Escuela Politécnica Superior de Jaén

Grado en Ingeniería Informática

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Jaén

Trabajo Fin de Grado

TEXT AT: DI ALGO DE ESTE LUGAR

Alumno: Sergio Ruiz Jurado

Tutor: Prof. D. Arturo Montejo Ráez
Dpto: Departamento de Informática

Junio, 2021

Contenido

1. Introducción:

Este documento es la memoria del Trabajo de Fin de Grado del alumno Sergio Ruiz Jurado, en la titulación de Ingeniería Informática. Este proyecto tiene como objetivo principal la planificación, diseño y desarrollo de una aplicación para dispositivos móviles.

El objetivo de la aplicación es permitir al usuario crear notas sobre la posición geográfica en la que se encuentre y guardarla como un marcador en un mapa. Dichas notas podrán ser públicas o privadas. De esta manera, el resto de usuarios podrán consultar las notas de los marcadores de aquellos sitios que se encuentren cerca de su posición, así como filtrarlas para realizar consultas de lugares específicos.

1.1. Novedad de la propuesta

Muchas de las aplicaciones más populares de hoy en día son aquellas que tienen un carácter social. Un gran ejemplo de este tipo de aplicaciones es Twitter, cuya popularidad viene de la capacidad que cualquier usuario tiene de poder compartir lo que quiera con toda la comunidad.

También existen muchas otras aplicaciones que trabajan con mapas, pero la gran mayoría tienen como objetivo la navegación, es decir, indicar al usuario como ir de un punto “A” a un punto “B”. Ejemplos de estas son Google Maps o Waze. Y si bien en estas también se pueden poner marcadores, su alcance es mucho más limitado, siendo usados como meros recordatorios de posiciones que el usuario crea importantes sin poder añadirle información, o teniendo que ser creados solo en negocios que hayan sido verificados por la propia compañía.

La mayor novedad de esta propuesta es la combinación de estos dos aspectos, la capacidad social de poder compartir cualquier ubicación con el resto de la comunidad sin complicaciones y sin los filtros que otras aplicaciones pondrían.

Además, la aplicación tendrá un carácter simple, teniendo el usuario unas pocas opciones bien definidas y sin forzar un tipo de uso predefinido. Por lo tanto, su usabilidad será sencilla y cada usuario podrá darle el uso que vea conveniente a las capacidades de la aplicación. Por ejemplo, un usuario puede usarla como una simple aplicación para crear recordatorios mientras que otro puede usarla para hacer cazas de tesoro.

1.2. Metodología y estructura de la memoria

La metodología que se va a seguir es el modelo de **desarrollo en cascada.**

Se trata de un procedimiento de desarrollo lineal, en el cual se divide los procesos del desarrollo de la aplicación en distintas fases iterativas, en la cual cada una de ellas se basa en la anterior y verifica los resultados conseguidos en esta. Cada una de las fases se sucede una detrás de la otra como en una cascada. Estas fases son:

- **Fase de análisis:**

  En esta fase se realiza una definición detallada de todos los requisitos que debe de tener la aplicación, así como que funciones o características debe de tener para cumplir estos requisitos. También se realiza un estudio de los costes que puede tener el desarrollo.

- **Fase de diseño**

  En esta fase se formula una solución específica en base a lo obtenido en la fase de análisis. En esta fase se obtiene un borrador preliminar con el plan de diseño de software así como planes de prueba para los distintos componentes.

- **Fase de implementación:**

  En esta fase se programa el software en base a lo obtenido en las dos fases anteriores. El resultado es una primera versión del producto software final.

- **Fase de Validación**

  En esta fase se comprueba si lo obtenido en la fase de implementación valida los requisitos definidos en la fase de análisis mediante pruebas. Si el producto software supera dichas pruebas, está listo para su lanzamiento.

(IONOS España S.L.U, 19)

Esta memoria se encuentra estructurada siguiendo cada una de las distintas fases del modelo de cascada. En cada apartado se detallará todo lo relacionado con esa fase del proyecto. Además, se incluirá un último apartado de conclusiones, donde se reflexionará sobre el trabajo realizado.

2. Análisis

En esta etapa del desarrollo en cascada, se debe determinar cuáles son las necesidades y objetivos a cumplir del proyecto según lo pedido por el cliente, y, a partir de estos, concretar todos los requisitos que se deben cumplir durante el desarrollo para que estos se cumplan.

2.1. Ámbito

Esta aplicación se encuentra en el ámbito de las aplicaciones móviles, debiendo ser compatible con la mayoría de sistemas posibles**.**

Debido al uso de mapas y la necesidad de conocer la localización del usuario, los dispositivos deberán disponer de un método de geolocalización preciso.

El carácter social de la aplicación lleva a la necesidad de que los usuarios deban de estar registrados en esta para poder darle uso.

2.2. Objetivos generales

El objetivo principal de esta aplicación es poder crear anotaciones asociadas a posiciones geográficas y poder consultar aquellas anotaciones cercanas a la posición actual del usuario.

Los usuarios podrán añadir a su vez etiquetas, similares a los hashtag usados en Twitter, que podrán ser usados para filtrar las anotaciones que aparezcan en el mapa.

Podrán restringir el acceso a las notas de dos maneras, haciendo que una nota sea privada y, por tanto, solo el usuario que la ha creado pueda acceder a ella y por distancia, limitando la distancia a la que otro usuario debe de estar para que dicha nota sea visible.

2.3. Requisitos funcionales:

Los requisitos funcionales son descripciones explicitas sobre el comportamiento que debe tener un programa y la información que debe manejar. Es decir, son las descripciones de los elementos y funcionalidades que deben programarse y desarrollar en el programa.

Para el caso de la aplicación que se va a desarrollar, tras el análisis, se han determinado los siguientes requerimientos funcionales:

REQ-1: Identificación

Código	RF-1	Prioridad	Alta
Título		Identificación	
Descripción:			
El usuario deberá identificarse utilizando una cuenta de Google o bien mediante correo electrónico y contraseña. Si el usuario no se ha autentificado, tendrá la opción de crear una nueva cuenta, introduciendo correo y contraseña.			

REQ-2: Localización

Código	RF-2	Prioridad	Alta
Título		Localización	
Descripción:			
Una vez que el usuario haya ingresado en la aplicación, esta deberá ser capaz de geolocalizarlo y mantener su posición en todo momento.			

REQ-3: Creación de notas

Código	RF-3	Prioridad	Alta
Título		Creación de notas	
Descripción:			

El usuario deberá ser capaz de crear notas de la posición en la que se encuentra en el momento de la creación de esta. A dichas notas se le podrán añadir etiquetas escribiendo el carácter '#' delante de la palabra que se quiera convertir en etiqueta. Podrá editar la privacidad de la nota, de tal manera que pueda elegir si la nota será pública o privada. También podrá modificar la distancia máxima a la que la nota será visible a otros usuarios dentro de un rango. Dichas notas serán desplegadas en el mapa como puntos.

REQ-4: Consulta de notas

Código	RF-4	Prioridad	Alta
Título		Consulta de notas	
Descripción:			
El usuario podrá consultar la información de sus propias notas privadas o cualquiera de las notas públicas si se encuentra en su radio de acción.			

REQ-5: Filtrado de notas

Código	RF-5	Prioridad	Media
Título		Consulta de notas	
Descripción:			
El usuario podrá realizar un filtrado de las notas que aparecen en el mapa. Dicho filtrado podrá ser por etiqueta o por usuario. Si se filtra por etiqueta, se mostrarán todas las notas que contengan dicha etiqueta en su descripción que se encuentren cercanas a la posición actual. Si se filtra por usuario, se mostrarán todas las etiquetas creadas por el usuario indicado que se encuentren cercanas a la posición actual. Solo puede haber un filtro activo en cada momento.			

REQ-6: Modificación de notas

Código	RF-6	Prioridad	Media
Título		Modificación de notas	
Descripción:			
El usuario podrá consultar, modificar o eliminar cualquiera de sus propias notas en todo momento.			

2.4. Requisitos No funcionales:

Los requerimientos no funcionales son aquellos que no especifican un funcionamiento específico que pueda ser programable, sino un comportamiento general independiente de estos.

Estos son los requisitos no funcionales que debe de tener la aplicación

RNF-1: Facilidad de uso

Código	RNF-1	Prioridad	Media
Título		Facilidad de Uso	
Descripción:			
La aplicación debe de ser fácil de usar para todo tipo de usuario, tanto experto como no experto.			

RNF-2: Dispositivo

Código	RNF-2	Prioridad	Media
Título		Dispositivos Móviles	
Descripción:			
La aplicación se ejecutará en dispositivos Android con versión 8.0 o superior o iOS versión 10 o superior.			

RNF-3: Conexión a internet

Código	RNF-1	Prioridad	Media
Título		Conexión a internet	
Descripción:			
El dispositivo debe de contar con conexión a Internet para poder acceder a todas las funciones de la aplicación.			

RNF-4: Capacidad de geolocalización

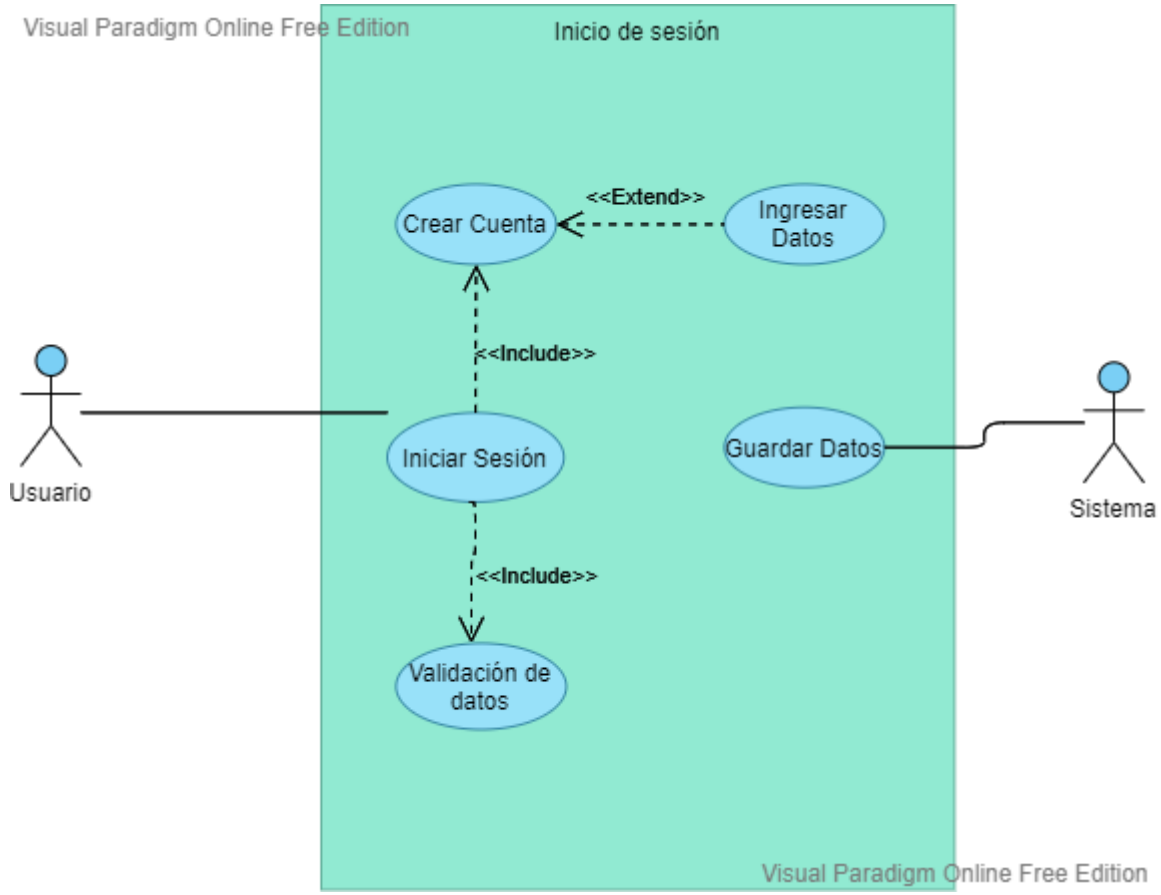
Código	RNF-1	Prioridad	Media
Título		Capacidad de gelocalización	
Descripción:			
El dispositivo debe de contar con un método de geolocalización para poder acceder a las funcionalidades de la aplicación.			

2.5. Casos de Uso:

Los casos de uso son la descripción de una acción o actividad, acompañados por un diagrama en el que se muestran todas las interacciones del sistema. Para cada caso de uso se da su diagrama, se indican que actores toman parte en esta actividad, su especificación y cuáles son los posibles cursos de las acciones que se pueden tomar.

Estos casos de uso son definidos a partir de los requisitos definidos en el apartado 2.a.

2.5.1.Inicio de sesión



Descripción de Actores

Usuario

Actor	Usuario	Identificador: Usu
Descripción	Usuario de la aplicación	

Sistema

Actor	Sistema	Identificador: Sis
Descripción	Sistema: Aplicación y parte de la nube	
Relación	El sistema se encarga de mostrar la información que el usuario inserta, así como de guardar la información que el usuario da.	

Especificación de Casos de Uso

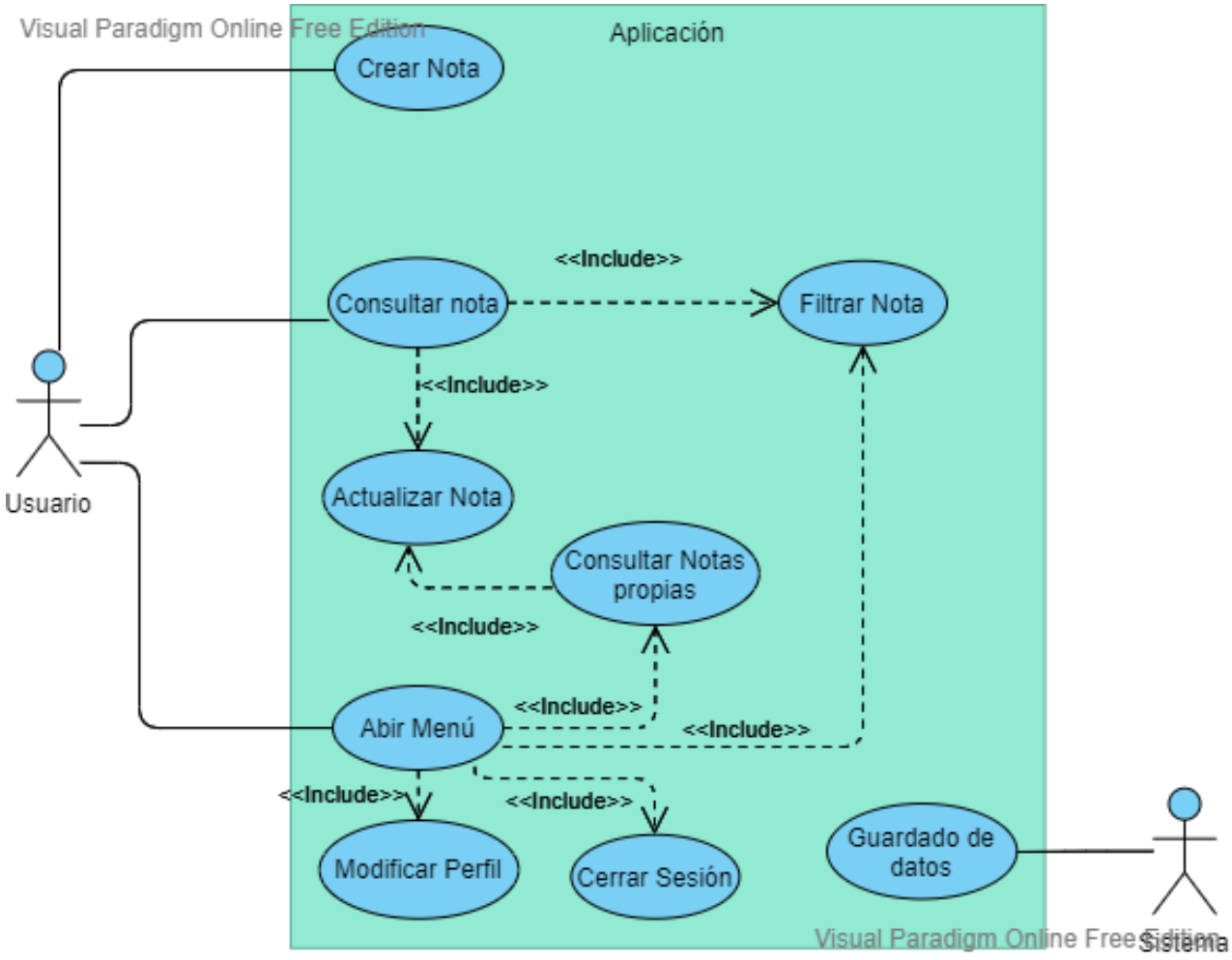
Inicio de sesión

Caso de Uso	Inicio de sesión	**Identificador:** CU1
Actores	Usuario y Sistema	
Tipo	Primario	
Precondición	El usuario debe de tener conexión a internet	
Postcondición	El usuario obtiene acceso a las funcionalidades de la aplicación	
Descripción	Inicio de sesión del usuario en la aplicación	

Curso Normal

Nro.	Ejecutor	Paso o Actividad
1	Usuario	El usuario entra a la aplicación
2	Sistema	Se le muestra al usuario la opción de iniciar sesión
3	Usuario	El usuario ingresa sus datos de cuenta
4	Sistema	Se validan los datos ingresados por el usuario. En caso de ser válidos, se da paso a la aplicación, en caso contrario se le vuelve a solicitar dichos datos
5	Sistema	Si es la primera vez que el usuario usa la aplicación, el sistema guarda la información

2.5.2. Aplicación



Descripción de Actores

Actor	Usuario	Identificador: Usu
Descripción	Usuario de la aplicación	

Actor	Sistema	Identificador: Sis
Descripción	Sistema: Aplicación y parte de la nube	

Especificación de Casos de Uso

Consultar Nota

Caso de Uso	Consultar nota	**Identificador:** CU2
Actores	Usuario y Sistema	
Tipo	Primario	
Precondición	El usuario debe de tener conexión a internet y haber iniciado sesión	
Postcondición	---	
Descripción	El usuario consulta la información de una de las notas que hay en el mapa y tiene la opción de puntuarla	

Curso Normal

Nro.	Ejecutor	Paso o Actividad
1	Usuario	El usuario pulsa sobre una de las notas del mapa
2	Sistema	Se le muestra la información de dicha nota al usuario

Cursos Alternos

Nro.	Descripción de acciones alternas
2	Durante la consulta, el usuario puede pulsar sobre una de las etiquetas para que se realice un filtro por etiqueta.
2	Durante la consulta, si la nota es del propio usuario, podrá acceder a la opción de actualizar dicha nota desde la propia consulta.

Crear Nota

Caso de Uso	Crear Nota	**Identificador:** CU3
Actores	Usuario y Sistema	
Tipo	Primario	
Precondición	El usuario debe de tener conexión a internet y haber iniciado sesión	
Postcondición	Se guarda la nota creada por el usuario y se despliega en el mapa	
Descripción	El usuario crea una nota de la posición en la que se encuentra en ese momento. Esta se guarda para su futura consulta.	

Curso Normal

Nro.	**Ejecutor**	**Paso o Actividad**
1	Usuario	El usuario pulsa sobre el botón de crear nota
2	Sistema	Se le muestra el formulario de creación de nota al usuario
3	Usuario	El usuario rellena la información del formulario.
4	Sistema	El sistema guarda la información de la nota.

Consultar notas propias

Caso de Uso	Consultar una nota propia	**Identificador:** CU4
Actores	Usuario y Sistema	
Tipo	Secundario	
Precondición	El usuario debe de tener conexión a internet y haber iniciado sesión.	
Postcondición	------	
Descripción	El usuario consulta sus propias notas.	

Curso Normal

Nro.	Ejecutor	Paso o Actividad
1	Usuario	El usuario pulsa sobre el botón del menú
2	Sistema	Se le muestra el menú al usuario
3	Usuario	El usuario pulsa sobre la opción de “notas propias”
4	Sistema	Se le muestra al usuario una lista de las notas que él ha creado.
5	Usuario	El usuario selecciona una de las notas
6	Sistema	Se le muestra la información de dicha nota al usuario

Cursos Alternos

Nro.	Descripción de acciones alternas
5	Tras consultar la lista, el usuario puede no seleccionar una nota, en cuyo caso volvería a la pantalla principal.
6	Tras consultar la información de la nota, el usuario podría modificar algún dato de dicha nota.

Filtrado de Notas

Caso de Uso	Filtrado de notas	**Identificador:** CU5
Actores	Usuario y Sistema	
Tipo	Secundario	
Precondición	El usuario debe de tener conexión a internet y haber iniciado sesión.	
Postcondición	Solo se muestran las notas que cumplan el filtro asignado por el usuario	
Descripción	El usuario agrega un filtro a la aplicación, de tal manera que solo se mostrarán en el mapa aquellas notas que cumplan dicho filtro	

Curso Normal

Nro.	Ejecutor	Paso o Actividad
1	Usuario	El usuario pulsa sobre el botón del menú
2	Sistema	Se le muestra el menú al usuario
3	Usuario	El usuario selecciona el campo de filtro y escribe el filtro que desea poner
4	Sistema	El sistema detecta el filtro y solo muestra aquellas notas que lo cumplan

Modificar Perfil

Caso de Uso	Modificar Perfil	**Identificador:** CU7
Actores	Usuario y Sistema	
Tipo	Secundario	
Precondición	El usuario debe de tener conexión a internet y haber iniciado sesión.	
Postcondición	Los datos del usuario son actualizados	
Descripción	El usuario modifica sus datos.	

Curso Normal

Nro.	Ejecutor	Paso o Actividad
1	Usuario	El usuario pulsa sobre el botón del menú
2	Sistema	Se le muestra el menú al usuario
3	Usuario	El usuario pulsa la opción de modificar perfil
4	Sistema	El sistema le muestra el formulario de modificación de perfil
5	Usuario	El usuario modifica aquellos datos que quiera cambiar.
6	Sistema	El sistema guarda los cambios

Cursos Alternos

Nro.	Descripción de acciones alternas
5	Tras entrar en el formulario, el usuario puede no modificar ningún dato, en cuyo caso volvería a la pantalla principal.

Cerrar Sesión

Caso de Uso	Cerrar Sesión	**Identificador:** CU8
Actores	Usuario y Sistema	
Tipo	Primario	
Precondición	El usuario debe de tener conexión a internet y haber iniciado sesión.	
Postcondición	Se cierra la sesión del usuario.	
Descripción	El usuario cierra su sesión en la aplicación	

Curso Normal

Nro.	Ejecutor	Paso o Actividad
1	Usuario	El usuario pulsa sobre el botón del menú
2	Sistema	Se le muestra el menú al usuario
3	Usuario	El usuario pulsa la opción de cerrar sesión
4	Sistema	El sistema cierra la sesión del usuario.

2.6. Tecnologías

Otro aspecto importante a detallar antes de proceder al resto de iteraciones es especificar las tecnologías a usar, las escogidas para este proyecto, y el motivo por el cual han sido escogidas:

2.6.1.Framework: Ionic-Angular.

Uno de los problemas en el desarrollo de aplicaciones móviles es el conseguir llegar al mayor porcentaje de usuarios posible. La mayor barrera es la de los sistemas operativos que tienen los dispositivos móviles.

En mayo de 2021, Android tenía un 72.72% del share mientras que iOS contaba con un 26.46%. En conjunto, ambos sistemas tienen más del 99% del share del mercado móvil (StatCounter, 2021), sin embargo son completamente incompatibles entre ellos en cuanto al desarrollo de aplicaciones respecta.

A la hora de desarrollar una aplicación móvil habría que elegir si crearla solo para uno de estos dos sistemas, de tal manera que los usuarios de uno de ellos no podrían usarla, o bien desarrollar la aplicación dos veces, una para Android y otra para iOS, lo cual conllevaría casi el doble de esfuerzo, así como tener que usar dos tecnologías y lenguajes de programación muy distintos, Kotlin y Android Studio como lenguajes y entorno de desarrollo en el caso de querer crear la aplicación en Android y Swift y Xcode en el caso de querer desarrollar para iOS.

No obstante, existe otra forma de desarrollar aplicaciones para dispositivos móviles de tal manera que solo se escribe código una vez y se puede ejecutar en diversas plataformas, se trata de las **Aplicaciones Webs Progresivas,** o **PWA** por sus siglas en inglés.

Una PWA es un "híbrido" entre una aplicación nativa, aquellas desarrolladas específicamente para un único sistema, y una aplicación web tradicional. Son, a grandes rasgos, aplicaciones web, que mediante el uso de tecnologías específicas, como Service Workers, funcionan más como una aplicación nativa que como una página web. (Vidal, 2019)

Gracias a esto, es posible crear una aplicación que se abra en el navegador del dispositivo, pero de una manera que el usuario final no llega a notarlo. Así, estas aplicaciones están disponibles en cualquier sistema con un navegador compatible y podemos saltarnos el problema del desarrollo nativo, siendo estas aplicaciones compatibles, en teoría, tanto con Android y iOS, e incluso con sistemas operativos de escritorio como Windows 10.

Puesto que uno de los requisitos era la compatibilidad tanto con sistemas Android como con sistemas iOS, esta solución será desarrollada como una PWA.

Se usará **IONIC** con base **Angular**, un framework ideado para la creación de aplicaciones web progresivas destinadas a ser usadas en plataformas móviles.

Angular es un framework diseñado para la creación de aplicaciones nativas, PWA y aplicaciones de escritorio en lenguaje HTML5 y Typescript. Cuenta con una gran cantidad de herramientas y documentación que facilitan en gran medida un rápido aprendizaje y desarrollo de las aplicaciones. (Angular, 2021)

Ionic es una Toolbox para la creación de interfaces gráficas móviles que puede trabajar sobre distintos frameworks, como en este caso, sobre Angular. Al igual que Angular, otorga una gran cantidad de documentación y plugins que facilitan el desarrollo de interfaces gráficas con apariencia de aplicación nativa. (Ionic, 2021)

2.6.2.Mapa: Mapbox.

Uno de los elementos más importantes de nuestra aplicación es el mapa, puesto que casi todas las acciones que realiza el usuario tienen lugar en este, debido a ello, la elección de la tecnología que nos dote de un mapa es de vital importancia.

La primera tecnología que a muchos desarrolladores les viene a la cabeza es **Google Maps**. Google ofrece una API muy potente y con muchas funciones que serían muy útiles para el desarrollo de nuestra APLICACIÓN. No obstante, antes de tener acceso a la clave que nos permite utilizar sus servicios en nuestra aplicación, hay que adelantar una forma de pago. Si bien no es necesario llegar a realizar un pago hasta llegar a una cierta cantidad de cargas de mapa y Google ofrece un crédito mensual de 200$ gratuito, que tiende a ser suficiente para este tipo de aplicaciones, se ha preferido buscar una opción que no requiera este adelanto de pago.

La opción gratuita más usada con diferencia es **OpenStreetMap**, se trata de un proyecto abierto, colaborativo y de código libre, donde los propios usuarios son los que cartografían, añaden datos y corrigen los posibles errores que existan en su base de datos. En los últimos años, OSM se ha convertido en una de las opciones para la creación de mapas más usadas por grandes empresas, como Aplicaciónle o Amazon, las cuales, juntos a los usuarios, están contribuyendo a la adición de mucha información en los mapas que OSM ofrece. (OpenStreetMap, 2021)

Para el desarrollo del mapa de la aplicación, he decidido usar **Mapbox**. Se trata de una tecnología desarrollada sobre OSM, siendo una de las mayores contribuidoras a su desarrollo. Mapbox ofrece una serie de APIs y de SDK que ayudan a desarrollar mapas muy personalizados, además de contar con una documentación muy detallada así como una gran cantidad de tutoriales que ayudan al desarrollo. Si bien se trata de una opción de pago, está disponible de forma gratuita y sin adelanto de forma de pago hasta 50000 cargas de mapas mensuales, más que suficiente para el alcance inicial de la aplicación. (Mapbox, 2021)

2.6.3.Backend – Firebase

El otro gran elemento importante de toda Aplicación y web es el Backend. Desarrollar toda esta parte de una aplicación es, posiblemente, la más complicada y que más tiempo consume.

Para lidiar con esto de la manera más rápida y efectiva se ha optado por usar una solución BaaS (Backend as a Service) en lugar de desarrollar un Backend desde cero para esta aplicación.

Dichas soluciones ofrecen entregan una serie de servicios, como bases de datos o identificación, que permiten a los desarrolladores prescindir totalmente de una API personalizada. (Alonso Vega, 2017)

Para el desarrollo de esta Aplicación, se usará Firebase, el BaaS de Google. Esta solución cuenta con herramientas como una base de datos con capacidad de almacenamiento y acceso a datos en tiempo real e incluso sin conexión, métodos de identificación tanto por correo como usando cuenta de Google o Hosting para el despliegue de la PWA.

Cuenta además con un plan gratuito, sin necesidad de adelanto de método de pago, que cubre sin problemas el alcance esperado de la aplicación.

Otras opciones alternativas a Firebase son AWS Amplify, la solución BaaS de Amazon, o Back4Aplicación, una solución de código abierto. No obstante se ha decidido por el uso de Firestore debido a que estas otras soluciones no contaban con algún elemento considerado como necesario, como el hosting o una mejor documentación y tutoriales que faciliten el desarrollo de la aplicación.

2.7. Tareas y estimación de tiempo

Una vez vistos los requisitos de la aplicación, así como las tecnologías que se van a utilizar, se divide el desarrollo la aplicación en distintas tareas que sirven a modo de hitos durante el desarrollo. Las tareas en las que se ha dividido el desarrollo de esta aplicación son:

Tarea 1: Diseño de los elementos gráficos.

Diseño de los diversos elementos gráficos de la aplicación de forma previa al desarrollo del software.

Tarea 2: Creación del proyecto en las diversas tecnologías

Preparación de los distintos elementos relacionados con las tecnologías que se van a usar en la creación de la solución software.

Tarea 3: Implementación del sistema de Login/Registro

Sistema de Registro y de Login, que permita al usuario crear una cuenta o usar su cuenta de Google para acceder a la aplicación, así como la familiarización con el funcionamiento del Backend escogido y su conexión con el framework.

Tarea 4: Implementación de las funcionalidades básicas de la aplicación.

Implementación de las distintas funcionalidades básicas de la aplicación, esta tarea se divide en las siguientes subtareas:

Integración del mapa: Creación del mapa mediante la tecnología escogida, así como dotación de controles y geolocalización.

Creación de notas: Capacidad de creación de las notas en el punto geográfico sobre el que se encuentra el usuario en el momento, así como el guardado de la información dada por este.

Consulta de notas: Mostrar las notas cercanas a la posición del usuario en el mapa y mostrar la información de estas.

La finalización de esta tarea supondría tener la base de la aplicación, con las capacidades mínimas necesarias para poder probar su funcionamiento.

Tarea 5: Implementación de las funcionalidades avanzadas de la aplicación.

Una vez implementadas todas las funcionalidades básicas de la aplicación y habiendo verificado que estas funcionan correctamente, se procederá a la implementación de aquellas otras funcionalidades que complementan a las básicas para dotar a la aplicación de todas aquellas funcionalidades descritas en el análisis de requisitos. La implementación de estas funcionalidades avanzadas se divide en las siguientes subtareas:

Modificación de notas: Capacidad del usuario de poder modificar los datos de sus propias notas así como borrarlas, tanto desde una opción en el menú como desde sus propias notas en el mapa.

Modificación del perfil: Capacidad del usuario de poder modificar la información de su perfil.

Filtrado de notas: Capacidad del usuario de poder añadir un filtro de tal manera que solo aparezcan en el mapa aquellas notas que cumplan dicho filtro.

Con todas las tareas y subtareas definidas, el tiempo estimado de desarrollo viene reflejado en la siguiente tabla:

Tarea	Tiempo (horas)	Tiempo(días)
Tarea 1: Diseño de los elementos gráficos	**16**	**2**
Diseño y creación de bocetos	16	2
Tarea 2: Creación del proyecto	**8**	**1**
Creación del proyecto en Ionic/Angular	2	0,25
Creación del proyecto en Mapbox	2	0,25
Creación del proyecto en Firebase	4	0,5
Tarea 3: Sistema de Login/Registro	**48**	**6**
Pantalla de Login	8	1
Pantalla de Registro	8	1
Servicio de Autentificación/Registro	16	2
Servicio de Base de datos	16	2
Tarea 4: Funcionalidades básicas	**52**	**6,5**
Creación del mapa		
Pantalla del mapa	8	1
Funcionalidades del mapa	4	0,5
Creación de notas		
Pantalla de creación de nota	8	1
Servicio de BBDD	8	1
Consulta de notas		
Pantalla de información de la nota	8	1
Servicio de BBDD	8	1
Funcionalidades del mapa	8	1
Tarea 5: Funcionalidades avanzadas	**44**	**5,5**
Modificación de notas		
Pantalla de modificación de notas	8	1
Modificación de pantalla de info de nota	4	0,5
Servicio de BBDD	4	0,5
Modificación de perfil		
Pantalla de modificación de perfil	8	1
Servicio de BBDD	4	0,5
Filtrado de notas		
Sistema de filtrado	8	1
Funcionalidades del mapa	8	1
TOTAL	**168**	**21**

2.8. Estimación de coste

Con la estimación de tiempo ya acabada, se estima que la duración del proyecto sea de aproximadamente 21 días, aunque existe la posibilidad de que, por dificultades que puedan aparecer, esta duración sea mayor.

El salario medio de un analista programador en España, a fecha de este proyecto, es de 14.46€/hora. Suponiendo un trabajo diario de 8 horas, durante los 21 días esperados de desarrollo, el coste salarial del desarrollo de la Aplicación sería de **2430€.** (Indeed, 2021)

El equipo a usar durante el desarrollo tiene un valor de **1500€**, con un tiempo de 2 años hasta que el hardware se considere obsoleto, por lo que la amortización del equipo durante el tiempo de uso es de **45.50€**

Firebase tiene un tanto una versión de pago como una versión gratuita. En esta última existen algunas limitaciones, como una cantidad de peticiones de consulta o borrado diarias limitadas. Analizando el alcance previsto de esta aplicación, se ha determinado que la versión gratuita cumple los requisitos necesarios para el desarrollo de esta aplicación. Por tanto se usará esta versión.

Mapbox, al igual que Firebase, tiene tanto una versión de pago como una versión gratuita. Esta última está limitada a 50000 cargas de mapa mensuales. No obstante, al igual que Firebase, se considera que esta versión gratuita cumple los requisitos necesarios con respecto al alcance esperado de la aplicación, por lo que se usará esta.

Con todo esto en cuenta, obtenemos el siguiente desglose:

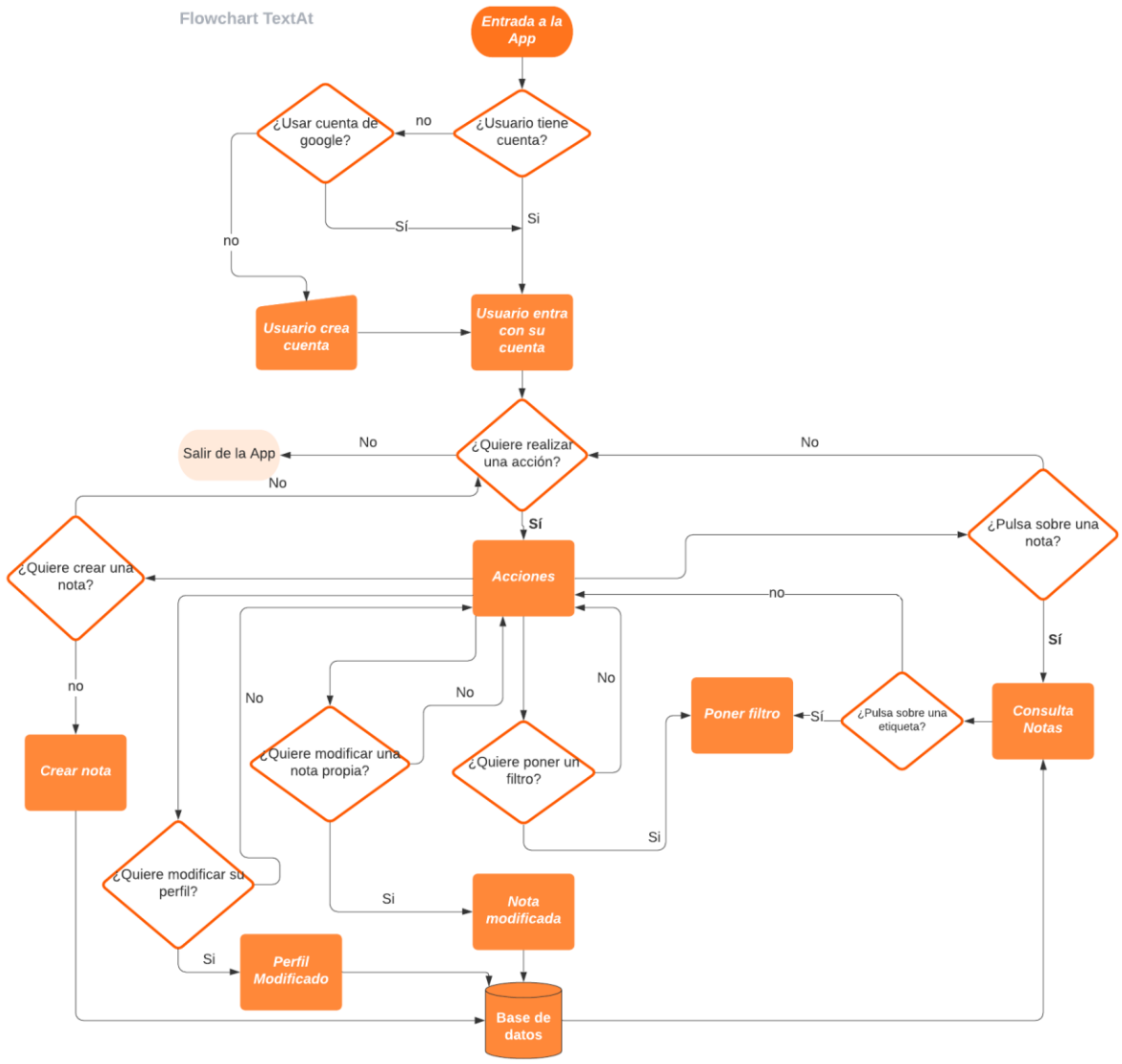
Elemento	Coste
Salario	**2.430,00€**
Amortización del Hardware	**45,50€**
Firebase	**0,00€**
Mapbox	**0,00€**
Total	**2.475,50€**

3. Diseño

La segunda fase del modelo en cascada es el Diseño de la aplicación a partir de lo especificado en la fase de análisis. En esta fase se especifica cómo debería funcionar la aplicación y se crean algunos bocetos de cómo debería verse la aplicación.

3.1. Diagrama de flujo

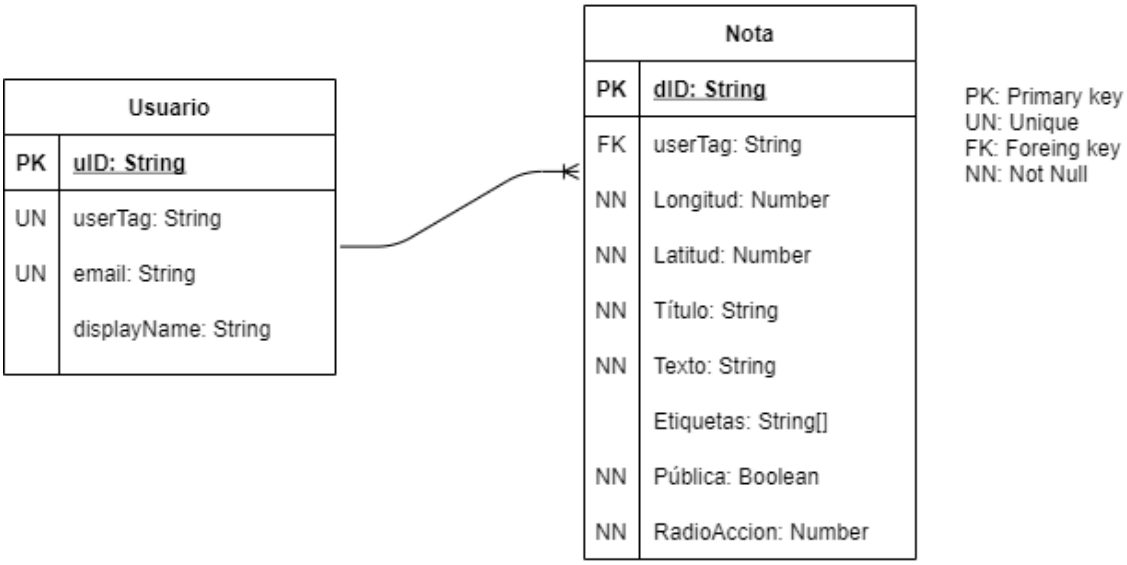
El primer paso sería especificar cuál sería el flujo de acciones de un usuario de la aplicación. Para ello, se ha creado el siguiente diagrama de flujo



3.2. Diseño de la base de datos

La base de datos que se va a usar es Firestore, la cual es una base de datos NOSQL, es decir, no tiene la estructura de una base de datos clásica. Se trata de una BBDD documental, que guarda los datos en documentos que tienen un formato idéntico a JSON. Estos documentos se encuentran agrupados en colecciones. Si bien estos documentos no tienen la estructura de una base de datos SQL, se puede "Forzar" a que lo tenga haciendo que cuando se guarde los datos en ella, estos siempre tengan el mismo formato.

Diagrama de flujo BBDD



Este esquema representa el formato que tendrán los datos en la aplicación, tanto uID como dID son claves que Firebase crea automáticamente para cada documento que se incluye en una colección.

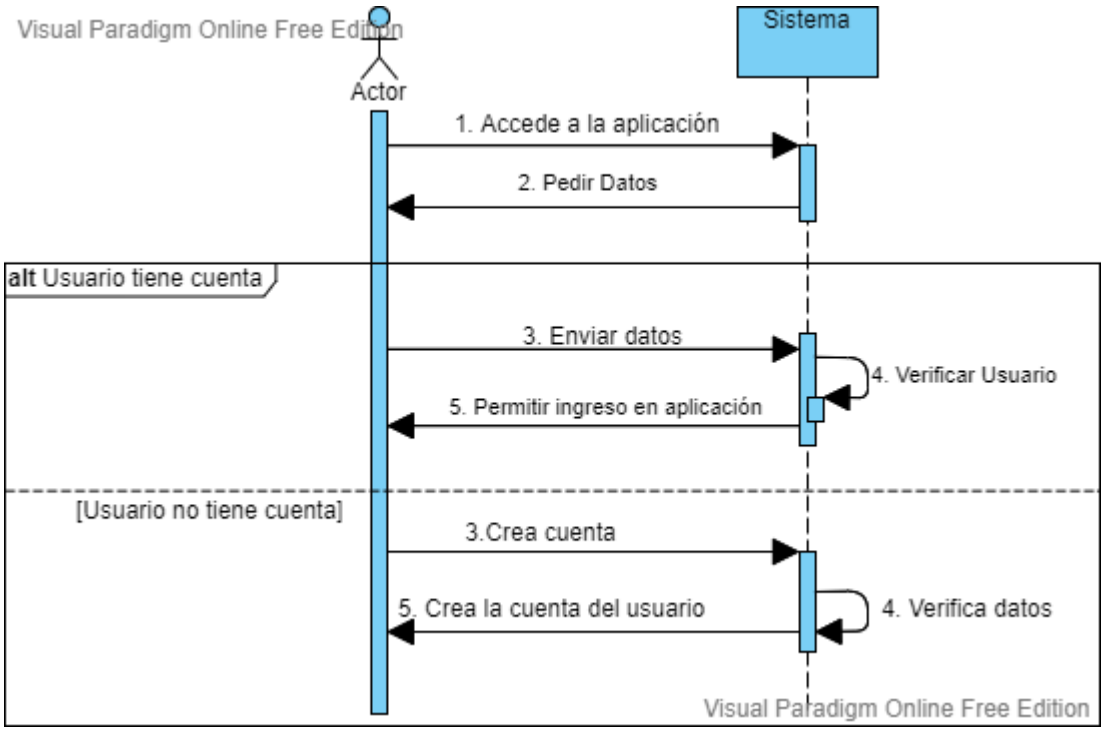
3.3. Diagramas de secuencia

Una vez se sabe cuál será el funcionamiento general de la aplicación, es conveniente especificar en qué orden deben ocurrir los distintos procedimientos durante la ejecución de esta para evitar posibles errores.

3.3.1. Inicio de sesión/Registro

El inicio de sesión es relativamente simple. En primer lugar se le solicitarán al usuario los datos. Si los datos que ingresa son válidos (o usa su cuenta de Google para identificarse), la aplicación validará estos datos y si son válidos, permitirá el ingreso en la aplicación.

En caso de no tener cuenta y no querer usar la de Google, el usuario podrá crear una cuenta. Si los datos para la creación son válidos, la aplicación creará la cuenta y le devolverá al principio para solicitar estos datos.



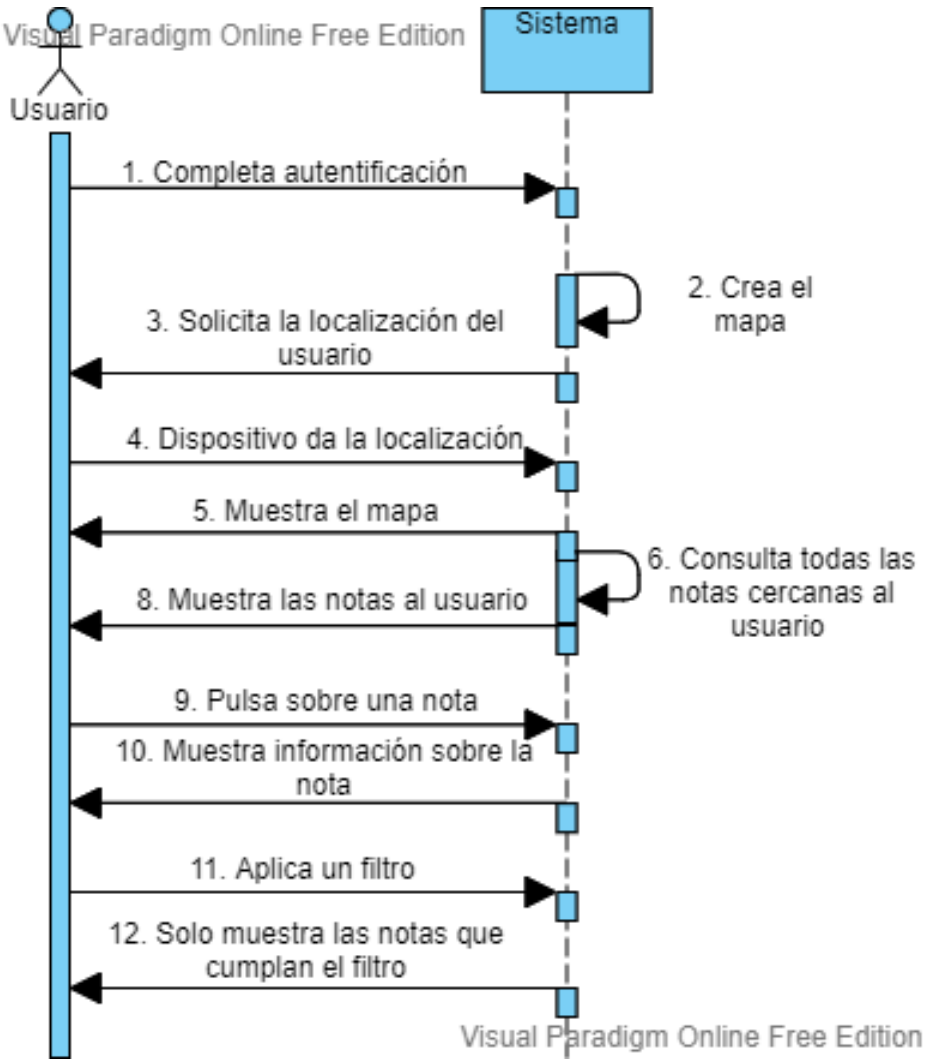
3.3.2. Aplicación

El funcionamiento de la aplicación una vez se ha ingresado es algo más complejo.

En primer lugar se creará el mapa, tras lo cual se solicitará la localización al dispositivo del usuario.

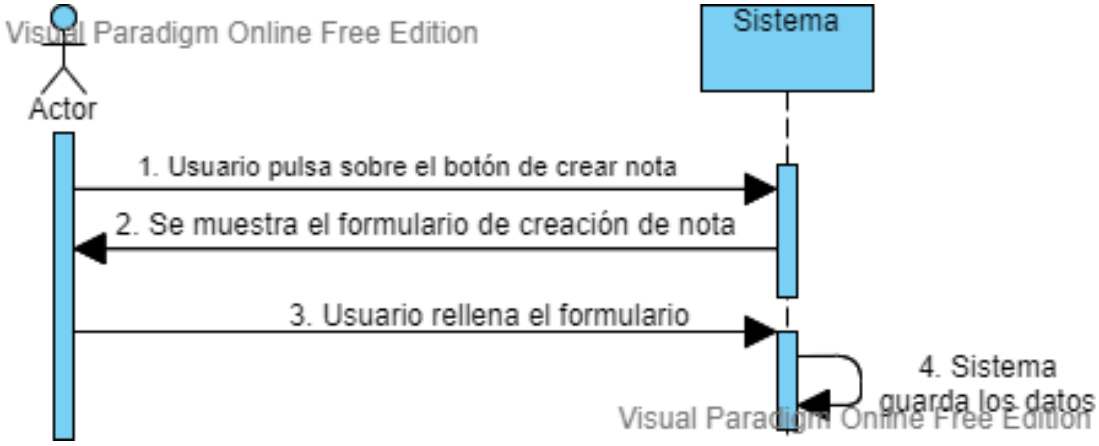
Una vez se sepa la localización, se le mostrará el mapa al usuario, se consultarán todas las notas que estén cercanas y se le mostrarán al usuario aquellas que hayan sido encontradas de tal manera que este esté en el radio de visión de estas notas.

Tras ello el usuario podrá consultar las notas que se le muestren pulsando sobre ellas. Esto hará que la aplicación le muestre la información de la nota al usuario. Si este aplica un filtro, solo se mostrarán las notas que cumplan dicho filtro.



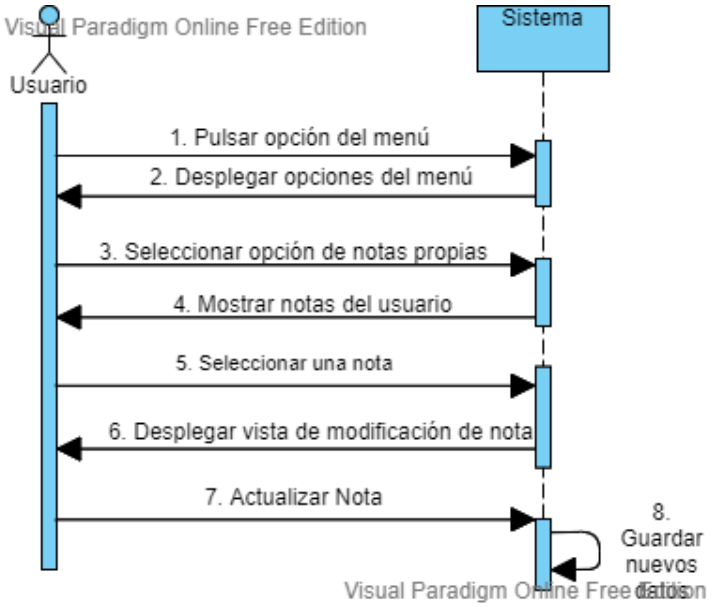
3.3.3. Creación de nota

La creación de las notas es un proceso muy simple. Si el usuario pulsa sobre el botón de creación de nota, se le lleva a la pantalla de creación, donde solo debe de rellenar los datos y enviarlos. Una vez enviados, la aplicación los guardará en la base de datos.



3.3.4. Consulta/Actualización de notas

La consulta y actualización de las notas creadas por el usuario es otro proceso simple. Para consultar sus notas, primero seleccionará dicha opción desde un menú. Tras esto, se le mostrará una lista con todas las notas que haya creado. Pulsar sobre una de ellas mostrará la información de esa nota, así como la capacidad de actualizarla. Si la actualiza, la aplicación guardará los datos actualizados en la base de datos



3.4. Bocetos de la interfaz

En este apartado, se realizarán algunos bocetos que serán usados como base para la creación de la interfaz durante el proceso de desarrollo e implementación del software. Se definen así algunas vistas

Vista 1: Login

Vista de la identificación. En ella encontrará los siguientes elementos tal como están enumerados en la imagen.

1 – Formulario para ingresar mediante correo y contraseña

2 – Opción de crear cuenta, llevará a la vista 2.

3 – Opción de iniciar sesión con Google, llevará a un Popup ajeno a la Aplicación.

Una vez identificado, el usuario será llevado a la vista 3.

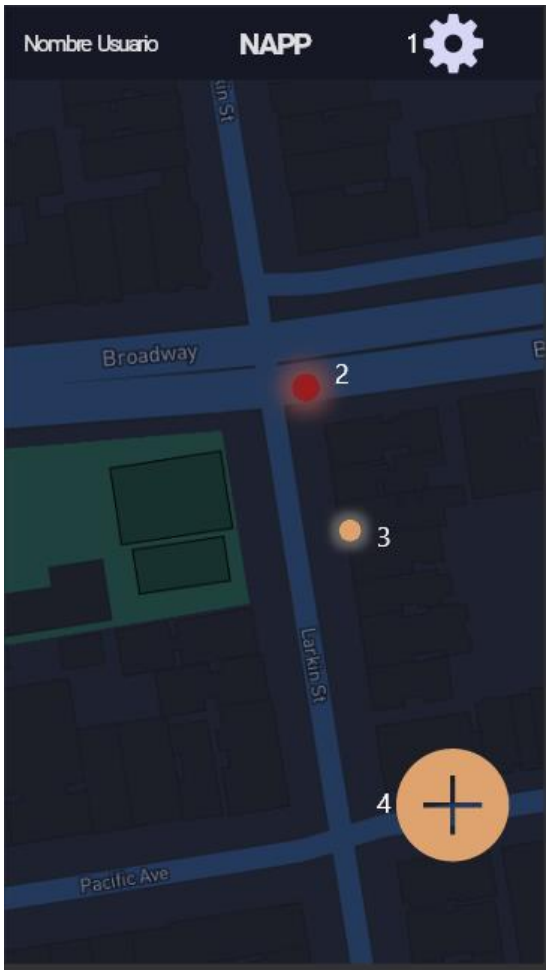
Vista 2: Creación de cuenta

En caso de querer crear una nueva cuenta, al usuario se le presenta esta vista.

En ella encuentra un formulario en el cual deberá indicar que correo y contraseña quiere usar.

Una vez rellenado los datos, el usuario será redirigido a la vista 1, para introducir la cuenta e ingresar en la aplicación.

Vista 3: Pantalla principal



Vista principal de la aplicación, dividida en 2 partes, en la superior un navbar y el resto de la pantalla ocupada por el mapa.

En esta vista encontramos 4 elementos principales:

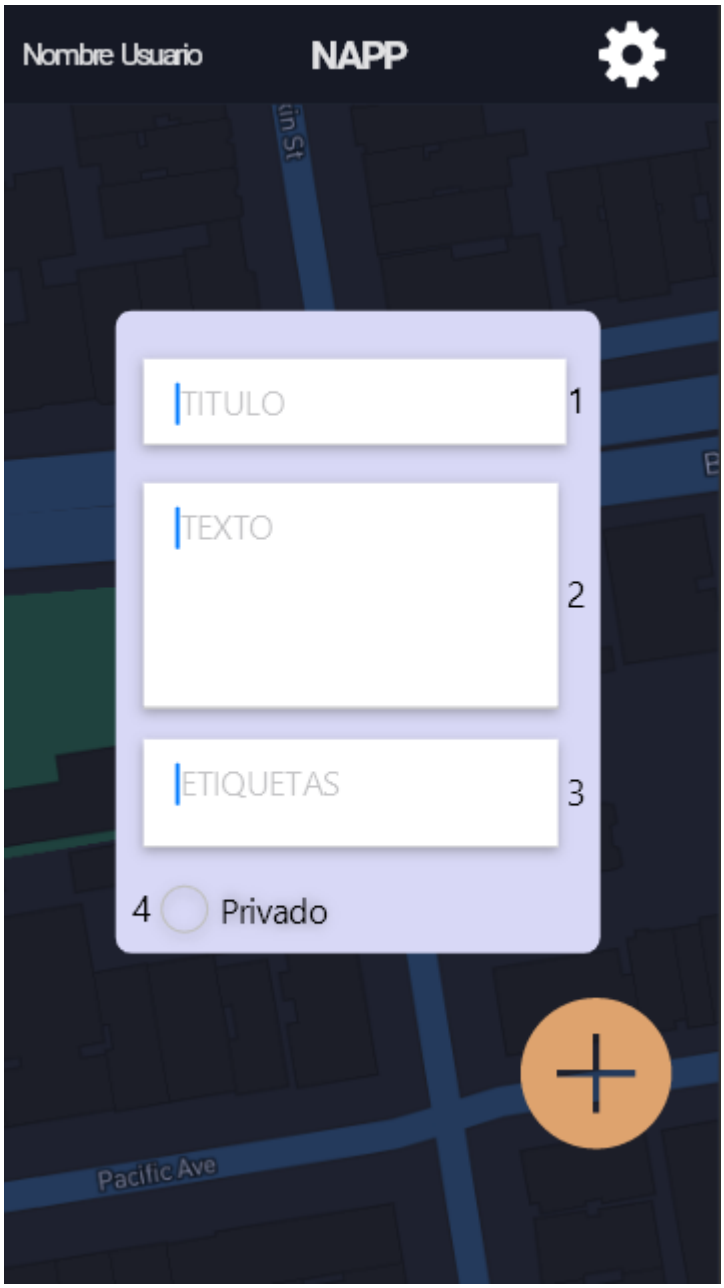
1 – Botón para el menú de opciones, lleva a la vista 6

2 – Marcador de posición, indica la posición geográfica del usuario en el mapa.

3 – Marcador de nota, indica que hay una nota en esa posición, pulsando sobre ella llevará a la vista 5

4 – Botón de creación de nueva nota, llevará a la vista 4.

Vista 4: Creación de nota



En el caso de pulsar sobre el botón de crear nota, aparecerá el siguiente Popup o posible ventana nueva, la cual incluirá un formulario con los siguientes elementos:

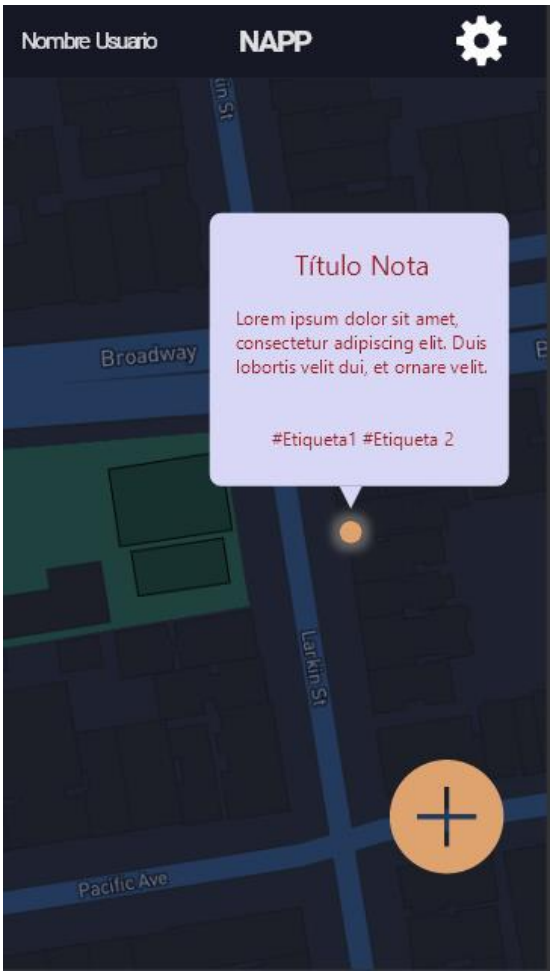
1 – Titulo de la nota.

2 – Texto que contiene la descripción del lugar.

3 – Etiquetas asociadas a la nota

4 – Boton radial/Toggle que indique si el usuario quiere que sea una nota privada o pública.

Vista 5: Vista de Nota



En el caso de que el usuario pulse sobre el marcador de una nota, como el indicado en la vista 3, se abrirá un Popup con la información de dicha nota.

Estas vistas servirán como base para la creación de la interfaz durante el desarrollo de la solución software.

4. Implementación

En este apartado se realizará una descripción más detallada sobre el funcionamiento de las tecnologías escogidas, así como una descripción de la organización y funcionamiento de cada uno de los módulos desarrollados para el proyecto.

4.1. Tecnologías

4.1.1. Framework: Ionic-Angular

El framework base a usar es **Angular**. Es un framework multiplataforma, con código en Typescript, en el cual se usa el **patrón Modelo-Vista-Vista-Modelo**, un patrón distinto al MVC tradicional, en el que la vistas son activas, conteniendo comportamientos, eventos, por lo que requieren conocimiento del modelo subyacente.

Angular se basa en la creación de distintos bloques para el desarrollo del programa. El principal es el **componente**, al cual llamaremos **página** cuando entremos en Ionic. Este componente es la pieza básica puesto que contiene los elementos de la UI así como los métodos de control requeridos para el funcionamiento.

A parte de los componentes, existen otros bloques que serán usados en este trabajo, como las **Interfaces** de Typescript, que sirven para crear objetos de datos fijos, o los **Servicios**, los cuales son clases con un objetivo muy bien definido y que deben de hacer algo específico y hacerlo bien.

Sobre Angular, se usará **Ionic.** Como se explicó en el punto 2.C.iii, Ionic es una Toolbox para la creación de interfaces de usuario en plataformas móviles creado para funcionar sobre diversos frameworks base, como Vue, React o, como en este caso, Angular. La base de Ionic son los **componentes**, bloques que permiten la creación de interfaces atractivas de una manera muy sencilla y veloz. Para evitar la confusión con los componentes de Angular, tal como se explicó antes, Ionic denomina a estos últimos **Páginas**, aunque su comportamiento es similar.

Junto a esto, Ionic tiene la capacidad de desarrollo multiplataforma, pudiendo subir el programa a un servicio de hosting para cargarlo como una web, o incluso pudiendo compilar el programa como una APK de Android o iOS.

4.1.2.Mapbox

La segunda tecnología a usar es Mapbox GL JS. Se trata de una biblioteca de JavaScript, creada por la compañía Mapbox, que utiliza WebGL para renderizar mapas interactivos. Esta biblioteca es solo una de las que ofrece Mapbox, puesto que también tiene SDKs nativos para aplicaciones de Android, iOS, o macOS, así como una serie de plugins para ampliar funcionalidades de los propios mapas y distintas APIs de servicios.

La biblioteca que se usa en esta aplicación nos permite la creación del mapa, así como dotarlo de **controles interactivos** para que el usuario pueda moverse por él. Además, también nos otorga la capacidad de colocar **marcadores** en los que poner notas, pieza clave de esta aplicación.

4.1.3.Firebase

Es el servicio BaaS que se usará para la parte del Backend de nuestra aplicación. Se usará **Angularfire**, la biblioteca oficial de Firebase para proyectos Angular.

Esta biblioteca nos permite acceder a los servicios que requiramos asociados al proyecto de Firebase de esta Aplicación. A través de ella se crearan los servicios de identificación/registro del usuario y el de relación con la base de datos.

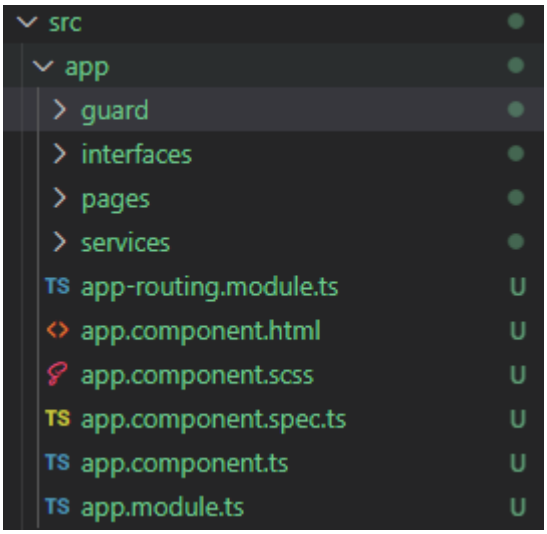
4.2. Organización del proyecto

Tanto los proyectos de Angular como Ionic se crean desde la interfaz de línea comandos (CLI, por sus siglas en inglés) mediante un comando específico. Esto genera ya un proyecto con todos los archivos base preparados para trabajar sobre ellos.

El trabajo se realiza dentro de la carpeta ‘src/app del proyecto, en la cual están incluidos todos los elementos relacionados con la web a crear. Aunque también es necesario añadir alguna información puntual en archivos externos a esta.

La generación de nuevos objetos se realiza también desde CLI, mediante comandos que generan todos los archivos de cada componente y modifican todos los elementos necesarios del proyecto para la integración de estos nuevos objetos.

El proyecto se ha organizado por tipo de elemento, así, si el elemento a crear es una página, se guarda en la carpeta ‘pages’, si es un servicio, se encuentra en la carpeta ‘services’, y así para cada tipo de elemento.



Carpeta proyecto.

Cada elemento se encuentra dentro de su carpeta correspondiente en 'src/Aplicación'

Carpeta a carpeta encontramos los siguientes módulos:

Guard: "auth.guard"

Interfaces: "Notas" y "Usuario"

Pages: "registrar", "login", "main", "nota", "nota-usuario", "create-marker" y "perfil"

Services: "auth.service", "firebase-service", geolocation.service".

De esta manera, mantenemos ordenados todos los elementos según su tipo, permitiendo trabajar de una manera más sencilla y veloz.

4.3. Módulos

En este apartado se realizará una descripción individual de cada módulo, debido a que algunos módulos requieren de otros para funcionar correctamente, comenzaré por las interfaces, seguido de los servicios y las páginas.

Siguiendo las ideas del patrón Modelo-vista-vista-Modelo, y debido a las funcionalidades que se describirán en los siguientes apartados, podemos considerar los servicios como los Modelos y las páginas como las vistas-modelos en este patrón.

i) Interfaces

4.3.1.1. Notas

Contienen la información sobre cada una de las notas en un solo objeto, guardando los siguientes parámetros:

- dID: Id única de la nota, idéntica a la ID que se le da en Firebase,
- título: Titulo de la nota.
- Texto: Texto de la nota
- Etiquetas: Etiquetas o Hastags de la nota, se extraen del texto.
- Longitud y Latitud: Posiciones geográficas donde está la nota
- Publica: Un booleano que indica si la nota es pública o no
- uID: ID del usuario que crea la nota, idéntica a la ID del usuario dada en Firebase.

4.3.1.2. Usuario

Guarda la información del usuario que está autentificado en la aplicación, tiene los siguientes parametros:

- uID: Id única del usuario, idéntica a la dada por Firebase
- email: Correo electrónico del usuario.
- displayName: Nombre o mote del usuario, si se ha autentificado por Google, se da automáticamente. Puede no tenerlo.

ii) Servicios

(1) Auth

Es el servicio encargado exclusivamente de todo lo relacionado con la identificación del usuario y el manejo de sus datos.

Incluye los métodos para la creación de una cuenta, Identificación, tanto por correo/contraseña como mediante cuenta de Google, cerrar la sesión en la aplicación y actualización de los datos del usuario. También es este servicio el que mantiene los datos del usuario activo en la aplicación, de tal manera que solo se pueden acceder a ellos a través de este.

Este servicio requiere de la biblioteca AngularFire para funcionar, necesitando específicamente los componentes AngularFireAuth, así como AngularFirestore y AngularFirestoreDocument de esta biblioteca. Además también requiere de la interfaz “Usuario”, descrita en el apartado anterior.

(2) Firebase

Es el servicio encargado de todo lo relacionado con Firestore, la base de datos de Firebase.

Incluye los métodos necesarios para las operaciones de creación, lectura, actualización y borrado de datos en Firestore, permitiendo a la aplicación crear notas en la base de datos, actualizarlas, borrarlas y realizar las consultas que devuelvan las notas.

Los métodos de actualización y creación de nota se encargan de estructurar la información que se guardará en la base de datos analizando los datos otorgados por el usuario. Estos métodos son los encargados de crear las etiquetas a partir del texto dado por el usuario.

Existen dos métodos de consulta, un primero que permite consultar las notas cercanas a la posición del usuario y otro para devolver todas las notas que el usuario haya creado.

Este servicio requiere la biblioteca AngularFire para funcionar, necesitando el componente AngularFirestore, así como la interfaz Notas descrita en el apartado 4.B.i.1 y el servicio Auth descrito en el apartado anterior.

(3) Geolocation

Este servicio se encarga de todo lo relacionado con la geolocalización y los cálculos de posiciones.

Contiene los métodos necesarios para la detección de la posición actual del dispositivo, así como los métodos para calcular la distancia entre dos puntos, para calcular la latitud y longitud a X distancia del dispositivo en cada dirección cardinal y un último método usado para obtener la circurferencia de la distancia de visión de cada nota.

Este servicio no requiere de ninguna biblioteca externa para funcionar.

(4) Notificaciones

Este servicio se encarga de la creación de ventanas emergentes usadas para darle notificaciones al usuario.

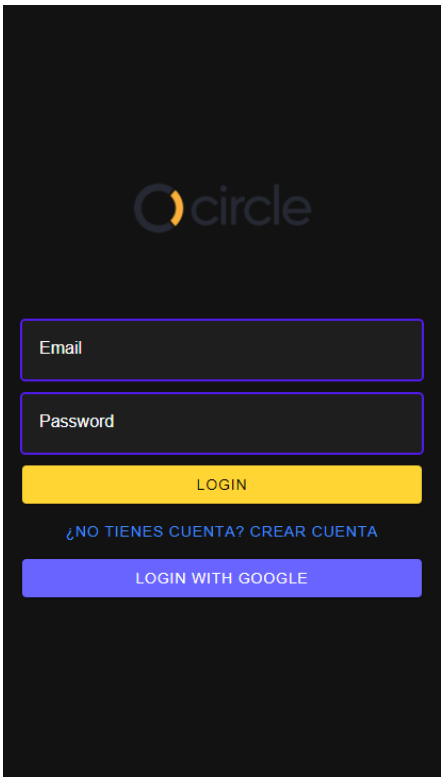
No requiere de ninguna biblioteca externa para funcionar.

iii) Páginas

(1) Login

Esta página contiene el modelo-vista relacionado con la identificación del usuario.

La vista incluye dos campos de texto, para introducir el correo y la contraseña, y un botón que autentifica al usuario si los datos introducidos son correctos. También incluye otro botón para permitir al usuario ingresar con su cuenta de Google si así lo quisiese. Por último, la vista incluye un botón que sirve a modo de enlace a la vista de registro.



Vista de login

El modelo contiene las funciones necesarias que invocan funciones del servicio Auth así como los elementos de routing para llevar al usuario a las diversas páginas a partir de estas.

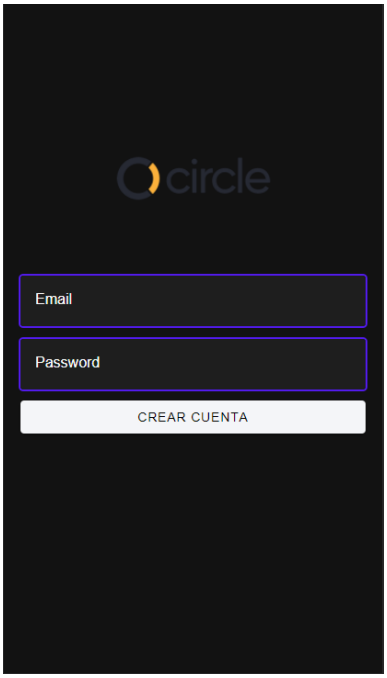
Además, comprueba si los datos introducidos son válidos, en caso de no serlos, manda una notificación mediante el servicio Notificaciones.

Esta página requiere los servicios Auth y Notificaciones para funcionar.

(2) Registrar

Esta página contiene el modelo-vista relacionado con la creación de una cuenta mediante correo/contraseña del usuario

La vista incluye dos campos de texto, donde el usuario introduciría su correo y contraseña, así como un botón para validarla



Vista Registrar

El modelo contiene las funciones necesarias para completar el registro, llamando a funciones necesarias del servicio Auth.

Al igual que Login, este modelo comprueba si los datos introducidos son válidos, en caso de no serlos, manda una notificación mediante el servicio Notificaciones.

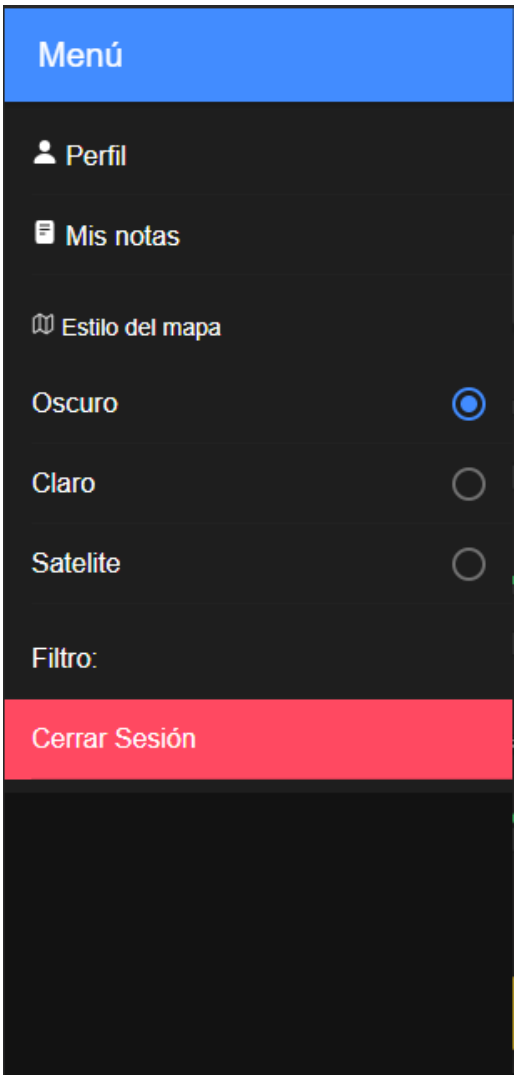
Esta página requiere de los servicios Auth y Notificaciones para funcionar.

(3) Main

Esta página contiene el modelo-vista principal de la aplicación. Todas las acciones que puede hacer el usuario, excepto las relacionadas con las dos páginas anteriores, se realizan a partir de esta.

La vista incluye tres elementos, una barra superior, un menú que por defecto no se muestra y el mapa.

La barra superior tiene el nombre de la aplicación y un botón para desplegar el menú en la parte izquierda.



El menú incluye dos botones que sirven de enlaces a las páginas de modificación de perfil y de las notas del usuario.

También incluye un selector del estilo del mapa, el cual permite al usuario elegir entre un estilo oscuro, que es el que viene por defecto, uno claro y una vista de satélite.

Incluye un campo de texto que es a través del cual se realiza la filtración de las notas que se muestran en el mapa.

Por ultimo tiene un botón que sirve para cerrar la sesión actual.

En el mapa podemos encontrar 5 elementos:

En la parte superior izquierda y parte inferior izquierda se encuentran el logo de Mapbox y el botón de atribución, los cuales deben de ser visibles en todo momento por los términos de uso de Mapbox.

En la parte superior derecha encontramos los botones de control.

Los dos botones superiores permiten dar o quitar zoom al mapa

El tercer botón reorienta el mapa, colocando el norte en la parte superior.

El último botón centra el mapa sobre la posición del usuario.

En la parte inferior derecha encontramos el botón de creación de nota. Pulsar sobre el lleva a la página de creación de nota.

Las notas están representadas por marcadores de tres colores. Cada color muestra la privacidad y propiedad de la nota. Un marcador rojo es una nota del usuario privada, un marcador morado es una nota del usuario pública y un marcador azul es una nota pública que no le pertenece al usuario.

La posición del usuario está representada por un círculo de color azul claro que parpadea.

Puesto que todos los elementos relacionados con el mapa solo se encuentran en esta página, se ha optado por incluir todos los métodos directamente en este, en lugar de crear un servicio a parte para su gestión.

Estos métodos incluyen el que crea el mapa, la consulta a la base de datos para obtener todas las notas en un rango de distancia según la posición actual del dispositivo y la creación de los marcadores, dotando a estos últimos de la lógica necesaria para que carguen las vistas de información de cada nota.

A parte de los métodos relacionados con el mapa, también incluye los métodos de redirección al resto de páginas de la aplicación y de mantenimiento de la posición del usuario.

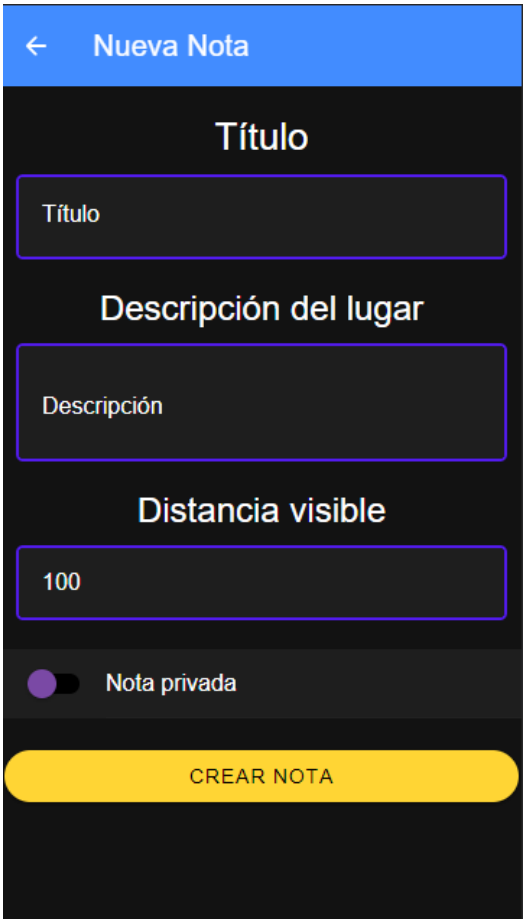
Cada vez que se carga esta vista, el modelo comprueba si la geolocalización del dispositivo está activa. Si no lo estuviese, manda un mensaje mediante el servicio de notificaciones para indicarle al usuario que debe de activarlo.

Esta página requiere de los servicios de geolocalización, Firebase, Auth y Notificaciones para funcionar.

(4) Crear Nota

Esta página incluye el modelo-vista correspondiente a la creación de una nueva nota.

La vista incluye un formulario con los datos que el usuario debe rellenar para crear la nota.



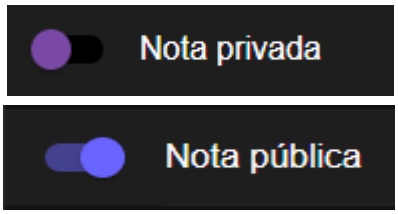
El formulario incluye los siguientes campos:

Título: Titulo que se le da a la nota.

Descripción del lugar: Descripción que el usuario da del lugar donde crea la nota.

Distancia visible: Distancia máxima a la que será visible la nota.

Privacidad de la nota: Botón que indica si la nota se creará como nota privada o pública. Pulsar sobre el cambiará el texto según la opción escogida.

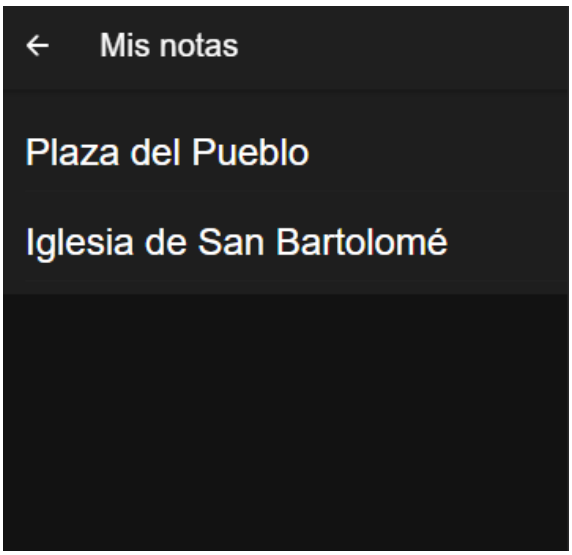


El modelo contiene un método que valida los campos están rellenados y, en el caso del campo de la distancia, los valores están en el rango correcto. Si la validación es incorrecta, lanza una notificación al usuario mediante el servicio Notificaciones indicando del problema. Si la validación es correcta, crea la nota y la envía al servicio Firebase para su guardado en la base de datos.

Esta página requiere de los servicios de geolocalización, Notificaciones y Firebase para funcionar.

(5) Notas del Usuario

Esta página contiene el modelo-vista en el cual se muestra al usuario todas las notas que ha creado.



En la vista se muestra la lista de las notas que ha creado, mostrando el título de la nota.

Pulsar sobre una nota llevará a la página de modificación de nota correspondiente a esta.

Si se mantiene sobre una nota y se desplaza hacia la derecha, se muestra el botón de borra nota.

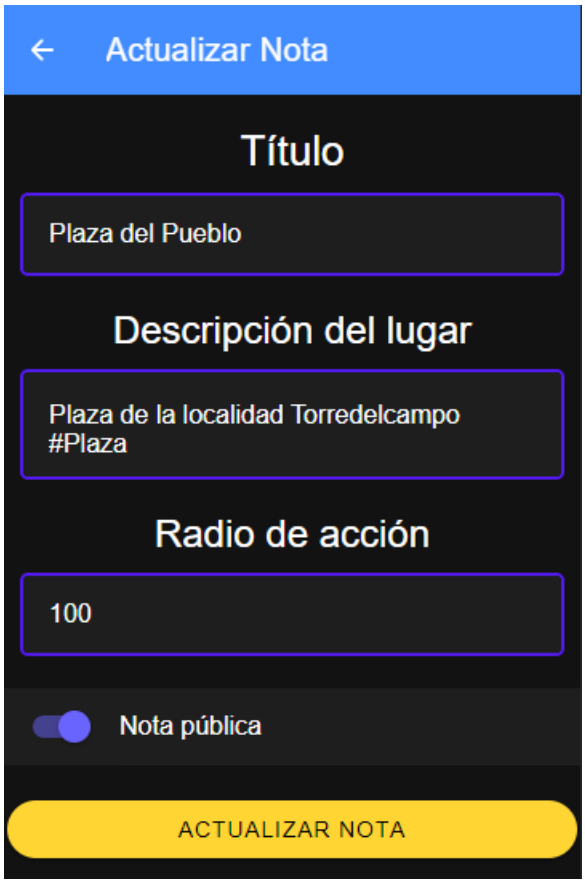
El modelo incluye un método que llama al servicio de Firebase para obtener todas las notas creadas por el usuario. También incluye otro método que prepara la nota seleccionada para pasársela a la página de modificación de nota.

Esta página requiere del servicio Firebase para funcionar.

(6) Actualizar Nota

Esta página contiene todo lo relacionado con mostrar y actualizar la información de una nota específica del usuario.

La vista es similar a la de la página “Crear Nota” descrita en el apartado 4 de esta misma sección, con la diferencia del texto del botón, que pasa a ser “Actualizar Nota”



El modelo sí varía con respecto al de “Crear Nota”, en este caso, el método que incluye no crea una nueva nota a partir de los campos de la vista, sino que asigna los datos de la nota ya existente a estos, valida que, si se han cambiado los datos, estos sean correctos, y le pasa la nota con los campos cambiados al método correspondiente del servicio Firestore. En caso de que un campo sea invalido, emite una notificación al usuario indicando el error mediante el servicio Notificaciones.

Esta página requiere los servicios Firestore y Notificaciones para funcionar.

(7) Nota

Esta página contiene el modelo-vista correspondiente a mostrar la información de una nota asignada a un marcador en el mapa. A diferencia de las otras páginas, esta solo se muestra a través de un componente “popover” de Ionic cuando se ha pulsado sobre un marcador.

La vista contiene la información de la nota que se ha seleccionado, mostrando su título y descripción.

En caso de que la nota pertenezca al usuario, en la parte inferior se mostrará un botón que le llevará a la página de actualización de dicha nota. Si la nota no es del usuario, en la parte inferior aparecerá el Tag del autor de la nota.

Si el usuario pulsa sobre una etiqueta (cualquier palabra que lleve una almohadilla o # al principio), filtrará las notas según esa etiqueta. Si la nota no es propia y pulsa sobre el autor, filtrará las notas según el autor de dicha etiqueta.

El modelo incluye los métodos que comprueba si el elemento del texto seleccionado por el usuario es una etiqueta o es un autor para enviarle esta información a la página “Main” y así colocar dicho filtro.

Esta página requiere del servicio Auth para funcionar.

(8) Perfil

Esta página contiene el modelo-vista relacionado con la información del usuario, así como su actualización si este quisiese cambiar algún dato.

La vista incluye un formulario en el cual se puede cambiar el nombre y el tag único del usuario.

También incluye un botón que permite al usuario actualizar los datos que haya modificado si así lo desea.

El modelo incluye un método que permite actualizar los datos indicados por el usuario. En caso de que un campo esté vacío, o el dato indicado en el campo Tag sea inválido, se lanzará una notificación por medio del servicio Notificaciones indicándole al usuario el problema.

Esta página requiere de los servicios de Notificación, Auth y Firebase para funcionar.

iv) Guardias

(1) Guard

En Angular, la navegación entre páginas se realiza mediante elementos llamados Rutas. Una guardia es un componente que se puede asignar a cada ruta, de tal manera que, cuando se solicita a la aplicación navegar a una página, se puede llamar a un método de la guardia.

En el caso de esta guardia, comprueba si el usuario se encuentra identificado en la aplicación. Si no se encuentra identificado, se redirige a la página de “Login”.

Todas las rutas de la aplicación, menos las correspondientes a “Login” y “Registrar” llevan esta guarda aplicada para impedir que se pueda acceder a las funciones si el usuario no se ha identificado.

5. Validación y Pruebas

La última etapa del modelo en cascada es la validación del software. Esta consiste en comprobar que todos los requisitos descritos en la fase de análisis se cumplen.

Se procederá a comprobar, requisito a requisito, si estos han sido implementados correctamente según lo descrito.

5.1. Validación

REQ-1: Identificación

Este primer requisito requería que el usuario fuese capaz de identificarse en la aplicación eligiendo identificarse mediante correo/contraseña o bien mediante cuenta de Google. En caso de no tener cuenta, se le da la opción de poder crear una.

Dicho requisito se cumple gracias a los módulos descritos en los apartados 4.C.ii.1, referente al servicio Auth, que otorga a la aplicación la capacidad de identificarse con los métodos indicados en el requisito, así como poder crear una cuenta mediante correo/contraseña, y en los apartados 4.C.iii.1 y 4.C.iii.2 referentes a las páginas que permiten al usuario acceder a dichas opciones.

REQ-2: Localización

Este requisito requiere que la aplicación pueda tener acceso a la localización del dispositivo en el cual se está ejecutando y mantenerla en todo momento.

Dicho requisito se cumple gracias al módulo descrito en el apartado 4.C.ii.3, referente al servicio de geolocalización. Dicho servicio se encarga de obtener acceso a los servicios de geolocalización del dispositivo en el que se está ejecutando la aplicación y mantiene actualizada la posición del usuario en todo momento.

REQ-3: Creación de notas

Este requisito requiere que el usuario sea capaz de crear una nota sobre la posición en la que se encuentra en el momento. Además, debe ser capaz de añadirle etiquetas a la nota, así como editar su privacidad y distancia máxima de visionado. Estas notas deben aparecer en el mapa.

La creación de la nota se valida con los módulos descritos en los apartados 4.C.iii.4, 4.C.ii.2 y 4.C.ii.3. El primero es la página en la cual el usuario puede detallar la información que quiere incluir en la nota a crear, indicando título, descripción, etiquetas, rango de visión y privacidad. El segundo apartado es referente al servicio que tiene las funcionalidades que permiten guardar la nota en la base de datos con el formato correcto. El tercero permite conseguir la posición actual del usuario para poder asignar la nota a esta posición.

La colocación de las notas en el mapa también se valida gracias al apartado 4.C.ii.2, puesto que el servicio incluye también las operaciones necesarias para obtener las notas desde la base de datos, así como en el apartado 4.C.iii.3, el cual describe la página en la que se encuentra el mapa y las funciones que permiten colocar los marcadores de las notas en este.

REQ-4: Consulta de notas

Este requisito requiere que el usuario tenga la capacidad de consultar la información de sus notas privadas o cualquier nota pública si se encuentra en su radio de acción.

Dicho requisito se cumple gracias a los módulos descritos en los apartados 4.C.iii.3 y 4.C.iii.7 y 4.C.ii.3.

El primer módulo describe la página donde se encuentra el mapa y los marcadores, que en conjunto con el tercer módulo, el cual describe el servicio de geolocalización, permite mostrar en el mapa solo aquellas notas cercanas al usuario. El segundo módulo describe la página en la cual se muestra la información de dicha nota.

REQ-5: Filtrado de notas

Este requisito requería que el usuario fuese capaz de filtrar las notas que aparecían en el mapa añadiendo un filtro. Dicho filtro podrá ser por etiqueta o por usuario y dependiendo del tipo se mostrarán las notas acordes a este.

Este requisito se cumple gracias a los módulos descritos en los apartados 4.C.iii.3 y 4.C.iii.7. En la página descrita en el primer apartado es en la que se muestra el mapa y los marcadores. Dicha página contiene los métodos que colocan los marcadores según los filtros dados. También otorga al usuario la capacidad de escribir manualmente un filtro.

La página descrita en el segundo apartado contiene la ventana de información de una nota. Dicha página tiene unos métodos que permiten al usuario pulsar sobre

una etiqueta o sobre un autor para mandarle dichos elementos a la página del mapa como filtros.

REQ-6: Modificación de notas:

Este requisito requiere que el usuario sea capaz de acceder, actualizar o borrar cualquiera de sus notas en todo momento.

Este requisito se cumple gracias a las páginas descritas en los apartados 4.C.iii.5, 4.C.iii.6 y 4.C.iii.7, así como el servicio descrito en el apartado 4.C.ii.2.

La primera página es la encargada de mostrarle en cualquier momento al usuario todas las notas que el haya creado, así como dotarle de la capacidad de borrarlas. La segunda página permite al usuario actualizar los datos de una de sus notas. La tercera página contiene la capacidad de redireccionar al usuario a la ventana de actualización de la nota descrita en esta.

El servicio indicado contiene las funciones para que, dada una nota actualizada, sea capaz de modificar los datos en la base de datos. También incluye el método que borra una nota de la base de datos si así lo quiere el usuario.

Dado todo lo anterior, la implementación de la solución software valida los requisitos funcionales detallados durante la fase de análisis.

5.2. Pruebas

Una vez comprobado que los requisitos han sido validados, se ha realizado una pequeña batería de pruebas con varios usuarios ajenos al desarrollo de la aplicación.

Dichas pruebas han sido diseñadas con el objetivo de comprobar el correcto funcionamiento de la aplicación en general así como para comprobar cómo se desenvuelve un usuario que nunca la ha usado. Las pruebas son las siguientes

5.2.1.Creación de cuenta

La primera prueba consiste en la creación de una nueva cuenta en la aplicación, de tal manera que los usuarios se familiaricen con las posibilidades de identificarse con los dos métodos que existen así como la creación de una cuenta.

Con esta prueba se busca ver si los usuarios encuentran alguna dificultad durante el proceso de identificación y, además, comprobar si existe algún error no encontrado durante el proceso de desarrollo.

5.2.2.Consulta de notas y control del mapa.

La segunda prueba consiste en hacer usar los distintos controles del mapa así como consultar varias notas. Esta prueba se realiza en un sitio escogido con antelación, donde previamente al inicio de la prueba se han creado diversas notas en la zona. Tras las consultas de las notas, se desplaza a los usuarios a una zona fuera del rango de visión, para que aprendan cómo funciona el sistema de radio de visión de las notas.

Con esta prueba se busca ver si los usuarios encuentran alguna dificultad con los controles del mapa o con la consulta de notas y comprobar si existe algún error no encontrado relacionado con estos aspectos durante el proceso de desarrollo.

5.2.3.Creación de notas

La tercera prueba consiste en hacer que el usuario cree una o más notas. A diferencia de la primera prueba, esta no tiene por qué ser en un sitio escogido con antelación, sino que puede ser en una localización escogida por el usuario.

Con esta prueba se busca ver si los usuarios encuentran dificultades durante el proceso de creación de notas y comprobar si existe algún error que no haya sido encontrado durante el proceso de desarrollo.

5.2.4. Consulta y actualización de notas propias

La cuarta prueba consiste en hacer que el usuario consulte sus propias notas y hacerle modificar alguna de las que haya creado en la prueba anterior.

Con esta prueba se busca ver si el usuario encuentra algún tipo de dificultad en el proceso de consulta de sus propias o durante el proceso de modificación de alguna de estas. También se busca comprobar si existe algún error relacionado con este concepto que no haya sido encontrado durante el proceso de desarrollo.

5.2.5.Uso del filtrado

Esta última prueba consiste en hacer al usuario usar el sistema de filtrado. Al igual que en la segunda prueba, esta se hace en una zona escogida de antemano, donde se han creado notas con etiquetas en la cercanía.

Los usuarios deben usar una serie de filtros que se corresponden a las notas creadas previamente, así como usar alguna etiqueta o usuario que no exista en el sistema de filtrado para que comprueben que de esa manera no aparece ninguna nota.

Con esta prueba se busca ver si los usuarios encuentran dificultades con el uso del sistema de filtrado, así como comprobar si existe algún error relacionado con este que no haya sido descubierto durante el proceso de desarrollo.

6. Conclusiones

Como se ha podido comprobar a lo largo de esta memoria, se han cumplido los objetivos planteados al principio de esta, el desarrollo de una aplicación móvil que permita a los usuarios guardar notas sobre una posición geográfica y consultarlas en un mapa cuando se encuentran cercas de ellas.

El concepto de la aplicación es uno que me ha parecido muy interesante, puesto que no existe otra aplicación que junte las características sociales de Twitter con conceptos de geolocalización, además de que se le pueden dar otros usos fuera de su aspecto social.

Este ha sido un proyecto mucho más largo y complejo que el de cualquier práctica que haya realizado a lo largo de la carrera, lo cual me ha hecho aprender mucho sobre muchos conceptos importantes en el desarrollo de software, como por ejemplo planificación a medio/largo plazo o diversos conceptos de análisis que no conocía en detalle previo a la realización de este. Incluso la propia redacción de esta memoria ha supuesto un gran reto en sí, puesto que hasta la fecha, nunca había realizado un documento de estas características.

El proyecto ha supuesto un gran esfuerzo, puesto que antes de comenzar, no estaba familiarizado con ninguna de las tecnologías usadas, lo cual supuso muchos bloqueos en diversos momentos del desarrollo, ya fuera porque no sabía cómo conectar alguna de las tecnologías entre sí o porque compliqué mucho más de la cuenta la creación de algunos módulos, lo cual me llevo a dar varios pasos hacia atrás y repetir trabajo en más de una ocasión.

No obstante, este esfuerzo ha hecho que me familiarice con estas tecnologías y abra la mente a probar otras parecidas, sobre todo React o Vue, otros frameworks destinados a la creación de aplicaciones muy parecidos a Angular. Además, creo que esta experiencia me vendrá bien en el futuro, puesto que estas tecnologías son punteras en el mercado, siendo muy codiciados por las empresas los programadores con conocimientos en esta.

El trabajo me ha llevado a recordar muchos conceptos que he aprendido durante el grado y conectarlos entre sí, enseñándome la importancia de algunos a los que en su momento les daba bastante poca, lo cual ha resultado muy gratificante, puesto que ha habido momentos en los que pensé que dichos conocimientos acabarían sin servir para nada.

Si volviera a empezar hoy a desarrollar esta aplicación otra vez, con los conocimientos adquiridos durante este periodo, seguramente enfocaría muchos elementos de esta de una manera muy distinta y creo que el resultado sería mucho mejor. No obstante estoy bastante contento con el desarrollo realizado. Creo que es una base muy buena sobre la que se podría seguir trabajando en un futuro. Mejorando algunos de los módulos y funcionalidades ya implementadas y agregando más podrían dar como resultado un producto muy interesante y que podría llegar a tener éxito.

Como conclusión final, el haber adquirido toda esta experiencia y conocimientos, así como el ver como la aplicación ha crecido de ser una simple página en blanco al principio a convertirse en un producto funcional ha sido muy gratificante y me ha hecho recordar por qué elegí estudiar este grado.

Referencias

Alonso Vega, A. (12 de 04 de 2017). *Backend as a Service o como prescindir de un backend developer*. Recuperado el 16 de 06 de 2021, de adrianalonso.es: https://adrianalonso.es/desarrollo-web/backend-service-o-como-prescindir-de-un-backend-developer/

Angular. (25 de 06 de 2021). *Caracteristicas: Angular*. Obtenido de Angular Latinoamerica: https://docs.angular.lat/features

Indeed. (04 de 06 de 2021). *Salario de Analista/Programador*. Recuperado el 04 de 26 de 2021, de Indeed Web Site: https://es.indeed.com/career/analista-programador/salaries

Ionic. (01 de 06 de 2021). *About: Ionic*. Obtenido de Ionic Website: https://ionic.io/about

IONOS España S.L.U. (11 de Marzo de 19). *El modelo en cascada: desarrollo secuencial de software*. Recuperado el 15 de 6 de 2021, de Ionos diguital guide web site: https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/el-modelo-en-cascada/

Mapbox. (25 de 06 de 2021). *About: Mapbox*. Obtenido de Mapbox web site: https://www.mapbox.com/about/company/

OpenStreetMap. (25 de 26 de 2021). *About: OpenStreetMap*. Obtenido de OpenStreetMap: https://www.openstreetmap.org/about

StatCounter. (1 de 06 de 2021). *Mobile Operating System Market Share Worldwide*. Recuperado el 16 de 06 de 2021, de Statcounter Web Site: https://gs.statcounter.com/os-market-share/mobile/worldwide

Vidal, M. (5 de 11 de 2019). *¿Qué son las Progressive Web Aps?* Recuperado el 16 de 06 de 2021, de iebSchool: https://www.iebschool.com/blog/progressive-web-apps-analitica-usabilidad/

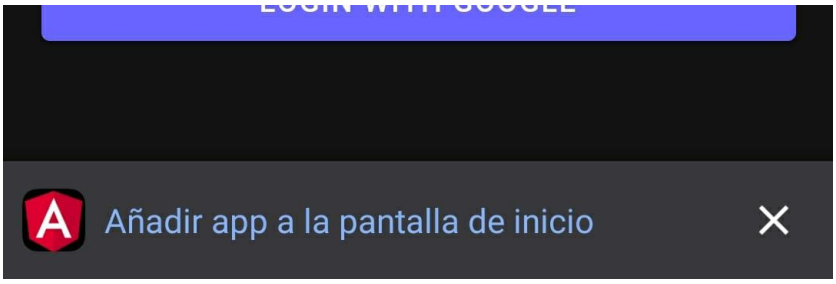
Anexos

Anexo I: Plan de despliegue:

Dispositivos Android

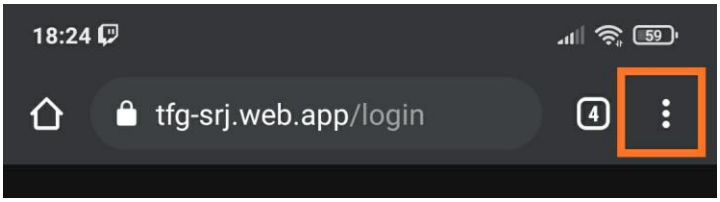
Para obtener la aplicación en un dispositivo Android, hay que introducir la siguiente URL en cualquier navegador del dispositivo: tfg-srj.web.Aplicación

Una vez se haya accedido a dicha dirección, aparecerá un popup preguntando si se quiere añadir la aplicación al escritorio. Pulsando sobre este, se instalará la aplicación y ya se podrá usar como una Aplicación nativa.



Popup en Google Chrome versión Android

En caso de que no aparezca dicho popup, se podrá instalar la aplicación usando el navegador Google Chrome de Android. Para ello primero se pulsa sobre el menú de opciones localizado en la parte superior derecha y se selecciona la opción de instalar aplicación.



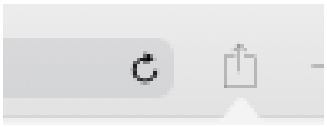
Menú de opciones de Google Chrome

Opción de instalar aplicación.

Dispositivos iOS

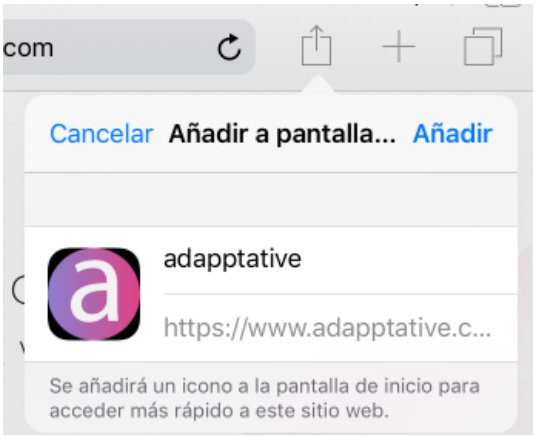
En el caso de querer instalar la aplicación en un dispositivo iOS hay que navegar a la misma url, tfg-srj.web.Aplicación, desde el navegador Safari.

Una vez en la página, seleccionamos la opción de compartir, localizada en la parte superior a la derecha del buscador.



Botón de compartir

En el menú desplegado, seleccionamos la opción de “añadir a la pantalla de inicio” y pulsamos sobre “añadir”



Opción de Añadir a pantalla de inicio

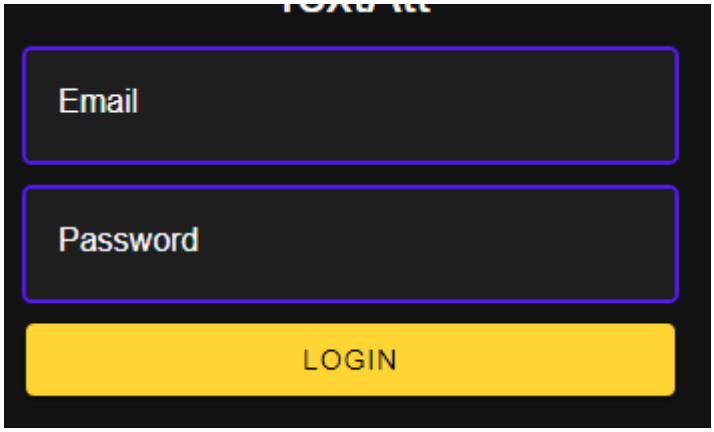
Una vez hecho esto, la aplicación será instalada y se podrá usar desde el icono en la pantalla de inicio.

Anexo II: Manual de usuario

Inicio de sesión:

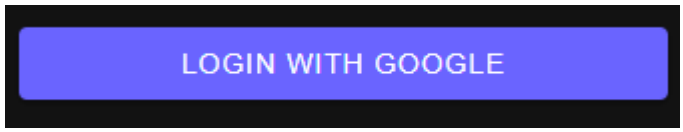
Cuando se ingresa en la aplicación, lo primero que hay que hacer es iniciar sesión. Para ello existen dos opciones:

La primera es ingreso mediante correo y contraseña, rellenando los campos específicos y pulsando el botón login.



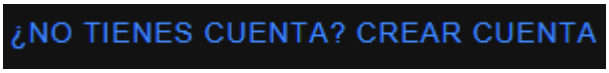
Campos de Correo/contraseña y botón de login.

La segunda opción es ingresar mediante cuenta de google. Para ello pulsamos sobre el botón de “Login with Google”. Si el dispositivo tiene una cuenta de Google vinculada, iniciará sesión con esta, si no, abrirá una nueva ventana en la que se podrá seleccionar con que cuenta se quiere ingresar.

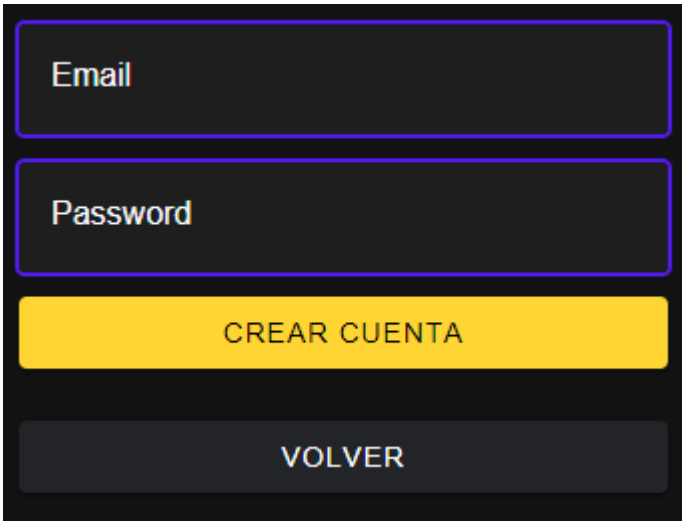


Crear cuenta

En caso de querer ingresar con correo y contraseña, pero no disponer de una cuenta en la aplicación, se pulsará sobre el texto de "¿No tienes cuenta? Crear nueva" localizado en la parte inferior, entre los botones de "Login" y "Login with Google"



Tras esto, se rellena los datos con el correo y contraseña deseados y se pulsa sobre el botón de crear cuenta. Si no se quiere crear una, solo hay que pulsar sobre el botón "Volver"



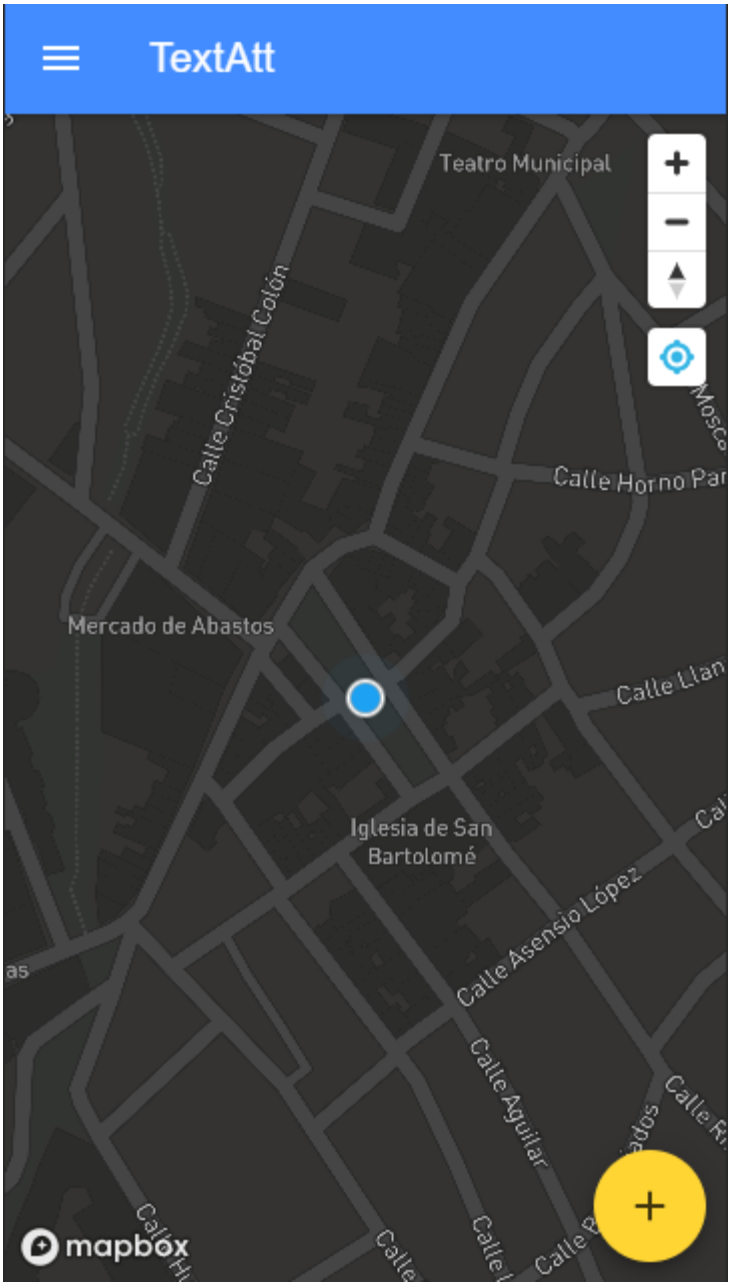
Rellenar datos:

Si es la primera vez que se ingresa con una cuenta en la aplicación, se pedirá que se indique el nombre del usuario (si se ha entrado con cuenta de Google, este será dado por la cuenta) y un tag para el usuario, este tag es único para cada usuario y, en caso de ingresar uno que ya exista para otro usuario, se pedirá que se ponga uno nuevo. Una vez rellenado los datos, se debe de pulsar en el botón “Actualizar Datos” para continuar.

Se deben de rellenar los datos “Nombre” y “Tag” para poder continuar.

Controles de la aplicación:

Una vez rellenados los datos, o si no es la primera vez que se ingresa con una cuenta, se nos colocará en la vista principal de la aplicación.



Vista principal de la aplicación.

Pulsando sobre cualquier punto del mapa y arrastrando cambiaremos la posición sobre la que el mapa está enfocado.

Pulsando con dos dedos y separando o uniéndolos, daremos o quitaremos zoom al mapa. También se puede hacer zoom pulsando dos veces rápidamente sobre el mapa, esto además cambiará la posición hacia la parte donde hemos tocado dos veces.

Pulsando con dos dedos y girándolos cambiaremos la orientación del mapa.

Existen 4 botones en la esquina superior derecha:

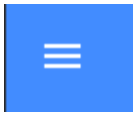
El primer botón sirve para añadir zoom al mapa.

El segundo botón quita zoom al mapa.

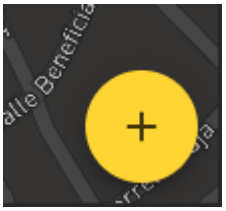
El tercer botón reorienta el mapa al modo por defecto, con el norte en la parte superior.

El cuarto botón centra la cámara sobre la posición actual del usuario.

Se puede desplegar el menú lateral pulsando sobre el botón de despliegue en la parte superior izquierda o arrastrando desde el borde izquierdo de la pantalla hacia la derecha.



Botón de despliegue del menú lateral.

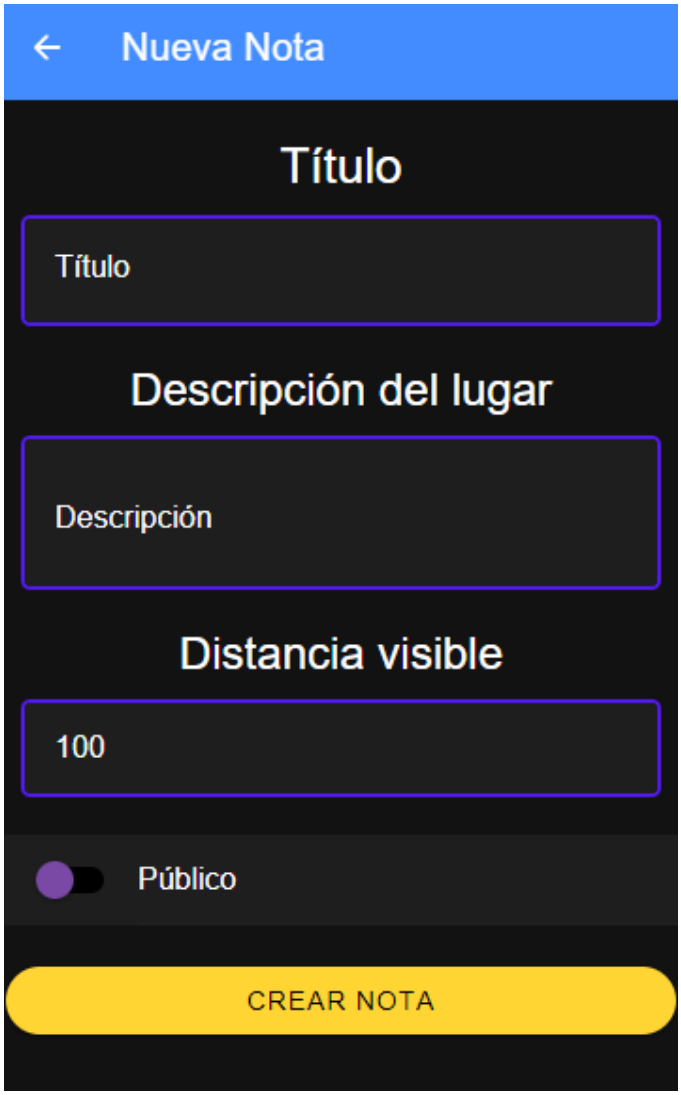


En la parte inferior derecha se encuentra el botón de crear marcador nuevo.

Al pulsar este botón se abrirá la ventana de Creación de nueva nota

En esta ventana, se deberá rellenar el formulario, indicando el título de la nota, la descripción del lugar, su distancia de visión en metros, que debe de tener un valor de entre 10 y 100, y mediante el botón de radio, indicar si la nota será pública o no. Por defecto la nota será privada. Para añadir etiquetas a esta nota, solo hace falta escribir con el símbolo # delante de la palabra que queramos convertir en etiqueta. Por ejemplo “#Monumento”.

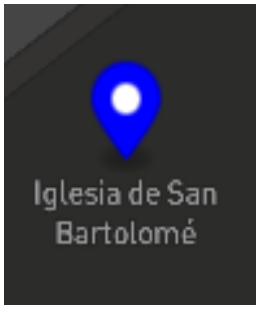
Una vez rellenados todos los datos, se pulsará sobre el botón de “Crear Nota”, lo cual nos devolverá a la ventana del mapa y la nota habrá sido creada.

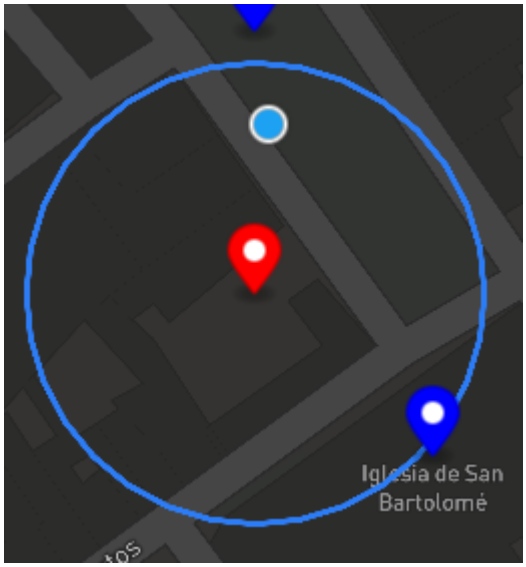


Nota: Es posible que tarde un poco de tiempo en esta nueva nota.

Las notas están representadas por marcadores colocados en el mapa.

El color del marcador indica el tipo de nota.
Rojo: Nota privada creada por el usuario.
Morado: Nota pública creada por el usuario.
Azul: Nota pública no creada por el usuario.

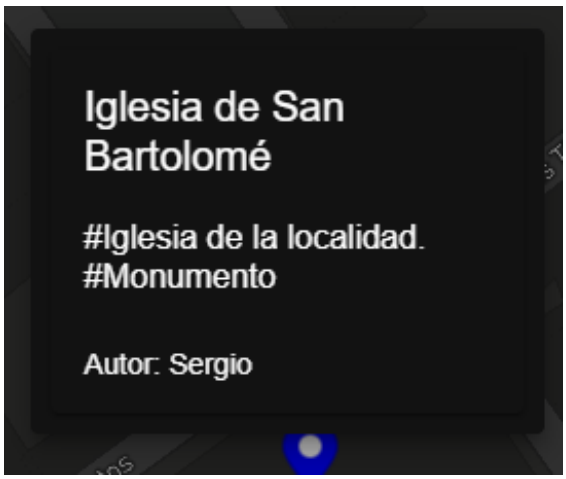




Cada marcador genera un círculo con el radio a partir del cual dicho marcador es visible sin filtros.

Pulsar sobre un marcador nos mostrará la información sobre la nota.

Si la nota es nuestra (marcador morado o rojo) en la parte inferior de esta aparecerá un botón que nos permitirá modificar la nota. Si la nota no es nuestra (marcador azul), aparecerá el tag del usuario que la creó en su lugar.



Pulsar sobre el botón de “Modificar Nota” nos llevará a una ventana que nos permitirá cambiar los datos de esta de una manera similar a la creación de nota.

Actualmente, en la aplicación se puede filtrar por etiqueta o por usuario. El filtrado permite ver las etiquetas que cumplan dicho filtro.

Se puede realizar la filtración pulsando sobre las etiquetas o el autor dentro de una nota, lo cual nos filtrará según la etiqueta seleccionada o el autor de dicha nota.

También se puede filtrar escribiendo el filtro que deseemos manualmente. Para ello desplegamos el menú lateral y pulsamos en el apartado de filtro, donde podremos escribir el filtro que deseemos.

Para filtrar por etiqueta, se debe escribir # y la etiqueta que queramos. Si queremos filtrar por usuario, solo hace falta escribir el tag del usuario del que queramos filtrar las notas.

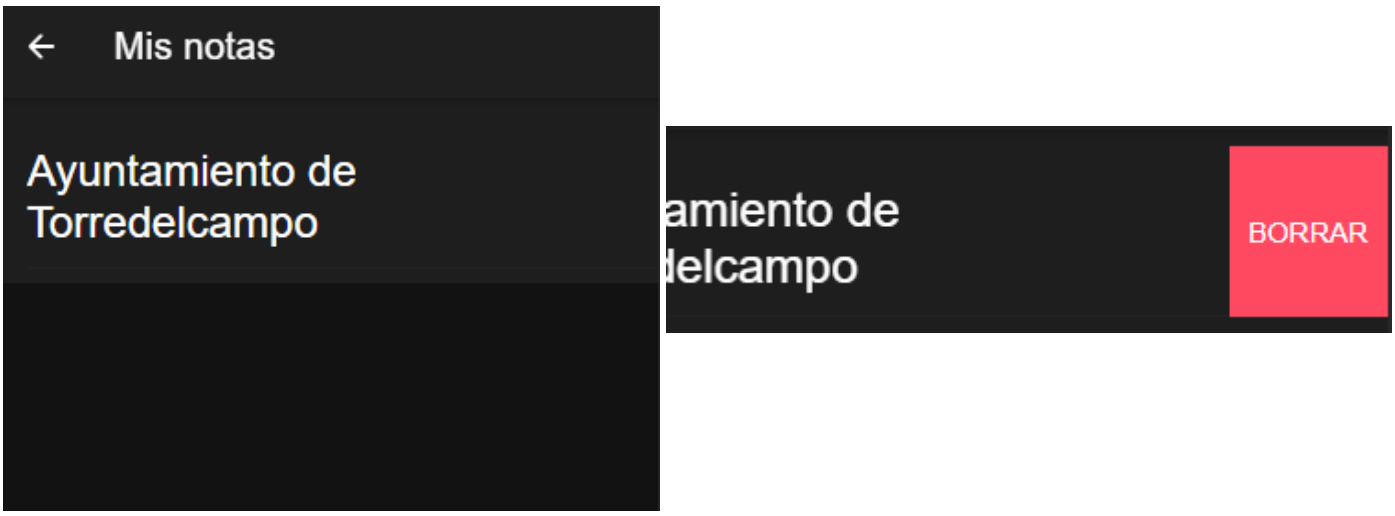
Si queremos quitar el filtro, solo hace falta borrar lo que está escrito en este campo o pulsar el botón “X” que aparecerá en la parte derecha si hay algo escrito.

En el menú lateral nos encontramos también las siguientes opciones:

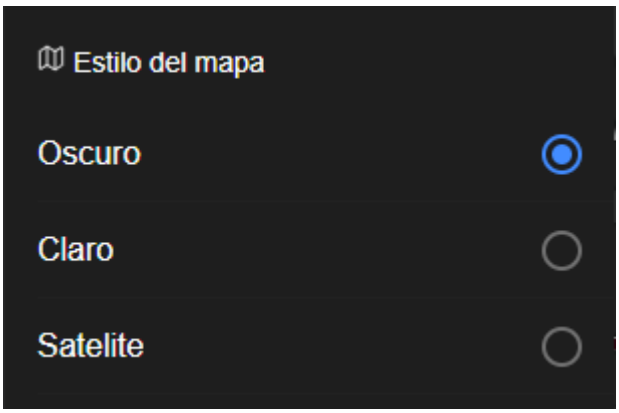
La opción perfil nos permite consultar o cambiar la información sobre nuestro perfil. Abre la misma ventana indicada en el apartado de rellenar datos.

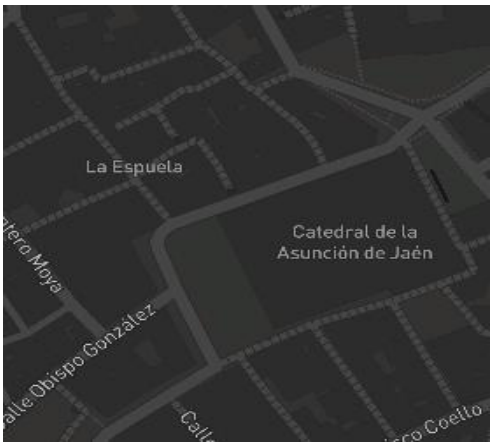
La opción “Mis Notas” permite consultar todas nuestras notas.

Pulsar sobre la nota nos llevará a la ventana de modificación de nota. Si mantenemos pulsado sobre la nota y deslizamos a la izquierda, desvelaremos la opción de borrar dicha nota. Pulsar sobre este nuevo botón borrará esta nota.



La opción estilo de mapa nos permitirá cambiar el estilo visual del mapa entre tres opciones, oscura, clara o Satélite. La opción por defecto es la oscura.





Modo Oscuro

Modo Claro

Modo Satelite

Es recomendable mantener el modo oscuro o claro, puesto que el modo satélite pierde calidad mientras más zoom se haga.

La última opción del menú lateral es la opción de cerrar sesión, pulsar sobre esta cierra la sesión del usuario y lo devuelve al menú de inicio de sesión.

Anexo III: Cuestionario a usuarios.

Las pruebas detalladas en el apartado 5.b de esta memoria fueron realizadas en el ámbito local de Torredelcampo (Jaén) y los usuarios que participaron en ellas fueron tanto familiares como amigos. Las pruebas fueron realizadas por 9 personas.

Las pruebas 5.b.ii y 5.b.v comenzaron en la zona de la plaza del pueblo, donde previamente se crearon diversas notas, algunas con etiquetas y otras sin ellas. Posteriormente, para enseñar el radio de funcionamiento de las notas, se les desplazó desde esta plaza al parque local, suficientemente alejado como para que las notas que aparecían en el primer sitio desaparezcan del mapa.

6.1. Preguntas

Una vez acabadas las pruebas, se les pidió a los usuarios que rellenaran un pequeño formulario, el cual contenía las siguientes preguntas:

¿Cuál es tu edad?:

Donde el usuario puede seleccionar 3 rangos de edad. Menor de 18 años, 18-40 años y mayor de 40. Estos rangos fueron escogidos en función a los usuarios que iban a tomar parte en las pruebas.

¿Cómo de interesante te ha parecido la aplicación?

Se le pregunta al usuario como de interesante le ha parecido la temática de la aplicación y sus posibles usos.

El usuario puede escoger entre 5 valores, donde el 1 es muy poco interesante y el 5 es muy interesante.

¿Cómo de fácil te ha parecido de usar la aplicación?

Se le pregunta al usuario como de difícil le ha parecido usar la aplicación una vez que se ha familiarizado con los controles y los conceptos de esta.

El usuario puede escoger entre 5 valores, donde el 1 es muy difícil y el 5 es muy fácil.

¿Cómo de intuitivo te ha parecido el manejo de los distintos elementos de la aplicación?

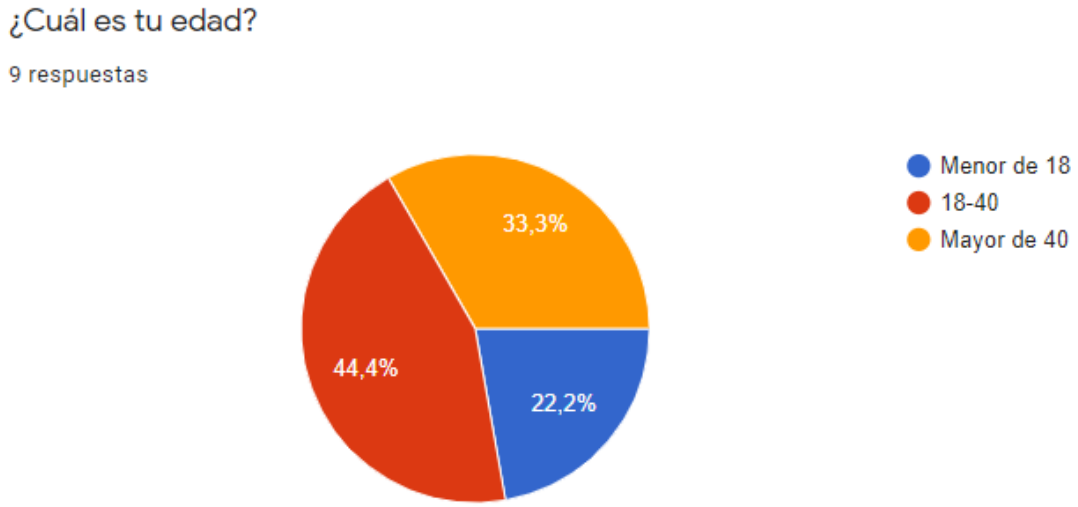
Se le pregunta al usuario como de intuitivos les ha parecido los controles de la aplicación durante su primer uso de esta.

El usuario puede escoger entre 5 valores, donde el 1 es muy poco intuitivo y el 5 es muy intuitivo.

Por último, se les daba la oportunidad de dejar un pequeño comentario sobre cualquier aspecto que quisieran comentar.

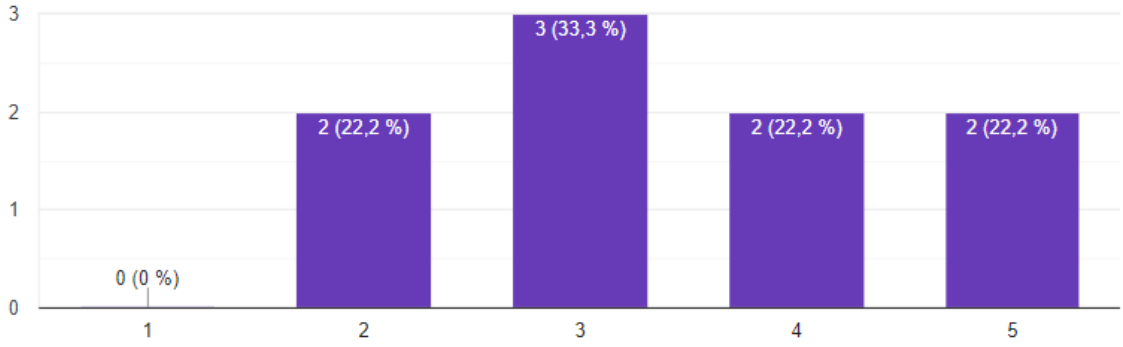
6.2. Datos

Este es el desglose de los usuarios por la edad:



De los nueve usuarios, dos eran menores de dieciocho años, cuatro tenían entre dieciocho y cuarenta años y tres eran mayores de cuarenta años.

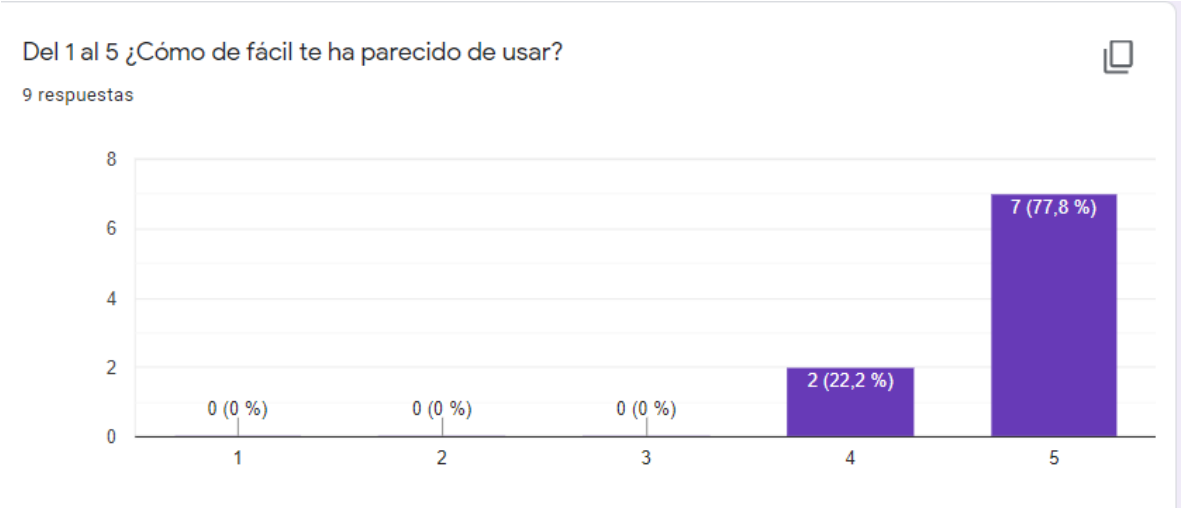
Respecto a la primera pregunta, relativa al interés sobre la temática de la aplicación, se ha recibido una cantidad de respuestas variadas.



El interés por la temática de la aplicación era relativamente alto, aunque a algunos de los encuestados no les pareció interesante.

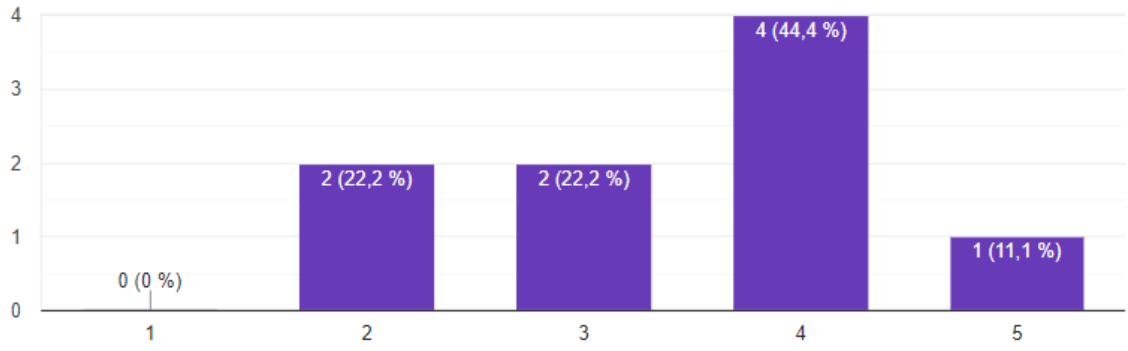
Curiosamente, en este caso, la diferencia de edad parece influir en la opinión, siendo las dos personas menores de dieciocho las que opinaban que la temática era poco interesante, mientras que los mayores de cuarenta tuvieron una opinión muy buena, puntuando dos de ellos con un 5 y uno con un 4. Los usuarios de entre dieciocho y cuarenta años se mostraron más apáticos con respecto a la temática, puntuando tres de ellos con un 3 la encuesta y uno si mostró más interés, puntuando con un cuatro.

Respecto a la segunda pregunta, relativa a la facilidad de uso de la aplicación, las respuestas fueron muy positivas



Todos los usuarios encontraron muy fácil usar la aplicación una vez que se familiarizaron con los conceptos y controles de esta, puntuando siete de ellos con un 5 este apartado y dos de ellos con un cuatro.

Respecto a la tercera pregunta, relativa a como de intuitivo les habían parecido los controles en su primer uso, las respuestas fueron muy variadas.



En este caso, la diferencia de edad parece que afecta en gran medida, puesto que todos los usuarios menores de dieciocho y todos los que tienen entre dieciocho y cuarenta años menos uno puntuaron esta pregunta con un 4 o 5. No obstante, dos de los usuarios de cuarenta o más puntuaron con un 2 y uno de ellos con un tres, por lo que parece que la brecha digital es palpable cuando se prueba una nueva

aplicación. Los usuarios más jóvenes y que tienen mucha experiencia manejando dispositivos digitales se adaptan y comprenden casi al instante el uso de los controles, mientras que a los usuarios mayores les cuesta más adaptarse a estas tecnologías.

Por último, algunos de los usuarios dejaron algunos comentarios, la mayoría relacionados con algunos problemas de usabilidad.

Un ejemplo de esto estaba durante la creación y actualización de notas, donde el botón que indica si la nota será pública o privada tan solo tenía como texto “Pública” y un slider que indicaba si sería pública o privada. El hecho de que la posición de dicho slider no cambiase el texto e indicase como sería la nota llevó a bastantes confusiones.

El otro gran problema que encontraron ocurrió cuando algunos de ellos denegaron los permisos de localización en sus dispositivos, provocando que la aplicación no funcionase correctamente ya que estos permisos son esenciales. Si ocurría este caso, la aplicación no indicaba al usuario que podía haber algún problema.

Se aprovecharon las opiniones y los elementos encontrados para aplicar cambios de calidad de vida en la aplicación, solucionando estos problemas para facilitar aún más la usabilidad de esta.