de capas que se pueden aplicar sobre dicho mapa. Además de esto, se ofrece la modificación de dichas capas y un sistema de identificación de usuario por medio de nombre de usuario y contraseña (login), en el que cada usuario tiene su propio mapa y puede gestionar sus propias capas. Todo ello implementado con la tecnología que la empresa nos dictó, además de ofrecernos una base de datos local para trabajar con ella.

OBJETIVOS

El proyecto TecnoMaps se basó en desarrollar una aplicación web que permita la visualización de mapas y capas a la que se acceda mediante un registro de usuario y contraseña (login), que además pueda gestionar capas, usuarios y

mapas, todo ello estructurado en Backend y Frontend. La empresa nos ha concedido acceso a una base de datos hecha solo y exclusivamente para este proyecto, a la que conectamos vía ssh y que seguimos utilizando a pesar de haber terminado las prácticas allí.

En mi caso, mi parte consistía en diseñar el Frontend, la parte más visual y accesible del proyecto que estaría conectada al Backend. Teniendo en cuenta los objetivos principales propuestos por la empresa, en los que se pedía un menú de login para el usuario, gestión y visualización de capas y gestión y visualización de usuarios, decidimos dividirlo en diferentes pantallas, que serían cada apartado de los objetivos propuestos mencionados anteriormente. En este caso la primera pantalla sería el login, que tendría lo necesario para que el usuario cree su perfil (sign in) o se registre (log in). Acto seguido aparecería la pantalla con el mapa base sobre el que se muestran las capas del usuario y el menú donde seleccionar los siguientes apartados, cada uno con su respectiva pantalla:

- Añadir capa: introduciendo determinados valores se crea una capa.
- Gestión de capas: listado de todas las capas que ha añadido el usuario, donde también se permite modificar o eliminar dichas capas.
- Gestión de usuarios: el administrador, el usuario con más permisos, es el único que puede acceder a esta página donde se muestran todos los usuarios registrados en la aplicación, junto con alguna información de su perfil y la opción de darles permisos o eliminarlos si fuera necesario.
- Perfil de usuario: el propio usuario puede añadir o editar información personal, además de poder eliminar su propia cuenta si lo desea.

METODOLOGÍA

Durante el desarrollo del proyecto se utilizaron dos metodologías diferentes: El modelo en cascada y la metodología ágil scrum.

Para la primera parte del proyecto en la que se trabajó en la empresa se utilizó un **modelo en cascada**, que incluye un desglose de las actividades del proyecto en una secuencia lineal de etapas, donde cada etapa depende de los entregables de la etapa anterior. En el desarrollo de software, a menudo es uno de los métodos

menos iterativos y flexibles, porque el progreso se lleva a cabo principalmente a través de una fase descendente, que incluye análisis de requisitos, diseño del sistema, codificación, prueba, implementación, evaluación del programa y mantenimiento.

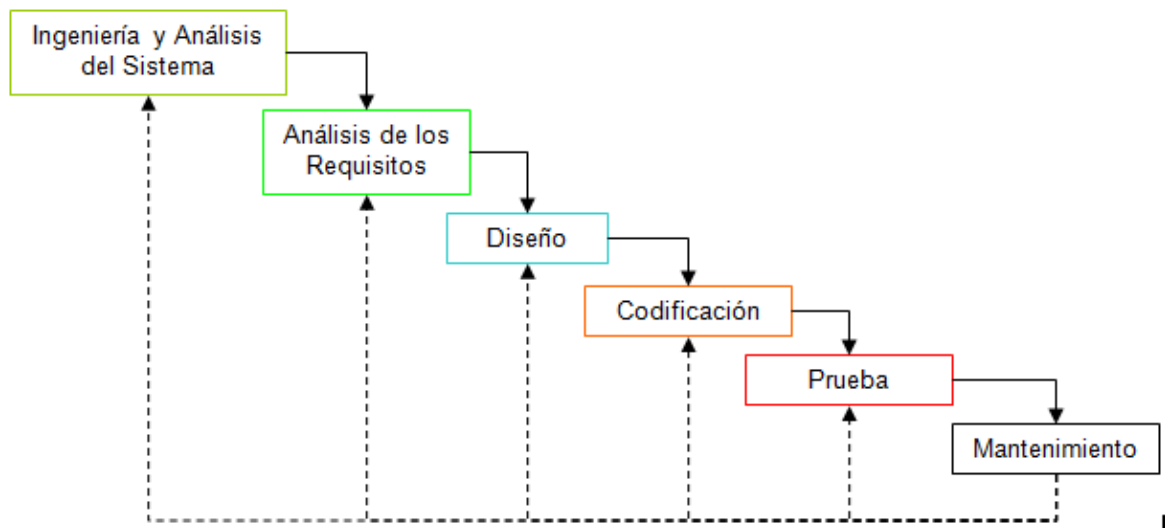


Figura 0.1.1 Esquemas de las fases del modelo en cascada. (Fuente: [1])

Nosotras elegimos este modelo al principio porque no conocíamos otras metodologías en aquel momento. Una vez que cursamos la asignatura de Ingeniería del Software II decidimos decantarnos por un marco de trabajo de una metodología ágil como es scrum, ya que nos aporta una visión de avance en menor tiempo.

En Scrum existen cuatro partes a destacar:

- **La pila de producto**, que es la parte esencial del Scrum, es, básicamente, una lista ordenada de historias o funcionalidades que el cliente quiere, descritas usando la terminología del cliente. Normalmente se denominan historias o elementos de la Pila.
- El **Sprint** es un periodo de corta duración (2-4 semanas) en el que se crea un producto, un prototipo operativo. Las características que van a implementarse en el Sprint provienen de la Pila del Producto (Product Backlog), que contiene una serie de historias de usuario ordenadas.

- **Reuniones:** se realizan varias reuniones para documentar todo el proceso de elaboración del producto, las historias realizadas, las no realizadas, incidencias, cosas a mejorar... Existe una reunión de planificación donde se elabora el plan de trabajo, se deciden qué historias se incluirán en cada sprint, etc. También se hacen reuniones antes y después de cada sprint en el que cada miembro pone en común con el resto de compañeros lo que se ha realizado, lo que no, problemas que han surgido, sugerencias...que son la reunión de planificación del Sprint y la reunión de retrospectiva. También se realizan unas reuniones diarias (daily) de no más de 15 minutos normalmente antes de empezar con el trabajo del día. Estas reuniones sirven para que cada participante ponga en común con el resto lo que ha hecho, lo que no ha hecho, dificultades o sugerencias...
- **Roles:** en scrum hay tres roles principales, el product owner que generalmente es un representante del cliente, el scrum master que es el que se encarga de liderar en la planificación de los Sprints al resto de desarrolladores y el equipo de desarrollo. En nuestro caso, ambas realizaremos todos los roles dado que no hay un cliente "real".

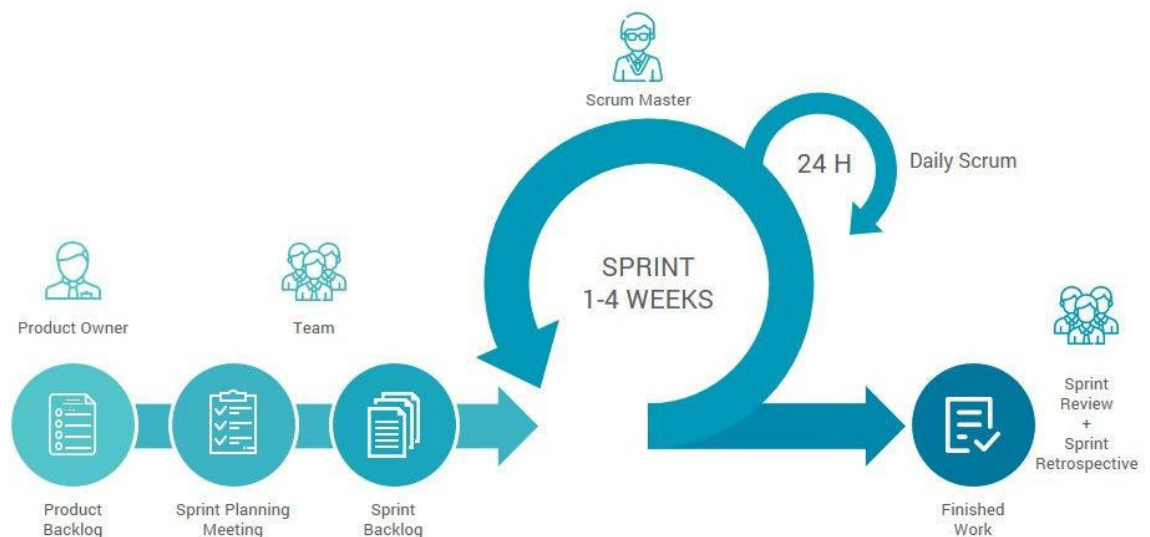


Figura 0.1.2 Esquema de la metodología Scrum. (Fuente: [2])

ESTRUCTURA DEL TRABAJO

El trabajo está estructurado en cuatro partes fundamentales que se corresponden a las diferentes fases del proyecto. Estas son:

- Un **Estudio del problema** donde se dará contexto al trabajo, cómo surgió, se mencionará el estado de la cuestión en torno a este tipo de aplicaciones o herramientas a usar en el trabajo que se pueden encontrar actualmente y, por último, se dará una definición del problema profundizando en qué se pretende resolver con este proyecto.
- En segundo lugar, se hablará de la **gestión del proyecto software**. En este apartado se realizará una simulación de cómo se gestionaría este producto software en el entorno empresarial, realizando un análisis del alcance del proyecto, definiendo el plan de trabajo con sus tareas, y de la gestión de recursos y de riesgos a lo largo del desarrollo de la aplicación.
- El tercer punto tratará de la **solución** final, donde se dará una descripción exhaustiva de esta: se explicará el proceso de desarrollo sobre el análisis de requisitos, funcionalidades, el diseño, la implementación, herramientas usadas, y finalmente se mostrará el producto del desarrollo por medio de una ejecución mostrando todas las funcionalidades de la aplicación detalladamente.
- En el cuarto punto se documentará la **evaluación** del producto software donde se observará la validez de la solución final. Se explicará el proceso de evaluación hablando de los métodos utilizados y mostrando los casos de prueba realizados para su verificación con su resultado. También se hará un análisis de los resultados obtenidos en dichas pruebas de evaluación y si se han cumplido los objetivos planteados para esta aplicación.

Finalmente se expondrán las conclusiones tanto personales como objetivas obtenidas después de realizar este proyecto, así como posibles mejoras de la aplicación en el futuro y los problemas que se han encontrado durante toda la elaboración del proyecto.

1. Estudio del problema

En este capítulo se explicará cómo surgió el proyecto, cuáles son los problemas a resolver con esta aplicación, de qué manera se decidió llevar a cabo y su posible solución, así como una revisión de la tecnología a usar y aplicaciones existentes en este campo.

1.1 EL CONTEXTO DEL PROBLEMA

El proyecto TecnoMaps surge cuando la empresa asignada para la asignatura de Prácticas en Empresa, Tecnosylva, nos propone realizar esta aplicación con unos requisitos concretos, concretamente, una aplicación web para la gestión y visualización de mapas y capas, junto con registro y gestión de usuarios.

Tecnosylva se dedica al sector forestal y de incendios forestales ofreciendo sistemas basados en GIS para la prevención de incendios, emergencias y planificar su posible respuesta a partir de datos obtenidos por sistemas de información geográfica (SIG), teledetección satelital y datos LiDAR. [3]

El trabajo en la empresa duró 3 meses, resultando en una aplicación totalmente funcional que cumplía con los requisitos que se pedían. Viendo que la aplicación nos resultó interesante, decidimos continuar con ella para presentarla como TFG y para seguir mejorándola e implementando más funciones que además nos servirían como trabajo final en la asignatura de 4º curso Ingeniería del Software II.

Dentro del equipo que formamos mi compañera Andrea y yo, mi tarea consiste en desarrollar la parte del Frontend, la parte visible de la aplicación y sobre la que haré hincapié a lo largo del trabajo.

1.2 EL ESTADO DE LA CUESTIÓN

1.2.1 Tecnología existente

El trabajo se empezó de cero, puesto que antes del proyecto nunca habíamos pensado realizar una aplicación gestora de mapas hasta estar haciendo prácticas con esta empresa. A partir de que nos dieran la propuesta de aplicación hicimos una pequeña búsqueda para encontrar herramientas que utilizaran los mismos métodos en internet o que la funcionalidad se asemejara a lo que pretendíamos implementar.

La empresa nos propuso trabajar con diferentes opciones de framework para Frontend, como Vue.js o Bootstrap. En mi caso, me pareció mejor opción Vue dado que es un framework que tanto Andrea como yo habíamos usado en aplicaciones que habíamos desarrollado en asignaturas anteriores y del que consideramos que tenemos experiencia, lo cual nos facilitaría el trabajo.

Además del framework tendríamos que usar un complemento para la visualización de mapas que pudiera ir sobre Vue, en este caso nos recomendaron OpenLayers, una biblioteca que permite mostrar mapas interactivos en páginas web. Así que tuvimos que buscar la manera de implementar mapas y capas dentro de la aplicación sobre esta plataforma. Por suerte existe VueLayers [4], una biblioteca de componentes de Vue.js que trae la API de OpenLayers [5]. Esta biblioteca puede mostrar mapas con capas de mosaico, ráster o vectoriales cargadas desde diferentes fuentes.

Una vez que tuvimos esta herramienta fue necesario buscar una fuente de donde obtener mapas y capas. La empresa nos recomendó Bing, ya que tienen material de código abierto y público al que es fácil acceder. Otra alternativa que buscamos fue Google Maps, pero fue rechazada porque no proporcionan mucha información pública al respecto.

Dado que la fuente para los mapas base sería Bing, las capas deberían ser compatibles con el proveedor. Aquí nos encontramos con el primer problema de la aplicación, y es que existen pocas capas que fueran públicas y que además pudiéramos utilizar en la aplicación, debido a que cada capa tenía un formato diferente y nuestra aplicación solo acepta uno. Afortunadamente, encontramos

algunas capas que funcionan perfectamente en la aplicación y que nos permitieron realizar pruebas en ella.

1.2.2 ArcGIS

Además de buscar las herramientas que usaríamos en la app, tratamos de buscar ejemplos de aplicaciones que pudieran ser similar (en una mayor escala) a lo que pretendíamos elaborar. Así fue como encontramos ArcGIS, un producto de software que trabaja con Sistemas de Información Geográfica, (SIG). [6]



Figura 1.1 Logo de ArcGIS (Fuente: [7])

Este producto desarrollado por Esri agrupa diferentes herramientas como ArcReader, ArcMap, ArcCatalog, ArcToolbox, ArcScene y ArcGlobe con el fin de capturar, editar, analizar, diseñar, tratar y publicar información geográfica. [7] Algunas de las herramientas permiten entre otras funciones:

- Crear, consultar y analizar datos ráster, combinar capas, realizar mediciones y cálculos, hallar ubicaciones
- Creación y visualización de datos en un entorno tridimensional
- Análisis geoestadístico
- Cálculo de rutas optimas

Como se puede apreciar, esta aplicación posee muchos usos y funciones diferentes lo cual nos podía ayudar a la hora de tratar de obtener más funcionalidades e ideas a implementar en nuestra aplicación. A pesar de ello, considero que sería algo irreal seguir este modelo de aplicación ya que es bastante más compleja que las tecnologías que nos han recomendado usar y nuestro propio conocimiento sobre el tema.

1.3 LA DEFINICIÓN DEL PROBLEMA

El problema que debíamos resolver con la aplicación propuesta por la empresa consistía en implementar una aplicación web que una vez creado un usuario y que se iniciara sesión, permitiera la creación, edición y visualización de capas en un mapa base, todo ello realizado con una gestión de usuarios y sus respectivas capas con una base de datos.

En nuestro caso, la solución que nos propusieron era bastante más sencilla y reducida que el trabajo que realizan en la misma empresa usando mapas, aunque la nuestra consistiría en resolver una pequeña parte del mismo problema, con la misma temática y con similares herramientas con las que trabajan, pero a menor escala.

El problema propuesto por la empresa, como ya se ha comentado consiste en crear una aplicación web que debe tener cinco partes diferenciadas:

- Una parte del login, que contiene el registro de nuevos usuarios y el inicio de sesión de los usuarios ya registrados.
- Una pantalla principal donde se mostrará el mapa base y se aplicarán las capas creadas por el usuario.
- Una pantalla para crear cada capa introduciendo ciertos valores que se nos indicaron.
- Una pantalla para gestionar las capas creadas por el usuario donde se podrán editar campos de las capas o eliminarlas si se desea.
- Una pantalla para gestionar los usuarios donde los usuarios con el rol de administrador podrán ver los datos de otros usuarios, así como eliminarlos.

La empresa nos dio mucha libertad para diseñar la aplicación, con solamente los requisitos mostrados anteriormente, por lo tanto, nosotras pudimos diseñar con total libertad la interfaz y la estructura de la aplicación.

Además de todo lo anterior y después de haber terminado las prácticas en la empresa, decidimos continuar con el proyecto, añadiendo más funcionalidad a la aplicación y mejorándola. Entre otras cosas:

- Crear una pantalla nueva de perfil de usuario donde este podrá añadir más datos sobre su perfil.
- Crear un apartado para poder subir una foto de perfil, a través de una URL de una imagen y que esta se almacene en la base de datos.
- Añadir otros campos opcionales de información de usuario como el nombre y apellidos, número de teléfono, dirección, foto de perfil, organización... dentro de la página del perfil del usuario.
- Botón para poder eliminar su cuenta el propio usuario, no solo a través del administrador.
- Pequeña mejora de la interfaz, así como mejorar el texto que aparece traducéndolo íntegramente al inglés.
- Posibilidad de editar el propio perfil de usuario pudiendo establecerlo como privado o público.
- Los administradores pueden visualizar los perfiles de otros usuarios. Se ha creado un rol intermedio (moderador) que se ve afectado por la privacidad de los usuarios estándar.
- Botón para que un administrador pueda dar permisos a otros usuarios de un rol inferior.

Todos estos problemas que debíamos solucionar con las funcionalidades mencionadas en este apartado fueron resueltos satisfactoriamente.

Las mayores limitaciones de este proyecto fueron la escasez de mapas y capas encontrados que fueran de dominio público. A pesar de ello, se consiguió obtener el principal objetivo, que era encontrar algunas capas que servirían para comprobar que la aplicación funcionaba correctamente.

2. Gestión de proyecto software

Debido a que este proyecto no tiene un fin comercial ni está destinado para ser usado por una empresa, se realizará una simulación empresarial de dicho proyecto, definiendo el alcance, objetivos, tareas, recursos, riesgos, plan de trabajo, presupuesto, etc.

2.1 ALCANCE DEL PROYECTO

2.1.1 Definición del proyecto

El proyecto consiste en desarrollar una aplicación web para la gestión y visualización de capas y mapas también pudiendo gestionar los usuarios de la misma. Los posibles usuarios de esta aplicación no requieren de ningún conocimiento básico de informática. Esta aplicación está pensada para hacer lo más simple y accesible posible las funcionalidades propuestas.

El público objetivo para la comercialización de esta aplicación serían empresas que necesitaran una herramienta para gestionar y visualizar mapas y capas, como puede ser Tecnosylva, una estación meteorológica, cartógrafos... también podría ser utilizada por aficionados ya que la aplicación solo requiere tener conocimientos básicos sobre mapas.

Sobre la interfaz, esta se realizará de la manera más intuitiva y accesible posible, facilitando a gente con alguna discapacidad visual o intelectual su uso y por supuesto tratando de seguir estándares de accesibilidad. El idioma utilizado en la interfaz de la aplicación será el inglés, dado que se pretende que esta pueda ser usada internacionalmente.

Dicho todo lo anterior, la aplicación también tendrá unos límites:

- Solo se asegura su funcionamiento para el sistema operativo Windows.
- Solo se encuentra en inglés.
- No se requerirá ningún otro programa o aplicación complementario para su uso.

2.1.2 Estimación de tareas y recursos

Una vez que se obtienen los principales objetivos de la aplicación se planteará una división de entregas teniendo en cuenta la posible dificultad de cada tarea y el tiempo que podría llevar realizarla para que estas divisiones sean equilibradas y tengan el mismo peso.

Dado que este proyecto cabalga entre un modelo en cascada y una metodología ágil como lo es el scrum, se ha decidido que para esta planificación se realice totalmente en scrum, ya que esto proporciona al cliente entregas regulares como son los Sprint, donde se puede ver el avance del proyecto y puede proporcionar retroalimentación por su parte.

Primeramente, se deberán escoger aquellas tareas que sean esenciales para la aplicación y que no dependan de alguna tarea anterior. El resto de tareas se intentarán colocar en cada Sprint según la dependencia que tenga con el resto de tareas y la prioridad que se les asigne.

En cuanto a recursos necesarios para el desarrollo del proyecto se debe tener en cuenta tres partes diferentes:

- El equipo y la tecnología a utilizar: (ordenadores, servidor, hardware en general, sistema operativo, licencias de otro tipo, etc.)
- Mano de obra: en este caso mi compañera Andrea y yo contaremos como programadoras.
- Tiempo: una estimación de ocho meses para la entrega final del proyecto (octubre 2019 – junio 2020)

En los siguientes apartados se tratará de desarrollar detalladamente la elaboración de un presupuesto final mediante la elaboración de un análisis de tareas, identificación de recursos y de riesgos que conlleva el desarrollo de este producto software.

2.2 PLAN DE TRABAJO

2.2.1 Identificación de tareas

En este apartado se tratará de identificar las diferentes fases y tareas del proyecto, para ello se realizará un análisis a grandes rasgos de las principales fases sin indagar en las fases más “profundas”, pues estas serán analizadas e identificadas más detalladamente en el siguiente apartado.

En este caso, no se tendrán en cuenta solo el proceso de desarrollo más técnico de la aplicación en sí, si no todo el proceso del proyecto, teniendo en cuenta el enfoque empresarial que se le quiere dar. Estas son las principales partes de este:

- Planificación del proyecto: en esta parte se dejarán claros los objetivos principales de la aplicación con el cliente tratando de determinar los requisitos y los casos de uso de esta por medio de reuniones, así como determinar la tecnología que se deberá usar.
- Desarrollo del proyecto: esta fase consiste en el desarrollo de la aplicación. Primero se realiza la planificación de los casos de uso e historias para cada Sprint que se realice, después se realiza cada iteración donde se implementa la aplicación.
- Revisión final: una vez realizada la implementación se pasa a revisar todo el proyecto por medio de pruebas como las de caja negra, casos de uso o cuestionarios para verificar que se cumplen todos los requisitos.

2.2.2 Estimación de tareas

Una vez identificadas las fases y las tareas se procede a definirlas y dar un análisis más detallado de estas realizando una estimación.

- Para la fase de **planificación del proyecto** se contará con las siguientes tareas: reunión con el cliente, análisis de requisitos y de casos de uso, aprendizaje de las tecnologías a emplear, identificación de los riesgos y de los recursos. En Scrum también se incluye una reunión de planificación de los Sprints.
- La fase del **desarrollo del proyecto** se compone de 8 iteraciones o Sprints, cada una con su parte de planificación, desarrollo y retrospectiva:

- **Sprint 1:** reunión de planificación del Sprint 1, instalación y puesta en punto del framework y del proyecto, diseño de la estructura de la aplicación web, diseño de las partes fijas, implementación de requisitos y casos de uso, productos entregables y finalmente reunión de revisión y reunión de retrospectiva del Sprint 1.
- **Sprint 2:** reunión de planificación del Sprint 2, continuación de la implementación de requisitos y casos de uso, mejora y continuación con los productos entregables para este sprint (creación de la página principal de registro de usuarios y la página principal del mapa...), estudio de tecnologías a usar, evaluación del entregable con los casos de uso y finalmente reunión de revisión y reunión de retrospectiva del Sprint 2.
- **Sprint 3:** reunión de planificación del Sprint 3, continuación de la implementación de los casos de uso y requisitos, implementación y mejora de los productos entregables para este sprint (login, capa principal de mapas, barras de menú, creación de la página de añadir capas...) con las respectivas historias de usuario, evaluación del entregable con los casos de uso, reunión de revisión y por último reunión de retrospectiva del Sprint 3.
- **Sprint 4:** reunión de planificación del Sprint 4, continuación de la implementación de requisitos y casos de uso, mejora e implementación de los productos entregables para este sprint (página de mapas, página de añadir capas, pruebas con capas, creación de la página de gestión de capas...) con las respectivas historias de usuario, evaluación del entregable con los casos de uso, reunión de revisión y reunión de retrospectiva del Sprint 4.
- **Sprint 5:** reunión de planificación del Sprint 5, continuación de la implementación de casos de uso y requisitos, implementación y mejora de los productos entregables para este sprint (mejora de página de añadir capas y gestionar capas, creación de la página de gestión de usuarios) con las respectivas historias de usuario, evaluación del entregable con los casos de uso, reunión de revisión y por último reunión de retrospectiva del Sprint 5.

- **Sprint 6:** reunión de planificación del Sprint 6, continuación de la implementación de requisitos y casos de uso, mejora e implementación de los productos entregables para este sprint (mejora de página de gestión de usuarios y retoques en login, y cambio de idioma) con las respectivas historias de usuario, evaluación del entregable con los casos de uso, reunión de revisión y finalmente reunión de retrospectiva del Sprint 6.
- **Sprint 7:** reunión de planificación del Sprint 7, continuación de la implementación de casos de uso y requisitos, implementación y mejora de los productos entregables para este sprint (creación de la página de perfil de usuario, y mejora general de interfaz) con las respectivas historias de usuario, evaluación del entregable con los casos de uso, reunión de revisión y reunión de retrospectiva del Sprint 7.
- **Sprint 8:** reunión de planificación del Sprint 8, continuación de la implementación de casos de uso y requisitos, mejora e implementación de los productos entregables para este sprint (continuación con la página de perfil de usuario, mejora general de interfaz y últimos retoques) con las respectivas historias de usuario, evaluación del entregable con los casos de uso, reunión de revisión y finalmente reunión de retrospectiva del Sprint 8.
- Para finalizar, se realiza una **revisión final del proyecto**, donde se realizará un análisis de los requisitos y casos de uso al entregable final por medio de una prueba de caja negra para comprobar que cumple satisfactoriamente con los requisitos que se debían implementar en la fase anterior. Por último, se realiza un informe con el contenido actualizado de los casos de usos y pruebas realizadas, incluyendo los objetivos que se han conseguido y los que no.

2.2.3 Planificación de tareas

2.2.3.1 Planificación general

Teniendo en cuenta el apartado anterior y que se usará una metodología ágil como lo es Scrum, es necesario planificar las posibles fechas de iteraciones (Sprint), entregas y reuniones que se harán durante la duración del proyecto.

Las fechas de estimación del proyecto son las siguientes, como fecha de inicio el 4 de octubre de 2019 y como final el 1 de junio de 2020, en total 9 meses de duración:

Reunión planificación general: 4 octubre 2019

Sprint 1:

- Reunión planificación Sprint 1: 7/10/2019
- Sprint 1: 7/10/2019– 31/10/2019
- Reunión diaria (daily): todos los días antes de comenzar a trabajar
- Entregable Sprint 1: 31/10/2019
- Reunión revisión Sprint 1: 31/10/2019
- Reunión retrospectiva Sprint 1: 31/10/2019

Sprint 2:

- Reunión planificación Sprint 2: 4/11/2019
- Sprint 2: 4/11/2019– 29/11/2019
- Reunión diaria (daily): todos los días antes de comenzar a trabajar
- Entregable Sprint 2: 29/11/2019
- Reunión revisión Sprint 2: 29/11/2019
- Reunión retrospectiva Sprint 2: 29/11/2019

Sprint 3:

- Reunión planificación Sprint 3: 2/12/2019
- Sprint 3: 2/12/2019– 20/12/2019
- Reunión diaria (daily): todos los días antes de comenzar a trabajar
- Entregable Sprint 3: 20/12/2019
- Reunión revisión Sprint 3: 20/12/2019
- Reunión retrospectiva Sprint 3: 20/12/2019

Sprint 4:

- Reunión planificación Sprint 4: 8/1/2020
- Sprint 4: 8/1/2020– 31/1/2020
- Reunión diaria (daily): todos los días antes de comenzar a trabajar
- Entregable Sprint 4: 31/1/2020
- Reunión revisión Sprint 4: 31/1/2020
- Reunión retrospectiva Sprint 4: 31/1/2020

Sprint 5:

- Reunión planificación Sprint 5: 3/2/2020
- Sprint 5: 3/2/2020– 28/2/2020
- Reunión diaria (daily): todos los días antes de comenzar a trabajar
- Entregable Sprint 5: 28/2/2020
- Reunión revisión Sprint 5: 28/2/2020
- Reunión retrospectiva Sprint 5: 28/2/2020

Sprint 6:

- Reunión planificación Sprint 6: 2/3/2020
- Sprint 6: 2/3/2020– 27/3/2020
- Reunión diaria (daily): todos los días antes de comenzar a trabajar
- Entregable Sprint 6: 27/3/2020
- Reunión revisión Sprint 6: 27/3/2020
- Reunión retrospectiva Sprint 6: 27/3/2020

Sprint 7:

- Reunión planificación Sprint 7: 13/4/2020
- Sprint 7: 13/4/2020– 30/4/2020
- Reunión diaria (daily): todos los días antes de comenzar a trabajar
- Entregable Sprint 7: 30/4/2020
- Reunión revisión Sprint 7: 30/4/2020
- Reunión retrospectiva Sprint 7: 30/4/2020

Sprint 8:

- Reunión planificación Sprint 8: 4/5/2020
- Sprint 8: 4/5/2020– 29/5/2020
- Reunión diaria (daily): todos los días antes de comenzar a trabajar
- Entregable Sprint 8: 29/5/2020
- Reunión revisión Sprint 8: 29/5/2020
- Reunión retrospectiva Sprint 8: 29/5/2020

Reunión para la revisión final del proyecto: 1/06/2020.

Entregable final: 1/06/2020.

2.3 GESTIÓN DE RECURSOS

En este apartado se tratará de hacer una estimación de los recursos humanos necesarios para realizar este proyecto también calculando el tiempo que han empleado para ello.

2.3.1 Especificación de recursos

Como recursos humanos, he considerado que no se requeriría de mucho personal puesto que la aplicación no es muy compleja. Se tendrán en cuenta las diferentes habilidades necesarias para realizar un proyecto de este tipo. Estos han sido los diferentes perfiles que se requerirán para este proyecto:

- 1 diseñador web para la parte de diseño de interfaz de la aplicación.
- 1 desarrollador web que implemente la estructura y el Backend de la aplicación.
- 1 analista de sistemas que se encargue de encontrar problemas o verificar que se cumplan las funciones previstas.
- 1 técnico de sistemas que se encargue del servidor, base de datos, etc.

En la siguiente tabla se puede ver un desglose de todos los recursos con lo que sería el salario de cada especialidad actualmente (2020).

El salario por hora ha sido calculado con la siguiente formula:

Salario por hora = Media Salario Anual / (251 días laborables * 8 horas diarias)

Tabla 2.1 Media nacional y cálculo por hora del salario de las especialidades requeridas

Recurso	Salario anual €	Salario/hora €
Diseñador web	20.243 €	10,12 € / h
Desarrollador web	23.531 €	11,76 € / h
Analista de sistemas	31.068 €	15,53 € / h
Técnico de sistemas	21.457 €	10,72 € / h

El salario medio anual de cada especialidad ha sido obtenido de la página “indeed”: para el Diseñador web [8], Desarrollador Web [9], Analista de sistemas [10] y Técnico de sistemas [11].

2.3.2 Asignación de recursos

Con toda la información de personal y salarios calculada, se tratará de hacer una estimación de las horas que le dedicará a la aplicación cada empleado según las tareas desarrolladas en el proyecto. La duración de cada trabajador en la empresa se ha realizado a mi propio juicio tratando de que la especialidad de cada trabajador se adecúe a las fases donde se requiere más su trabajo.

Las horas trabajadas totales se han calculado multiplicando los días trabajados por el número de horas trabajadas al día.

Tabla 2.2 Periodo de tiempo trabajado y cálculo de las horas trabajadas

Recurso	Comienzo	Fin	Días trabajados	Horas trabajadas
Diseñador web	7/10/2019	22/05/2020	155	1240h
Desarrollador web	7/10/2019	29/05/2020	160	1280h
Analista de sistemas	7/10/2019	1/06/2020	161	1288h
Técnico de sistemas	4/11/2019	1/06/2020	142	1136h

2.4 GESTIÓN DE RIESGOS

En este punto se detalla los posibles riesgos del proyecto, la probabilidad de que ocurran, el impacto que tendrían de darse y las estrategias propuestas para reducir los riesgos.

2.4.1 Identificación de riesgos

En la siguiente tabla se especifican los riesgos de mayor importancia que podrían darse en este proyecto. Para ello se han tenido en cuenta diferentes modalidades de riesgos como los riesgos de proyecto, riesgos técnicos, riesgos de negocio...

Tabla 2.3 Especificación de riesgos de la aplicación

Nombre riesgo	Riesgo	Impacto	Tipo
R-1	Fallo técnico de los ordenadores	Muy alto	Genérico
R-2	Fallo de la base de datos	Alto	Específico
R-3	Modificación de interfaz	Bajo	Específico
R-4	Cambio de requerimientos	Bajo	Genérico
R-5	Retraso de entregables	Medio	Genérico
R-6	Falta de experiencia	Medio	Específico
R-7	Aumento del presupuesto	Bajo	Genérico

2.4.2 Análisis de riesgos

Ahora se procederá a analizar detalladamente cada riesgo indicado en la tabla anterior.

Tabla 2.4 Riesgo 1 - Fallo técnico del equipo.

[R-1] Fallo técnico de los ordenadores	
Categoría	Técnico
Precondición	Mal uso o fallo impredecible de algún dispositivo hardware.
Consecuencia	No se podrá trabajar en ese equipo o equipos hasta que el técnico solucione el problema. Posibles retrasos en la entrega de producto software.

Tabla 2.5 Riesgo 2 - Fallo de la base de datos.

[R-2] Fallo de la base de datos	
Categoría	Técnico
Precondición	Mal uso o fallo impredecible de la base de datos
Consecuencia	No se podrá continuar con el desarrollo de la aplicación hasta que el técnico arregle el problema. Posibles retrasos en la entrega de producto software.

Tabla 2.6 Riesgo 3 - Modificación de interfaz.

[R-3] Modificación de interfaz	
Categoría	Proyecto
Precondición	El cliente solicita un cambio de diseño
Consecuencia	Cambio de diseño y estructura de la interfaz por parte del diseñador web. Se deberán cambiar la planificación de tareas y costes. Un cambio muy grande podría conllevar a que la parte del diseño se atrase y hubiera un retraso en la entrega de producto software.

Tabla 2.7 Riesgo 4 - Cambio de requerimientos.

[R-4] Cambio de requerimientos	
Categoría	Proyectos
Precondición	El cliente solicita un cambio de requisitos
Consecuencia	Los desarrolladores deberán adaptar la aplicación a los nuevos requisitos requeridos. Se deberán cambiar la planificación de tareas y costes Si este cambio es muy brusco podría conllevar retrasos en la entrega de producto software.

Tabla 2.8 Riesgo 5 - Retraso de entregables.

[R-5] Retraso de entregables	
Categoría	Proyecto
Precondición	Mala planificación, fallos técnicos, cambios en la aplicación, etc. que requieran más tiempo del calculado.
Consecuencia	El proyecto sufrirá un retraso en la entrega de producto software, se deberán cambiar la planificación de tareas y costes y este podría ser arrastrado hasta la entrega del producto final.

Tabla 2.9 Riesgo 6 - Falta de experiencia.

[R-6] Falta de experiencia	
Categoría	Proyecto
Precondición	Falta de estudio de las tecnologías, desarrolladores junior...
Consecuencia	La parte del trabajador con falta de experiencia llevará más tiempo realizarla y tendrá cierto retraso que requerirá de ayuda de otros trabajadores o de la contratación de más personal para realizar dichas tareas

Tabla 2.10 Riesgo 7 - Aumento del presupuesto.

[R-7] Aumento del presupuesto	
Categoría	Negocio
Precondición	Necesidad de contratar a más personal, mala predicción del presupuesto, algún motivo impredecible.
Consecuencia	El cliente deberá tomar la decisión de cancelar este proyecto con sus respectivas consecuencias o continuar con ello.

Como se puede apreciar en las tablas anteriores, los riesgos descritos para esta aplicación son bastante comunes y genéricos en productos software. Todo ello no evita que puedan surgir más problemas a lo largo del desarrollo que conlleven ciertos riesgos diferentes a los aquí descritos.

3. Solución

En este capítulo se tratará de explicar detenidamente el proceso del desarrollo de la solución, detallando las fases de análisis de requisitos, diseño, implementación y pruebas. También se desarrollará una explicación sobre el producto final, su estructura, flujo de la aplicación, etc.

3.1 EL PROCESO DE DESARROLLO

A continuación, se detallarán todos los pasos seguidos para el desarrollo de la solución: el análisis de requisitos, el diseño del sistema, la implementación realizada y las pruebas realizadas en esta.

3.1.1 Análisis

En la fase de análisis se fijarán los requisitos que se esperan implementar en la aplicación para solucionar el problema propuesto. Estos se diferenciarán entre requisitos funcionales y no funcionales.

3.1.1.1 Especificación de requisitos funcionales

A continuación, se especificará en unas tablas cada requisito funcional que se pretenderá implementar en la aplicación.

Los requisitos funcionales son aquellos que se refieren a los comportamientos o funciones específicas que deberá entregar el sistema

Tabla 3.1 Requisito funcional 1 - Pantalla de registro de usuarios.

[RF-1]: Pantalla de registro de usuarios	
Descripción	El sistema mostrará la pantalla donde se encuentra el registro de nuevos usuarios y de usuarios ya registrados
Prioridad	Alta
Estado	Desarrollado

Tabla 3.2 Requisito funcional 2 - Registro de usuarios.

[RF-2]: Registro de usuarios	
Descripción	El sistema permitirá el registro de nuevos usuarios en la aplicación.
Prioridad	Alta
Estado	Desarrollado

Tabla 3.3 Requisito funcional 3 - Log in de usuarios.

[RF-3]: Log in de usuarios	
Descripción	El sistema permitirá el acceso a la aplicación de usuarios ya registrados
Prioridad	Alta
Estado	Desarrollado

Tabla 3.4 Requisito funcional 4 - Log out de usuarios.

[RF-4]: Log out de usuarios	
Descripción	El sistema permitirá finalizar la sesión de usuario-
Prioridad	Alta
Estado	Desarrollado

Tabla 3.5 Requisito funcional 5 - Visualización de capas.

[RF-5]: Visualización de capas	
Descripción	El sistema permitirá a los usuarios mostrar las capas creadas sobre el mapa base.
Prioridad	Alta
Estado	Desarrollado

Tabla 3.6 Requisito funcional 6 - Creación de capas.

[RF-6]: Creación de capas	
Descripción	El sistema permitirá que a través de introducir ciertos parámetros se cree una capa y se añada a su perfil.
Prioridad	Alta
Estado	Desarrollado

Tabla 3.7 Requisito funcional 7- Edición de capas.

[RF-7]: Edición de capas	
Descripción	El sistema permitirá que los usuarios puedan editar los campos de las capas de su perfil.
Prioridad	Alta
Estado	Desarrollado

Tabla 3.8 Requisito funcional 8 - Eliminar capas.

[RF-8]: Eliminar capas	
Descripción	El sistema permitirá la eliminación de las capas que el usuario tiene registradas.
Prioridad	Alta
Estado	Desarrollado

Tabla 3.9 Requisito funcional 9 - Edición de la información de usuario.

[RF-9]: Edición de la información de usuario	
Descripción	El sistema permitirá a los usuarios editar los campos con información de su perfil.
Prioridad	Alta
Estado	Desarrollado

Tabla 3.10 Requisito funcional 10 - Gestión de usuarios.

[RF-10]: Gestión de usuarios	
Descripción	El sistema permitirá a los usuarios que tengan el rol de administrador visualizar y editar los roles del resto de usuarios.
Prioridad	Alta
Estado	Desarrollado

Tabla 3.11 Requisito funcional 11 - Roles de usuario.

[RF-11]: Roles de usuario	
Descripción	El sistema permitirá editar a los usuarios con el rol de administradores los roles del resto de usuarios.
Prioridad	Alta
Estado	Desarrollado

Tabla 3.12 Requisito funcional 12 -Eliminar cuenta de usuario.

[RF-12]: Eliminar cuenta de usuario	
Descripción	El sistema permitirá eliminar a los usuarios su propia cuenta y a los administradores cualquier cuenta de usuario.
Prioridad	Alta
Estado	Desarrollado

3.1.1.2 Especificación de requisitos no funcionales

A continuación, se especificará en unas tablas cada requisito no funcional que se pretenderá implementar en la aplicación.

Los requisitos funcionales son aquellos que se refieren a las propiedades emergentes de este, características generales y restricciones de la aplicación o sistema.

Tabla 3.13 Requisito no funcional 1 - Diseño web adaptable.

[RNF-1]: Diseño web adaptable	
Descripción	La aplicación deberá adaptarse a los diferentes formatos de pantallas de ordenador.
Prioridad	Media
Estado	Desarrollado

Tabla 3.14 Requisito no funcional 2 - Accesibilidad.

[RNF-2]: Accesibilidad	
Descripción	La aplicación deberá facilitar el acceso y el manejo de la aplicación a personas con discapacidades.
Prioridad	Media
Estado	Desarrollado

Tabla 3.15 Requisito no funcional 3 - Compatibilidad con navegadores.

[RNF-3]: Compatibilidad con navegadores	
Descripción	El sistema deberá permitir que se pueda acceder a la aplicación desde diferentes navegadores como Google Chrome, Mozilla Firefox, etc.
Prioridad	Media
Estado	Desarrollado

Tabla 3.16 Requisito no funcional 4 - Acceso a la aplicación.

[RNF-4]: Acceso a la aplicación	
Descripción	El sistema deberá permitir que solo los usuarios que se ha registrado en la plataforma puedan acceder al contenido de esta.
Prioridad	Alta
Estado	Desarrollado

Tabla 3.17 Requisito no funcional 5 - Empleo de Vue.

[RNF-5]: Empleo de Vue	
Descripción	El diseño de la interfaz de la aplicación web deberá estar desarrollado usando el Framework de JavaScript Vue.
Prioridad	Alta
Estado	Desarrollado

3.1.2 Diseño

En este apartado de la fase de diseño se explicará detalladamente el diseño del sistema, donde se desarrollará sobre la arquitectura del sistema y las tecnologías utilizadas en esta, y un apartado del diseño detallado de la interfaz.

3.1.2.1 Diseño de sistema

En la siguiente figura se muestra un esquema general de la arquitectura de la aplicación, en ese caso de una Single Page Application, este tipo de arquitectura consiste en que la aplicación, que como dice su nombre, se muestra en una sola página, permita que la experiencia del usuario sea mucho más fluida que con la estructura tradicional. La mayor ventaja de este tipo de aplicaciones es que no requieren que se actualice o cargue la página entera continuamente, si no una sola vez, lo que también mejora el rendimiento y la velocidad de carga de esta. A continuación, se explicarán las funciones principales de la parte del Frontend de esta arquitectura.

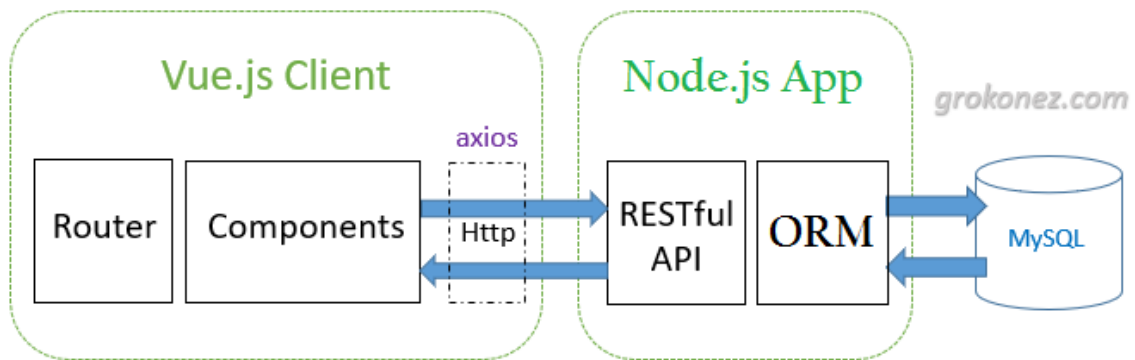


Figura 3.1 "Arquitectura con Vue.js + Nodejs RestAPIs – Sequelize ORM + MySQL"
(Fuente: [12])

Por un lado, está la parte del cliente, implementada usando Vue.js, el cual está compuesto de diferentes componentes que componen las diferentes páginas de la interfaz de la aplicación. Estos componentes están enrutados gracias a un Router, en nuestro caso con el módulo de Vue-Router. La comunicación entre cliente y servidor se realiza con Axios, un cliente HTTP basado en promesas que funciona tanto en el cliente (navegador) como en el servidor (en nuestro caso, Node.js). [13].

En cuanto a REST, este término se refiere a una arquitectura software que describe una interfaz entre sistemas que utiliza HTTP para realizar peticiones. Sus principales características son que el protocolo cliente/servidor no tiene estados, utiliza operaciones bien definidas como POST, GET, PUT, DELETE, y usa una sintaxis universal, entre otras. [14]

Para implementar esta aplicación se decidió usar un framework, una herramienta web que se compone de un conjunto de módulos que permiten desarrollar una aplicación de una manera ágil mediante librerías y funcionalidades ya existentes.

La gran ventaja de usar un framework ya elaborado es que no es necesario implementarlo, lo que permite ahorrar tiempo en el resto de desarrollo. Además, estos cuenta con una gran cantidad de librerías, una API y ayuda que permiten un desarrollo mucho más rápido y fluido. Este a su vez proporciona seguridad que un framework empezado desde cero no proporcionaría.

El gran inconveniente de usar un framework existente es que este requiere de experiencia, saber usarlo con fluidez. Si se desconociera el funcionamiento del

framework, se requeriría de una gran cantidad de tiempo empleado en el aprendizaje del mismo que se restaría del resto del tiempo del proyecto. También requiere de gran rendimiento, pues en el caso del framework que hemos elegido, este tiene una gran cantidad de librerías y recursos que muchas veces no son utilizados.

3.1.2.2 Diseño detallado

A continuación, se describirá el diseño y la estructura de la interfaz del sistema.

El Frontend, el cual se encarga de la parte visual de la interfaz, está estructurado en diferentes ficheros .vue llamados componentes ya que Vue se basa en los Single File Components. Cada componente, en nuestro caso, es una página diferente de la aplicación. Todos los ficheros .vue siguen la misma estructura:

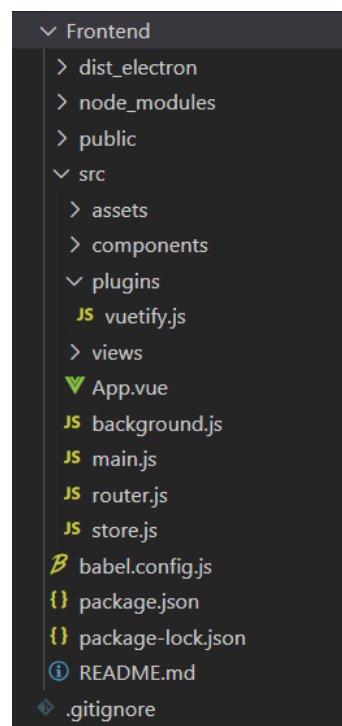


Figura 3.2 Directorio con todas las carpetas y archivos de la implementación.

- Una parte del “template” donde aparece todo el código relacionado íntegramente con la estructura y el diseño de la aplicación entre las etiquetas `<template></template>`. En esta parte no se manejan ni almacenan datos.

- La parte del “script” donde se guardan variables o datos y se elaboran funciones con estos además de enviar los datos al Backend. Entre las etiquetas `<script></script>`
- La parte puramente de estilo css que en nuestro caso apenas se usa y viene dada por las etiquetas `<style></style>`

Para nuestra aplicación son necesarios también otros ficheros:

- router.js, que es el que se encarga de enrutar las diferentes vistas o componentes de la aplicación.
- main.js, donde se encuentran las librerías usadas y se declaran los componentes.
- store.js, que utiliza Vuex para almacenar datos como las sesiones en el Frontend.
- vuetify.js, el componente de Vue que se encarga de la parte del diseño de componentes utilizando Material Design.

En mi caso, me centraré en la parte del diseño gráfico de la aplicación, pues fue mi contribución más importante a este proyecto.

Una de las grandes ventajas que posee Vuetify es que su diseño se realiza mediante pequeños componentes prefabricados, como, por ejemplo, tablas, campos de información para formularios, botones, etc. Altamente customizables y que permiten una amplia libertad a la hora de escoger el estilo de la aplicación. Además, posee un sistema grid que permite que las aplicaciones sean “responsive” y se adapten a cualquier tipo de formato de pantalla usando “containers”, filas, columnas y alineación para un diseño limpio, exacto y accesible.

Para TecnoMaps se decidió entre mi compañera y yo que la aplicación siguiera un estilo minimalista y que usara una paleta de colores lo más pequeña posible, pues así no resultaría recargada a la vista, facilitaría al usuario su uso y sería mucho más accesible y visual. El color destacado de la aplicación es un azul grisáceo con la nomenclatura hexadecimal #607D8B y todas sus tonalidades. Decidimos este color porque creíamos que hacía un buen contraste con los colores de los mapas en general. Como color de fondo se usó el blanco.

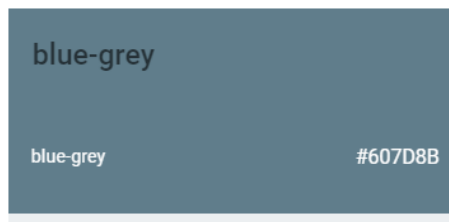


Figura 3.3 color destacado de la aplicación.

En el apartado siguiente se mostrarán capturas de pantalla de la aplicación donde se puede observar el diseño de la interfaz.

3.1.3 Implementación

3.1.3.1 Herramientas y tecnología utilizadas

En mi parte del trabajo que es la del Frontend, se utilizaron las herramientas mostradas a continuación, además de tener que usar alguna vez alguna función del Backend para realizar pruebas con el programa.

- **Visual Code Studio:** Usado como entorno de programación. Es un editor de código desarrollado por Microsoft. Este incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis. Es de código abierto totalmente gratuito. [15]
- **Vue:** un framework progresivo de código abierto para construir interfaces el cual tiene una gran cantidad de módulos, en nuestro caso se utilizaron los siguientes: [16]
 - **Vue-Layers**, que utiliza OpenLayers para visualizar los mapas.
 - **Vuex**, para almacenar datos en Frontend y permitir una conexión directa entre componentes. [17]
 - **Vue-Router** que permite hacer conexiones entre páginas por medio de enrutamientos o vistas anidadas. [18]

Todo ello posibilita el crear aplicaciones web de una manera sencilla, intuitiva y con un diseño actual. Vue es muy fácil de instalar por consola y lleva menos de 10 minutos crear un proyecto.

- **Vuetify**: un framework de Vue que permite acelerar el desarrollo de aplicaciones web complejas, incorporando una gran cantidad de componentes "listos para usar" utilizando la estética de Material Design. [19]

Esta herramienta fue usada para la parte del Frontend y en especial para la realización de la interfaz gráfica. En mi caso he elegido estas herramientas por el hecho de que ya he trabajado con ellas y me han facilitado mucho la creación de aplicaciones web, además de que sus componentes ofrecen un diseño minimalista y cuidado para la interfaz que se valora bastante.

- Para mostrar la aplicación en ejecución a tiempo real se utilizó **electron.js**, otro framework para aplicaciones web muy sencillo de usar y que se puede incorporar a Vue. Permite ejecutar y construir la aplicación en un ejecutable .exe en vez de acceder mediante localhost:8080. [20]
- **GitHub** usado para el control de versiones (ver *ANEXO A: Control de Versiones*)

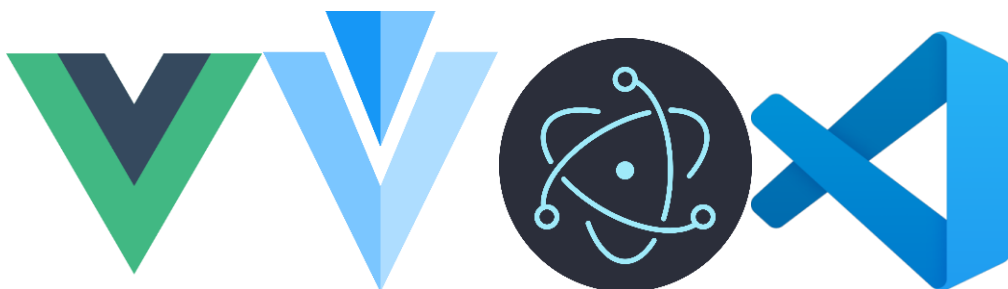


Figura 3.4 - Logos de (Vue, Vuetify, Electron y Visual Code Studio)

3.1.3.2 Organización del proyecto

Los proyectos de Vue suelen tener una estructura fijada. Es muy fácil crear proyectos usando Vue CLI 3, el cual genera el propio proyecto automáticamente, creando los ficheros y carpetas pertinentes. A continuación, se explicarán las partes principales que tiene nuestra implementación, centrándome en la estructura del Frontend:

- **Node_modules** contiene todos los módulos o librerías necesarios para la implementación del proyecto.
- Dentro de **src** se encuentra todo lo relacionado con los componentes de la implementación. **Assets** contiene imágenes estáticas, en **components** se almacenan los componentes personalizados, **plugins** contiene los plugins usados, en este caso, el de Vuetify.
- En la carpeta de **views** se almacenan los diferentes ficheros **.vue** que componen las vistas o páginas de la aplicación y que serán explicados cada uno más detenidamente.
- **App.vue** es el componente raíz que contiene la vista principal de la aplicación.
- Además de todos los ficheros anteriores, las configuraciones del proyecto se realizan en otros archivos como **router.js**, que es el que se encarga de enrutar las diferentes vistas o componentes de la aplicación, **main.js**, donde se encuentran las librerías usadas y se declaran los componentes o **store.js**, que utiliza Vuex para almacenar datos como las sesiones en el Frontend.

El proyecto de Vue está montado sobre Electron-builder, una librería usada para construir, empaquetar y desarrollar aplicaciones web. Esta aplicación nos ha permitido visualizar la interfaz de la aplicación mientras que la programábamos, pues además proporciona una actualización continua que muestra al momento los cambios realizados en el programa.

3.1.3.3 Breve descripción del despliegue de la aplicación

Dado que la aplicación está montada en una base de datos en la empresa, es necesario acceder a ella usando ssh, por lo que se ha requerido usar una herramienta que nos permita realizar esa conexión a la base de datos. En nuestro caso hemos usado Putty. Para ello fue necesario introducir el HostName y el puerto del servidor, así como otros ajustes que aparecen en la figura mostrada. Una vez que está configurado se debe introducir un login y contraseña para conectarse.

The image shows the PuTTY configuration window. At the top, it says 'Specify the destination you want to connect to'. Below this, there are two input fields: 'Host Name (or IP address)' with the value 'practicass.tecnosylva.es' and 'Port' with the value '22'. Under 'Connection type:', there are five radio buttons: 'Raw', 'Telnet', 'Rlogin', 'SSH' (which is selected), and 'Serial'. Below this is a section titled 'Options controlling SSH port forwarding'. Inside, there's a 'Port forwarding' section with two checkboxes: 'Local ports accept connections from other hosts' and 'Remote ports do the same (SSH-2 only)', both of which are unchecked. Below these is a 'Forwarded ports:' section with a text box containing '4L3305 localhost:3306' and a 'Remove' button. At the bottom, there's an 'Add new forwarded port' section with 'Source port' and 'Destination' input fields, and a set of radio buttons for 'Local', 'Remote', 'Dynamic', 'Auto', 'IPv4' (selected), and 'IPv6'. There is also an 'Add' button next to the 'Source port' field.

Figura 3.5 Configuración de la conexión ssh en Putty.

A continuación, el despliegue de la aplicación se realizaría como un proyecto común de Vue. En un terminal situados en la ruta del Backend se introduciría el siguiente comando de Node:

```
node app.js
```

Este comando desplegaría el Backend. Por último, se lanzaría el Frontend usando electron-builder:

```
npm run electron:serve
```

Para empaquetar el proyecto se usaría el siguiente comando:

```
npm run electron:build
```

Si no se dispusiera de electron-builder, esta se podría desplegar en el navegador usando el siguiente comando y entrando en <http://localhost:8080/>:

```
npm run serve
```

3.2 EL PRODUCTO DEL DESARROLLO

En este apartado se mostrará detalladamente el funcionamiento de la aplicación, mostrando el flujo y la forma de ejecución de esta junto con capturas de pantalla.

3.2.1 Flujo de la aplicación

En este capítulo me centraré en explicar con profundidad la interfaz de la aplicación, así como sus diferentes pantallas y funcionalidades. En primer lugar, en la siguiente figura se puede ver un esquema del flujo de la aplicación con las diferentes pantallas y accesos que existen de manera simplificada y de los cuales haré hincapié más adelante.

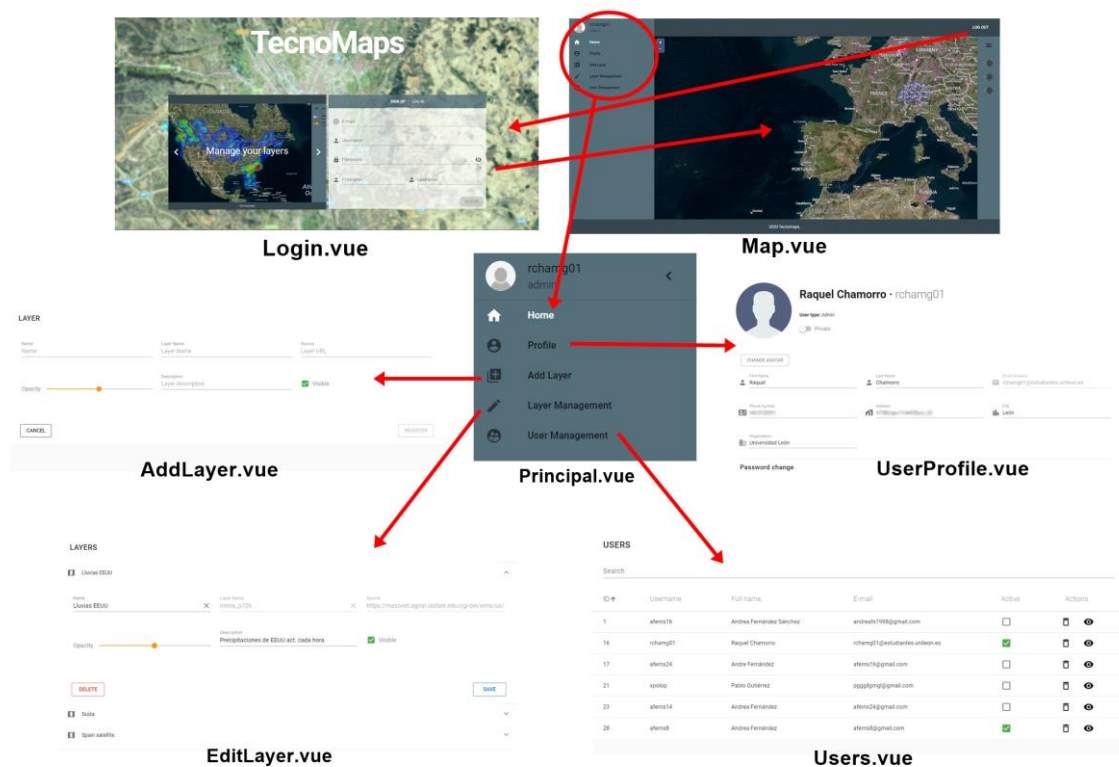


Figura 3.6 Flujo de la aplicación TecnoMaps

Como se puede apreciar en el la Figura 3.6. se podría decir que la arquitectura de la aplicación es de la de Single Page Application ya que no requiere recargar la página para que se actualicen los datos y se puede acceder a cualquier página de la aplicación desde cualquier punto gracias al menú de la barra.

3.2.2 Análisis de cada componente

La aplicación está compuesta por siete componentes diferentes:

- **Login.vue**: la primera pantalla donde se registran los usuarios.
- **Principal.vue**: vista permanente con los menús de la aplicación por los que se accede al resto de componentes.
- **Map.vue**: el componente del mapa sobre el que se muestran las capas
- **AddLayer.vue**: pantalla en la el usuario crea y añade las capas.
- **EditLayer.vue**: gestión de capas del usuario donde estas se pueden editar, eliminar...
- **Users.vue**: lugar de gestión de usuarios donde los administradores pueden visualizar los usuarios y eliminarlos.
- **UserProfile**: pantalla donde el propio usuario puede modificar sus datos de perfil, así como cambiar la contraseña o elimina su cuenta de usuario.

A continuación, explicaré con todo detalle cada componente de la aplicación, insistiendo en su funcionalidad y en la interfaz.

3.2.2.1 Login

El Login, donde se registran e identifican los usuarios, se compone de dos partes fundamentales: el “carrusel” de imágenes, que son las diapositivas sobre las funcionalidades de la aplicación que se muestran a la izquierda de la pantalla y, por otra parte, a la derecha está el apartado de la identificación de usuarios en sí.

En esta parte existen dos tipos de formulario: Sign up, para que los usuarios que no tengan cuenta se creen una y puedan acceder a la aplicación y el Log in que permite identificar al usuario que ya tiene cuenta en la aplicación. Las funciones del Login en las que se profundizará algo más serán el Sign up y el log in:

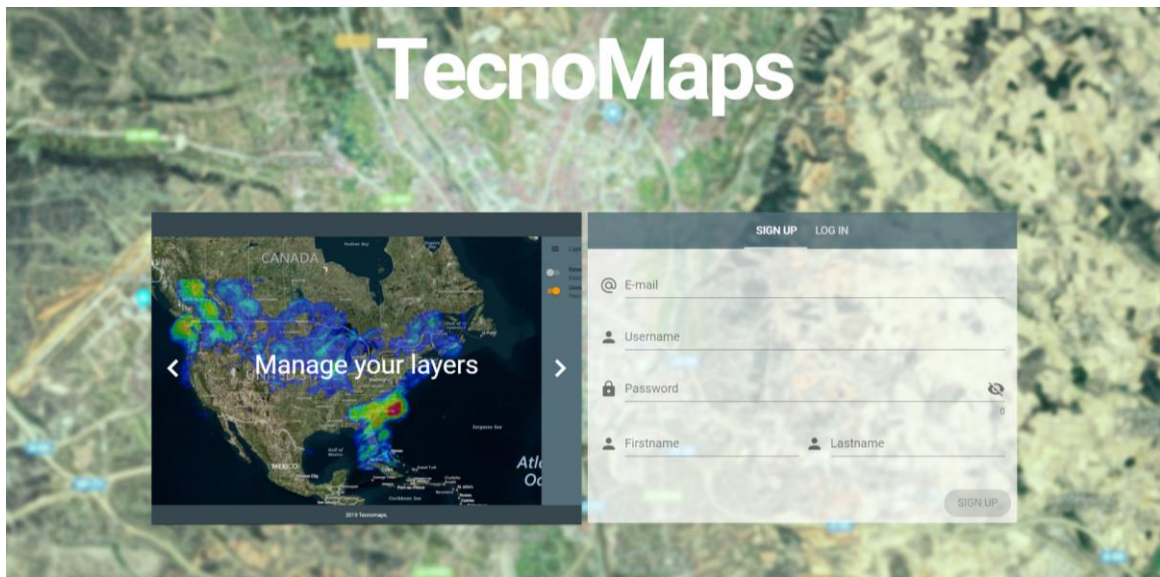


Figura 3.7 Vista del login

- Sign in: en este paso el usuario crea la cuenta para acceder a las funcionalidades de la aplicación. Esta especie de formulario donde se introducen los datos cuenta con tres campos: **E-mail**, **Username**, **Password**, **First name** y **Last name**.
- Log in: aquí el usuario ya debería estar registrado previamente en la base de datos de la aplicación. En este caso, solo se requieren de dos campos: **Username** y **Password**.

Tanto los campos de Sign in como los de Login in tienen las mismas características en cuanto a formato e introducción de caracteres.

- **E-mail**: campo obligatorio. El correo electrónico es meramente informativo puesto que no se va a requerir más durante el proceso de identificación del usuario, pero que si podría servir en futuras ampliaciones de la aplicación.
- **Username**: campo obligatorio. El nombre de usuario permite la identificación de cada usuario una vez que se registra. Este campo controla que un usuario no se pueda registrar con un nombre de usuario que ya esté en la base de datos de la aplicación. Este campo se puede cambiar una vez iniciada la sección en la vista de *Perfil de usuario*.
- **Password**: campo obligatorio. Existe una restricción de caracteres, pues la contraseña tiene que tener un mínimo de 8 caracteres. Este campo también cuenta con un sistema que oculta la contraseña, puesto que es un

dato privado. El usuario también puede hacerla visible para su comprobación haciendo click en el icono del ojo (ver Figura 3.8). Una vez iniciada la sesión, el usuario puede cambiar la contraseña si lo desea.

- **First name:** campo obligatorio. El nombre real del usuario. Es meramente informativo. Una vez iniciada la sesión, el usuario puede cambiar el nombre si lo desea.
- **Last name:** campo obligatorio. El/los apellido/s del usuario. Es meramente informativo. Una vez iniciada la sesión, el usuario puede cambiar el apellido si lo desea.

Figura 3.8 Detalle del formulario de Sign Up y Login In.

Una vez que se hace click en el botón derecho inferior (del Sign in y del Log in), la base de datos hace sus comprobaciones asegurándose de que estos campos son correctos, no estén duplicados o no cumplan las características correspondientes. En este caso aparecerá una pequeña notificación avisando del error o de si falta algún campo por rellenar, en las que aparecen los campos en rojo y con un mensaje avisando de que son obligatorios y no se han rellenado.

Todo lo mencionado anteriormente relacionado con el control y autenticación de usuario y su notificación al usuario se realiza parcialmente en funciones dentro del Frontend ya que según sean correctos los datos o no, la interfaz mostrará notificaciones de diferentes tipos, así como ciertos controles de elementos de esta. Un ejemplo de esa función es la que controla que los campos introducidos en el Log in se encuentren ya dentro de la base de datos.

```

285   register: function() {
286     if (
287       this.signName == null ||
288       this.signName == "" ||
289       this.firstname == null ||
290       this.firstname == "" ||
291       this.lastname == null ||
292       this.lastname == "" ||
293       this.signPass == null ||
294       this.signPass == "" ||
295       this.email == null ||
296       this.email == ""
297     ) {
298       this.alertContent = "Rellene todos los campos";
299       this.alert = true;
300     } else {
301       let data = {
302         username: this.signName,
303         email: this.email,
304         password: this.signPass,
305         firstname: this.firstname,
306         lastname: this.lastname
307       };
308       this.$store
309         .dispatch("register", data)
310         .then(() => {
311           this.$router.push({ name: "Map" });
312         })
313         .catch(err => {
314           if (err.message.includes("404")) {
315             this.alertContent = "Ese nombre de usuario ya está registrado";
316           } else if (err.message.includes("Network")) {
317             this.alertContent = "Hubo un problema en el envío";
318           }
319           this.alert = true;
320         });
321     }
322   }

```

Figura 3.9 Función de control de registro de usuarios.

3.2.2.2 Principal

Esta vista contiene las dos barras de opciones existentes en la aplicación.

Por un lado, la barra del menú principal (Figura 3.10), que es desplegable y permanece siempre visible en el lado izquierdo en toda la aplicación. Esta muestra todas las demás vistas a las que se puede acceder haciendo click sobre el nombre, además de mostrar información sobre el usuario (su Username y el tipo de usuario). Esto se consigue gracias a Vue-Router, que crea una ruta para cada uno de los elementos del menú (Figura 3.12).

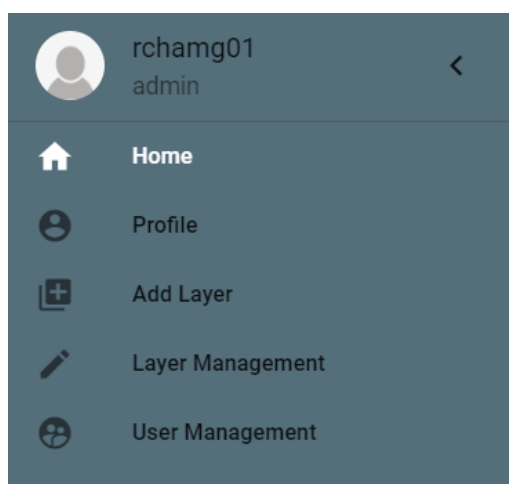


Figura 3.10 Detalle del menú principal.

La barra de menú de capas desplegable, que se encuentra en el lado derecho de la aplicación (Figura 3.13), nos muestra las capas existentes que tiene almacenadas el usuario en la base de datos. Cada usuario tiene las suyas propias. Esta barra solo se muestra en la vista del mapa, ya que permite controlar a tiempo real la visibilidad de dichas capas en el mapa. Cuando el control está de color naranja quiere decir que son visibles. Si estuviera gris, no se mostraría en el mapa.

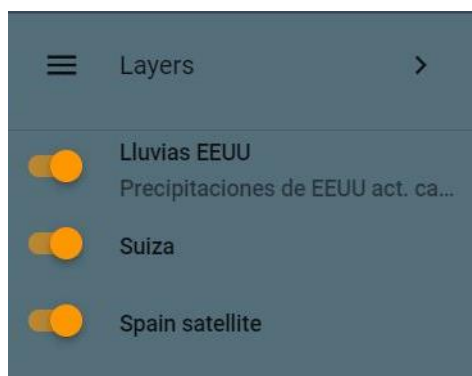


Figura 3.11 Detalle del menú de capas.

Además de estos elementos de menú, la aplicación cuenta también con una barra horizontal permanente en la parte superior con la opción Log out, que permite al usuario cerrar sesión y un “footer” en la parte inferior con el nombre del proyecto.

```

15 routes: [
16   {
17     path: "/",
18     name: "Login",
19     component: Login
20   },
21   {
22     path: "/principal",
23     name: "Principal",
24     component: Principal,
25     children: [
26       {
27         path: "/",
28         name: "Map",
29         component: Map
30       },
31       {
32         path: "/addlayer",
33         name: "AddLayer",
34         component: AddLayer
35       },
36       {
37         path: "/editlayer",
38         name: "EditLayer",
39         component: EditLayer
40       },
41       {
42         path: "/users",
43         name: "Users",
44         component: Users
45       },
46       {
47         path: "/userprofile",
48         name: "UserProfile",
49         component: UserProfile
50       }
51     ]
52   }
53 ]

```

Figura 3.12 Array con los enlaces del Router a cada componente.

3.2.2.3 Mapa

Una vez que el usuario se registra correctamente en el Login se pasa a la siguiente pantalla que es la principal en la aplicación y contiene el mapa base.

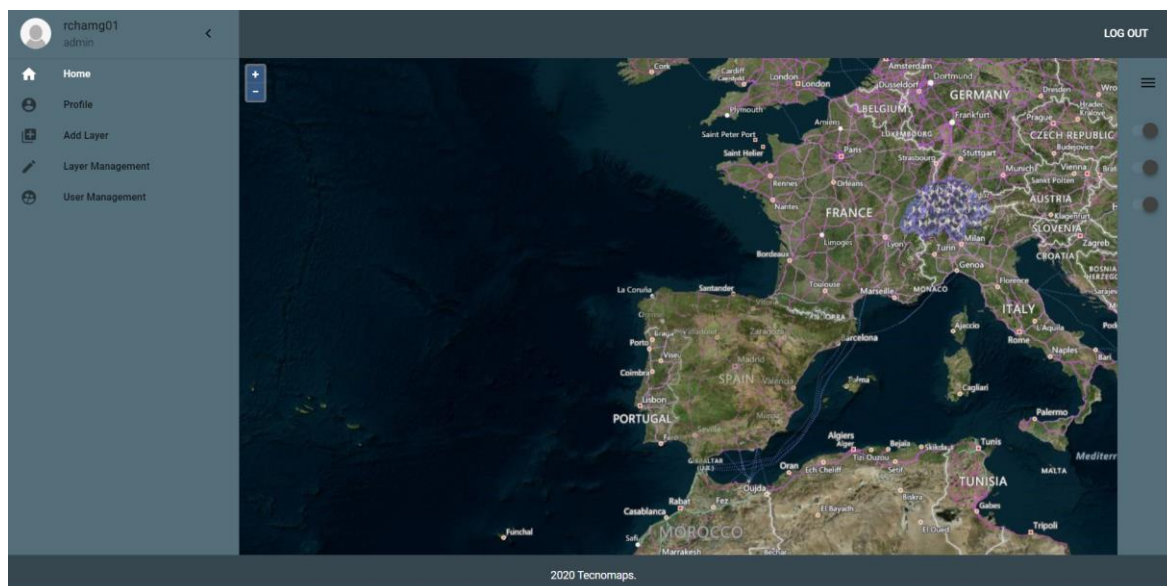


Figura 3.13 Vista del mapa.

La visualización del mapa en la aplicación se realiza mediante Vue-Layers. En este caso hemos utilizado como modelo de mapa base un mapa satélite obtenido de Bing listo para mostrar sobre él las capas que el usuario cree.

En el siguiente fragmento de código mostrado en la figura 3.14, se puede apreciar cómo se realiza el añadido de capas al mapa base junto con los parámetros recibidos al crear la capa en la vista de *AddLayer*.

```

<vl-layer-tile>
  <vl-source-bingmaps :api-key="apiKey" :imagery-set="imagerySet"></vl-source-bingmaps>
</vl-layer-tile>
<vl-view
  :zoom.sync="zoom"
  :center.sync="center"
  :rotation.sync="rotation"
  :max-zoom="maxZoom"
></vl-view>

<vl-layer-tile
  v-for="layer in layers"
  v-bind:key="layer.id"
  :opacity="layer.opacity/100"
  :visible="visible"
>
  <vl-source-wms
    :url="layer.url"
    :params="layer.params"
    :layers="layer.layerName"
    v-if="layer.visible"
  ></vl-source-wms>
</vl-layer-tile>

```

Figura 3.14 Código del mapa


3.2.2.4 AddLayer

A esta pantalla se accede desde el menú de vistas que se encuentra a la izquierda de la pantalla. Aquí es donde el usuario puede crear las capas que después podrá usar en el mapa. Para ello, se necesita completar los pasos que se ven en la Figura 3.15:

- **Name:** nombre de la capa. Es un campo obligatorio y puede ser cualquier nombre, preferiblemente corto.
- **Layer Name:** es el nombre de la capa “oficial”. Es obligatorio y tiene que ser el nombre de la capa especificado en la fuente de donde se ha obtenido dicha capa. Si es diferente, la capa no se podría visualizar en el mapa.
- **Source:** la URL de la capa. Campo obligatorio y también tiene que ser específico de esa capa ya que si es diferente no se podrá visualizar la capa.
- **Opacity:** la opacidad que tendrá la capa en el mapa. Varía de 0 a 100 y tiene un valor por defecto de 50.
- **Description:** una breve descripción de la capa. No es obligatorio.

- **Visible:** checkbox para indicar si esa capa está visible en el mapa. Este valor se puede cambiar también en la barra de menú de la Figura 5.6. Está por defecto marcado como visible.

LAYER

Name Name	Layer Name Layer Name	Source Layer URL
Opacity 	Description Layer description	<input checked="" type="checkbox"/> Visible

CANCEL
REGISTER

Figura 3.15 Vista de añadir capa.

Todos estos campos del formulario poseen el mismo tipo de presentación que los del Login, ya explicados, con sus respectivas notificaciones si estos campos no están completados (el campo se cambia a color rojo y aparece un aviso).

```
methods: {
  resetForm() {
    this.form = Object.assign({}, this.defaultForm);
    this.$refs.form.reset();
  },
  submit() {
    this.resetForm();
  },
  register() {
    if (this.form.opacity == null) this.form.opacity = 50;
    var data = {
      name: this.form.name,
      layerName: this.form.layerName,
      url: this.form.layerSource,
      opacity: this.form.opacity,
      desc: this.form.desc,
      visible: this.form.visible,
      idUser: this.$store.getters.getUser.id
    };
    this.$store
      .dispatch("registerLayer", data)
      .then(() => (this.snackbarSuccess = true))
      .catch(err => {
        this.snackbarError = true;
      });
  }
}
```

Figura 3.16 Métodos para crear y guardar una capa.

Además de estos campos en la parte inferior del formulario aparecen dos botones: Cancel, que borra los campos y Register, que añade la capa. Este último solo se activará si todos los campos obligatorios están completados. Después de añadir la capa, esta aparecerá en el menú de capas mostrado anteriormente (Figura 3.11) y en la vista de gestión de capas. En la Figura 3.16 se muestran los diferentes métodos necesarios para enviar el formulario (los valores introducidos por el usuario en los campos) donde se pueden ver las variables utilizadas para almacenar dichos campos del formulario.

3.2.2.5 Gestión de capas

En este apartado de la aplicación se puede visualizar un listado de todas las capas que tiene guardadas el usuario, además de modificarlas. La vista es similar a la de EditLayer, solo que tiene el añadido de la lista desplegable que muestra los detalles de la capa como se puede apreciar en la Figura 3.17. La principal diferencia en la edición de los datos de la capa es que los dos campos “Layer name” y “Source” no se pueden editar, ya que si se modifican la capa dejaría de ser visible por error de parámetros.

The screenshot displays the 'LAYERS' management interface. At the top, there's a header 'LAYERS'. Below it, a list of layers is shown. The first layer is 'Lluvias EEUU', which is expanded to show its details. The details include:

- Name:** Lluvias EEUU (with a close icon)
- Layer Name:** mrms_p72h (with a close icon)
- Source:** https://mesonet.agron.iastate.edu/cgi-bin/wms/us/ (with a close icon)
- Opacity:** A slider control.
- Description:** Precipitaciones de EEUU act. cada hora
- Layer visibility minimum scale:** 1: (with a search icon)
- Visible:** A checkbox that is checked.
- Buttons:** A red 'DELETE' button and a blue 'SAVE' button.

Below the expanded layer, there are two more layers listed: 'Suiza' and 'Spain satellite', each with a dropdown arrow.

Figura 3.17 Vista de gestión de capas.

Además, también aparece una nueva opción que es la de “Delete”, para eliminar la capa. Cuando se hace click en este botón aparece un cuadro de dialogo comprobando la eliminación voluntaria de la capa (Figura 3.18).

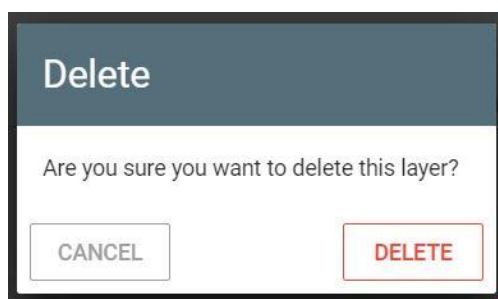


Figura 3.18 Cuadro de diálogo del borrado de capa.

Los cambios que se realizan en la capa se aplican al momento y se muestra una notificación del estado de guardado de la capa (si el guardado ha sido correcto o erróneo) como se muestra en la Figura 3.19.



Figura 3.19 Notificaciones del estado de guardado de capas.

3.2.2.6 Gestión de usuarios

La vista de gestión de usuario contiene un listado a modo de tabla en la que se muestran todos los usuarios registrados en la aplicación con posibilidad de edición. Esta página solo es visible para los usuarios de tipo administrador.

USERS

Search













ID ↑	Username	Full name	E-mail	Active	Actions
1	aferns16	Andrea Fernández Sánchez	andrea1998@gmail.com	<input type="checkbox"/>	 
16	rhamg01	Raquel Chamorro	rhamg01@estudiantes.unileon.es	<input checked="" type="checkbox"/>	 
17	aferns24	Andre Fernández	aferns16@gmail.com	<input type="checkbox"/>	 
21	xpolop	Pablo Gutiérrez	pgggllgmgl@gmail.com	<input type="checkbox"/>	 
23	aferns14	Andrea Fernández	aferns24@gmail.com	<input type="checkbox"/>	 
28	aferns8	Andrea Fernández	aferns8@gmail.com	<input checked="" type="checkbox"/>	 

Figura 3.20 Vista de Gestión de usuarios.

La tabla cuenta con 5 campos de información sobre el usuario:

- ID: el número de identificación de usuario. El número es asignado según el orden de creación de dicho usuario siendo el 1 el primer usuario registrado en la aplicación.
- Username: el nombre de usuario con el que el usuario se registra.
- Contraseña: contraseña del usuario.
- Active: campo que indica si el usuario está usando la aplicación en ese mismo momento

El administrador no puede editar ninguno de estos campos, pero en un posible futuro de la aplicación sí que se podría implementar esta función. Además del listado, el administrador también cuenta con una barra de búsqueda donde se pueden buscar por cualquiera de estos campos el usuario. Función útil cuando hay una gran cantidad de usuarios en la base de datos.

USERS

Search
rhamg01



ID ↑	Username	Full name	E-mail	Active	Actions
16	rhamg01	Raquel Chamorro	rhamg01@estudiantes.unileon.es	<input checked="" type="checkbox"/>	 

Figura 3.21 Ejemplo de búsqueda de usuario.

Volviendo a la tabla de información, en ella se encuentra también el icono de la papelera, que es el botón de borrado de usuario. Los usuarios con el rol de administrador son los únicos que pueden acceder a este botón y que pueden eliminar a otros usuarios de la aplicación. Además, este botón cuenta con confirmación de borrado, como en la gestión de capas. Figura 3.22.

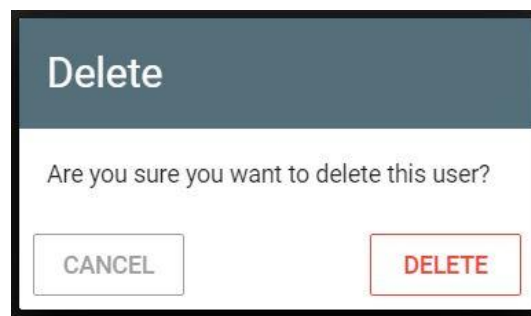


Figura 3.22 Cuadro de diálogo de confirmación de borrado de usuario.

El fragmento de código que se muestra a continuación permite al administrador la eliminación de usuarios. Como se puede apreciar, deleteUser envía los datos correspondientes al Backend para que se elimine el usuario en la base de datos. En este caso los usuarios no se eliminaban como tal si no que se dejaban de mostrar, tal como nos ordenó el tutor de la empresa.

```
deleteUser() {
  var data = {
    id: this.dialogId
  };
  this.$store
    .dispatch("deleteUser", data)
    .then(() => {
      this.snackbarSuccessDeleted = true;
      this.requestUsers();
    })
    .catch(err => {
      this.snackbarError = true;
    });
}
```

Figura 3.23 Función para la eliminación de usuarios.

El icono de un ojo donde pulsando aparece una ventana emergente con información sobre el perfil del usuario seleccionado. En este pop up se permite cambiar el rol del usuario siempre y cuando se tenga el rol de Administrador. Dentro de los usuarios existen tres tipos de roles:

- Administradores: pueden acceder a todas las vistas de la aplicación y realizar acciones restringidas para otros roles como visualizar información de otros usuarios, acceder a la vista de gestión de usuarios, eliminar a otros usuarios, cambiar el rol de los usuarios...
- Moderadores: pueden acceder a todas las vistas de la aplicación, incluida la de gestión de usuarios, pero solo podrán visualizar la información de los usuarios que tengan activada la opción de perfil público. Además, no pueden realizar acciones como eliminar usuarios ni cambiar el rol de estos.
- Estándar: el rol que menos permisos tiene, no puede acceder a la vista de gestión de usuarios, ni visualizar perfiles de otros usuarios ni cambiar el rol ni eliminar a otros usuarios.

rchamg01

Raquel Chamorro · rchamg01

User type:

- admin
- standard
- moderator

First name: Raquel

Email: rchamg01@estudiantes.unileon.es

Phone Number: [Redacted]

Address: [Redacted]

City: León

Organization: Universidad León

CANCEL SAVE

Figura 3.24 Ejemplo de ventana emergente con la información del perfil de usuario.

3.2.2.7 Perfil de usuario

En esta vista, el usuario que ha iniciado sesión en la aplicación podrá editar su información de usuario, así como añadir más campos de información.

En la última versión de la aplicación se ha añadido la opción de cambiar foto de perfil. Al pulsar el botón de *Change Avatar*, se abriría una ventana emergente para la selección de archivos en el dispositivo, que permitiría elegir una foto de formato .png o .jpg. Actualmente, esta opción no es funcional como se comenta en el apartado de “Problemas Encontrados” pero se ha dejado el icono para una futura implementación de este.

Raquel Chamorro · rchamg01

User type: Admin

☐ Private

CHANGE AVATAR

First Name
Raquel

Last Name
Chamorro

Email Address
rchamg01@estudiantes.unileon.es

Phone number
[Redacted]

Address
[Redacted]

City
León

Organization
Universidad León

Password change

Figura 3.25 Vista del perfil de usuario.

En el apartado superior aparece, además de la foto de perfil, Información como el nombre completo y el nickname del usuario. También aparece el tipo de usuario (rol) y la opción de cambiar el perfil a público o privado con un “switch”:

- **Privado:** solo el propio usuario y los administradores pueden ver la información de perfil de este usuario.
- **Público:** el propio usuario, los administradores y los moderadores podrán ver la información de perfil del usuario.

A continuación, está la sección de la información, donde se pueden editar los siguientes campos:

- **First Name y Last Name:** al igual que en el login, ambos son obligatorios, pero se pueden editar.
- **E-mail:** campo obligatorio, pero no se puede eliminar, puesto que es el identificador del usuario. Si se quisiera cambiar este campo se tendría que eliminar la cuenta y crear otra nueva.
- **Phone number:** número de teléfono. Campo optativo
- **Address:** campo optativo. Dirección del domicilio, empresa u organización.
- **City:** campo optativo. Localidad del domicilio, empresa u organización.
- **Organization:** campo optativo. Nombre del domicilio, empresa u organización.

Debajo de estos campos recién mencionados, se encuentra la sección de cambio de contraseña. Esta cuenta con dos campos similares a los campos de contraseñas que se han mostrado en vistas anteriores:

- **Old Password:** Campo obligatorio. Es la contraseña a cambiar. Si se introduce una contraseña errónea o no se introduce ningún valor, esta no se podrá cambiar.
- **New Password:** Campo obligatorio. Es la contraseña nueva.

Las contraseñas no serán visibles a no ser que se pulse en el icono del ojo y deben tener al menos 8 caracteres.

Password change







	Old password			New password	
					

Figura 3.26 Detalle de la sección de cambio de contraseña.

Debajo de la sección del cambio de contraseña se encuentran los botones “Save” y “Delete”.

Una vez que se han cambiado los campos de información oportunos en el perfil, se debe pulsar el botón de “Save” para guardar los cambios, de lo contrario, si no se guardan o se sale de la página, estos cambios no se harán efectivos.

El botón “Delete” sirve para eliminar la cuenta de usuario. Una vez pulsado, aparecerá una ventana de confirmación para asegurarse de que el usuario quiere borrar la cuenta. Si esta se borra, se cerrará la sesión al usuario automáticamente y aparecerá en la pantalla inicial del registro de usuario.

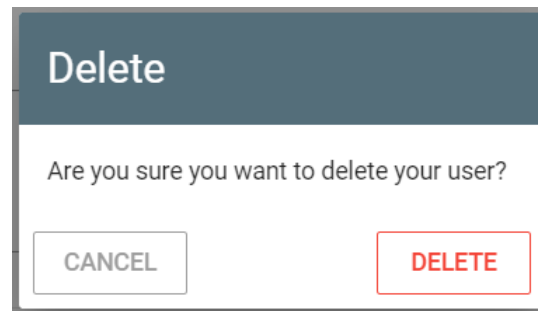


Figura 3.27 Ventana de confirmación de eliminación del propio usuario.

En el código que se muestra a continuación aparece la función de guardar los datos del formulario que incluyen toda la información del perfil y donde se pueden observar las condiciones para el cambio de contraseña.

```

252 save() {
253   if (
254     (this.oldPass != "" &&
255       this.oldPass == this.$store.getters.getUser.password) ||
256     this.oldPass == ""
257   ) {
258     var data = {
259       id: this.$store.getters.getUser.id,
260       pass:
261         this.newPass != ""
262         ? this.newPass
263         : this.$store.getters.getUser.password,
264       firstName: this.form.name,
265       lastName: this.form.lastName,
266       phone: this.form.phone,
267       address: this.form.address,
268       city: this.form.city,
269       organization: this.form.organization,
270       privacy: this.form.privacy == true ? "public" : "private"
271     };
272     this.$store
273       .dispatch("updateUser", data)
274       .then(() => {
275         this.snackbarSuccess = true;
276       })
277       .catch(() => {
278         this.snackbarError = true;
279       });
280   } else this.snackbarPass = true;
281 },

```

Figura 3.28 Extracto de código de Users.vue

4. Evaluación

En este apartado se tratará de mostrar cómo ha sido el proceso de evaluación del proyecto, qué tipo de pruebas se han realizado para evaluarlo y las conclusiones que se obtienen de los resultados de dicha evaluación.

4.1 PROCESO DE EVALUACIÓN

4.1.1 Forma de evaluación

La aplicación ha sido evaluada de dos maneras distintas: por un lado, para evaluar la interfaz, se ha elaborado un cuestionario en Google Forms donde se pide a personas de nuestro entorno que utilicen la aplicación siguiendo unos pasos y que contesten el cuestionario.

Las personas que han realizado el cuestionario han tenido que ser personas que convivan con nosotros o que hayan sido proporcionadas el software de la aplicación. Muchos de estos usuarios han tenido que realizar la prueba desde nuestro equipo puesto que no tienen el permiso o equipos necesarios para realizarlo. He hecho especial hincapié en que participen personas de diferentes edades, géneros y profesiones para tener una evaluación más diversa y desde más puntos de vista. Además, he podido contactar con personas con diferentes discapacidades para que participen en la encuesta y evalúen la accesibilidad de la aplicación y si está adaptada para su tipo de discapacidad.

El objetivo principal de este cuestionario es evaluar la accesibilidad, el diseño y la funcionalidad de la aplicación, pero centrándose en especial en la interfaz.

La otra forma de evaluar este proyecto ha sido por depuración. Una vez que teníamos completos unos objetivos o componentes, redactábamos sus correspondientes casos de prueba, que consisten en realizar unos supuestos de ejecución y documentar cuál es el estado en el que se encuentra el software, qué se debe introducir o qué se debe realizar, unas precondiciones y la salida que se debería obtener.

4.1.2 Casos de prueba

Casos de pruebas realizados:

Tabla 4.1 Caso de prueba 1 – Login.

Test Case 1: Login
Precondiciones:
La aplicación debe haber sido iniciada
Flujo:
<ul style="list-style-type: none"> - Introducir nombre de usuario - Introducir contraseña - Hacer clic en el botón de 'Login'
Resultado esperado:
La aplicación mostrará la pantalla principal si el usuario es correcto.
Resultado obtenido:
Se ha obtenido el resultado esperado.

Tabla 4.2 Caso de prueba 2 - Sign in.

Test Case 2: Sign in
Precondiciones:
La aplicación debe haber sido iniciada
Flujo:
<ul style="list-style-type: none"> - Introducir nombre de usuario - Introducir contraseña - Introducir nombre y apellido(s) - Hacer clic en el botón de 'Sign in'
Resultado esperado:
El usuario se creará y la aplicación mostrará la pantalla principal si los datos son correctos.
Resultado obtenido:
Se ha obtenido el resultado esperado.

Tabla 4.3 Caso de prueba 3 - Añadir capa.

Test Case 3: añadir capa	
Precondiciones:	
El usuario debe estar registrado y haber hecho login.	
Flujo:	
<ul style="list-style-type: none"> - Desplegar el menú y pulsar el apartado “Add Layer” - Completar los campos necesarios para crear una capa - Pulsar el botón “register” para crear la capa - Aparece notificación de que se ha creado correctamente 	
Resultado esperado:	
Se crea una capa con los datos introducidos si la información es correcta.	
Resultado obtenido:	
Se ha obtenido el resultado esperado.	

Tabla 4.4 Caso de prueba 4 - Editar capa.

Test Case 4: editar capa	
Precondiciones:	
El usuario debe haber hecho login y creado al menos una capa.	
Flujo:	
<ul style="list-style-type: none"> - Desplegar el menú y pulsar el apartado “Layer management” - Elegir y desplegar la capa a editar de la lista - Editar los campos que precisen de ello - Pulsar el botón “save” para guardar cambios - Aparece notificación de que se ha cambiado correctamente 	
Resultado esperado:	
Se cambian los datos especificados de una capa.	
Resultado obtenido:	
Se ha obtenido el resultado esperado.	

Tabla 4.5 Caso de prueba 5 - Eliminar capa.

Test Case 5: eliminar capa
Precondiciones:
El usuario debe haber hecho login y creado al menos una capa.
Flujo:
<ul style="list-style-type: none"> - Desplegar el menú y pulsar el apartado “Layer management” - Elegir y desplegar la capa a eliminar de la lista - Pulsar el botón “Delete” - En el pop up de confirmación pulsar “Delete” - Aparece notificación de que se ha eliminado correctamente
Resultado esperado:
Se elimina la capa especificada.
Resultado obtenido:
Se ha obtenido el resultado esperado.

Tabla 4.6 Caso de prueba 6 - Editar perfil del propio usuario.

Test Case 6: editar perfil del propio usuario
Precondiciones:
El usuario debe haber hecho login.
Flujo:
<ul style="list-style-type: none"> - Desplegar el menú y pulsar el apartado “Profile” - Cambiar la información que precise - Pulsar el botón “Save” - Aparece notificación de que se han guardado los cambios
Resultado esperado:
Se cambian los datos especificados del perfil.
Resultado obtenido:
Se ha obtenido el resultado esperado.

Tabla 4.7 Caso de prueba 7 - Eliminar perfil del propio usuario.

Test Case 7: eliminar perfil del propio usuario	
Precondiciones:	
	El usuario debe haber hecho login.
Flujo:	
	<ul style="list-style-type: none"> - Desplegar el menú y pulsar el apartado "Profile" - Una vez en la pantalla pulsar el botón "Delete" - En el pop up de confirmación pulsar "Delete" - Se cierra la sesión y aparece la pantalla del login
Resultado esperado:	
	Se elimina el perfil del usuario.
Resultado obtenido:	
	Se ha obtenido el resultado esperado.

Tabla 4.8 Caso de prueba 8 - Visualizar información de otros usuarios.

Test Case 8: Visualizar información de otros usuarios	
Precondiciones:	
	El usuario debe haber hecho login y ser administrador.
Flujo:	
	<ul style="list-style-type: none"> - Desplegar el menú y pulsar el apartado "User Management" - Aparece la tabla con los usuarios e información - Pulsar en el icono de ojo del usuario - Aparece un pop up con la información del usuario
Resultado esperado:	
	Se visualiza la información de cada usuario registrado
Resultado obtenido:	
	Se ha obtenido el resultado esperado.

Tabla 4.9 Caso de prueba 9 - Eliminar otros usuarios.

Test Case 9: Eliminar otros usuarios	
Precondiciones:	
	El usuario debe haber hecho login y ser administrador.
Flujo:	
	<ul style="list-style-type: none"> - Desplegar el menú y pulsar el apartado "User Management" - Aparece la tabla con los usuarios e información - Pulsar el botón con icono de cubo de basura del usuario a eliminar - En el pop up de confirmación pulsar "Delete" - Aparece una confirmación de que se ha eliminado correctamente
Resultado esperado:	
	Se elimina el perfil del usuario deseado.
Resultado obtenido:	
	Se ha obtenido el resultado esperado.

Tabla 4.10 Caso de prueba 10 - Cambiar de rol a otros usuarios.

Test Case 10: Cambiar de rol a otros usuarios	
Precondiciones:	
	El usuario debe haber hecho login y ser administrador.
Flujo:	
	<ul style="list-style-type: none"> - Desplegar el menú y pulsar el apartado "User Management" - Aparece la tabla con los usuarios e información - Pulsar en el icono de ojo del usuario - Elegir el nuevo rol del usuario en el pop up - Pulsar el botón "Save" - Aparece una notificación de que se han guardado los cambios
Resultado esperado:	
	Se cambia el rol del usuario deseado.
Resultado obtenido:	
	Se ha obtenido el resultado esperado.

Tabla 4.11 Caso de prueba 11 - Visualizar las capas en un mapa.

Test Case 11: Visualizar las capas en un mapa
Precondiciones:
El usuario debe haber hecho login y haber creado al menos una capa.
Flujo:
<ul style="list-style-type: none"> - Desplegar el menú a la derecha en la pantalla de bienvenida o “Home” - Pulsar en la capa que se desea visualizar en el mapa - Pulsar otra vez para hacer invisible la capa
Resultado esperado:
Se visualiza la capa en el mapa.
Resultado obtenido:
Se ha obtenido el resultado esperado.

Tabla 4.12 Caso de prueba 12 - Log out.

Test Case 12: Log out
Precondiciones:
El usuario debe haber hecho login
Flujo:
<ul style="list-style-type: none"> - Pulsar desde cualquier pantalla el botón de “Log Out” que aparece en la esquina superior derecha
Resultado esperado:
Se cierra la sesión y se muestra la pantalla de Login
Resultado obtenido:
Se ha obtenido el resultado esperado.

Tabla 4.13 Caso de prueba 13 - Editar privacidad del usuario.

Test Case: editar privacidad del usuario
Precondiciones:
El usuario debe haber hecho login.
Flujo:
<ul style="list-style-type: none"> - Desplegar el menú y pulsar el apartado "Profile" - Cambiar en el switch la privacidad ya sea a "public" o "private" - Pulsar el botón "Save" - Aparece notificación avisando de que se han guardado los cambios
Resultado esperado:
Se cambia la privacidad del perfil.
Resultado obtenido:
Se ha obtenido el resultado esperado.

Tabla 4.14 Caso de prueba 14 - Añadir foto de perfil.

Test Case: añadir foto de perfil
Precondiciones:
El usuario debe haber hecho login.
Flujo:
<ul style="list-style-type: none"> - Desplegar el menú y pulsar el apartado "Profile" - Pulsar en el botón "Change Avatar" - Aparece un pop up donde poner la URL de la imagen que se desee - Pulsar "Save avatar" - Aparece notificación de que se han guardado los cambios
Resultado esperado:
Se cambia la foto del perfil.
Resultado obtenido:
No se cambia la foto de perfil.

Tabla 4.15 Caso de prueba 15 - Cambiar contraseña.

Test Case: cambiar contraseña	
Precondiciones:	
El usuario debe haber hecho login.	
Flujo:	
<ul style="list-style-type: none"> - Desplegar el menú y pulsar el apartado "Profile" - Introducir la contraseña actual - Introducir la nueva contraseña - Pulsar "Save" - Aparece notificación de que se han guardado los cambios 	
Resultado esperado:	
Se cambia la contraseña del perfil.	
Resultado obtenido:	
Se ha obtenido el resultado esperado.	

4.1.3 Cuestionarios de evaluación

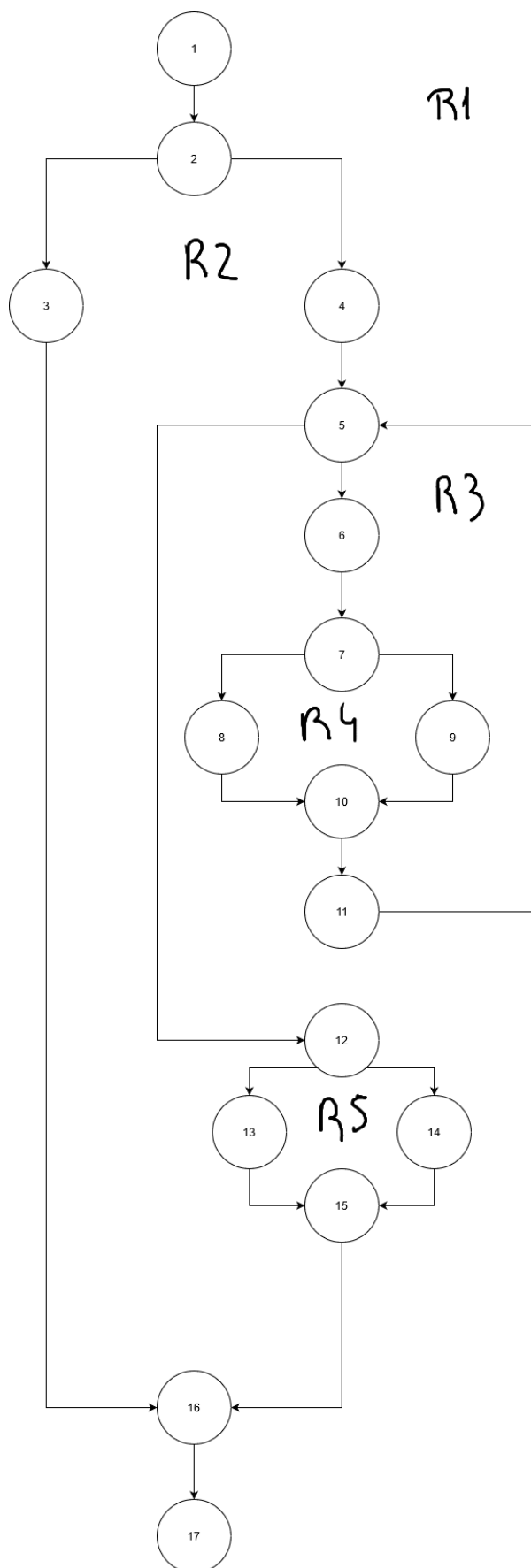
Ver ANEXO C: Cuestionarios de evaluación.

4.1.4 Test de caja blanca

El siguiente código es un extracto de la vista inicial del login que no aparece directamente en la aplicación, sino que ha sido modificado para tener más sentencias if y while.

```
procedimiento login()
inicio
1 datos = recoger_datos()
2 si datos == null
3 entonces
    alert = true
4 sino
    found = null
    i = 0
5 mientras (!found)
6 hacer
7     si (datos == usuario[i].datos)
8     entonces
        found = usuario[i].datos

9     sino
        i++
10    finsi
11 finfor
12 si found == null
13 entonces
    alert = true
14 sino
    mostrar_pagina_principal()
15 finsi
```



Complejidad ciclomática

- 1) $V(G) = R$ (Regiones) = 5
- 2) $V(G) = E - N + 2$ (E aristas, n nodos) = $20 - 17 + 2 = 5$
- 3) $V(G) = P + 1$ (P nodos con bifurcaciones) = $4 + 1 = 5$
(los 3 'if' y el 'while')

Conclusión: 5 caminos diferentes.

Camino 1 (datos nulos): 1, 2, 3, 16, 17

Camino 2 (found a la primera): 1, 2, 4, 5, 6, 7, 8, 10, 11, 5, 12, 14, 15, 16, 17

Camino 3 (not found con 1 usuario en Usuarios): 1, 2, 4, 5, 6, 7, 9, 10, 11, 5, 12, 13, 15, 16, 17

Camino 4 (found a la segunda): 1, 2, 4, 5, 6, 7, 9, 10, 11, 5, 6, 7, 8, 10, 11, 5, 12, 13, 15, 16, 17.

Camino 5 (found, pero datos nulos): 1, 2, 4, 5, 6, 7, 8, 10, 11, 5, 12, 13, 15, 16, 17

Casos de prueba:

- **Camino 1:** datos == null

Caso de prueba:

datos = null

Resultado: Se muestra alerta de 'Datos no introducidos'.

- **Camino 2:** datos != null, usuarios[0].datos = { JSONdeDatos }, found != null

Caso de prueba:

datos = {JSONdeDatos},

usuarios[0].datos = {JSONdeDatos}

Resultado: El usuario se hace login correctamente y se muestra la página principal

- **Camino 3:** datos != null, usuarios[0].datos = { JSONdeDatos }, found == null

Caso de prueba:

datos = { OtroJSONdeDatos }

usuarios[0].datos = { JSONdeDatos }

Resultado: Se muestra alerta de 'Usuario no encontrado'

- **Camino 4:** datos != null, usuarios[1].datos = { JSONdeDatos }, found != null

Caso de prueba:

datos = { JSONdeDatos },

usuarios[0].datos = { JSONdeDatos }

Resultado: El usuario hace login correctamente y se muestra la página principal

- **Camino 5:** datos != null, usuarios[0].datos = { null }, found != null

datos = { JSONdeDatos },

usuarios[0].datos = { null }

Resultado: Se muestra alerta de 'Error de base de datos'

4.1.5 Test de caja negra

A continuación, se muestra una prueba de clases de equivalencia en base al registro de capas de la aplicación.

Tabla 4.16 - Especificación de clases de equivalencia.

Sec.	Condición entrada	Tipo	Clases válidas		Clases no válidas	
			Entrada	Código	Entrada	Código
1	Nombre Capa	Valor	Cadena <= 45 caracteres	CEV<01>	Cadena mayor a 45 caracteres	CENV<01>
2	URL Capa	Valor	Cadena <= 200 caracteres	CEV<02>	Cadena mayor a 200 caracteres	CENV<02>
3	Opacidad	Rango	0 <= Opacidad <= 100	CEV<03>	Opacidad > 100	CENV<03>
					Opacidad < 0	CENV<04>
4	Descripción	Valor	Cadena <= 200 caracteres	CEV<04>	Cadena mayor a 200 caracteres	CENV<05>
5	Visible	Lógico	Sí	CEV<05>	Valor numérico o carácter	CENV<06>
			No	CEV<06>		

Tabla 4.17 - Pruebas de clases de equivalencia.

ID CP / Clases		Clases de equivalencia					Resultado
		Nombre capa	UPL capa	Opacidad	Descripción	Visible	
CP1	CEV<01>, CEV<02>, CEV<03>, CEV<04>, CEV<05>	“EEUU”	“unaURL”	80	“Temperaturas”	True	Capa añadida.
CP2	CEV<01>, CEV<02>, CEV<03>, CEV<04>, CEV<06>	“EEUU”	“unaURL”	80	“Temperaturas”	False	Capa añadida.
CP3	CEV<01>, CEV<02>, CEV<03>, CEV<04>, CENV<06>	“EEUU”	“unaURL”	80	“Temperaturas”	5	Error en el registro.
CP4	CEV<01>, CEV<02>, CEV<03>, CENV<04>, CEV<05>	“EEUU”	“unaURL”	80	[Cadena > 200]	True	Error. ‘Descripción’ fuera de rango.
CP5	CEV<01>, CEV<02>, CENV<03>, CEV<04>, CEV<06>	“EEUU”	“unaURL”	120	“Temperaturas”	5	Error. ‘Opacidad’ fuera de rango.
CP6	CEV<01>, CEV<02>, CENV<04>, CEV<04>, CEV<06>	“EEUU”	“unaURL”	-3	“Temperaturas”	5	Error. ‘Opacidad’ fuera de rango.
CP7	CEV<01>, CENV<02>, CEV<03>, CEV<04>, CEV<05>	“EEUU”	[Cadena >200]	-3	“Temperaturas”	5	Error. ‘URL’ fuera de rango.
CP8	CENV<01>, CEV<02>, CEV<03>, CEV<04>, CEV<05>	[Cadena > 45]	“unaURL”	-3	“Temperaturas”	5	Error. ‘Nombre’ fuera de rango.

4.2 ANÁLISIS DE RESULTADOS

4.2.1 Casos de prueba

Debido a que la aplicación no es muy compleja y no tiene muchas posibles salidas, el número de pruebas es reducido y la evaluación ha sido fácil de realizar. En el proceso de redacción de los casos de prueba aparecieron algunos casos en los que no habíamos pensado y sobre los que no había una solución clara, como fue el caso de los roles de usuario y los permisos que tenían, pues había un “vacío” que permitía acceder al gestor de usuarios a usuarios con el rol estándar. Este fue solucionado rápidamente.

También apareció un error similar en la opción de privacidad del usuario donde se puede poner público o privado el perfil y que afecta principalmente a los usuarios con el rol de moderador. Este error también fue solucionado.

Como conclusión podemos decir que la evaluación de casos de prueba ha sido satisfactoria puesto que nos ha permitido encontrar errores que nos hubiera costado encontrar de otra forma y comprobar que el resto funciona perfectamente. A pesar de esto, hay que tener en cuenta que no probablemente no se hayan realizado todos los casos de prueba posibles, por lo que podría seguir existiendo errores de funcionabilidad.

En cuanto a usabilidad se podría decir que la aplicación cumple con los requisitos planteados originalmente, por lo que es totalmente útil.

4.2.2 Cuestionarios de evaluación

Los cuestionarios de evaluación (ver ANEXO C: Cuestionarios de evaluación) dadas las circunstancias, solo se realizaron a 10 personas, el cual no es un número que permita hacer una evaluación muy informativa o real pero que sí que permite realizar un pequeño análisis de los resultados.

El cuestionario cuenta con 24 preguntas, todas ellas sobre el diseño y la usabilidad de la interfaz. El rango de edad de las personas varía entre 19 y 55, lo que nos da una perspectiva de la aplicación para diferentes edades.

Entre los encuestados había dos personas con discapacidad: dos con daltonismo y una con miopía y astigmatismo. Es importante saber si estas personas pudieron

ver perfectamente la aplicación puesto que podría haber algún color o tamaño de texto que dificultara la visión a estas personas.

El resultado general de las respuestas ha sido satisfactorio, obteniendo un 5/5 en la mayoría de las cuestiones donde se preguntaba la dificultad de acceder a ciertas pantallas o realizar alguna acción como crear una capa. Aún así, se obtuvieron algunos resultados negativos que son los que se comentarán a continuación:

La parte peor valorada ha sido la de creación de capas, pues, aunque no haya tenido una valoración por debajo de 3/5 es la que menos nota tiene de todo el cuestionario. Las principales dificultades que han tenido son que apenas se encuentran URLs de capas válidas y que si hay una equivocación al introducir los campos de las capas esta no se ve. Ambas respuestas son razonables puesto que nosotras mismas tampoco encontramos muchas capas que pudieran funcionar con la aplicación y si hay un error en los campos introducidos, esta capa no se verá puesto que no existe ningún mecanismo en la app que avise si los campos son erróneos.

En la parte del cambio de contraseña hemos recibido un comentario que comenta que a pesar de que el cambio de contraseña salió satisfactoriamente, le apareció una notificación de que había salido mal. Probablemente se deba a algún error en la comunicación con la base de datos que debería ser revisado en el futuro.

En cuanto a los puntos fuertes y ventajas de la aplicación, la mayoría de las personas están de acuerdo en que la interfaz es sencilla e intuitiva y está bien estructurada lo cual indica que la aplicación ha conseguido el objetivo de crear una interfaz sencilla y “user-friendly” que sea accesible.

En el apartado de aspectos a mejorar se repiten muchas propuestas, sobre todo la de que se debería traducir la aplicación al español, cosa que podría hacerse fácilmente en el futuro. También se comentó que el tamaño de la letra es muy pequeño, algo que también tendría fácil solución.

4.2.3 Caja negra y caja blanca

En este apartado se ha tratado de evaluar la funcionalidad y la complejidad del código analizando el flujo de ejecución y la estructuración de la clase con las

pruebas de la caja blanca, y evaluar los posibles escenarios de ejecución con las pruebas de caja negra.

En primer lugar, la prueba de caja blanca se realizó sobre un fragmento de código de la vista del login y el registro de usuarios y se transformó a pseudocódigo puesto que era necesario para su evaluación. De esta se obtuvieron los posibles recorridos que haría el código para llegar a la salida y se evaluó su complejidad. Los resultados fueron satisfactorios puesto que el flujo de la aplicación permite llegar a cualquier solución sin problemas.

En la caja negra se realizaron pruebas sobre las salidas posibles que se podrían obtener en la creación de una capa. Estas fueron las esperadas, por lo que se puede concluir que el resultado, al menos en esa sección de la aplicación, es satisfactorio.

Conclusión

En este capítulo final del trabajo se darán un conjunto de conclusiones que he sacado del resultado obtenido de realizar el trabajo y todo el proceso que ha conllevado. Primero se explicarán los objetivos conseguidos en este trabajo, que recuerdo, ha sido en equipo junto con mi compañera Andrea. También hablaré sobre posibles modificaciones o trabajos futuros que se podrían realizar en esta aplicación para mejorarla o hacerla más funcional, problemas que nos hemos encontrado a lo largo del proyecto

OBJETIVOS CONSEGUIDOS

De los objetivos propuestos en un principio por la empresa y más tarde por nosotras mismas, se han conseguido llevar a cabo con éxito la gran mayoría de ellos. Los principales eran crear una aplicación funcional que requiriera de un registro de usuarios y que permitiera la creación y gestión de capas sobre un mapa base con control de usuarios con diferentes roles. Todo ello implementado con una interfaz de usuario sencilla, intuitiva y visual que facilitara todas las utilidades de la aplicación.

De la documentación del seguimiento del trabajo (ver ANEXO B: Seguimiento de Proyecto fin de Carrera) y de las posteriores evaluaciones realizadas sobre la aplicación y nuestra propia experiencia con el trabajo podemos concluir que efectivamente la aplicación cumple con su cometido, es totalmente funcional y la interfaz cumple todos los requisitos mencionados a lo largo del trabajo.

A pesar de ello, todavía ha quedado alguna pequeña idea que se planteó como objetivo y que no se pudo realizar, debido a la falta de tiempo y de investigación, como es la opción de añadir fotos de perfil de usuario, o realizar un sistema que permitiera crear otro tipo de capas que requirieran de otros parámetros, en conclusión, hacerla más “universal”. A pesar de ello, estas opciones junto con otras más forman el siguiente apartado de trabajos futuros donde se mencionan posibles añadidos y actualizaciones a la aplicación en un futuro.

TRABAJOS FUTUROS

- Debido a que las aplicaciones para móviles son algo que una empresa de software considera indispensable hoy en día, una buena idea de futuro para este proyecto podría ser adaptar la aplicación actual a un formato móvil para poder ejecutarla desde un smartphone sin problemas de formato de pantalla o de interfaz.
- Hablando de la interfaz, esta requiere de ciertas reformas para hacerla más vistosa y “actual”. Algo que se podría realizar para solucionar este problema podría ser actualizar toda la aplicación a una versión más actual de Vuetify, dado que la anterior tiene objetos menos actuales y algunos están incluso obsoletos y no son del todo funcionales.
- El hecho de que la aplicación solo este hecha en inglés podría ser un inconveniente para atraer a los “futuros clientes”. Se podría trabajar en dar la opción de poner la aplicación disponible en español o incluso en otros idiomas.
- Como tarea pendiente nos queda hacer más búsqueda de información sobre cómo se muestran los mapas en otras plataformas como Google maps o Bing, para poder añadir coordenadas, u otro tipo de capas que no tienen el formato que hemos incluido en la aplicación.
- Otro enfoque que se le podría dar a la aplicación y sobre el que he reflexionado ha sido el poder hacer de TecnoMaps una plataforma donde los usuarios pudieran compartir sus mapas y capas con el resto de usuarios, llegando a hacer de esta una “red social de mapas”. Es una funcionalidad bastante ambiciosa y complicada, pero podría realizarse si se sigue poniendo empeño en ella.
- Si esta aplicación realmente fuera puesta en el mercado, probablemente no cumpliría con la política de datos actual, y tampoco tendría un sistema de seguridad lo suficientemente robusto, así que otros objetivos a mejorar podrían ser adaptar esta aplicación a las leyes vigentes, así como implementar un mayor nivel de seguridad en esta, por ejemplo, realizando una encriptación de la contraseña.

PROBLEMAS ENCONTRADOS

- Versión anticuada de Vue y Vuetify, que dio problemas de compatibilidad y otros errores de componentes a la hora de diseñar elementos y dar funcionalidad a ciertas áreas de la aplicación. Se podría actualizar, pero llevaría mucho tiempo realizar las correcciones debidas en el código.
- La base de datos no es muy estable, puesto que a lo largo del desarrollo se ha “caído” a causa de desconfiguraciones, falta de actualizaciones u otros problemas totalmente ajenos a nosotras. La base de datos, como he mencionado anteriormente, se encuentra localizada en la empresa y fue configurada por un trabajador de esta para nuestro uso exclusivo.
- Ciertos problemas a la hora de guardar la imagen de perfil en la base de datos, pues la foto se puede visualizar en la vista previa del icono, pero no se guarda en la base de datos. Este problema está sin resolver actualmente.
- Dificultad para encontrar mapas y capas de libre uso. Además, cada tipo de mapa requiere de unos atributos y formatos diferente que deberían ser implementados para cada uno, lo que haría más compleja la programación de la aplicación, además de ser costoso de tiempo.
- En la primera parte del trabajo que se realizó en la empresa, no se siguió una metodología estrictamente, a pesar de que se siguiera la estructura del modelo en cascada, lo cual trajo algún que otro problema de tiempo y de planificación, aunque al ser una aplicación poco compleja se pudo atajar perfectamente.
- La poca experiencia en el diseño de interfaces sumado a los problemas de compatibilidad de objetos con la versión de Vuetify hizo que el diseño de la aplicación quedara poco profesional y llevara más tiempo de elaboración del debido.
- La aplicación en sí no tiene implementado ningún método de seguridad ni sabemos si cumpliría con las leyes de protección de datos si esta aplicación se hiciera pública o fuera comercializada. Dado que no tenemos mucha experiencia en cómo realizar esto y además dimos por hecho que esta aplicación no iba a ser comercializada, no le dimos prioridad a esta

cuestión ni dedicamos mucho tiempo a investigar cómo hacerlo, lo que podría ser un futuro problema si se siguiera adelante con este proyecto.

OPINIONES PERSONALES

Con este proyecto y en concreto con la elaboración de la aplicación saco bastantes conclusiones y lecciones, además de experiencia. Durante todo este periodo de varios meses (octubre - junio) he aprendido a trabajar aún mejor con las herramientas explicadas en otras asignaturas anteriores, herramientas para elaboración de aplicaciones web (Vue, Node.js Vuetify...), herramientas para la gestión de proyectos, de productos software y en general a ganar experiencia programando, administrando y aprender de errores. Además, he tenido que aprender a usar otras herramientas como Electron o Vue-Layers o metodologías ágiles como scrum, las cuales me servirán bastante en el futuro laboral ya que están a la orden del día.

Estoy muy contenta con el proyecto en general porque se han conseguido realizar la mayoría de las tareas pendientes, que es el principal objetivo de la aplicación, y que a pesar de que posiblemente no tenga futuro esta aplicación, se ha conseguido resolver un problema real que nosotras desconocíamos y que es una pequeña base de lo que se hace en empresas como Tecnosylva.

La experiencia de poder trabajar con mi compañera Andrea en un equipo ha sido muy satisfactoria ya que nos entendemos muy bien, sabemos nuestros puntos fuertes y débiles. Además, ella siempre está dispuesta a explicar y ayudar con los problemas que surgen y se molesta en que aprendamos de ellos, incluido su apartado de la aplicación que era el Backend. Ha sido todo un placer trabajar con ella.

Lista de referencias

- [1] “Modelo en cascada” [Online]. Available: http://todounidad3fsi.blogspot.com/2014/04/31-modelo-en-cascada_2303.html
- [2] “How to run Scrum efficiently in 2019? Quick guide for beginners”. [Online]. Available: <https://habr.com/en/company/hygger/blog/455022/>
- [3] “Tecnosylva | Soluciones avanzadas para incendios forestales”. [Online]. Available: <https://tecnosylva.es/>
- [4] “vuelayers - npm”. [Online]. Available: <https://www.npmjs.com/package/vuelayers>
- [5] “OpenLayers” [Online]. Available: <https://openlayers.org/>
- [6] “ArcGIS Online”. [Online]. Available: <https://www.arcgis.com/index.html>
- [7] “ArcGIS - Wikipedia, la enciclopedia libre”. [Online]. Available: <https://es.wikipedia.org/wiki/ArcGIS>
- [8] “Salarios para empleos de Diseñador/a web en España” [Online]. Available: <https://es.indeed.com/salaries/dise%C3%B1ador-web-Salaries>
- [9] “Salarios para empleos de Programador/a web en España” [Online]. Available: <https://es.indeed.com/salaries/programador-web-Salaries>
- [10] “Salarios para empleos de Analista de sistemas en España” [Online]. Available: <https://es.indeed.com/salaries/analista-de-sistema-Salaries>
- [11] “Salarios para empleos de Técnico/a de sistemas en España” [Online]. Available: <https://es.indeed.com/salaries/t%C3%A9cnico-de-sistemas-Salaries>
- [12] “Vue.js + Nodejs/Express RestAPIs – Sequelize ORM + MySQL CRUD example” [Online]. Available: <https://grokonez.com/frontend/vue-js/vue-js-nodejs-express-restapis-sequelize-orm-mysql-crud-example>
- [13] “axios - npm” [Online]. Available: <https://www.npmjs.com/package/axios>

- [14] “Transferencia de Estado Representacional - Wikipedia, la enciclopedia libre”. [Online]. Available: https://es.wikipedia.org/wiki/Transferencia_de_Estado_Representacional
- [15] “Visual Code Studo - Code editing. Redefined.”. [Online]. Available: <https://code.visualstudio.com/>
- [16] “Vue.js - The Progressive JavaScript Framework”. [Online]. Available: <https://vuejs.org/>
- [17] “What is Vuex? | Vuex”. [Online]. Available: <https://vuex.vuejs.org/>
- [18] “Introduction | Vue Router”. [Online]. Available: <https://router.vuejs.org/>
- [19] “Vue Material Design Component Framework – Vuetify.js”. [Online]. Available: <https://vuetifyjs.com/en/>
- [20] “Electron | Build cross-platform desktop apps with JavaScript, HTML, and CSS”. [Online]. Available: <https://www.electronjs.org/>
- [21] “GitHub.” [Online]. Available: <https://github.com/>

ANEXO A: Control de Versiones

Como controlador de versiones hemos utilizado GitHub, ya que es el que usábamos cuando empezamos el proyecto en la empresa y además tenemos ya bastante experiencia usándolo para otros trabajos a lo largo del grado.

GitHub [21] proporciona un control de versiones que permite registrar todo tipo de cambio realizado en el proyecto, haciendo posible el cambio a versiones antiguas del código si es necesario, evitar pérdidas de código o para realizar un seguimiento del mismo. Todo ello sin que sea necesario almacenar cada versión del proyecto.

Para que las acciones (commit, push, pull, fetch...) sean mucho más fácil de controlar y prácticas hemos usado la herramienta Fork, que es mucho más visual ya que tiene una interfaz con la que interactuar y permite visualizar de una manera esquemática todos los commits y demás acciones del repositorio.

En el proyecto hicimos uso de ramas, ya que facilitan la creación de una sección de funcionalidad concreta en un entorno alternativo y aislado que evita que la rama principal cuente con posibles errores. En nuestro caso se crearon tres ramas diferentes, una para la funcionalidad de visualización de perfil, otra para la foto de perfil del usuario y otra rama principal alternativa donde unir las ramas anteriores.

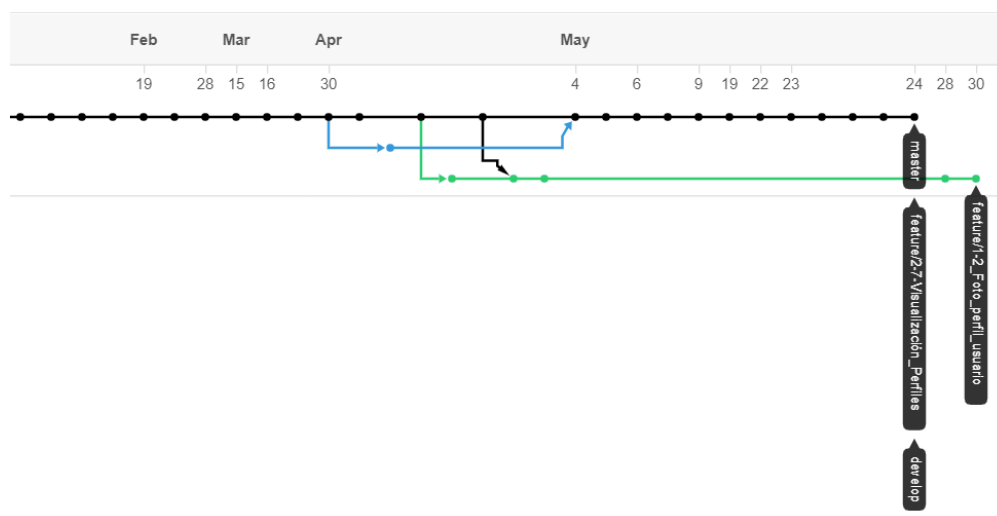


Figura Anexo A - 1 - Ramas del repositorio

En las siguientes capturas se pueden ver estadísticas con gráficos con los diferentes commits que hemos realizado a lo largo del proyecto tanto grupal como individual.

El enlace del repositorio es: <https://github.com/rchamg01/TecnoMaps>



Figura Anexo A - 3 – Frecuencia de commits propios al repositorio

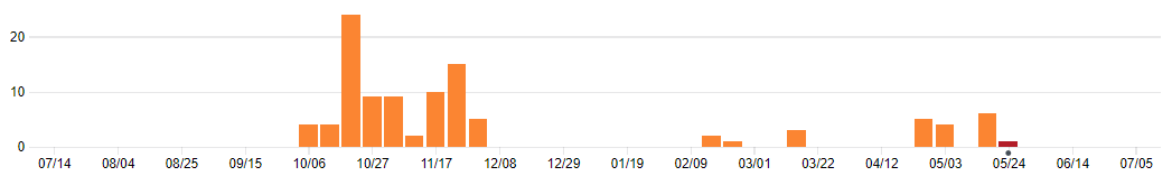


Figura Anexo A - 2 - Commits grupales en el repositorio

ANEXO B: Seguimiento de Proyecto fin de Carrera

El seguimiento del proyecto se divide en dos etapas:

- La **primera parte** se realizó para la asignatura de Prácticas en Empresa.
- La **segunda parte** para la asignatura Ingeniería del Software II y el TFG.

Seguimiento de la primera parte

Al principio no se pensó que este proyecto, que era un trabajo mandado por la empresa donde se realizaron las prácticas, llegaría a ser un Trabajo de Fin de Grado, por lo que no hay un seguimiento muy estricto o formal de este hasta que tomamos la decisión de llevarlo a TFG. A pesar de esto, sí que contamos con varios informes de seguimiento que se realizaron para el tutor de las prácticas. Estos informes detallan el objetivo de la aplicación, qué se ha realizado hasta ese momento y qué se realizará en el futuro. También se planearon reuniones con el tutor que nos llevaba el seguimiento. Las fechas más importantes del proyecto fueron:

- Comienzo del proyecto: 07/10/19
- 1ª reunión de seguimiento: 29/10/19
- 2ª reunión de seguimiento: 03/12/19
- Finalización de las prácticas: 02/12/19

A continuación, se muestra un extracto del informe final de las prácticas en la empresa.

Para la parte de planificar el proyecto, teniendo en cuenta el plazo del que disponíamos y haciéndolo lo más repartido posible, decidimos dividir la aplicación en tres “entregas” o “plazos” para distribuirnos bien en el tiempo y entre nosotras mismas. Andrea se encargaba de hacer el Backend y yo simultáneamente de realizar la respectiva parte del Frontend. Todo esto, aun así, lo fuimos planeando sobre la marcha y no fue algo concreto desde el principio. Las divisiones del trabajo que fuimos realizando fueron:

- Login y página principal, que nos llevó todo octubre.*
- Gestión de capas y usuarios en prácticamente noviembre entero.*
- Para diciembre ya teníamos todo lo que se pedía y en este tiempo restante nos dedicamos a corregir errores menores y perfeccionar y pulir la interfaz.*

[...] Con nuestro tutor de las prácticas en la universidad nos reunimos en dos ocasiones (la 1ª reunión el 5 de noviembre y la 2ª reunión el 3 de diciembre).

Seguimiento de la segunda parte

La segunda parte del proyecto se realizó durante el segundo cuatrimestre en la asignatura de Ingeniería del Software II. En ella seguimos una planificación mucho más estricta puesto que la asignatura lo requería, además de utilizar una metodología ágil. En este punto ya sabíamos que este proyecto se usaría también como TFG.

A continuación, se adjunta el documento de planificación de los Sprints que se realizó, junto con el diagrama de Gantt de esta etapa:

TecnoMaps	ACTA DE REUNIÓN DE PLANIFICACIÓN
-----------	----------------------------------

1. Información General

Fecha de realización: 24 de abril de 2020
Preparación de Sprints
Asistentes: Raquel Chamorro Giganto, Andrea Fernández Sánchez

2. Objetivos de la reunión

- Iniciación del proyecto
- Roles
- Recursos/programas externos

2.1 Iniciación del proyecto

Debido a que este proyecto está comenzado, se crearán nuevas historias de usuario para añadir funcionalidad a TecnoMaps. Se partirá de la ya creada y desarrollada en el documento de 'Nueva Funcionalidad', del cual obtendremos las historias que formarán el Product Backlog.

Para comenzar, necesitaremos documentar el trabajo en progreso, empezando por un archivo Project para llevar el control de las reuniones y Sprints. Las fechas a plasmar son:

Nombre	Fecha
Reunión de Planificación Sprint 1	24 de abril 2020
Sprint 1	27 de abril 2020 a 11 de mayo de 2020
Reunión Revisión Sprint 1	12 de mayo de 2020
Reunión Retrospectiva Sprint 1	12 de mayo de 2020
Reunión Planificación Sprint 2	12 de mayo de 2020
Sprint 2	12 de mayo 2020 a 26 de mayo de 2020
Reunión Revisión Sprint 2	27 de mayo de 2020
Reunión Retrospectiva Sprint 2	27 de mayo de 2020

Los días de comienzo y final están ambos incluidos en el Sprint, por lo que tendrían una duración de 2 semanas cada uno.

Por cada reunión, se realizará un Acta donde se recogerá todo lo decidido en ella.

2.2 Roles

Dado que somos dos personas y los roles totales suman más de dos, desarrollaremos tareas de todos los roles, Product Owner, Scrum Master y equipo de desarrollo, ya que nuestro objetivo ahora es aprender sus funciones.

2.3 Recursos/programas externos

Para ayudarnos a organizar los Sprints, el código y la comunicación, tendremos en cuenta las siguientes aplicaciones:

- **Trello** – Ideal para organizar tareas y controlar las pendientes, en progreso y hechas. Además, permite añadir comentarios a las tarjetas de tareas por si hubiera algún problema o sugerencia.
- **WhatsApp** – La usaremos a modo de plataforma de comunicación, especialmente para cuestiones urgentes y concretas.
- **Discord** – Complementa a WhatsApp en labores de comunicación, añadiendo la posibilidad de realizar videollamadas y compartición de pantalla para resolución de errores.
- **Fork** – Herramienta de control de versiones de Git. Su interfaz clara y fácil de usar nos ayudará a agilizar el proceso de creación de ramas, mergearlas, eliminarlas...

A la hora de crear features para desarrollar, hemos acordado que tendrán la siguiente nomenclatura:

feature/[NºSPRINT]-[ID]_[Nombre de la historia]

Por ejemplo:

feature/1-1_Eliminar_cuenta_usuario

TecnoMaps	ACTA DE REUNIÓN DE PLANIFICACIÓN DEL SPRINT 1
-----------	---

1. Información General

Fecha de realización: 24 de abril de 2020
Número de Sprint: 1
Asistentes: Raquel Chamorro Giganto, Andrea Fernández Sánchez

2. Objetivos de la reunión

- Preparación Sprint 1

2.1 Preparación Sprint 1

Primero, seleccionaremos las historias de usuario del Product Backlog para crear el Sprint Backlog del Sprint 1.

Las historias seleccionadas para el Sprint 1 serían: 1, 2, 3, 5, 7, 11, 13, 14, 15.

El reparto de historias de usuario será de esta manera:

	Andrea Fernández	Raquel Chamorro
ID 1	X	X
ID 2	X	X
ID 3	X	X
ID 5		X
ID 7	X	X
ID 11	X	X
ID 13	X	
ID 14	X	X
ID 15	X	X

Muchas de las historias tienen su parte en Frontend y Backend, por lo que las desarrollaremos conjuntamente para asegurar una buena coordinación.

La velocidad estimada para este Sprint es 235.

TecnoMaps	ACTA DE REUNIÓN DE REVISIÓN DEL SPRINT 1
-----------	--

1. Información General

Fecha de realización: 12 de mayo de 2020
Número de Sprint: 1
Asistentes: Raquel Chamorro Giganto, Andrea Fernández Sánchez

2. Objetivos de la reunión

- Analizar Sprint
- Puntos completados

2.1 Análisis del Sprint

El desarrollo del Sprint se ha llevado de forma bastante uniforme, es decir, las tareas se han realizado a lo largo de las 2 semanas. La comunicación ha sido muy buena, al igual que la coordinación en la creación y mergeo de ramas.

2.2 Puntos completados

En esta tabla, se expone cada tarea y si se ha llegado a completar o no:

Historia	Completada	Puntos
1. Eliminar cuenta usuario	Sí	15/15
2. Foto de perfil usuario	No	15/30
3. Más campos en perfil de usuario	Sí	20/20
5. Traducciones	Sí	5/5
7. Visualización de perfiles	No	0/50
11. Completar relaciones	Sí	50/50
13. Añadir campos a Usuario	Sí	15/15
14. Añadir tabla Acciones	Sí	20/20
15. Investigar QuickMaps	No	0/30
Total		140/235

Podemos observar que se completaron, en su mayoría, las tareas. La tarea de Foto de perfil de Usuario se llegó a empezar, pero debido a su dificultad y su excesivo empleo de tiempo en ella, se acabó abandonando y dejando para más adelante. Visualización de perfiles e Investigar QuickMaps no se llegaron a empezar.

Sacamos la conclusión de que, debido a la carga de trabajo externa y el no poder completar 3 tareas, en el próximo Sprint nos asignaríamos una cantidad menor de historias.

TecnoMaps	ACTA DE REUNIÓN DE RETROSPECTIVA DEL SPRINT 2
-----------	--

1. Información General

Fecha de realización: 27 de mayo de 2020
Número de Sprint: 2
Asistentes: Raquel Chamorro Giganto, Andrea Fernández Sánchez

2. Objetivos de la reunión

- Fortalezas
- Debilidades

2.1 Fortalezas

El hecho de tener menos tareas nos ha dado más ‘cancha’ a la hora de realizarlas: menos presión, más tranquilidad. Al parecer, eso ha aumentado los puntos completados. Andrea ha podido aplicar parte de los conocimientos que ha adquirido en su día a día en el trabajo, lo que ha agilizado un poco más el desarrollo. Además, el hecho de desarrollar un Test Plan va a ser bastante enriquecedor, especialmente para Raquel, ya que podrá usarlo en su Trabajo de Fin de Grado.

2.2 Debilidades

Puede que hayan sido pocas tareas para un Sprint de 2 semanas, a lo mejor podríamos haber añadido una o un par más cuya duración o prioridad no fuesen muy altas.

TecnoMaps	ACTA DE REUNIÓN DE PLANIFICACIÓN DEL SPRINT 2
-----------	--

1. Información General

Fecha de realización: 12 de mayo de 2020
Número de Sprint: 2
Asistentes: Raquel Chamorro Giganto, Andrea Fernández Sánchez

2. Objetivos de la reunión

- Preparación Sprint 2

2.3 Preparación Sprint 2

Como expusimos en el Acta de Retrospectiva del Sprint 1, en esta ocasión no nos asignaremos tanta carga de trabajo para reforzar la parte teórica de la asignatura. Consecuencia de ello, es posible que queden historias sin realizar del Product Backlog, pero siempre tenemos la posibilidad de realizarlas después del fin del Sprint 2.

Las historias seleccionadas para el Sprint 2 serían: 6, 7, 8, 9, 16

El reparto de historias de usuario será de esta manera:

	Andrea Fernández	Raquel Chamorro
ID 6	X	X
ID 7	X	X
ID 8	X	X
ID 9	X	X
ID 16	X	X

Al igual que en el Sprint 1, muchas de las historias tienen su parte en Frontend y Backend, por lo que las desarrollaremos conjuntamente para asegurar una buena coordinación.

La velocidad estimada para este Sprint es 190.

TecnoMaps	ACTA DE REUNIÓN DE REVISIÓN DEL SPRINT 2
-----------	--

1. Información General

Fecha de realización: 27 de mayo de 2020
Número de Sprint: 2
Asistentes: Raquel Chamorro Giganto, Andrea Fernández Sánchez

2. Objetivos de la reunión

- Analizar Sprint
- Puntos completados

2.1 Análisis del Sprint

Ha sido un Sprint con menos tareas, lo que nos ha permitido poder llevarlas mejor y de forma más tranquila, además de poder completar más porcentaje de ellas respecto al Sprint anterior.

2.2 Puntos completados

En esta tabla, se expone cada tarea y si se ha llegado a completar o no:

Historia	Completada	Puntos
6. Modificación de rol	Sí	25/25
7. Visualización de perfiles	Sí	50/50
8. Nuevo rol intermedio	Sí	40/40
9. Opción perfil público/privado	Sí	15/15
16. Test Plan	Sí	50/50
Total		180/180

Podemos observar que se completaron todas las tareas. Han sido 4 tareas menos que en el Sprint anterior, pero las hemos podido desarrollar al 100%. La última vez, se completaron 140/235 puntos, así que, aunque la cantidad de tareas haya disminuido, la productividad ha sido mayor.

Cabe mencionar que el Test Plan nos descubrió un error en la aplicación: al eliminar un usuario propio en la pestaña de User Management, se puede permanecer logeado hasta salir de la aplicación por cuenta propia. Las soluciones propuestas son:

- Cuando el usuario esté en la pestaña de User Management, no figurará en la lista de usuarios.
- El usuario estará en la lista, pero el icono de eliminar desaparecerá en su fila.

TecnoMaps	ACTA DE REUNIÓN DE RETROSPECTIVA DEL SPRINT 2
-----------	---

1. Información General

Fecha de realización: 27 de mayo de 2020
Número de Sprint: 2
Asistentes: Raquel Chamorro Giganto, Andrea Fernández Sánchez

2. Objetivos de la reunión

- Fortalezas
- Debilidades

2.1 Fortalezas

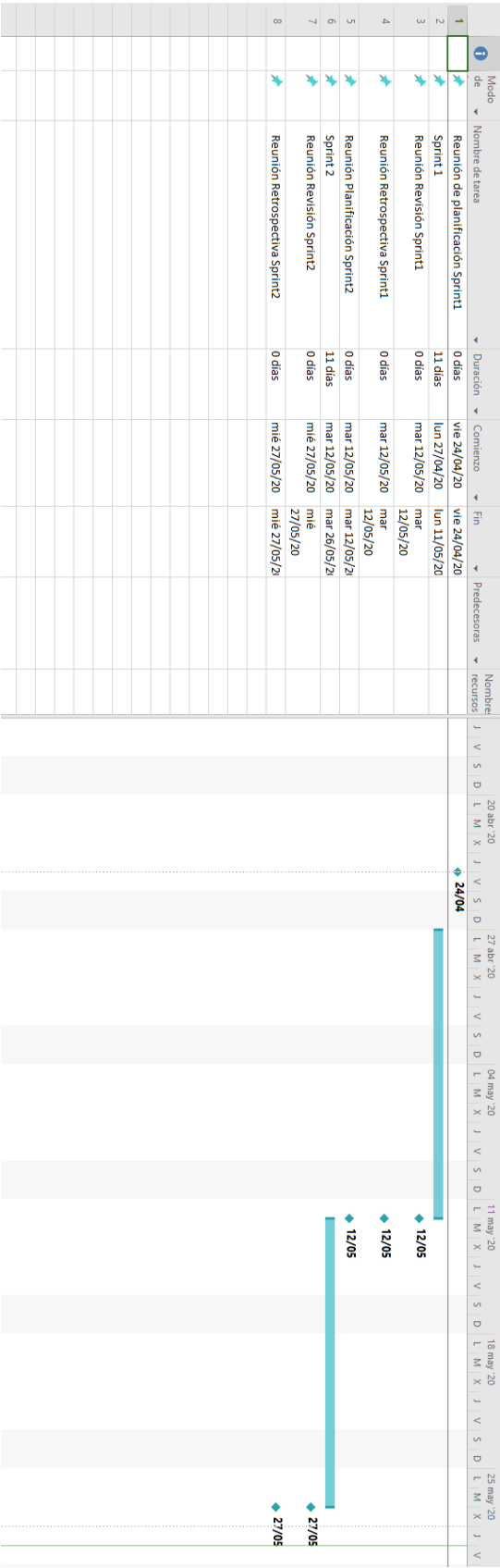
Durante el desarrollo de este Sprint, Raquel realizó un Test Plan, es decir, un conjunto de pequeñas pruebas respecto a la funcionalidad del Frontend, como, por ejemplo, comprobar que un usuario pueda logearse. Uno de los test cases nos permitió descubrir un error en la eliminación de usuario.

Por otra parte, Andrea pudo aplicar lo aprendido en el trabajo en varias ocasiones, lo que agilizó la duración del desarrollo en gran medida.

2.2 Puntos débiles

Puede que escogiéramos una cantidad pequeña de historias de usuario para ser 2 semanas, a lo mejor podríamos haber añadido una o un par más. De todas formas, no nos quisimos arriesgar porque suponíamos que la segunda quincena de mayo estaríamos más ocupadas, y así acabó siendo.

DIAGRAMA DE GANTT



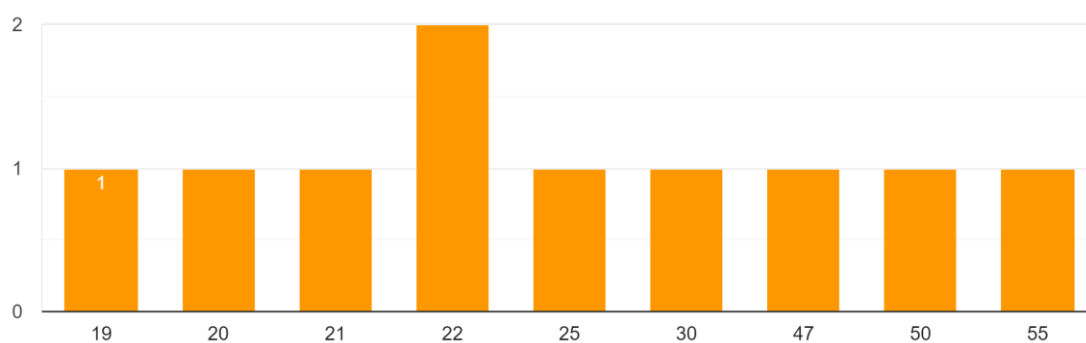
ANEXO C: Cuestionarios de evaluación

Para evaluar la aplicación se realizó un cuestionario en Google Forms a 10 personas de nuestro entorno. A continuación, se muestran las estadísticas de las respuestas obtenidas.

El enlace del cuestionario es: <https://forms.gle/dWG6UKN8Wp94HiWKA>

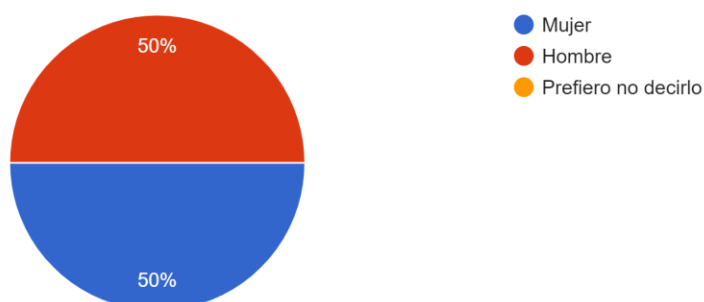
Edad

10 respuestas



Sexo

10 respuestas



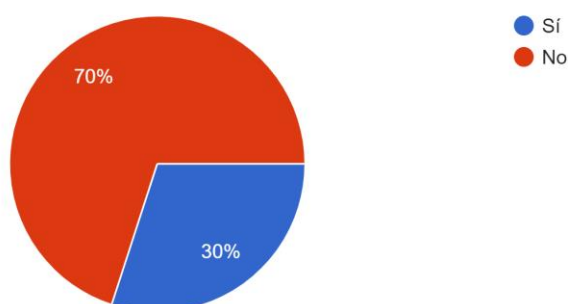
Ocupación

10 respuestas



¿Tiene alguna discapacidad? Nota: si padece daltonismo, marcar "Sí"

10 respuestas



En caso afirmativo, indique cuál

3 respuestas

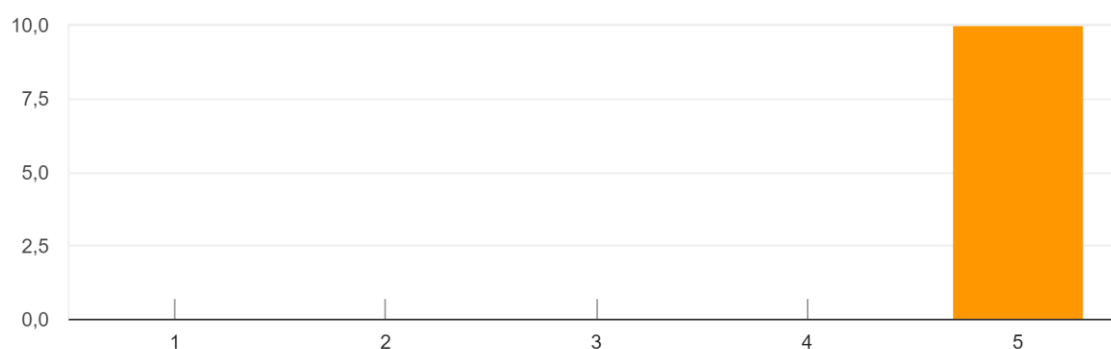
Miopía y astigmatismo

Daltonismo

Daltonismo

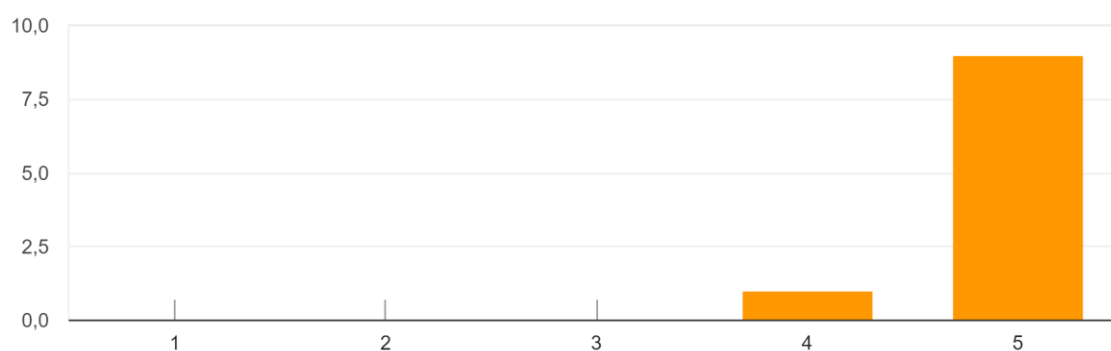
¿Cómo de fácil le ha resultado crearse una cuenta o iniciar sesión en la aplicación?

10 respuestas



¿Le parece apropiado el tamaño, color y formato de las imágenes y el texto que se muestra?

10 respuestas



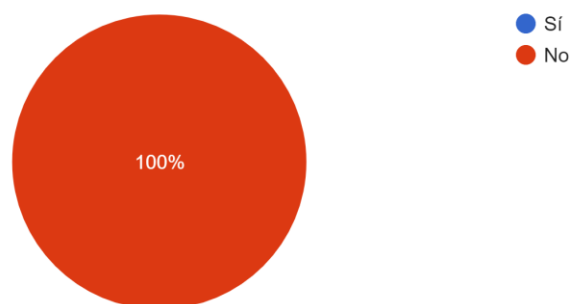
Si ha tenido algún problema al crearse una cuenta o iniciar sesión, indíquelo:

0 respuestas

Aún no hay respuestas para esta pregunta.

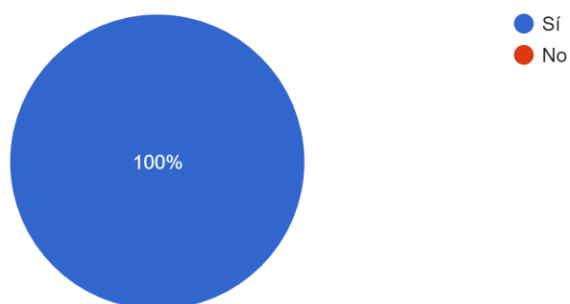
¿Ha tenido alguna dificultad a la hora de visualizar el mapa y las dos barras laterales?

10 respuestas



¿Le parece apropiado el tamaño y formato de las imágenes y el texto que se muestra?

10 respuestas



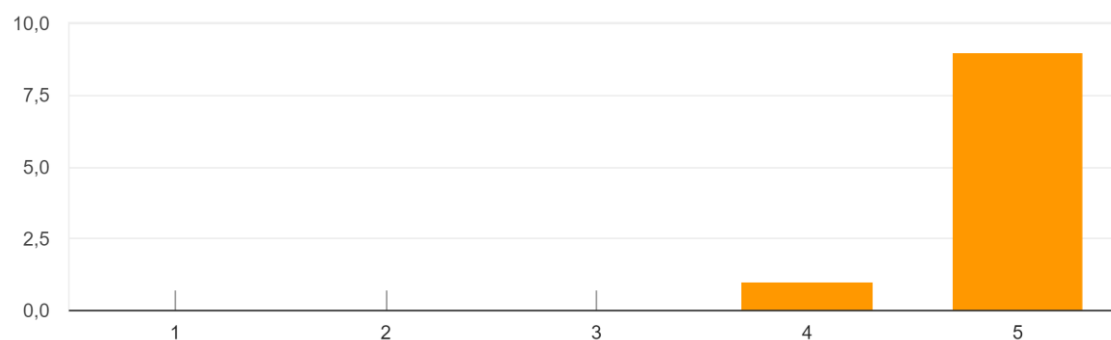
Si ha tenido alguna dificultad, indíquelo:

0 respuestas

Aún no hay respuestas para esta pregunta.

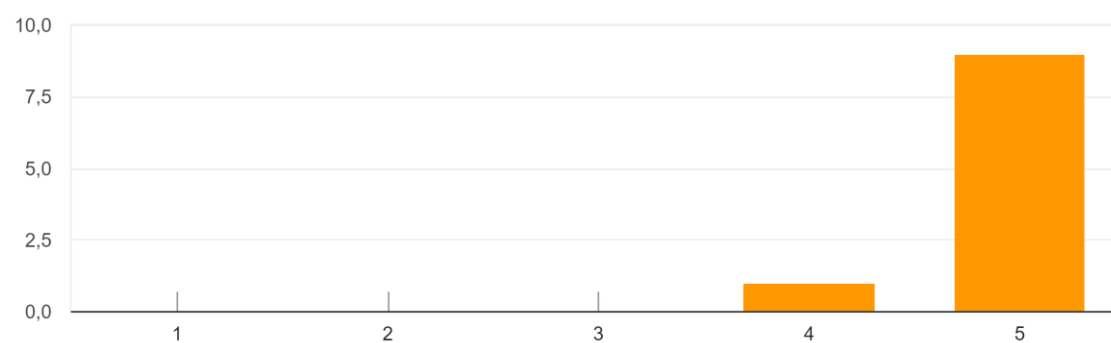
¿Cómo de sencillo cree que es visualizar la capa creada en el mapa?

10 respuestas



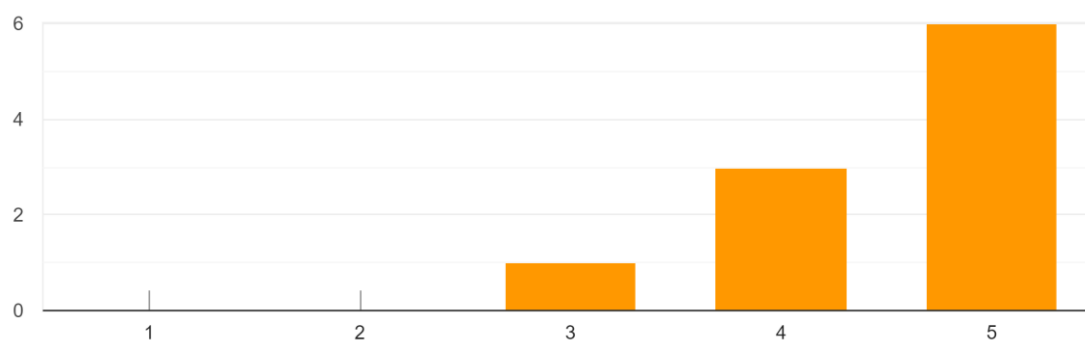
¿Cómo de sencillo cree que es visualizar el listado de capas creadas?

10 respuestas



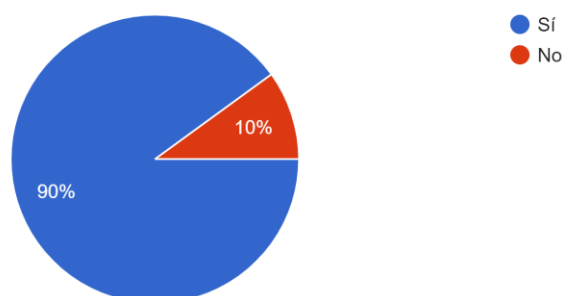
¿Cómo de sencillo cree que es crear una capa?

10 respuestas



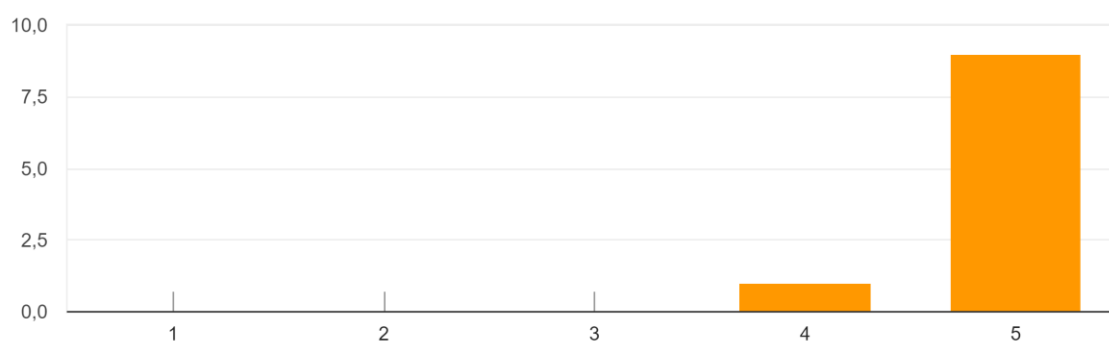
¿Le parece apropiado el tamaño y formato de las imágenes y el texto que se muestra?

10 respuestas



¿Cómo de sencillo cree que es eliminar o editar la capa?

10 respuestas



Si ha tenido alguna dificultad, indíquelo:

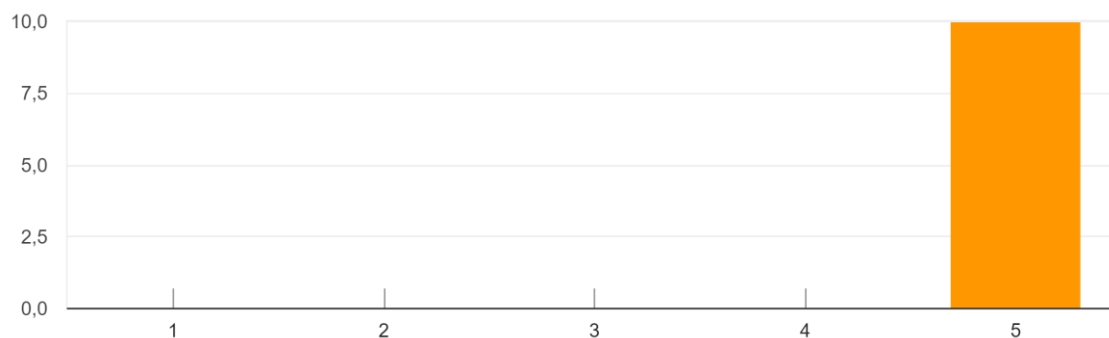
2 respuestas

La principal dificultad, al margen de la accesibilidad, es encontrar URLs de capas válidas

No se veía la capa, lo volví a intentar y pude verla

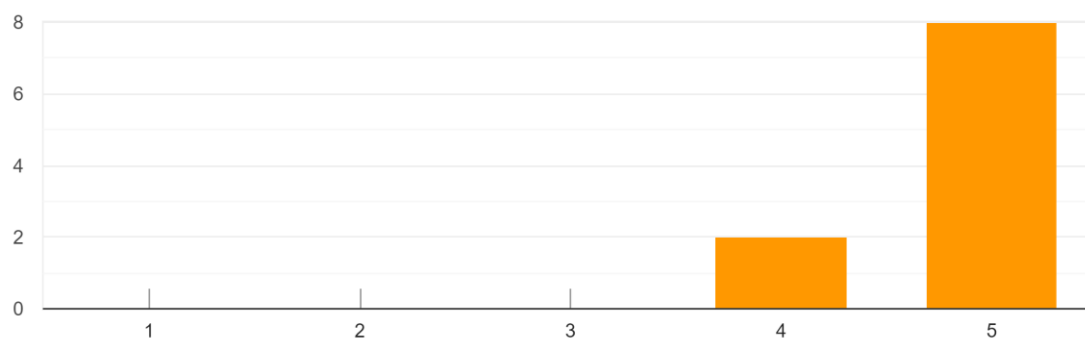
¿Cómo de fácil ha sido añadir información de usuario?

10 respuestas



¿Cómo de fácil ha sido cambiar la contraseña del usuario?

10 respuestas



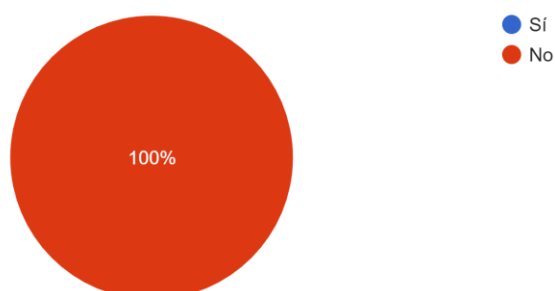
Si ha tenido alguna dificultad, indíquelo:

1 respuesta

Ha ido bien, pero cuando cambié la contraseña me indicó que era incorrecta aunque luego salió bien

¿Tuvo problemas al navegar entre las diferentes pantallas?

10 respuestas



En caso afirmativo, indique cuáles

0 respuestas

Aún no hay respuestas para esta pregunta.

Indique los puntos fuertes o ventajas de la aplicación respecto a su interfaz

10 respuestas

Es bastante sencilla

Simpleza, facilidad, minimalismo, buen aspecto.

Muy simple y fácil de manejar

Es sencilla, muy fácil de usar

Muy intuitiva y user-friendly. Me ha gustado que la cantidad de información que proporciona el mapa (info geográfica y topográfica)

La sencillez en el uso, práctica

Es fácil de usar incluso si no sabes de qué va

La creación de diversas capas es muy sencilla, y el poder activar/desactivarlas lo hace muy práctico, así como la posibilidad de modificar el nombre de capa en caso de necesitarlo.

Bien estructurada, diseño sobrio

Intuitiva y fácil de usar incluso para alguien que no sabe de mapas

Indique los aspectos que propondría mejorar

8 respuestas

Poca funcionalidad.

Más opciones

Que el texto más grande en general

Sería buena idea permitir cambiar el lenguaje entre español e inglés

El tamaño de las letras y de los símbolos del menú son pequeños, los colores son poco atractivos

La manera de introducir los datos de la capa

Solo está en inglés, se podrían poner otros idiomas

La letra del menú es algo pequeña