



Facultad de Ingeniería
Universidad de Deusto

Ingeniaritza Fakultatea
Deustuko Unibertsitatea

Máster Universitario en Ingeniería Informática

Informatikako Ingeniaritzako Unibertsitate masterra

Proyecto fin de máster
Master amaierako proiektua

Resumen

El proyecto consiste en el desarrollo de una aplicación nativa para Android para que los usuarios puedan compartir vehículo a diario para ir a trabajar. Con ello se busca conseguir un doble objetivo: que los usuarios puedan ahorrar dinero en sus desplazamientos diarios para ir a trabajar sin perder por ello la comodidad o rapidez de utilizar el vehículo particular y, segundo, reducir los atascos y la contaminación provocadas por el uso individual del vehículo. Para ello, la aplicación permitirá a los conductores crear rutas donde indicarán el lugar exacto y la hora a la que realizarán los desplazamientos, así como la tarifa que deben abonar los usuarios. Después, los usuarios pueden buscar aquellas rutas que se encuentren a una distancia máxima y unirse.

Descriptores

- Android
- Vehículo
- Carpooling
- Ir al trabajo
- Contaminación

Índice

1. INTRODUCCIÓN	1
1.1 Presentación del documento.....	1
1.2 Motivación	2
1.2.1 La contaminación aérea	2
1.2.2 El vehículo para ir a trabajar	3
1.3 Aplicaciones Similares	3
1.3.1 Móvil	4
1.3.2 Web	5
1.4 Android	5
2. DEFINICIÓN Y OBJETIVOS DEL PROYECTO.....	7
2.1 Visión general.....	7
2.2 Definición de objetivos	7
2.2.1 Objetivos.....	7
2.2.2 Alcance.....	7
2.3 Producto final	7
2.4 Descripción de la realización.....	8
2.4.1 Método de desarrollo.....	8
2.4.2 EDT	9
2.4.3 Definición de tareas.....	10
2.4.4 Productos intermedios.....	11
2.5 Organización	11
2.5.1 Esquema organizativo.....	11
2.5.2 Plan de recursos humanos.....	12
2.6 Condiciones de ejecución	12
2.6.1 Entorno de trabajo.....	12
2.6.2 Control de cambios	12
2.6.3 Recepción de los productos	13
2.7 Planificación	13
2.7.1 Plan de trabajo	13
2.7.2 Diagrama de precedencias	14
2.7.3 Diagrama de Gantt	15
2.7.4 Reparto de cargas de trabajo por recurso humano del proyecto.....	16
2.7.5 Cargas de trabajo para las diferentes tareas del proyecto	16
2.7.6 Presupuesto	17

3. ESPECIFICACIÓN DE REQUISITOS	19
3.1 Visión general	19
3.2 Especificación de requisitos de la aplicación Android.....	19
3.3 Casos de uso.....	20
3.3.1 Registro	21
3.3.2 Añadir coche.....	23
3.3.3 Crear ruta.....	24
3.3.4 Unirse a una ruta	26
4. TECNOLOGÍAS UTILIZADAS	27
4.1 Visión general	27
4.2 Android	27
4.2.1 Razón de uso.....	28
4.3 Java	28
4.4 Firebase.....	29
4.4.1 Razón de uso.....	29
4.5 NodeJS	30
4.5.1 Express.....	30
4.5.2 Mongoose	30
4.5.3 Razón de uso.....	31
4.6 JavaScript.....	31
4.7 MongoDB.....	31
4.7.1 Razón de uso.....	32
4.8 JSON	32
4.9 Google Cloud SQL	33
4.10 Google Maps	33
5. ESPECIFICACIÓN DEL DISEÑO.....	35
5.1 Visión general	35
5.2 Entorno de desarrollo	35
5.2.1 Android Studio	35
5.2.2 GitHub.....	36
5.2.3 NotePad++.....	36
5.2.4 Sublime Text.....	37
5.2.5 TortoiseGit	37
5.2.6 Visual Paradigms.....	37
5.2.7 Microsoft Project.....	37
5.3 Modelo de la base de datos	38
5.3.1 User	38

5.3.2	Ride	38
5.3.3	Car	38
5.3.4	Messaging	38
5.3.5	UserDays.....	38
5.4	Arquitectura de la aplicación	39
6.	IMPLEMENTACIÓN	41
6.1	Visión general.....	41
6.2	Implementación de la Base de Datos MongoDB	41
6.3	Implementación del servidor NodeJS	42
6.3.1	Servidor REST	42
6.3.2	Servidor Messaging.....	43
6.4	Implementación de Firebase	44
6.4.1	Authentication.....	44
6.5	Implementación de la aplicación Android.....	45
6.5.1	Firebase.....	45
6.5.2	Activities	46
6.5.3	Indicativo de coche eléctrico	46
6.5.4	Búsqueda de rutas	47
6.5.5	Google Maps	47
6.5.6	Preferencias	48
6.5.7	Multilenguaje	48
7.	HERRAMIENTAS DE ANÁLISIS	49
7.1	Visión general.....	49
7.2	Firebase Analytics	49
7.3	Crash reporting.....	50
8.	MANUAL DE USUARIO	53
8.1	Visión General.....	53
8.2	Login.....	53
8.3	Registro	53
8.4	Menu.....	54
8.5	Coche	55
8.6	Crear ruta	55
8.7	Localización en Google Maps	56
8.8	MI Ruta	57
8.9	Resultados	57
8.10	Detalle de la ruta	58
8.11	Perfil usuario	59

8.12	Notificaciones	59
8.13	Ajustes	60
9.	CONCLUSIONES Y LÍNEAS FUTURAS	61
9.1	Visión general	61
9.2	Conclusiones	61
9.3	Incidencias	61
9.4	Líneas futuras	62
9.4.1	Monetización	62
9.4.2	Solución en caso de ausencia del conductor	62
9.4.3	Valoración entre usuarios	62
10.	BIBLIOGRAFÍA.....	63

Índice de figuras

Ilustración 1.1: Porcentaje de las versiones de Android, a mayo 2017	6
Ilustración 2.1 Modelo de desarrollo incremental	8
Ilustración 2.2: EDT	9
Ilustración 2.3. Organización.....	11
Ilustración 2.4 Plan de trabajo.....	13
Ilustración 2.5 Diagrama de Precedencias	14
Ilustración 2.6 Diagrama de Gantt	15
Ilustración 2.7 Cargas de trabajo	16
Ilustración 2.8: Tareas.....	16
Ilustración 3.1 Casos de uso	20
Ilustración 3.2: Registro.....	21
Ilustración 3.3: Secuencia del Registro.....	22
Ilustración 3.4: Añadir coche	23
Ilustración 3.5: Crear ruta.....	24
Ilustración 3.6: Secuencia de crear ruta.....	25
Ilustración 3.7: Unirse a una ruta	26
Ilustración 4.1: Android.....	27
Ilustración 4.2: Firebase	29
Ilustración 4.3: NodeJS	30
Ilustración 4.4: MongoDB	31
Ilustración 4.5: Google Maps	33
Ilustración 5.1: GitHub.....	36
Ilustración 5.2: Notepad	36
Ilustración 5.3: SublimeText 3.....	37
Ilustración 5.4: Base de datos de la aplicación.....	38
Ilustración 5.5: Arquitectura de la aplicación	39
Ilustración 6.1: Documento en MLab	41
Ilustración 6.2: Métodos REST	42

Ilustración 6.3: Implementación REST en Android	42
Ilustración 6.4: Implementación REST en Android	43
Ilustración 6.5: MessagingService.....	44
Ilustración 6.6: Usuarios registrados en Firebase	44
Ilustración 6.7: Registro de un usuario	45
Ilustración 6.8: Enviar email para resetear la contraseña	45
Ilustración 6.9: Butterknife	46
Ilustración 6.10: Símbolo indicativo de coche eléctrico.....	46
Ilustración 6.11: Fichero XML con la clave de Google Maps	47
Ilustración 6.12: Seekbar para seleccionar la distancia máxima	48
Ilustración 6.13: Multilenguaje	48
Ilustración 7.1: Analíticas de Firebase sobre nuestros usuarios ..	49
Ilustración 7.2: Análisis de Firebase sobre la procedencia de los usuarios y sus dispositivos ..	50
Ilustración 7.3: Informe sobre los fallos que se han producido	51
Ilustración 7.4: Email diario con los errores detectados.....	51
Ilustración 8.1: Login	53
Ilustración 8.2: Formulario de registro.....	53
Ilustración 8.3: Menú de la aplicación.	54
Ilustración 8.4: Menú lateral	54
Ilustración 8.5: Añadir coche	55
Ilustración 8.6: Crear ruta	55
Ilustración 8.7: Reloj para indicar la hora	56
Ilustración 8.8: Lista para indicar los días	56
Ilustración 8.9: Google Maps.....	56
Ilustración 8.10: Vista de la ruta desde la perspectiva del conductor	57
Ilustración 8.11: Resultados de la búsqueda	57
Ilustración 8.12: Ruta desde la perspectiva del pasajero.....	58
Ilustración 8.13: Lista de solicitudes y usuarios unidos	58
Ilustración 8.14: Perfil de usuario	59

Ilustración 8.15: Notificación	59
Ilustración 8.16: Distancia y tiempo máximo.....	60
Ilustración 8.17: SeekBar para indicar la distancia máxima	60

Índice de tablas

Tabla 2.1: Coste de los recursos humanos	17
Tabla 2.2: Coste del Hardware y Software	17
Tabla 2.3: Coste total del proyecto	17

1. INTRODUCCIÓN

1.1 PRESENTACIÓN DEL DOCUMENTO

En este documento se presenta la aplicación DriveJob, desarrollada en Android y cuyo objetivo es tratar de animar a los usuarios a que comparten coche para realizar sus desplazamientos diarios, describiendo los objetivos del proyecto, la planificación, el diseño y la implantación:

- Definición y objetivos del proyecto

En este apartado se definen los objetivos del proyecto. Además, se define el método de desarrollo del proyecto y las tareas a realizar para lograr el objetivo del proyecto. También se define la composición del equipo de trabajo durante el desarrollo y sus funciones, y las condiciones necesarias para ejecutar el proyecto. Por último, se detalla la planificación y el presupuesto necesario.

- Especificación de requisitos

Especificación de los requisitos que debe satisfacer nuestra aplicación y los principales casos de uso.

- Tecnologías utilizadas

Exposición de las diferentes tecnologías que se han empleado durante el desarrollo del proyecto y la razón de su uso.

- Especificación del diseño

Diseño de los diferentes componentes de la aplicación.

- Implementación

Detalle sobre las consideraciones a tener cuenta durante la implementación del proyecto.

- Herramientas de análisis

Herramientas para conocer el funcionamiento de nuestra aplicación en el mercado y detectar los problemas que sufren los usuarios.

- Manual de usuario

Explicación de las funcionalidades más importantes de la aplicación, para que cualquier tipo de usuario pueda hacer uso de ellas.

- Conclusiones y líneas futuras

Exposición las conclusiones y el aprendizaje obtenido tras la realización del proyecto, y los posibles cambios y mejoras que se pueden realizar en un futuro

1.2 MOTIVACIÓN

1.2.1 La contaminación aérea

Uno de los grandes problemas a los que se enfrenta hoy en día la sociedad es la contaminación. Este problema se ha visto agravado en los últimos años debido al progreso tecnológico que ha originado diversas formas de contaminación. Esto está teniendo consecuencias en el medio ambiente como el calentamiento global, el derretimiento de los polos y glaciares, ... Aunque la contaminación se puede producir de diversas maneras, en este estudio vamos a centrarnos más concretamente en la contaminación aérea.

Uno de los principales causantes de la contaminación aérea son los gases que emiten los vehículos a través del tubo de escape. [1] El automóvil es el medio de transporte que más energía consume por persona transportada y kilómetro recorrido. Se calcula que los automóviles son responsables del 80% de emisiones de NO₂ debidas al tráfico y del 60% de emisiones de partículas.

[2] Además, este problema es más grave en Europa, debido a la gran popularidad de los coches diésel. Estos vehículos resultan beneficiosos y para el conductor debido a su menor consumo de combustible; pero son más contaminantes que los vehículos de gasolina, debido a que emiten hasta cuatro veces más partículas en suspensión y liberan a la atmósfera partículas dañinas para la salud humana, como los óxidos de nitrógeno -NO_x-, los de azufre, el monóxido de carbono y las partículas de hollín. Para tratar de reducir este problema, diversos países imponen restricciones en cuanto a las emisiones máximas que pueden emitir, aunque como se ha podido comprobar recientemente esta medida parece ser inútil debido a que los fabricantes manipulan el funcionamiento de sus coches durante las pruebas para maquillar las cifras de emisiones contaminantes a la baja.

[3] Esta contaminación tiene graves consecuencias, tanto a corto como a largo plazo, para la salud de las personas. Según la Organización Mundial de la Salud (OMS), la contaminación atmosférica urbana aumenta el riesgo de padecer enfermedades respiratorias agudas, como la neumonía, y crónicas, como el cáncer del pulmón y las enfermedades cardiovasculares. Estos efectos son más graves en os grupos más vulnerables, como los niños, los ancianos o las personas ya enfermas. Además, mediante la disminución de los niveles de contaminación del aire, los países pueden reducir la carga de morbilidad derivada de accidentes cerebrovasculares, cánceres de pulmón y neumopatías crónicas y agudas, entre ellas el asma. Cuanto más bajos sean los niveles de contaminación del aire mejor será la salud cardiovascular y respiratoria de la población, tanto a largo como a corto plazo.

[4] Para solucionar este problema, la Agencia Europea de Medio Ambiente aconseja reducir la velocidad y restringir el tráfico en el centro de las ciudades ayuda para reducir las emisiones de contaminantes. Según este organismo, las limitaciones de 80km/h ayudan a bajar los niveles de óxidos de nitrógeno de los motores diésel y las partículas en suspensión. Además de esta solución, las ciudades europeas con mayor contaminación han adoptado diferentes medidas:

- Londres: se aplica una tasa de congestión. Se trata de un cargo a los vehículos que se mueven en la zona de peaje de lunes a viernes en horario laboral, es decir, entre las 7:00h y las 18:00h. En cambio, no se paga ni los fines de semana, ni el día de Navidad ni el de año nuevo. El resto de días la tasa es de 12 libras al día por acceder al centro. Pero hay rebajas para quien quiera abonar antes la cantidad: 10 libras para el que lo haga el día antes y 9 si se registra en el sistema de auto pago.

- Roma: sólo se deja pasar al centro de la ciudad a residentes y trabajadores si pagan una cuota anual y a personas con algún tipo de discapacidad. En cambio, los motos y ciclomotores pueden circular gratis. La zona restringida es de 4,2 kilómetros cuadrados y está vigente de lunes a viernes de 6.30 a 18h y de 14h a 18 horas el sábado.
- París: se paga para aparcar en todas las zonas de la ciudad. Pagan tanto residentes como visitantes, que pueden hacerlo un máximo de dos horas antes de cambiar su coche.
- Oslo: el objetivo de la capital noruega es prohibir la entrada de los vehículos en la ciudad a partir del año 2019
- Madrid: si la contaminación sube en exceso, se empiezan a aplicar diferentes medidas para restringir el tráfico, desde reducir los límites de velocidad hasta permitir el tráfico dependiendo de si la matrícula es par o impar.

1.2.2 El vehículo para ir a trabajar

[5] Uno de los principales motivos de esta contaminación son los desplazamientos para ir a trabajar. Para realizar estos, el 62% de los trabajadores opta por utilizar un vehículo propio, mientras que para los desplazamientos de ocio los usuarios prefieren desplazarse andando o en bicicleta. [6] De los trabajadores que se desplazan a diario en coche, solo el 11% los comparten, aunque casi el 50% se muestra en disposición de llevar pasajeros, siempre que eso no suponga ninguna desviación de su ruta.

[7] Además de la contaminación, los conductores pierden mucho tiempo en los atascos producidos durante el horario de entrada y salida del trabajo. En España en 2014, Barcelona más congestionada, ya que los usuarios estuvieron de media 25 horas anuales en atascos. Por su parte, Madrid sitúa la espera en 22 horas. Completan este ranking Sevilla (con 18 horas, 2 más que en 2013), Bilbao (con 16 horas, 8 menos que el año pasado), Zaragoza (12 horas) y Valencia (11 horas al año). Sin embargo, ninguna está entre las 25 ciudades con mayores atascos. Lidera el ranking Londres con 96 horas al año.

Además de compartir gastos, ir dos pasajeros en un mismo vehículo puede permitir circular por carriles especializados solo para vehículos de alta ocupación y ahorrar tiempo en atascos. [8] O si no hay otra opción, tratar de usar una muñeca para utilizar el carril.

Además de perder tiempo, ir en coche al trabajo produce estrés a los trabajadores. [9] En una encuesta realizada por Ford a 5.500 de sus trabajadores europeos expone que una de cada de tres trabajadores consideran los desplazamientos más estresantes que la jornada laboral. En Roma, por ejemplo, la mayoría de encuestados destaca como fuente de estrés estos desplazamientos por encima de mudarse de casa

1.3 APPLICACIONES SIMILARES

Antes de empezar a hablar de la situación del mercado actual, me gustaría matizar los términos relacionados con compartir vehículo. La traducción más sencilla para traducir 'compartir coche' podrá parecer 'carsharing', sin embargo, este término hace referencia a alquilar el vehículo propio con otra persona, es decir, compartir el coche de manera literal. En inglés se utiliza el término 'carpooling' para indicar dos personas que comparten trayecto. Además, en este análisis, me he centrado en escoger solo aquellas herramientas que no generan lucro para ninguno de los usuarios.

1.3.1 Móvil

1.3.1.1 Blablacar

[10] Blablacar es una aplicación francesa para carpooling, que hace posible que las personas que quieren desplazarse al mismo lugar al mismo momento puedan organizarse para viajar juntos. Es una de las aplicaciones más populares para compartir viaje con más de diez millones de descargas en Android, contando en España incluso con anuncios en televisión. En ella, un usuario puede crear un viaje y el resto de usuarios deberán pagar una pequeña cantidad para unirse al viaje, que viene limitada para evitar abusos. Su modelo de negocio se basa en cobrar un pequeño porcentaje del pago que realizan esos usuarios.

Ventajas:

- Gran número de usuarios
- Fácil de usar e intuitiva
- Seguridad
- Ofrece reembolso en caso de cancelación
- Seguro adicional en caso de accidente

Diferencias con nuestra aplicación

- Solo permite crear un viaje cada ocasión
- El lugar de origen y destino se indica solo con el nombre de la localidad, por lo que en grandes ciudades como Madrid, puede que tengas que recorrer varios kilómetros para llegar al punto de encuentro

1.3.1.2 Amovens

[11] Amovens es una aplicación española que realiza en mayor medida las mismas acciones que Blablacar. El momento de mayor popularidad de la app se produjo cuando Blablacar decidió empezar a cobrar comisión por viajes, mientras que ellos ofrecían el servicio gratis. Sin embargo, su negocio parece estar más dirigido hacia el carsharing, es decir, el alquiler de coches entre usuarios, servicio por el que si cobran comisión.

Ventajas:

- No cobra comisiones en el pago en efectivo y hasta un euro si el pago se realiza online
- Permite indicar el lugar de origen y destino de una manera más precisa, pero el usuario debe escribir la dirección completa
- Permite indicar puntos intermedios.

Diferencias con nuestra aplicación:

- Solo permite crear un viaje cada ocasión
- Aunque permite indicar de manera precisa el lugar de origen y destino, por las búsquedas realizadas, los usuarios apenas hacen uso de dicha funcionalidad, por lo que en grandes ciudades como Madrid, puede que tengas que recorrer varios kilómetros para llegar al punto de encuentro
- El negocio de carpooling parece ser secundario, ya que se centran sobre todo en el alquiler de coches particulares.

1.3.1.3 Poolmyride

[12] Poolmyride es una aplicación india que permite crear rutas, tanto mensuales como ocasionales. Su mayor problema es que cuenta con muy pocos usuarios y la mayoría son indios.

Ventajas:

- Combina tanto viajes diarios como ocasionales

Diferencias con nuestra aplicación:

- El lugar de origen y destino se indica solo con el nombre de la localidad, por lo que, en grandes ciudades como Madrid, puede que tengas que recorrer varios kilómetros para llegar al punto de encuentro.
- Interfaz confusa

1.3.2 Web

En la web se han encontrado algunas opciones para compartir viaje, como Shareling o Carpling, pero uso es poco claro e intuitivo, además de que, por las pruebas realizadas, no parece que tengan usuarios en la actualidad.

1.3.2.1 Liftshare

[13] Liftshare es una página web británica que permite compartir vehículo, tanto para diario como para una sola vez. De momento, su negocio se basa en la web y no cuenta con aplicación para dispositivos móviles.

Ventajas:

- Combina ambos modelos de negocio: viajes diario y viajes ocasionales
- Al realizar una búsqueda, muestra aquellos resultados que pasan por los lugares de salida y llegada indicados.

Diferencias con nuestra aplicación:

- Solo cuenta con aplicación web

1.4 ANDROID

Como se ha comprobado durante el análisis anterior, el mercado de compartir vehículo es prácticamente 100% móvil. [14] Esto es debido a la gran penetración de los smartphones en el mercado, y principalmente, en España, el país con más «smartphones» por habitante del mundo, donde el 92% de los habitantes poseen uno. [15] Esto ha provocado que, por primera vez, en 2016, el móvil haya sido el dispositivo preferente elegido por los usuarios para acceder a Internet.

Entre todas las alternativas posibles, se ha escogido Android ya que [16] es el sistema operativo más utilizado en el mundo y donde nuestro desarrollador cuenta con más conocimientos para realizar el proyecto. Sin embargo, uno de los motivos de este éxito han sido los dispositivos extremadamente baratos de algunos fabricantes con [17] versiones del sistema operativo antiguas. Debido a estos terminales asequibles, han podido acceder a un smartphone gente con capacidades económicas limitadas, por lo que, [18] a la hora de gastar dinero en aplicaciones, los usuarios de iOS invierten una mayor cantidad. Por ello, si en el futuro se desea rentabilizar la

aplicación con pequeñas comisiones, sería interesante desarrollar una versión para los dispositivos de Apple.

Por último, al depender del fabricante las actualizaciones del sistema, estas no se realizan y actualmente quedan en el mercado demasiadas versiones que se pueden considerar obsoletas, por lo que la aplicación a desarrollar no podrá tener las últimas novedades. Como se puede observar en la siguiente tabla, alrededor de un 30% de los dispositivos Android, a fecha de mayo de 2017, aún tienen un sistema operativo con más de 4 años de antigüedad, y la versión más reciente, Nougat lanzada en octubre de 2016, todavía tiene un porcentaje simbólico. Por este motivo, es importante conocer la versión de los dispositivos Android de los usuarios, porque es posible que, al querer mejorar la app, se le añada una funcionalidad que provoque que un gran número de usuarios no pueda hacer uso de ella.

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	1.0%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.8%
4.1.x	Jelly Bean	16	3.2%
4.2.x		17	4.6%
4.3		18	1.3%
4.4	KitKat	19	18.8%
5.0	Lollipop	21	8.7%
5.1		22	23.3%
6.0	Marshmallow	23	31.2%
7.0	Nougat	24	6.6%
7.1		25	0.5%

Ilustración 1.1: Porcentaje de las versiones de Android, a mayo 2017

2. DEFINICIÓN Y OBJETIVOS DEL PROYECTO

2.1 VISIÓN GENERAL

En este apartado se detallan los objetivos que debe satisfacer nuestra aplicación, así como el método de desarrollo, las tareas y la planificación para llevarlo a cabo.

2.2 DEFINICIÓN DE OBJETIVOS

2.2.1 Objetivos

El objetivo de este proyecto es desarrollar una aplicación en Android que permita a las personas que se tienen que desplazar a diario a sus lugares de trabajo compartir vehículo. De esta manera, se quiere ofrecer a los usuarios la posibilidad de ahorrar dinero, y reducir el tráfico diario en las horas de entrada y salida del trabajo, reduciendo con ello la contaminación. Para ello, se permitirá a los usuarios decidir entre crear una ruta (conductor) o unirse a uno ya existente (pasajero). En caso de decidir que quiere ser pasajero, se tratará de ofrecer al usuario aquellas rutas que mejor se adapten a sus necesidades, tratando de reducir las diferencias entre la experiencia de viajar en coche propio o compartir vehículo. Además, como compensación al conductor por sus gastos en gasolina y mantenimiento del vehículo y su labor de conducir, los pasajeros deberán abonar una cantidad variable dependiendo de la distancia recorrida y su periodicidad.

Los objetivos secundarios del proyecto serán conectarse a Google Maps para que el usuario pueda indicar con precisión tanto el punto de partida como el de llegada. Además, la aplicación debe ser intuitiva y fácil de usar y, también, multilenguaje (español e inglés de momento), con el objetivo de ser accesible al mayor número de usuarios posibles.

2.2.2 Alcance

De acuerdo a los objetivos previamente señalados en el apartado anterior, el proyecto constará de los siguientes aspectos:

- Una aplicación Android nativa, intuitiva y fácil de usar por los usuarios.
- Permitir a los usuarios que se registren y logueen en la aplicación
- Permitir a los usuarios crear y unirse a rutas.
- Un sistema de búsqueda que permita al usuario encontrar aquellas rutas que mejor se adaptan a sus necesidades.
- Cambiar de idioma dependiendo de las preferencias del usuario.
- Conexión con Google Maps para obtener datos precisos sobre la localización de los usuarios.

2.3 PRODUCTO FINAL

El producto final del proyecto consiste en una aplicación Android, que permita a los usuarios crear o unirse a rutas para compartir vehículo diariamente para ir a trabajar.

Para ello, la aplicación permitirá registrarse en ella a las personas interesadas en su uso, a través de un nombre de usuario único, una contraseña y una dirección de correo electrónico para poder contactar en caso de necesidad. Esta funcionalidad se realizará a través de Firebase.

El aplicativo contará con una base de datos que guarde toda la información de las rutas creadas por los usuarios y con un buscador que encuentre las rutas que mejor se adaptan a las necesidades de cada usuario. La base de datos se encontrará en un servidor externo especializado (mlab.com [19]). Situar la base de datos en un servidor externo ofrece la ventaja de que la base de datos estará siempre en funcionamiento y online, lo que permitirá que se pueda acceder a ella desde cualquier dispositivo.

Además, la aplicación deberá conectarse con GoogleMaps para obtener localizaciones precisas de los usuarios.

La aplicación también deberá permitir el envío de notificaciones entre los usuarios, para mantenerles informados de lo que sucede con sus rutas.

El segundo producto, serán dos servidores de NodeJS. El primero debe permitir la comunicación entre la aplicación Android y la base de datos online. El segundo será un intermediario entre nuestra aplicación y el servidor de Firebase para el envío de notificaciones.

2.4 DESCRIPCIÓN DE LA REALIZACIÓN

2.4.1 Método de desarrollo

El método utilizado para desarrollar este proyecto es el iterativo e incremental [20]. Este modelo divide el proyecto en varias partes y, en cada una de ellas realiza un desarrollo completo, desde el análisis hasta la pruebas e implementación: tras el cual, se obtiene una parte completamente funcional del proyecto. Este proceso se repite en varias ocasiones hasta lograr el producto final.

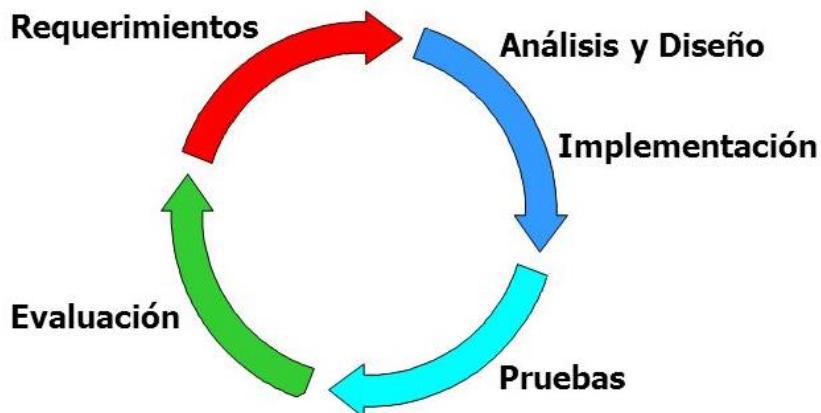


Ilustración 2.1 Modelo de desarrollo incremental

Este proceso de desarrollo intenta solucionar los defectos y problemas ocasionados por el modelo en cascada, que obligaba a completar una fase antes de comenzar con la siguiente y eso provocaba que no se adecuase a las necesidades específicas del desarrollo de software. Gracias a las iteraciones, este proceso permite encontrar y solucionar problemas de desarrollo en las fases iniciales y solventarlos del modo más efectivo posible. Además, tras cada desarrollo se obtiene un feedback y un aprendizaje, de cara a mejorar el proceso en las próximas iteraciones.

2.4.2 EDT

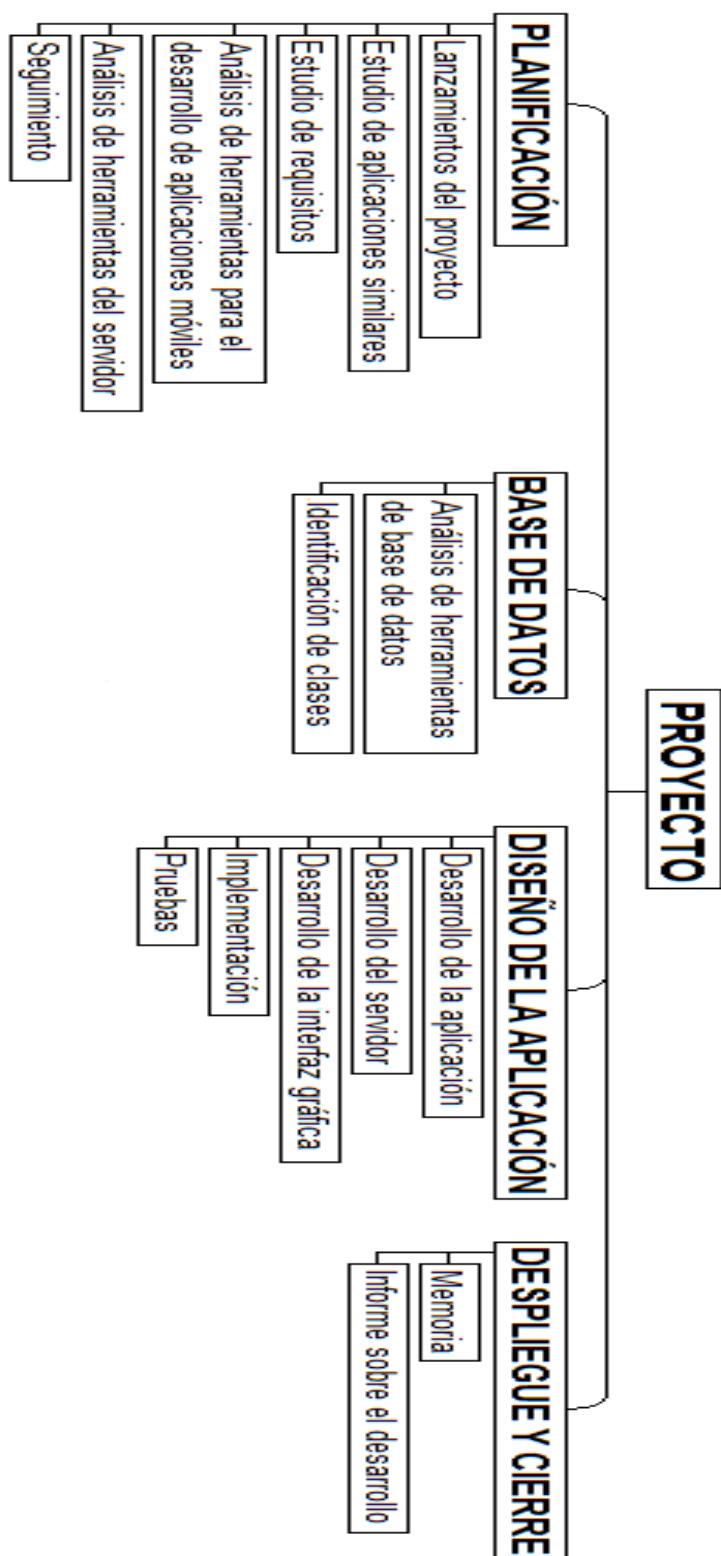


Ilustración 2.2: EDT

2.4.3 Definición de tareas

2.4.3.1 Planificación

- Lanzamiento del proyecto: aceptación del proyecto, análisis del mismo y selección del equipo de desarrollo.
- Estudio de aplicaciones similares: investigación y análisis de las herramientas disponibles actualmente en el mercado para dispositivos móviles o la web, que cubran, total o parcialmente, las necesidades de los usuarios que pretendemos satisfacer con el desarrollo de nuestra aplicación, así como el número de usuarios que las utilizan, sus funcionalidades básicas y los vacíos que podemos cubrir con nuestro proyecto.
- Análisis de requisitos: estudio sobre las necesidades básicas que nuestra aplicación debe satisfacer para ofrecer una experiencia óptima al usuario.
- Análisis de herramientas para el desarrollo de la aplicación móvil: investigación de los diferentes sistemas operativos que existen actualmente en el mercado móvil, así como las diferentes tecnologías de desarrollo disponibles.
- Análisis de herramientas del servidor: investigación de las herramientas actuales para el desarrollo de servidores.
- Seguimiento: control del progreso del proyecto.

2.4.3.2 Base de datos

- Análisis de herramientas de bases de datos: investigar las diferentes tecnologías disponibles para la persistencia de los datos y escoger en base a necesidades futuras, como la escalabilidad.
- Identificación de las clases: análisis de las necesidades de la aplicación y diseño del sistema.

2.4.3.3 Diseño de la aplicación

- Desarrollo de la aplicación: diseño e implementación de la aplicación móvil.
- Desarrollo del servidor: diseño e implementación de la lógica del servidor.
- Desarrollo de la interfaz de usuario: diseño de la interfaz gráfica siguiendo los patrones de diseño existentes.
- Implementación: puesta en marcha del proyecto en fase beta.
- Pruebas: ejecución de unas pruebas bien definidas para comprobar el funcionamiento del software.

2.4.3.4 Despliegue y Cierre

- Memoria: realización del manual de usuario y de una memoria sobre la realización del proyecto.
- Informe sobre el desarrollo: reunión del equipo para registrar el conocimiento adquirido durante la realización del proyecto sobre los problemas surgidos y la manera de afrontarlos y superarlos, para generar un documento de guía para futuros casos.

2.4.4 Productos intermedios

- Estudio actual del mercado: documento de estudio sobre el estado actual de la contaminación producida por los vehículos, las restricciones en la normativa de circulación debido a la polución y las herramientas de las que disponen los usuarios para compartir vehículo para ir al trabajo, como el transporte público o las diferentes aplicaciones específicas.
- Planificación: documento de planificación para el desarrollo del proyecto.
- Modelo de datos: diseño y creación de la base de datos que dará soporte a la aplicación.
- Diseño de la aplicación: desarrollo e implementación de la aplicación.
- Despliegue y cierre del proyecto: análisis y recogida de datos sobre los problemas del proyecto.

2.5 ORGANIZACIÓN

2.5.1 Esquema organizativo

- Director de proyecto: es el encargado de controlar y gestionar el buen funcionamiento y coordinación del grupo de trabajo, así como, orientar en el desarrollo de proyecto y definir las tareas que se deberán realizar. Por último, deberá aprobar las entregas realizadas en cada iteración.
- Equipo de trabajo: grupo multidisciplinar encargado del desarrollo e implementación del proyecto con las funcionalidades acordadas y siguiendo la planificación previa.

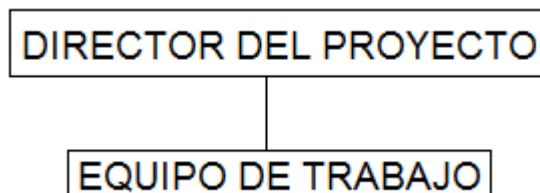


Ilustración 2.3. Organización

- Reuniones de Seguimiento: se celebrarán cada dos semanas al mes y en las que se analizará, basándose en los Informes de Seguimiento, la marcha del proyecto, detectando los posibles problemas que puedan surgir en su evolución y planteando alternativas de solución a los mismos. La celebración de dichas reuniones es competencia del Director del Proyecto.
- Hitos de Validación: tras la primera reunión de Lanzamiento del Proyecto, el director validará la finalización de cada una de las fases del proyecto en la medida que se cumplimenten todas y cada una de sus actividades. El director dispondrá de cinco días laborables para proporcionar un informe de validación, con la conformidad o no del producto. Este objetivo es fácilmente asumible ya que la implantación es progresiva y se realiza con la directa participación de los usuarios implicados. La no-formalización de este informe de validación en el plazo indicado dará por validada la fase.
- Recepción: se considera como recepción al último hito de validación, y se corresponde con el final del período de seguimiento de la explotación.

2.5.2 Plan de recursos humanos

El equipo de trabajo idóneo estaría formado por los siguientes perfiles:

- Director del proyecto: es el encargado de organizar el proyecto, coordinar al equipo de desarrollo y controlar que se cumpla la planificación.
- Administrador de BD: su función es la de diseñar, programar y mantener la base de datos del proyecto.
- Programador Android: su función es el diseño y desarrollo de la aplicación móvil.
- Diseñador gráfico: es el encargado del diseño y creación de las interfaces gráficas sencillas y fáciles para el usuario.
- Programador del servidor: su función es el diseño y desarrollo del servidor de la aplicación.

2.6 CONDICIONES DE EJECUCIÓN

2.6.1 Entorno de trabajo

El lugar de trabajo será el hogar del director del proyecto.

El horario será variable debido a la situación actual del desarrollador del proyecto, estudiante; dependiendo de las tareas que tenga que realizar para otras asignaturas. Eso provocará que se tenga que trabajar tanto en días laborales, como fin de semanas y festivos.

Los medios informáticos de los que se disponen son los siguientes:

- HARDWARE
 - Ordenador portátil Sony Vaio
 - Smartphone Sony Xperia.
- SOFTWARE
 - Android Studio
 - Microsoft Project
 - Sublime Text

El director del proyecto es responsable de la seguridad de los módulos de software, datos y documentación presentes en sus locales, incluso durante la fase de desarrollo.

2.6.2 Control de cambios

Todas las peticiones formuladas durante la ejecución del proyecto, que impliquen cambios en las especificaciones, diseños o desarrollos ya realizados, o en el entorno de desarrollo que se describe en el siguiente procedimiento:

1. Comunicación formal al director del proyecto sobre las modificaciones solicitadas.
2. Valoración por el equipo del proyecto de las repercusiones, tanto técnicas como de planificación
3. Presentación de una oferta valorada.
4. Aprobación o rechazo de la oferta.
5. En caso afirmativo, modificación del plan de trabajo y del presupuesto.

2.6.3 Recepción de los productos

Para la recepción de cada producto software, que constituyan todo o parte del conjunto a desarrollar, se definirán previamente las pruebas que se consideren necesarias. Si se obtienen unos resultados satisfactorios en dichas pruebas, los productos se entregarán al director del proyecto para su implementación. En caso negativo, se comunicarán dichos fallos al equipo de trabajo en un plazo máximo de cinco días laborables para su análisis y corrección.

Los documentos tales como estudios, diseños, especificaciones funcionales, documentación, análisis, etc. deberán ser entregados y aprobados por el director del proyecto.

El director dispondrá de cinco días laborables para comunicar formalmente su aceptación o rechazo del módulo. Si en el plazo antes citado no se recibe respuesta alguna, los módulos se considerarán recibidos.

2.7 PLANIFICACIÓN

En este apartado se detallan las tareas y la planificación del proyecto.

2.7.1 Plan de trabajo

En la siguiente tabla, se muestran las tareas a realizar especificadas anteriormente, el recurso asignado a cada tarea y las fechas de comienzo y fin, así como el orden de precedencia de las tareas.

Nombre de tarea	Duración	Comienzo	Fin	Predecesoras	Nombres de los recursos
1 Lanzamiento del proyecto	1 día	mar 08/11/16	mar 08/11/16		Andrés (Director de proyecto)
2 Estudio de aplicaciones similares	3 días	mié 09/11/16	vie 11/11/16	1	Andrés (Director de proyecto)
3 Análisis de requisitos	2 días	sáb 12/11/16	dom 13/11/16	2	Andrés (Director de proyecto)
4 Análisis de herramientas para el desarrollo de la aplicación móvil	1 día	lun 14/11/16	lun 14/11/16	3	Andrés (Desarrollador Android)
5 Análisis de herramientas del servidor	2 días	mar 15/11/16	mié 16/11/16	4	Andrés (Programador web)
6 Análisis de herramientas de bases de datos	2 días	jue 17/11/16	vie 18/11/16	5	Andrés (Administrador de BD)
7 Identificación de las clases	1 día	sáb 19/11/16	sáb 19/11/16	6	Andrés (Administrador de BD)
8 Desarrollo de la aplicación	66 días	dom 20/11/16	vie 02/06/17	7	Andrés (Desarrollador Android)
9 Desarrollo del servidor	66 días	dom 20/11/16	vie 02/06/17	7	Andrés (Programador web)
10 Desarrollo de la interfaz de usuario	66 días	dom 20/11/16	vie 02/06/17	7	Andrés (Diseñador gráfico)
11 Pruebas	37 días	lun 19/12/16	dom 04/06/17	7	Andrés (Programador web);Andrés (Desarrollador Android)
12 Implementación	1 día	lun 05/06/17	lun 05/06/17	11;10;9;8	Andrés (Desarrollador Android);Andrés (Programador web)
13 Memoria	3 días	mar 06/06/17	jue 08/06/17	12;28	Andrés (Director de proyecto)
14 Informe sobre el desarrollo	1 día	vie 09/06/17	vie 09/06/17	13	Andrés (Director de proyecto)
15 Seguimiento	1 día	lun 21/11/16	lun 21/11/16	1	Todos
16 Seguimiento	1 día	lun 05/12/16	lun 05/12/16	15	Todos
17 Seguimiento	1 día	lun 19/12/16	lun 19/12/16	16	Todos
18 Seguimiento	1 día	lun 02/01/17	lun 02/01/17	17	Todos
19 Seguimiento	1 día	lun 16/01/17	lun 16/01/17	18	Todos
20 Seguimiento	1 día	lun 30/01/17	lun 30/01/17	19	Todos
21 Seguimiento	1 día	lun 13/02/17	lun 13/02/17	20	Todos
22 Seguimiento	1 día	lun 27/02/17	lun 27/02/17	21	Todos
23 Seguimiento	1 día	lun 13/03/17	lun 13/03/17	22	Todos
24 Seguimiento	1 día	lun 27/03/17	lun 27/03/17	23	Todos
25 Seguimiento	1 día	lun 10/04/17	lun 10/04/17	24	Todos
26 Seguimiento	1 día	lun 24/04/17	lun 24/04/17	25	Todos
27 Seguimiento	1 día	lun 08/05/17	lun 08/05/17	26	Todos
28 Seguimiento	1 día	lun 22/05/17	lun 22/05/17	27	Todos

Ilustración 2.4 Plan de trabajo

2.7.2 Diagrama de precedencias

En el siguiente gráfico, se muestra el orden en el que se realizarán las tareas

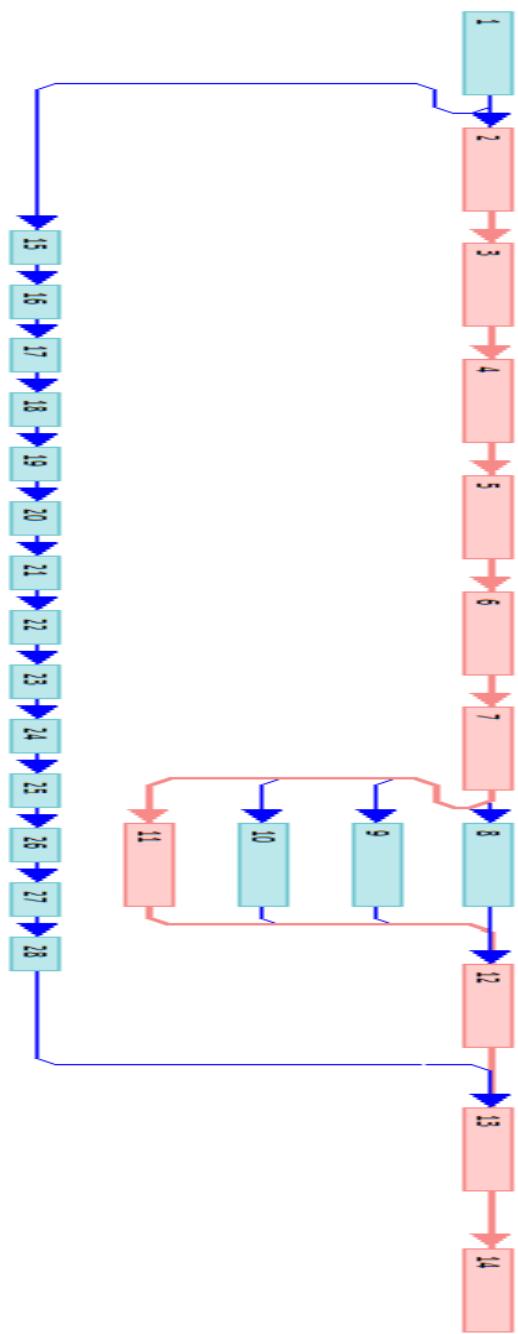


Ilustración 2.5 Diagrama de Precedencias

2.7.3 Diagrama de Gantt

En este apartado se muestra el diagrama de Gantt

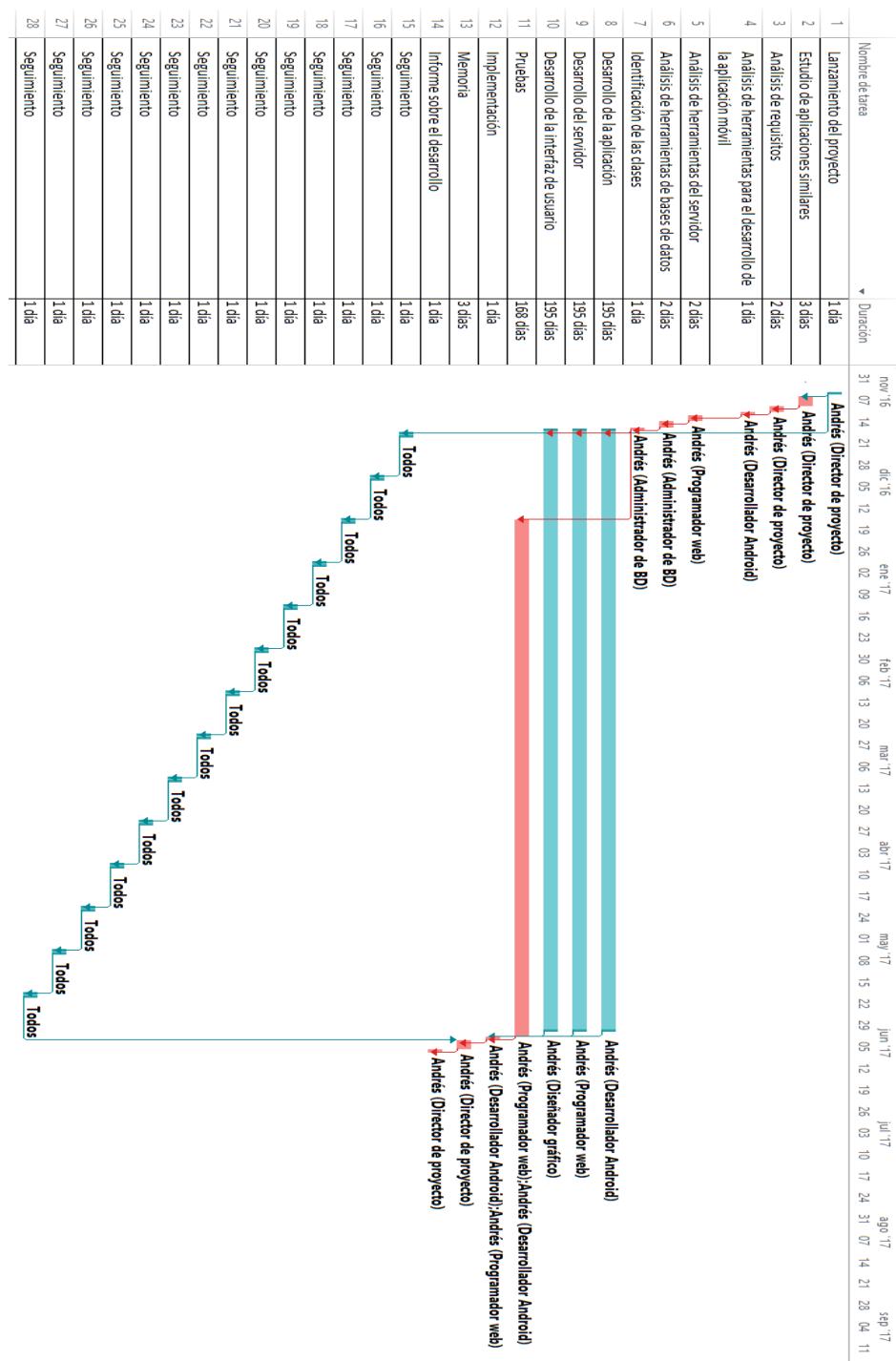


Ilustración 2.6 Diagrama de Gantt

2. DEFINICIÓN Y OBJETIVOS DEL PROYECTO

2.7.4 Reparto de cargas de trabajo por recurso humano del proyecto

Nombre del recurso	Trabajo	07	14	21	28	05	12	19	26	02	09	16	23	30	06	13	20	27	03	10	17	24	01	08	15	22	29	05	11
Andrés (Director de proyecto)	30 horas	18h																											12h
Lanzamiento del proyecto	3 horas	3h																											
Estudio de aplicaciones similares	9 horas	9h																											9h
Análisis de requisitos	6 horas	6h																											3h
Memoria	9 horas																												
Informe sobre el desarrollo	3 horas																												9h
Andrés (Administrador de BD)	7 horas	7h																											3h
Análisis de herramientas de bases de datos	5 horas	5h																											
Identificación de las clases	2 horas	2h																											
Andrés (Programador web)	270,5 horas	8h	9h	9h	9h	9h	9,5h	6,5h	1,5h																				
Análisis de herramientas del servidor	6 horas	6h																											
Desarrollo del servidor	251 horas	2h	9h	6h																									
Puebas	12 horas																												
Implementación	1,5 horas																												1,5h
Andrés (Desarrollador Android)	267,5 horas	5h	9h	9h	9h	9h	9h	9,5h	6,5h	1,5h																			
Análisis de herramientas para el desarrollo de la aplicación móvil	3 horas	3h																											
Desarrollo de la aplicación	251 horas	2h	9h	6h																									
Puebas	12 horas																												
Implementación	1,5 horas																												1,5h
Andrés (Diseñador gráfico)	61 horas	1h	3h	3h	3h	3h	2h																						
Desarrollo de la interfaz de usuario	61 horas	1h	3h	3h	3h	3h	2h																						
Todos	7 horas	0,5h																											

Ilustración 2.7 Cargas de trabajo

2.7.5 Cargas de trabajo para las diferentes tareas del proyecto

Nombre de tarea	Trabajo	07	14	21	28	05	12	19	26	02	09	16	23	30	06	13	20	27	03	10	17	24	01	08	15	22	29	05	11
Lanzamiento del proyecto	3 horas	3h																											
Estudio de aplicaciones similares	9 horas	9h																											
Análisis de requisitos	6 horas	6h																											
Análisis de herramientas para el desarrollo de la aplicación móvil	3 horas	3h																											
Análisis de herramientas del servidor	6 horas	6h																											
Análisis de herramientas de bases de datos	5 horas	5h																											
Identificación de las clases	2 horas	2h																											
Desarrollo de la aplicación	251 horas	2h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	6h	
Desarrollo del servidor	251 horas	2h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	9h	6h	
Desarrollo de la interfaz de usuario	61 horas	1h	3h	3h	3h	3h	2h																						
Puebas	24 horas																												
Implementación	3 horas																												3h
Memoria	9 horas																												9h
Informe sobre el desarrollo	3 horas																												3h
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas																												
Seguimiento	0,5 horas				</td																								

2.7.6 Presupuesto

Presupuesto total del proyecto, incluyendo el coste de los recursos humanos y las necesidades de hardware y software para el desarrollo de la aplicación.

Director de proyecto	30 horas	25 €/hora	750,00 €
Administrador BD	7 horas	15 €/hora	105,00 €
Programador Web	95,75 horas	15 €/hora	1.436,25 €
Desarrollador Android	92,75 horas	15 €/hora	1.391,25 €
Diseñador gráfico	23,5 horas	15 €/hora	352,50 €
	249 horas		4.035,00 €

Tabla 2.1: Coste de los recursos humanos

Ordenador portátil	1.000 €
Smartphone	500 €
Mlab	0 €
Firebase	0 €
Google CloudSQL	0 €
Google Maps	0 €
	1.500 €

Tabla 2.2: Coste del Hardware y Software

Recursos humanos	4.035 €
Hardware y Software	1.500 €
	6.158 €

Tabla 2.3: Coste total del proyecto

3. ESPECIFICACIÓN DE REQUISITOS

3.1 VISION GENERAL

En este apartado se detallan los requisitos que el proyecto desarrollado debe satisfacer y que definen el funcionamiento de todo el software del mismo. Además, se exponen los casos de usos de la aplicación y se muestran los diagramas de secuencia aquello que se consideran más complejos. Este apartado se divide en los siguientes bloques:

- Especificación de requisitos de la aplicación Android: requisitos que debe cumplir la aplicación web.
- Casos de uso

3.2 ESPECIFICACIÓN DE REQUISITOS DE LA APLICACIÓN ANDROID

Los requisitos funcionales de la aplicación son:

- RF.1** La aplicación debe permitir a los usuarios que se registren y que inicien sesión
- RF.2** La aplicación de permitir a los usuarios crear rutas
- RF.3** La aplicación debe permitir al usuario buscar las rutas que más le puedan ser útiles, en base a diferentes criterios
- RF.4** La aplicación debe permitir a los usuarios solicitar unirse a una ruta
- RF.5** La aplicación debe permitir al conductor aceptar o rechazar las solicitudes realizadas por los usuarios.
- RF.6** La aplicación debe permitir expulsar a los usuarios unidos a su ruta
- RF.7** La aplicación debe permitir a los usuarios salir de las rutas a las que se han unido
- RF.8** La aplicación debe permitir al conductor editar su ruta
- RF.9** La aplicación debe permitir al conductor eliminar su ruta.
- RF.10** La aplicación debe permitir al usuario añadir sus coches.
- RF.11** La aplicación debe destacar aquellas rutas con coche eléctrico
- RF.12** La aplicación debe permitir a los usuarios ver el perfil de sus compañeros.
- RF.13** La aplicación debe editar a los usuarios modificar su perfil.
- RF.14** La aplicación debe poder enviar mensajes entre usuarios para notificar los eventos más destacados.
- RF.15** La aplicación debe permitir a los usuarios establecer sus preferencias sobre la distancia y el tiempo máximo para encontrar una ruta.
- RF.16** La aplicación debe conectarse a Google Maps para que el usuario pueda indicar el lugar de salida y de vuelta.

Los requisitos no funcionales de la aplicación son:

- RNF.1** La aplicación debe ser intuitiva y fácil de usar
- RNF.2** La aplicación debe ser multilenguaje

3.3 CASOS DE USO

A continuación, se muestran los principales casos de uso de la aplicación:

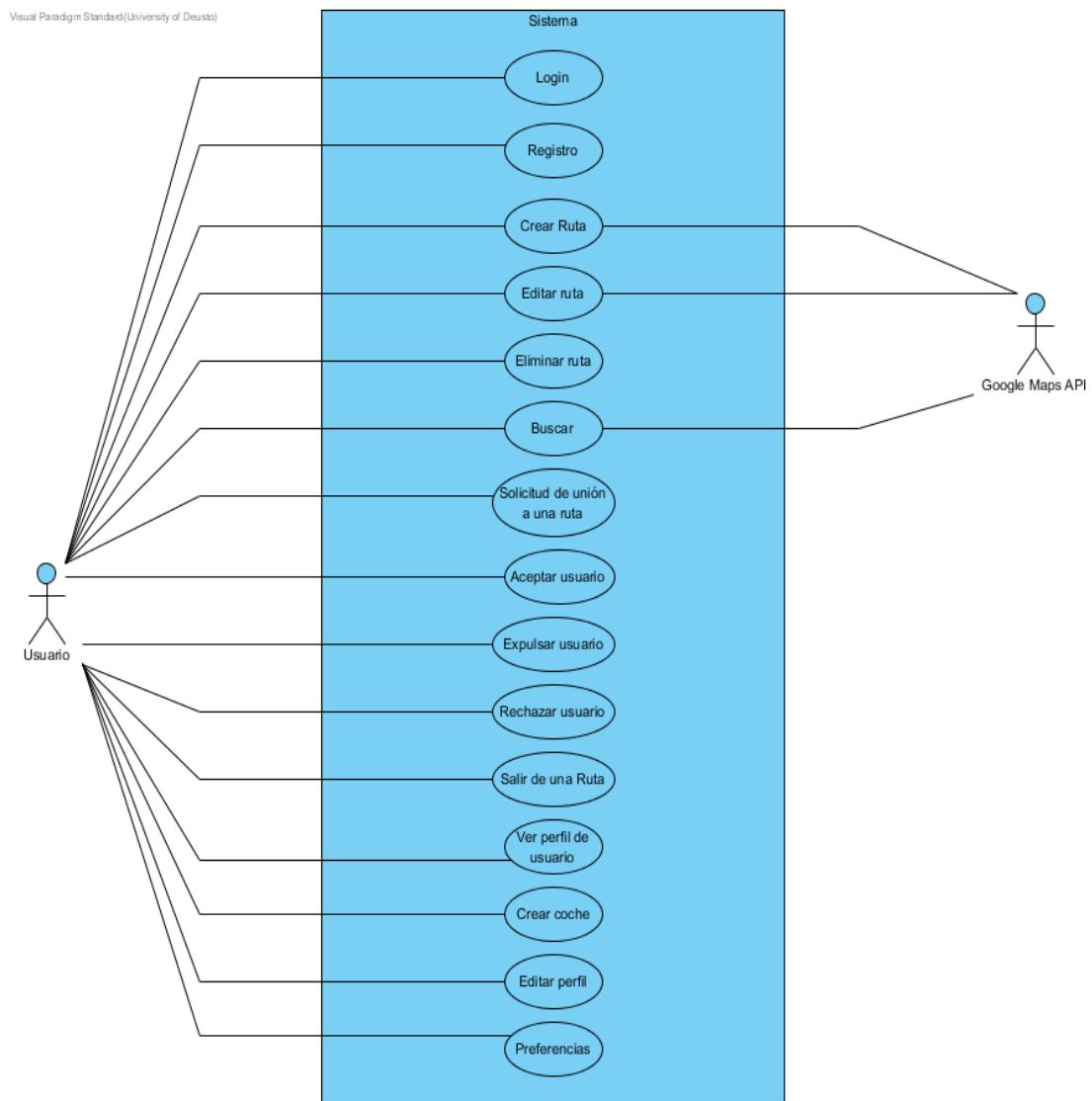


Ilustración 3.1 Casos de uso

3.3.1 Registro

Este caso de uso representa el registro inicial y el terminar registro por parte del usuario en la aplicación.

- **Nombre:** Registro
- **Propósito:** El objetivo de este caso de uso es completar todos los pasos del registro del usuario en la aplicación.
- **Actores:** El usuario
- **Pos condición:** el usuario estará registrado en la aplicación.
- Escenario principal:
 1. El usuario abre la aplicación.
 2. El usuario pulsa el botón 'Registro'.
 3. El usuario rellena el formulario.
 4. El usuario pulsar en registrar
 5. Entrar en la aplicación
- Escenarios alternativos:
 4. Volver al paso 3.

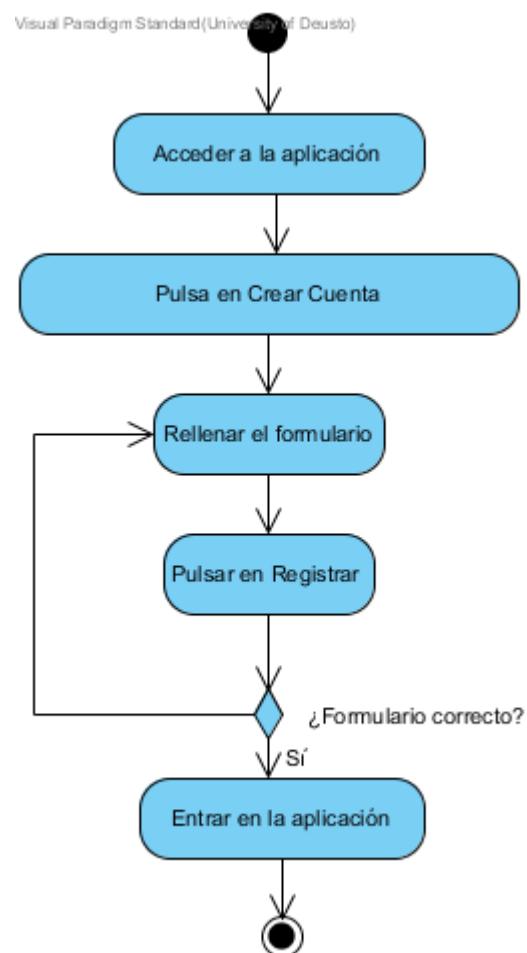


Ilustración 3.2: Registro

Para registrar un nuevo usuario en nuestra aplicación, primero, mediante la API de Firebase, se crea un usuario, con su email y su contraseña. Tras ello, se realiza una petición POST al servidor de NodeJS para crear el usuario en nuestra base de datos mlab.com.

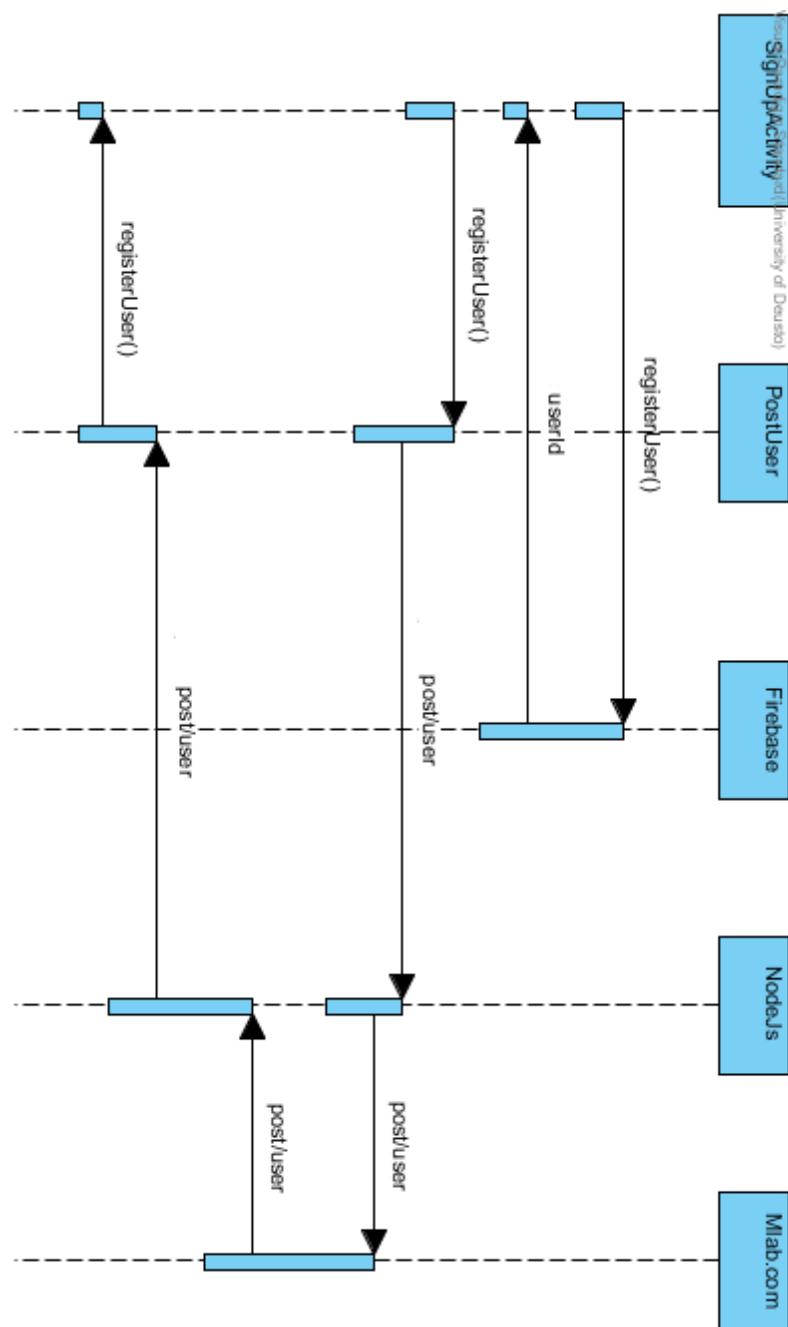


Ilustración 3.3: Secuencia del Registro

3.3.2 Añadir coche

Este caso de uso representa como añadir un coche para poder crear una ruta.

- **Nombre:** Añadir coche
- **Propósito:** El objetivo de este caso de uso es que el usuario pueda añadir su coche a la aplicación.
- **Actores:** El usuario
- **Precondición:** el usuario está logueado.
- **Pos condición:** el usuario posee al menos un coche.
- Escenario principal:
 1. El usuario accede la aplicación.
 2. El usuario pulsa el botón 'Mis coches'.
 3. El pulsa en 'Añadir coche'
 4. El usuario rellena el formulario.
 5. El usuario pulsar en 'Añadir'
- Escenarios alternativos:
 5. Volver al paso 4.

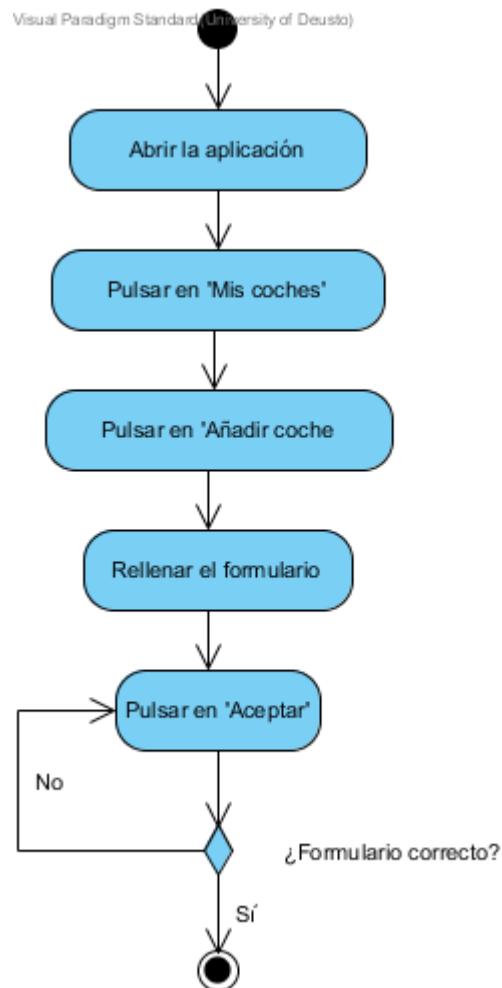


Ilustración 3.4: Añadir coche

3.3.3 Crear ruta

Este caso de uso representa al usuario creando una ruta

- **Nombre:** Unirse a una ruta
- **Propósito:** El objetivo de este caso de uso es permitir al usuario crear una ruta
- **Actores:** El usuario
- **Precondición:** el usuario está logueado.
- **Pos condición:** se crea una ruta nueva.
- **Escenario principal:**
 1. El usuario pulsa el botón 'Crear ruta'.
 2. El usuario rellena el formulario.
 3. El usuario pulsa 'Aceptar'
 4. Se redirige al detalle de la ruta.
- **Escenarios alternativos:**
 1. Si el usuario no ha añadido ningún coche, se finaliza.
 3. Si el formulario no está correcto, se vuelve al paso 2.

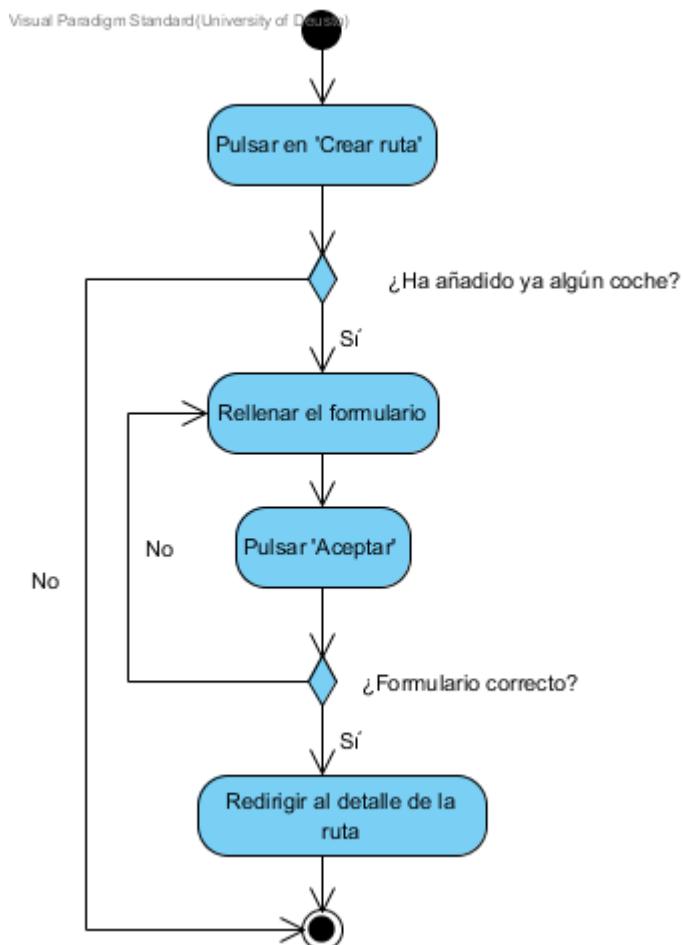


Ilustración 3.5: Crear ruta

Para crear una ruta, se crea una petición POST desde la Activity de crearRuta que se envía al servidor NodeJS y este se encarga de guardarla en la base de datos en mlab.com. Si la petición se realiza satisfactoriamente, se guarda la ruta también en el servidor de SQL.

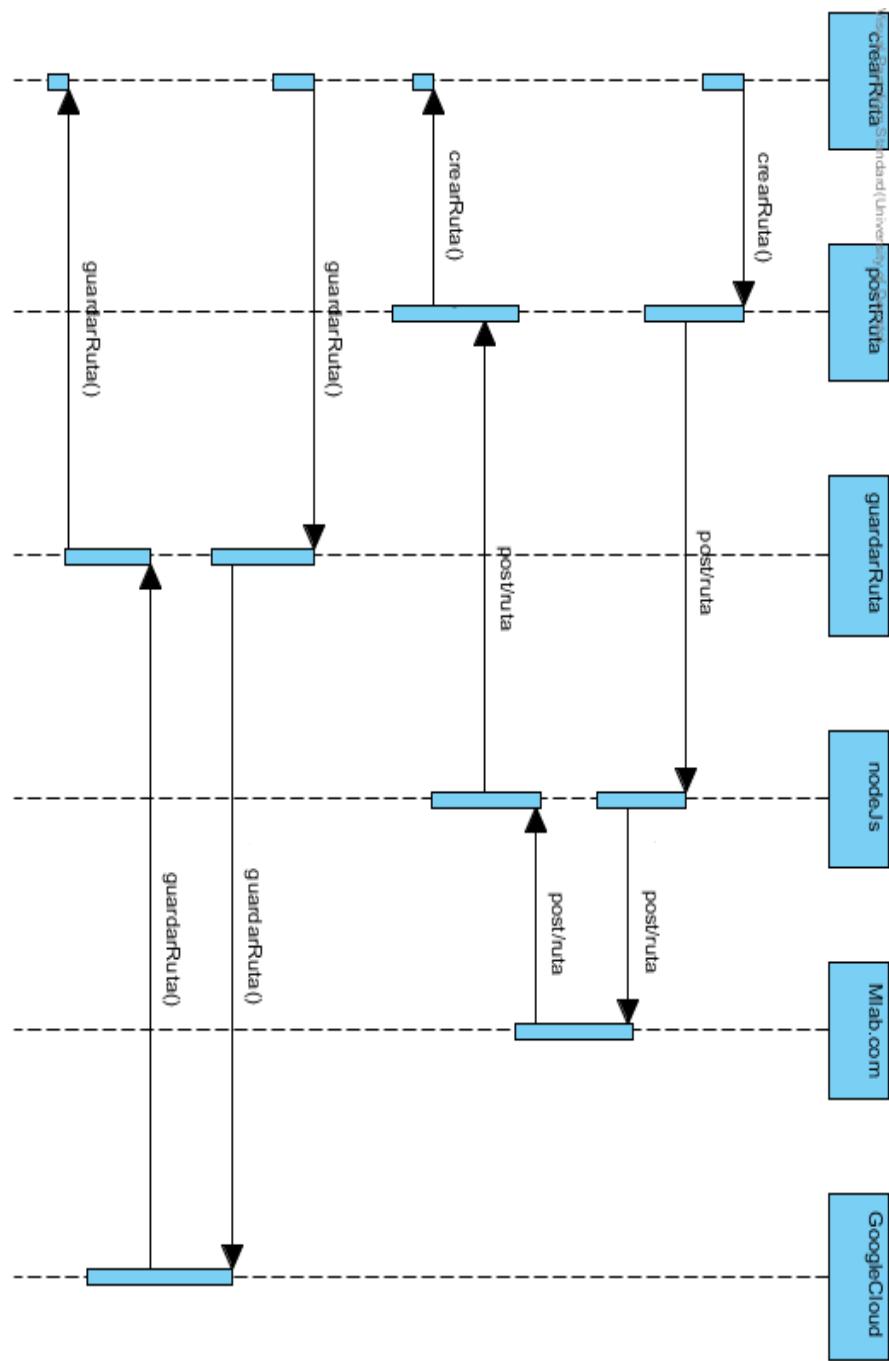


Ilustración 3.6: Secuencia de crear ruta

3.3.4 Unirse a una ruta

Este caso de uso representa al usuario uniéndose a una ruta ya creada

- **Nombre:** Unirse a una ruta
- **Propósito:** El objetivo de este caso de uso es permitir al usuario unirse a una ruta ya existente.
- **Actores:** El usuario
- **Precondición:** el usuario está logueado.
- **Pos condición:** el usuario queda pendiente de ser aceptado en una ruta.
- **Escenario principal:**
 1. El usuario pulsa el botón 'Buscar ruta'.
 2. El usuario rellena los campos disponibles
 3. El usuario pulsa en el botón 'Buscar'.
 4. Mostrar las rutas disponibles al usuario.
 5. El usuario selecciona una ruta.
 6. El usuario pulsa el botón 'Unirme'.
 7. Se envía una notificación al conductor
 8. Se decrementa en uno el número de pasajeros
- **Escenarios alternativos:**
 3. Volver al paso 2 si el formulario no está correcto.
 4. Si no existen rutas, se vuelve al paso 2.
 5. Si la ruta no es acorde a las necesidades del usuario, vuelve atrás.

VISU Paradigm Standard (University of Deusto)

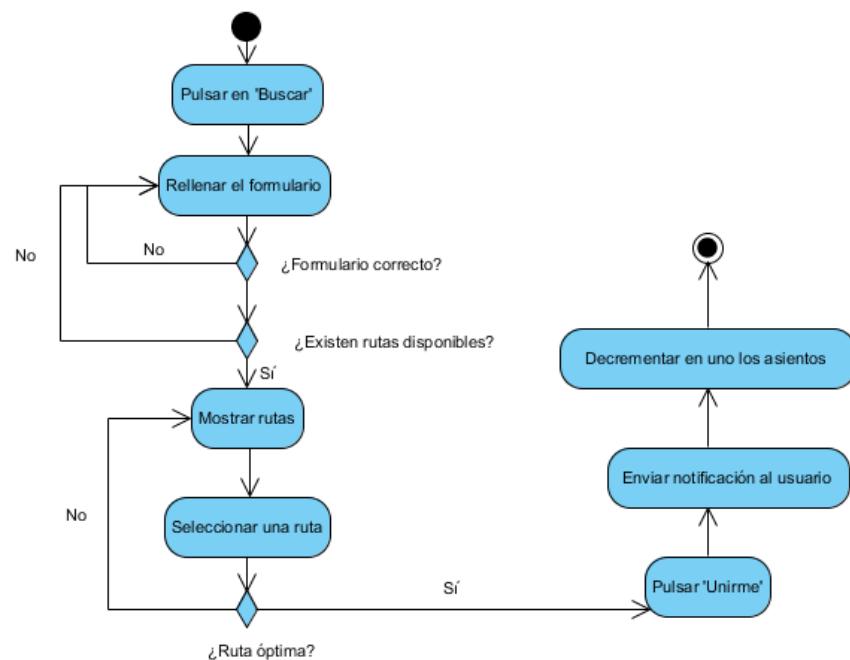


Ilustración 3.7: Unirse a una ruta

4. TECNOLOGÍAS UTILIZADAS

4.1 VISIÓN GENERAL

En este capítulo se exponen las principales tecnologías utilizadas para el desarrollo de la aplicación y su uso dentro del proyecto.

4.2 ANDROID



Ilustración 4.1: Android

[21] Android es un sistema operativo basado en el núcleo Linux. Fue diseñado principalmente para dispositivos móviles con pantalla táctil, como teléfonos inteligentes, tablets o teléfonos; y también para relojes inteligentes, televisores y automóviles. Inicialmente fue desarrollado por Android Inc., empresa que Google respaldó económicamente y más tarde, en 2005, compró. Las ventas de los dispositivos basados en Android son superiores a las ventas combinadas de Windows Phone e IOS.

Android es la primera plataforma móvil completa, abierta y gratuita, y está basado en el lenguaje de programación Java:

- Completa: Android es una plataforma móvil robusta, segura y fácilmente actualizable, con un framework completo. También proporciona programas de compatibilidad y certificación, por lo que los fabricantes pueden diseñar dispositivos altamente compatibles.
- Abierta: Android ha sido desarrollado y proporcionado bajo los términos de licencia de código abierto de Apache. Los desarrolladores tienen pleno acceso a las características y servicios del dispositivo mientras desarrollan aplicaciones.
- Libre: Android no cobra ninguna licencia, royalty o membresía para el desarrollo de aplicaciones en la plataforma. El código fuente de Android y sus kits de desarrollo se proporcionan de forma gratuita. La plataforma de desarrollo está disponible en muchos sistemas operativos de escritorio, permitiendo a los desarrolladores de aplicaciones desarrollar las aplicaciones usando el sistema operativo de su elección.

[22] Los componentes principales de Android son:

- Activity: representan el componente principal de la interfaz gráfica de una aplicación Android.
- View: son los componentes básicos con los que se construye la interfaz gráfica de la aplicación. De inicio, Android pone a nuestra disposición una gran cantidad de controles básicos, como cuadros de texto, botones, listas desplegables o imágenes.

- Service: son componentes sin interfaz gráfica que se ejecutan en segundo plano. En concepto, son similares a los servicios presentes en cualquier otro sistema operativo. Los servicios pueden realizar cualquier tipo de acciones, por ejemplo, actualizar datos, lanzar notificaciones, o incluso mostrar elementos visuales (p.ej. actividades) si se necesita en algún momento la interacción con del usuario
- Content Provider: es el mecanismo que se ha definido en Android para compartir datos entre aplicaciones.
- Widget: elementos visuales, normalmente interactivos, que pueden mostrarse en la pantalla principal del dispositivo Android y recibir actualizaciones periódicas.
- Intent: es el elemento básico de comunicación entre los distintos componentes Android.

4.2.1 Razón de uso

Como ya se ha explicado anteriormente, se escogió Android como base para el proyecto, debido a que está basado en Java y tenía ciertos conocimientos previos, y a su gran popularidad y el número de usuarios potenciales.

4.3 JAVA

[23] Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo (conocido en inglés como WORA, o "write once, run anywhere"), lo que quiere decir que el código que es ejecutado en una plataforma no tiene que ser recompilado para correr en otra. Java es uno de los lenguajes de programación más populares en uso. Tiene una sintaxis muy similar a C y C++, con el defecto de que no tiene un acceso directo a la programación de bajo nivel como la gestión de memoria

Las características principales son:

- Simple: basado en el lenguaje C++, pero donde se eliminan muchas de las características, creaban problemas frecuentes a los programadores.
- Orientado a objetos: Java es un lenguaje orientado a objetos puro. Esto significa que en un programa Java, todo es un objeto y todo desciende de una clase del objeto raíz.
- Independencia de la plataforma: significa que programas escritos en el lenguaje Java pueden ejecutarse igualmente en cualquier tipo de hardware. Este es el significado de ser capaz de escribir un programa una vez y que pueda ejecutarse en cualquier dispositivo. Para ello, se compila el código fuente escrito en lenguaje Java, para generar un código conocido como "bytecode". El bytecode es ejecutado entonces en la máquina virtual (JVM), un programa escrito en código nativo de la plataforma destino, que interpreta y ejecuta el código.
- Recolector de basura: el entorno en tiempo de ejecución de Java (Java runtime) es el responsable de gestionar el ciclo de vida de los objetos. El programa, u otros objetos pueden tener localizado un objeto mediante una referencia a éste. Cuando no quedan referencias a un objeto, el recolector de basura de Java borra el objeto, liberando así la memoria
- Applet Interface: además de poder crear aplicaciones independientes, los desarrolladores de Java pueden crear programas que se pueden descargar de una página web y ejecutarse en un navegador de cliente.

4.4 FIREBASE



Ilustración 4.2: Firebase

[24] Firebase es un proveedor de servicios en la nube y backend como servicio. Sus características fundamentales están divididas en varios grupos, las cuales podemos agrupar en:

- Analytics: Firebase Analytics es una solución de medición gratuita que proporciona información sobre el uso de las aplicaciones y el compromiso del usuario. Permite comprender la manera en que las personas usan la app de iOS o Android. El SDK captura de manera automática un número de eventos y propiedades, y también permite a los desarrolladores definir sus propios eventos personalizados a fin de medir lo que únicamente importa para tu negocio. Una vez capturados, los datos quedan disponibles en un panel a través de la consola de Firebase. Este panel proporciona información detallada sobre los datos; desde datos de resumen, como usuarios activos e información demográfica, hasta datos más detallados, como la identificación de tus elementos más comprados. Firebase Analytics te permite comprender el comportamiento de los usuarios, para tomar decisiones fundamentadas sobre la comercialización de tu app.
- Autenticación: Firebase Auth es un servicio que puede autenticar a los usuarios utilizando sólo código en el cliente. Es compatible con los principales proveedores de servicios de login como, Facebook, GitHub, Twitter y Google.
- RealTime Database: Firebase proporciona una base de datos en tiempo real alojada en la nube. Los datos se almacenan en formato JSON y se sincronizan en tiempo real con cada cliente conectado. El servicio proporciona a los desarrolladores una API que permite sincronizar los datos de las aplicaciones entre los clientes y almacenarlos en la nube de Firebase.
- Hosting: Firebase permite almacenar y recuperar contenido generado por los usuarios, como imágenes, audio y video, directamente desde el SDK del cliente de Firebase.
- AdMob: Firebase facilita la monetización de la aplicación, permitiendo a los desarrolladores añadir AdMob, el servicio de publicidad en línea de Google, que muestra a los usuarios anuncios en diferentes formatos (vídeos, banners) mientras navegan por la aplicación.

4.4.1 Razón de uso

Se decidió utilizar Firebase en el desarrollo de la aplicación porque ofrece una base de datos en la nube fácil de integrar en un proyecto Android y facilita la autenticación de los usuarios gestionando las contraseñas y los registros. Además, Firebase ofrece una versión gratuita del servicio, que cubre actualmente todos los requisitos para el funcionamiento de la aplicación y, en caso de un aumento del número de usuarios, facilita la escalabilidad y permite acceder al plan que mejor se adapte las necesidades de la aplicación en cada momento.

4.5 NODEJS



Ilustración 4.3: NodeJS

[25] NodeJS es un entorno de ejecución en JavaScript del lado del servidor, multiplataforma, de código abierto, asíncrono, con I/O de datos en una arquitectura orientada a eventos. Está basado en el motor V8 de Google, lo que permite a NodeJS proporcionar un entorno de ejecución del lado del servidor que compila y ejecuta Javascript a velocidades increíbles. El aumento de velocidad es importante debido a que V8 compila Javascript en código de máquina nativo, en lugar de interpretarlo o ejecutarlo como bytecode. Fue creado con el enfoque de ser útil en la creación de programas de red altamente escalables, como, por ejemplo, servidores web.

[26] Aunque Javascript tradicionalmente ha sido relegado a realizar tareas menores en el navegador, es actualmente un lenguaje de programación completo, tan capaz como cualquier otro lenguaje tradicional como C++, Ruby o Java. Además, Javascript tiene la ventaja de poseer un excelente modelo de eventos, ideal para la programación asíncrona. Javascript también es un lenguaje omnipresente, conocido por millones de desarrolladores. Esto reduce la curva de aprendizaje de NodeJS, ya que la mayoría de los desarrolladores no tendrán que aprender un nuevo lenguaje para empezar a construir aplicaciones usando NodeJS.

NodeJS funciona con un modelo de evaluación de un único hilo de ejecución, usando entradas y salidas asíncronas las cuales pueden ejecutarse concurrentemente en un número de hasta cientos de miles sin incurrir en costos asociados al cambio de contexto. Este diseño de compartir un único hilo de ejecución entre todas las solicitudes atiende a necesidades de aplicaciones altamente concurrentes, en el que toda operación que realice entradas y salidas debe tener una función callback.

4.5.1 Express

[26] Express, es un framework para aplicación web para NodeJS, publicado como software libre y de código abierto bajo la Licencia MIT. Está diseñado para el desarrollo de aplicaciones web y API. Es el framework de servidor estándar de facto para NodeJS

4.5.2 Mongoose

[27] Se trata de un ODM (Object Data Mapper) en JavaScript diseñado para aplicaciones que usen MongoDB sobre NodeJS. Posee un API intuitivo para definir los modelos de datos y el almacenamiento asíncrono. Además, mediante la creación de Schemas permite definir una estructura en colecciones dinámicas que no posee ninguna estructura.

4.5.3 Razón de uso

NodeJS se utiliza en el proyecto como servidor web para conectarse a la base de datos que se encuentra en www.mlab.com. Se decidió utilizar esta tecnología debido a su gran popularidad y a la facilidad para crear una API Rest para conectarse con la base de datos.

4.6 JAVASCRIPT

[28] JavaScript es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico. Se utiliza principalmente en su forma del lado del cliente (client-side), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas.

JavaScript se diseñó con una sintaxis similar al C, aunque adopta nombres y convenciones del lenguaje de programación Java. Sin embargo, Java y JavaScript no están relacionados y tienen semánticas y propósitos diferentes. Todos los navegadores modernos interpretan el código

JavaScript integrado en las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del Document Object Model (DOM).

4.7 MONGODB



Ilustración 4.4: MongoDB

[29] MongoDB (de la palabra en inglés “humongous” que significa enorme) es un sistema de base de datos [30] NoSQL orientado a documentos, desarrollado bajo el concepto de código abierto. En lugar de guardar los datos en tablas como se hace en las bases de datos relacionales, MongoDB guarda las estructuras de datos en documentos similares a JSON con un esquema dinámico (MongoDB utiliza una especificación llamada BSON), haciendo que la integración de los datos en ciertas aplicaciones sea más fácil y rápida.

MongoDB está formado por un conjunto de colecciones, similares a una tabla para las bases de datos relacionales. Estas colecciones están compuestas de documentos. Los Documentos son un conjunto de líneas en formato JSON. Cada línea JSON está compuesta de atributos y cada atributo está compuesto de un par clave-valor

Las características principales serían:

- Consultas Ad hoc: permite búsqueda de campos, consulta de rangos y expresiones regulares
- Indexación: cualquier campo en un documento de MongoDB puede ser indexado y es posible realizar índices secundarios, similar a las bases de datos relacionales.

- Replicación: MongoDB proporciona una alta disponibilidad gracias a un conjunto de réplicas. Un conjunto de réplicas consta de dos o más copias de los datos. Cada miembro del conjunto de réplicas puede actuar como una réplica primaria o secundaria en cualquier momento. Todas las escrituras y lecturas se realizan en la réplica primaria de forma predeterminada y las réplicas secundarias mantienen una copia de los datos. Cuando una réplica primaria falla, el conjunto de réplicas lleva a cabo automáticamente un proceso para escoger una nueva réplica primaria
- Facilidad de escalabilidad horizontal

Sin embargo, también tiene desventajas importantes:

- El tamaño de los datos en MongoDB es típicamente más alto debido a que cada documento tiene nombres de campo almacenados
- Menos flexibilidad en consultas
- Sin soporte para las transacciones
- No cumple ACID: Atomicidad, Consistencia, Aislamiento y Durabilidad.
- MongoDB bloquea la base de datos a nivel de documento ante cada operación de escritura. Sólo se podrán hacer operaciones de escritura concurrentes entre distintos documentos.

4.7.1 Razón de uso

Se ha escogido MongoDB como Base de Datos del proyecto debido a su gran popularidad actualmente para el desarrollo de aplicaciones móviles y sociales. La Base de Datos del proyecto se encuentra en <https://mlab.com/>

4.8 JSON

[31] JSON, acrónimo de JavaScript Object Notation, es un formato de texto ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript. JSON es un formato de texto que es completamente independiente del lenguaje, pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes C, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos.

JSON está constituido por dos estructuras:

- Una colección de pares de nombre/valor. En varios lenguajes esto es conocido como un objeto, registro, estructura, diccionario, tabla hash, lista de claves o un arreglo asociativo.
- Una lista ordenada de valores. En la mayoría de los lenguajes, esto se implementa como arreglos, vectores, listas o secuencias.

Estas son estructuras universales; virtualmente todos los lenguajes de programación las soportan de una forma u otra.

JSON se utiliza en el proyecto para enviar y recibir datos al servidor, tanto a Firebase como a NodeJS

4.9 GOOGLE CLOUD SQL

[32] Google Cloud SQL es un servicio de base de datos totalmente gestionado que facilita la configuración, el mantenimiento y la gestión de bases de datos MySQL en Google Cloud Platform.

Google Cloud Platform es un servicio de cloud computing de Google que ofrece alojamiento en la misma infraestructura de soporte que Google utiliza internamente para productos como Google Search y YouTube. Cloud Platform proporciona a los desarrolladores productos para crear una gama de programas desde sitios web sencillos hasta aplicaciones complejas.

Google Cloud Platform forma parte de un conjunto de servicios empresariales de Google Cloud y proporciona un conjunto modular de servicios basados en la nube con una serie de herramientas de desarrollo. Por ejemplo, alojamiento y computación, almacenamiento en la nube, almacenamiento de datos, API de traducción y API de predicción.

4.10 GOOGLE MAPS



Ilustración 4.5: Google Maps

[33] Google Maps es un servidor de aplicaciones de mapas en la web. Ofrece imágenes de mapas desplazables, así como fotografías por satélite del mundo e incluso la ruta entre diferentes ubicaciones o imágenes a pie de calle con Google Street View.

Para acceder a las funcionalidades de Google Maps e interactuar con ello, existe una API oficial. Mediante el uso de esta API, es posible: integrar el sitio de Google Maps en un sitio web externo, superponer datos específicos, obtener las coordenadas de un lugar, mostrar rutas, ...

La API de Google Maps es gratuita para uso comercial, siempre y cuando el sitio en el que se está utilizando sea accesible al público y no cobre por acceso y no genere más de 25.000 accesos al mapa al día. Los sitios que no cumplan estos requisitos pueden adquirir una licencia de Google Maps.

Google Maps se utiliza en el proyecto para que el usuario indique de la manera más precisa posible el lugar de salida y de vuelta.

5. ESPECIFICACIÓN DEL DISEÑO

5.1 VISIÓN GENERAL

El objetivo de este capítulo es describir la labor de diseño realizada para desarrollar el proyecto, así como las herramientas y entornos que han sido necesarios para su realización. Para ello, contiene los siguientes apartados:

- Entorno de desarrollo
- Arquitectura de la aplicación
- Modelo de la Base de Datos

5.2 ENTORNO DE DESARROLLO

5.2.1 Android Studio

[34] Android Studio es el IDE oficial para desarrollar aplicaciones basadas en Android. Está basado en el IDE IntelliJ IDEA y está diseñado específicamente para desarrollar en Android.

Sus ventajas son:

- Instant Run: cuando se hace clic en Run o Debug, la función Instant Run aplicará los cambios en el código y los recursos de la aplicación en ejecución. Esta interpreta de manera inteligente los cambios y a menudo los entrega sin reiniciar tu app ni volver a compilar tu APK, para que puedas ver los efectos de inmediato
- Editor de código inteligente: al ofrecer compleción avanzada de código, refactorización y análisis de código, el editor de código inteligente permite escribir un código más eficaz, trabajar más rápido y ser más productivo.
- Un emulador rápido y cargado de funciones: Android Emulator se instala e inicia las apps más rápidamente que en un dispositivo real. También permite crear prototipos de la app y probarlos en todas las configuraciones de dispositivos Android: teléfonos, tablets y dispositivos Android Wear y Android TV. Además, se pueden simular varias funciones del hardware, como la localización por GPS, la latencia de red o las funciones multitáctiles.
- Sistema de compilación sólido y flexible: ofrece automatización de compilaciones, administración de dependencias y configuración de compilaciones personalizables.
- Diseñado para equipos: Android Studio se integra con herramientas de control de versión, como GitHub y Subversion, para poder mantener al equipo actualizado respecto a los cambios que se produzcan en los proyectos.
- Herramientas y frameworks de prueba: Android Studio proporciona una gran cantidad de herramientas y frameworks para facilitar las pruebas de las apps.
- Plantillas de código y apps de ejemplo: en Android Studio se incluyen plantillas de proyectos y código que facilitan la creación de algunos elementos, como el panel lateral de navegación y un paginador de vistas.

5.2.2 GitHub



Ilustración 5.1: GitHub

[35] Git es un sistema de control distribuido con énfasis en la velocidad. Este sistema de control de versiones es muy popular en proyectos open source, y su autor original fue Linus Torvald. Al igual que la mayoría de sistemas de control de revisiones distribuidos, cada directorio de trabajo Git es un repositorio con la historia completa y las plena capacidades de un sistema completo de seguimiento de versiones, independientemente del acceso a la red o de la existencia de un servidor central.

Su principal ventaja es su gran rendimiento, ya que, al tratarse de un sistema distribuido, las búsquedas son mucho más eficaces lo que supone una gran rapidez para detectar diferencias entre archivos. Además, su sistema de ramas o branches es muy flexible, pudiendo crear ramas locales independientes de un repositorio centralizado.

GitHub es un servicio de alojamiento repositorio Git basado en la web GitHub, que ofrece todas las funcionalidades de Git sobre el control de revisión distribuida y la gestión de código fuente, así como añade sus propias características. A diferencia de Git, GitHub ofrece una interfaz gráfica, tanto en la web como en el escritorio, con integración para móviles. También proporciona control de acceso y varias características de colaboración como wikis, gestión de tareas y seguimiento de errores y peticiones para cada proyecto.

5.2.3 Notepad++



Ilustración 5.2: Notepad

[36] Notepad++ es un editor de texto y de código fuente libre con soporte para varios lenguajes de programación. Incluye opciones más avanzadas que pueden ser útiles para usuarios avanzados como desarrolladores y programadores. Se distribuye bajo los términos de la Licencia Pública General de GNU. Soporta resaltado de sintaxis y plegado de código de más de 50 lenguajes de programación, scripting, y marcado. Trata de detectar automáticamente el lenguaje que utiliza un fichero determinado, utilizando una lista modificable de extensiones de archivo. El programa también soporta autocompletado para un subconjunto de la API de algunos lenguajes de programación.

5.2.4 Sublime Text



Ilustración 5.3: SublimeText 3

[37] Sublime Text es un editor de texto y código, y está cargado de funcionalidades útiles y cómodas desde el punto de la usabilidad y eficiencia. Las características más importantes del editor son:

- Da soporte nativo para 43 lenguajes de programación y texto plano.
- Auto completado
- Marcado de llaves
- Resalta las expresiones propias de la sintaxis del lenguaje, facilitando su lectura.
- Permite la edición simultánea, lo que facilita los cambios de nombre de una variable

5.2.5 TortoiseGit

[38] TortoiseGit es una interfaz del Shell de Windows para Git, basado en TortoiseSVN. Es de código abierto y está bajo la licencia de GNU General Public License.

Puesto que no está integrado en un IDE específico como Visual Studio, Eclipse u otro, puede ser utilizarlo con las herramientas de desarrollo el usuario desee y con cualquier tipo de archivo. En el Explorador de Windows, además de mostrar los comandos de Git en el menú contextual, TortoiseGit muestra iconos superpuestos en los archivos, para indicar el estado de los archivos de trabajo en Git.

5.2.6 Visual Paradigms

[39] Visual Paradigms for UML es una herramienta CASE que soporta el modelado mediante UML y proporciona asistencia a los ingenieros de software y desarrolladores, durante todos los pasos del Ciclo de Vida de desarrollo de un Software. Además del soporte para el modelado, proporciona la generación de informes y es capaz de generar código automático.

En este proyecto se ha utilizado para generar el modelo de la base de datos, la arquitectura de la aplicación y los casos de uso.

5.2.7 Microsoft Project

[40] Microsoft Project (o MSP) es un software de administración de proyectos diseñado, desarrollado y comercializado por Microsoft para asistir a administradores de proyectos en el desarrollo de planes, asignación de recursos a tareas, dar seguimiento al progreso, administrar presupuesto y analizar cargas de trabajo.

El software Microsoft Office Project en todas sus versiones es útil para la gestión de proyectos, aplicando procedimientos descritos en el PMBoK del Project Management Institute.

5.3 MODELO DE LA BASE DE DATOS

En el proyecto se ha utilizado la base de datos MongoDB, alojada en mlab.com.

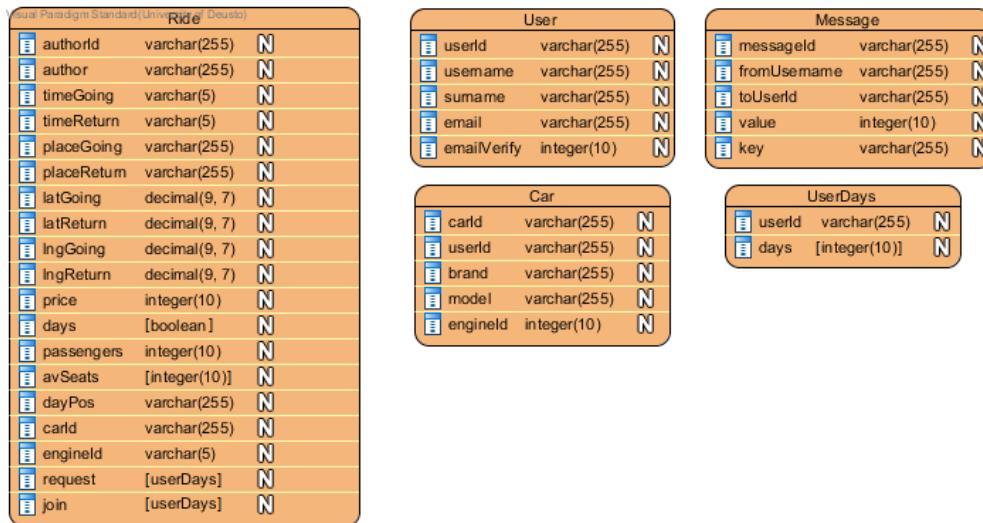


Ilustración 5.4: Base de datos de la aplicación

5.3.1 User

En la colección User se guarda todo lo relacionado con los usuarios de la aplicación. Sin embargo, no incluye la contraseña porque es Firebase la que se encarga del registro y logueo de los usuarios.

5.3.2 Ride

En la colección Ride se almacenan las rutas creadas por los usuarios. Para ello, el usuario debe llenar un formulario. El nombre de los lugares y las coordenadas, se obtienen desde Google Maps y las horas, a través de un reloj.

5.3.3 Car

En la colección Car se guarda la información relacionada con el coche, como la marca, el modelo y el tipo de motor.

5.3.4 Messaging

En la colección Messaging, se guardan de manera temporal las notificaciones entre usuarios, de manera que si el receptor está desconectado cuándo se la envían, pueda recibirla al iniciar la aplicación.

5.3.5 UserDays

En la colección UserDays se guarda la información relacionada con las solicitudes y uniones de un usuario a una ruta, y los días que este ha seleccionado.

5.4 ARQUITECTURA DE LA APLICACIÓN

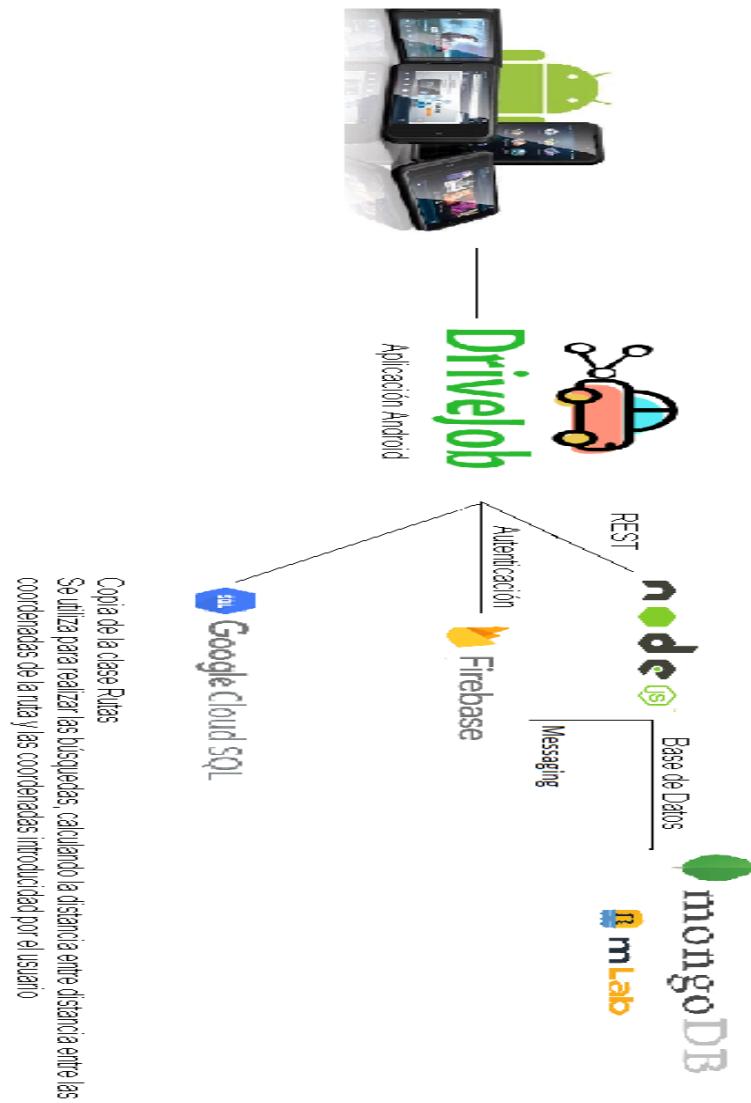


Ilustración 5.5: Arquitectura de la aplicación

Nuestra aplicación es compatible con los dispositivos Android, desde donde los usuarios acceden a nuestros servicios. Para realizar el registro y la autenticación de los usuarios en nuestra aplicación, se utiliza Firebase y, de esta manera, se reducen los riesgos por el robo de cuentas.

Además, como nuestra aplicación trata de conectar a los usuarios, la base de datos ha de estar disponible en un servidor online, en este caso, mlab.com. Para acceder a ella, nuestra aplicación se conecta a través de REST a un servidor propio y este se encarga de realizar las consultas.

Para el envío de notificaciones entre los usuarios, primero se envían a un servidor NodeJS, con el objetivo de ocultar la clave secreta proporcionada por Firebase. Después, se reenvía la información al servidor de Firebase encargado de gestionar el envío de mensajes, a través del módulo Firebase Cloud Messaging.

Por último, debido a ciertas limitaciones de NodeJS y MongoDB, para encontrar las rutas que mejor se adaptan a las necesidades del usuario, la consulta se realiza en un servidor SQL de Google Cloud Platform.

6. IMPLEMENTACIÓN

6.1 VISIÓN GENERAL

Este capítulo contiene las consideraciones más importantes sobre cómo se han implementado las tecnologías utilizadas en el desarrollo del proyecto.

6.2 IMPLEMENTACIÓN DE LA BASE DE DATOS MONGODB

Para guardar la base de datos de la aplicación, se ha decidido escoger un servicio online fácil de configurar y de acceder desde nuestro servidor. Por ello, se decidió escoger MLab (mlab.com), un servicio de base de datos en la nube totalmente administrado que aloja bases de datos MongoDB. MLab se ejecuta en los proveedores de la nube Amazon, Google y Microsoft Azure y se ha asociado con proveedores de plataforma como servicio.

Para empezar a utilizarlo, simplemente hay que registrarse en la página y crear una base de datos, seleccionando el proveedor y el lugar en el que quieras que se aloje. Tras ello, MLab te ofrece una url a la que simplemente hay que añadir el nombre de usuario y la contraseña de un usuario con permisos en la base de datos, y ya está listo para realizar consultas.

Por último, mencionar que para las necesidades de nuestra aplicación el plan gratuito se amoldaba perfectamente, pero en caso de necesitar un aumento de capacidad, MLab ofrece diferentes planes de pago.

The screenshot shows the MongoDB Compass interface. At the top, it says "Collection: rides" with a dropdown arrow. Below the header are tabs: "Documents" (which is selected and highlighted in blue), "Indexes", "Stats", and "Tools". Under the "Documents" tab, there's a button to "Delete all documents in collection" and another to "Add document". A search bar says "-- Start new search -- ▾". Below that, it says "All Documents". It shows "Display mode: list" and "records / page: 10". There is one document listed with the following data:

```

{
  "timeReturn": "12:38",
  "placeGoing": "Urkixo Zumarkalea, 46",
  "placeReturn": "Kalea Labayru, 29A",
  "latGoing": 43.2606545,
  "latReturn": 43.2564,
  "lngGoing": -2.9384879,
  "lngReturn": -2.94067
}

```

At the bottom, it says "records / page: 10" and "[1 - 1 of 1]".

Ilustración 6.1: Documento en MLab

6.3 IMPLEMENTACIÓN DEL SERVIDOR NODEJS

6.3.1 Servidor REST

El servidor en NodeJS es un simple intermediario entre la aplicación de Android y la base de datos en MLab. El servidor se conecta con la base de datos utilizando mongoose a través de la url que proporciona la página. Después, se crean los Schemas de los objetos que se desean guardar en la base datos. Por último, gracias a node-restful, solo hace falta indicar los métodos Rest con los que se podrá manipular los objetos.

```
//getModels
var Rides = require('../models/ride');

//routes
Rides.methods(['get', 'post', 'put', 'delete']);
Rides.register(router, '/ride');
```

Ilustración 6.2: Métodos REST

Después, el cliente Android se conecta al servidor a través de la URL proporcionada por Heroku, añadiendo la clase correspondiente. Para ello, se utiliza una AsyncTask, para realizar el proceso en background. Para enviar los datos al servidor, se crea un objeto JSON, al que se le añaden los atributos correspondientes. Tras ello, se crea un *HttpURLConnection* para conectarse al servidor y se indica el método REST que se va a utilizar.

```
//Create data to send to server
JSONObject dataToSend = new JSONObject();
dataToSend.put("authorID", ride.getAuthorID());
dataToSend.put("timeGoing", ride.getTimeGoing());
dataToSend.put("timeReturn", ride.getTimeReturn());
try {
    //Initialize and config request, the connect to server
    URL url = new URL(uriPath);
    HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();
    urlConnection.setRequestMethod("POST");
    urlConnection.setRequestProperty("Content-Type", "application/json");
    urlConnection.connect();
    //Write data into server
    OutputStream outputStream = urlConnection.getOutputStream();
    bufferedWriter = new BufferedWriter(new OutputStreamWriter(outputStream));
    bufferedWriter.write(dataToSend.toString());
    bufferedWriter.flush();
    //Read data response from server
    InputStream inputStream = urlConnection.getInputStream();
    bufferedReader = new BufferedReader(new InputStreamReader(inputStream));
    String line;
} finally {
    if (bufferedWriter != null) { bufferedWriter.close(); }
    if (bufferedReader != null) { bufferedReader.close(); }
}
```

Ilustración 6.3: Implementación REST en Android

6.3.2 Servidor Messaging

Uno de los requisitos más importantes de la aplicación era enviar notificaciones a los usuarios cuando se realizase alguna acción en las rutas que han creado o en las que se han añadido. Para ello, se decidió utilizar la herramienta de Cloud Messaging de Firebase. Esta herramienta permite enviar mensajes entre usuarios de una manera sencilla y sin apenas código. Para implementar esta funcionalidad no es necesaria la creación de un servidor propio. Sin embargo, se decidió crear uno para ocultar la clave secreta proporcionada por Firebase, debido a que, si se dejaba en la aplicación Android, cualquier usuario tendría acceso a ella y podría realizar un uso malintencionado, pudiendo enviar los mensajes que quisiese a todos los usuarios de la aplicación.

El servidor de messaging es el encargado de recibir los mensajes que se crean cuando un usuario realiza alguna acción que afecta a otro usuario, y los reenvía al servidor de Firebase Cloud Messaging encargado de hacer llegar el mensaje al usuario específico.

Para enviar los mensajes desde la aplicación Android, se sigue el mismo procedimiento realizado con los mensajes enviado al servidor REST de base de datos.

Cuando una petición llega al servidor, se encarga de generar una nueva petición POST, llenando el cuerpo del mensaje con los datos recibidos, y enviarla al servidor de Firebase, con la clave proporcionada.

```
function sendNotificationToUser(req) {
  request({
    url: 'https://fcm.googleapis.com/fcm/send',
    method: 'POST',
    headers: {
      'Content-Type' : 'application/json',
      'Authorization': 'key=' + API_KEY
    },
    body: JSON.stringify({
      data: {
        messageId: req.body.messageId,
        fromUsername: req.body.fromUsername,
        value: req.body.value,
        key: req.body.key
      },
      to : '/topics/' + req.body.toUserId
    })
}
```

Ilustración 6.4: Implementación REST en Android

Por último, para que la aplicación Android pueda recibir estos mensajes, se crearon dos servicios. El primero, *InstanceIdService*, se encarga de obtener el token de registro que proporciona Firebase, para identificar al usuario.

El segundo servicio, *MessagingService*, se ejecuta cuando la aplicación recibe un nuevo mensaje desde el servidor de Firebase, tanto cuando la aplicación está activa como cuando se encuentra en segundo plano, y se encarga de extraer los datos del mensaje y, basándose en el atributo 'value', crear la notificación correspondiente.

```

public class MessagingService extends FirebaseMessagingService {

    private static final String TAG = "MessagingService";

    //Object representing the message received from Firebase Cloud Messaging.
    @Override
    public void onMessageReceived(RemoteMessage remoteMessage) {
        Log.e(TAG, "From: " + remoteMessage.getFrom());

        // Check if message contains a data payload.
        if (remoteMessage.getData().size() > 0) {
            int value = Integer.parseInt(remoteMessage.getData().get("value"));
            if(value == 0){ //User request
                notUserRequest(remoteMessage.getData().get("username"), remoteMessage.getData().get("key"));
            }
        }
    }
}

```

Ilustración 6.5: MessagingService

6.4 IMPLEMENTACIÓN DE FIREBASE

6.4.1 Authentication

Uno de los principales motivos por los que se escogió Firebase como Backend de la aplicación es que ofrece la opción de registrar y loguear a los usuarios de manera rápida y sencilla y se encarga de mantener seguras las contraseñas, además de que es compatible con los principales proveedores de servicios de login, como Gmail o Facebook. Otra de las ventajas de esta plataforma es que, al registrar los usuarios en ella, permite realizar un seguimiento de los usuarios activos, lo que permite conocer como los usuarios interactúan con la aplicación y conocer las debilidades y fortalezas. Asimismo, cuenta con una documentación muy exhaustiva y varios ejemplos de código de cada una de sus funcionalidades, lo que facilita el aprendizaje de la herramienta.

Correo electrónico	Proveedores	Fecha de creación	Inicio de sesión	UID de usuario ↑
fran10480@gmail.com	✉	14 dic. 2016	25 ene. 2017	JCgjAxHDvfbFr3W9UICXYaHU...
fran@gmail.com	✉	25 ene. 2017	25 ene. 2017	OaLLRufCMVezPkLYY7aU3AB...
fonky_128gcc@hotmail.com	✉	4 dic. 2016	25 ene. 2017	WUoCk0z1jxQiMxnuLDJaRErrE...
fonkyc@hotmail.com	✉	25 ene. 2017	25 ene. 2017	iteK7jPSFN gj0nH1yoH9Y276I...

Ilustración 6.6: Usuarios registrados en Firebase

Para implementar esta funcionalidad, simplemente hay que crear un proyecto en su web y descargarse el fichero con la información necesaria para que nuestra aplicación pueda acceder al servidor. Tras ello, hay que añadir las dependencias de Firebase necesarias a nuestra aplicación para poder acceder a las funciones que ofrece el servicio. Por último, solo hace falta añadir el código requerido en las Activities correspondientes.

```
// Auth user and email
mAuth.createUserWithEmailAndPassword(email, password)
    .addOnCompleteListener(this, (task) -> {
        progressDialog.dismiss();
        if (task.isSuccessful()) {
            onSignupSuccess(task.getResult().getUser(), name, surname);
        } else {
            Toast.makeText(LoginSignupActivity.this, "Sign Up Failed",
                Toast.LENGTH_SHORT).show();
        }
    });
}
```

Ilustración 6.7: Registro de un usuario

Además, está funcionalidad permite enviar a los usuarios emails, tanto para verificar el email del usuario como para cambiar la contraseña, en caso de haberla olvidado.

Para recuperar la contraseña, se ha añadido la opción en la página de Login. Al pulsar el usuario sobre dicha opción, se le pedirá que introduzca su email. Después, Firebase se encarga de enviar un email a la dirección de correo especificada, cuya plantilla se puede editar, con una dirección URL personalizada. Al hacer click sobre ella, redirige a una plantilla en la que pide introducir la nueva contraseña.

```
String email = input.getText().toString();
FirebaseAuth.getInstance().sendPasswordResetEmail(email)
    .addOnCompleteListener((task) -> {
        if (task.isSuccessful()) {
            Toast.makeText(LoginActivity.this, "Email sent", Toast.LENGTH_LONG).show();
        }
    });
}
```

Ilustración 6.8: Enviar email para resetear la contraseña

Para recuperar el email, el procedimiento es exactamente el mismo. Indicando el email del usuario, Firebase envía un email con una url. Al pulsar, el email del usuario quedará validado. Esta opción permite ofrecer mayor seguridad, porque los usuarios podrán saber si sus compañeros de viaje son reales.

6.5 IMPLEMENTACIÓN DE LA APLICACIÓN ANDROID

6.5.1 Firebase

Para comenzar el desarrollo, se empezó implementando Firebase en la aplicación para realizar el registro y el logueo de los usuarios, así como para permitir la comunicación entre usuarios mediante el envío de notificaciones. Esta tarea resultó ser bastante sencilla gracias a las facilidades que ofrece Firebase y a la extensa documentación proporcionada por la página.

En un principio se decidió crear la base datos también con Firebase, porque era fácil de acceder y se encontraba en la nube. Sin embargo, debido a las limitaciones en las consultas, se descartó esta posibilidad.

6.5.2 Activities

Las Activities son los componentes de la aplicación con las que los usuarios pueden interactuar para realizar una acción. Por ello, se ha creado una Activity para cada una de las pantallas de la aplicación.

Además, para obtener un código más claro y limpio, se ha hecho uso de la librería ButterKnife. Esta librería permite eliminar todos los `findViewById` que pueden resultar muy molestos y sustituirllos por `@Bind(R.id.recurso) TextView txtView` en la declaración de los elementos de la vista. Para hacer uso de esta funcionalidad, simplemente hay que añadir en las dependencias del gradle `compile 'com.jakewharton:butterknife:7.0.1'` y después `Butterknife.bind(this);` en el método `onCreate()` de las Activities que hagan uso de esta funcionalidad.

```
@Bind(R.id.btn_signup) Button signupButton;
@Bind(R.id.link_login) TextView loginLink;

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_login_signup);
    ButterKnife.bind(this);
```

Ilustración 6.9: Butterknife

Para crear ListView personalizadas para mostrar las rutas encontradas en las búsquedas o los usuarios añadidos a una ruta, se han creado Adapter personalizados. Para ello, se crea un objeto de layout con el aspecto que debe tener cada uno de los ítems de la lista y, después, se importa en el layout de la Activity. Además, ha sido necesario implementar OnClickListener en aquellos layout que tenían imágenes que realizaban acciones al ser pulsada, como los botones de aceptar (checkmark) y rechazar (x) en la lista de solicitudes para unirse a una ruta.

6.5.3 Indicativo de coche eléctrico

Uno de los objetivos de la aplicación es reducir la contaminación aérea provocada por la emisión de gases contaminantes debido a los desplazamientos diarios que tienen que realizar las personas para llegar a su lugar de trabajo. Por ello, se decidió que aquellas rutas cuyo conductor utilizase un coche eléctrico, al ser la opción que menos gases contaminantes genera, deberían destacar y favorecerse en las rutas.

Para ello, es obligatorio añadir un coche antes de que un usuario pueda crear una ruta nueva, indicando el tipo de motor. Además, en las búsquedas, las rutas con coche eléctrico se muestran con un símbolo indicativo, de manera que sean fácilmente visibles e identificables para los usuarios.



Ilustración 6.10: Símbolo indicativo de coche eléctrico

6.5.4 Búsqueda de rutas

El objetivo en el momento de buscar las rutas es ofrecer al usuario solo aquellas que mejor se adapten a sus necesidades, según lo indicado en el formulario. Para ello, se siguen los siguientes parámetros:

- Distancia: la búsqueda solo devolverá aquellos resultados que se encuentren dentro de la distancia máxima indicada por el usuario en sus preferencias. Para calcular la distancia entre el lugar de la ruta y el lugar indicado por el usuario, se utiliza la siguiente fórmula: $3959 * \text{acos}(\cos(\text{radians}(lat1)) * \cos(\text{radians}(lat2)) * \cos(\text{radians}(lon2) - \text{radians}(lon1)) + \sin(\text{radians}(lat1)) * \sin(\text{radians}(lat2))$.
- Tiempo: la búsqueda solo devolverá las rutas que se encuentre dentro del tiempo máximo indicado por el usuario en las preferencias, tanto por arriba como por abajo.
- Días: la búsqueda solo devolverá las rutas que dispongan de al menos un asiento libre para los días indicados por el usuario.

6.5.5 Google Maps

6.5.5.1 Obtención de la clave

Para permitir a los usuarios indicar de manera precisa los lugares de partida y destino, se ha decidido utilizar Google Maps, debido a su gran popularidad y a su fácil integración con Android. Para ello, lo primero que hay que hacer es crear una Google Maps Activity. Tras ello, se genera automáticamente un fichero XML, al que hay que añadir una clave gratuita que te da Google. Para obtenerla, hay que copiar la url que aparece en el documento y que te dirige a la consola de desarrolladores de Google, donde es obligatorio loguearse con una cuenta de Gmail. Tras ello, simplemente hay que crear un nuevo proyecto o seleccionar uno ya existente. Por último, hay que copiar la clave obtenida y pegarla en el lugar indicado en el fichero XML. Esta licencia ofrece en Android consultar ilimitadas a Google Maps, por lo que no es necesario pagar por su uso.

```
<resources>
    <string name="google_maps_key" templateMergeStrategy="preserve" translatable="false">
        AIzaSyCWFb4F2zAhC02s7Adolh14IjsJhDmUTM8
    </string>
</resources>
```

Ilustración 6.11: Fichero XML con la clave de Google Maps

6.5.5.2 Implementación

Google Maps se inicia cuando un usuario está rellenando el formulario para crear o buscar una ruta, y pulsa sobre el campo “Desde” o “Hacia”. Cuando se inicia la actividad, se solicita los permisos de ubicación de localización y activar la ubicación. Si el usuario acepta, el mapa se centrará en las coordenadas del usuario.

También, se ha implementado un buscador para que el usuario introduzca la localización y pueda buscar el lugar, en vez de estar moviéndose por el mapa.

Cuando el usuario ha seleccionado un lugar y pulsa en aceptar, la aplicación guarda la dirección del lugar y su latitud y longitud. Estos dos últimos valores, se utilizan en las búsquedas para calcular la distancia en metros entre el lugar seleccionado y el destino.

6.5.6 Preferencias

En la pantalla de ajustes se permite al usuario modificar los valores de 'Distancia máxima' y 'Tiempo máximo'. Estos dos valores se utilizan en las búsquedas que realiza el usuario para determinar la distancia máxima que el usuario desea recorrer entre el punto que ha indicado y el punto de salida o llegada de una ruta, y el margen de tiempo que considere aceptable. De esta manera, solo se mostrará al usuario aquellas rutas que mejor se adapten a sus necesidades. Para ello, se ha implementado una *SeekBar*, para que el usuario pueda escoger de una manera rápida y sencilla sus preferencias.

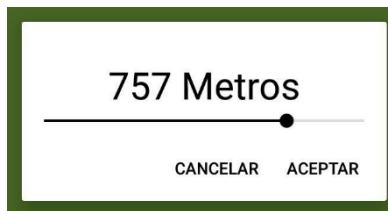


Ilustración 6.12: Seekbar para seleccionar la distancia máxima

6.5.7 Multilenguaje

Para facilitar la expansión de la aplicación a otros países, se ha decidido añadir la opción de multilenguaje, la cual permite que se muestre al usuario la aplicación en su idioma. De esta manera, se facilita la accesibilidad de los usuarios en la aplicación y se evita que la rechacen por el simple hecho de no estar localizada.

Antes de empezar, es muy importante asegurarse de que todos los textos que se vayan a mostrar por pantalla en nuestra aplicación se encuentren en el fichero strings.xml, con su id única. Después, para traducir los textos al nuevo idioma escogido, simplemente hay crear un nuevo fichero de 'values' con las iniciales del idioma deseado. Para evitar que el desarrollador se olvide de traducir alguna palabra y pueda provocar incoherencias, Android provee un editor en el que se muestra para cada id, su traducción. Por último, es el sistema el que escoge el idioma basándose en las preferencias generales del usuario.

En nuestra aplicación, debido a que de momento solo se va a lanzar en España, se ha decidido ofrecer la aplicación en español y en inglés, siendo este último el idioma por defecto.

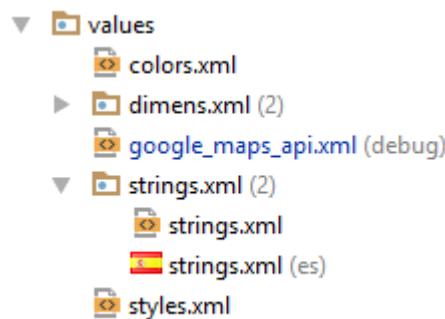


Ilustración 6.13: Multilenguaje

7. HERRAMIENTAS DE ANÁLISIS

7.1 VISION GENERAL

En este apartado se describen aquellas herramientas que permiten conocer la forma en la que los usuarios interactúan con nuestra aplicación y los problemas que sufren, de manera que se pueden subsanar de manera rápida y eficaz, y que tengan el menor impacto posible en su experiencia con la aplicación.

7.2 FIREBASE ANALYTICS

Firebase Analytics es una solución de medición gratuita que proporciona información sobre el uso de las aplicaciones y el compromiso de los usuarios. Gracias a esta información obtenida, podremos conocer desde el primer día aquellas funcionalidades que aportan un valor diferencial a nuestra aplicación y atraen a los usuarios, o sobre qué segmento de la población tiene más éxito y realizar una publicidad mejor orientada y más efectiva.

Estas funcionalidades pueden resultar básicas en un mercado tan volátil y fragmentado como es el de las aplicaciones móviles, ya que permite orientar mejor el desarrollo de la aplicación y puede marcar la diferencia entre las aplicaciones que estén tratando en ese momento de encontrar su hueco en el mercado.

Para utilizar esta funcionalidad de Firebase, solo hay que registrar a los usuarios de la aplicación utilizando Firebase Authentication. Desde ese momento, Firebase empezará a recoger estadísticas sobre el uso de la aplicación y a monitorizar la actividad de los usuarios. De este modo, podremos obtener información tan valiosa como el número de veces que los usuarios entran en nuestra aplicación, así como el tiempo medio por sesión que pasan en ella. Esta información se muestra en la sección de Analytics en el panel de control de Firebase, en forma de gráficos, lo que facilita su estudio y comprensión. Para obtener toda esta información, no es necesaria ninguna línea adicional de código, a parte del uso del módulo de autenticación de usuarios.

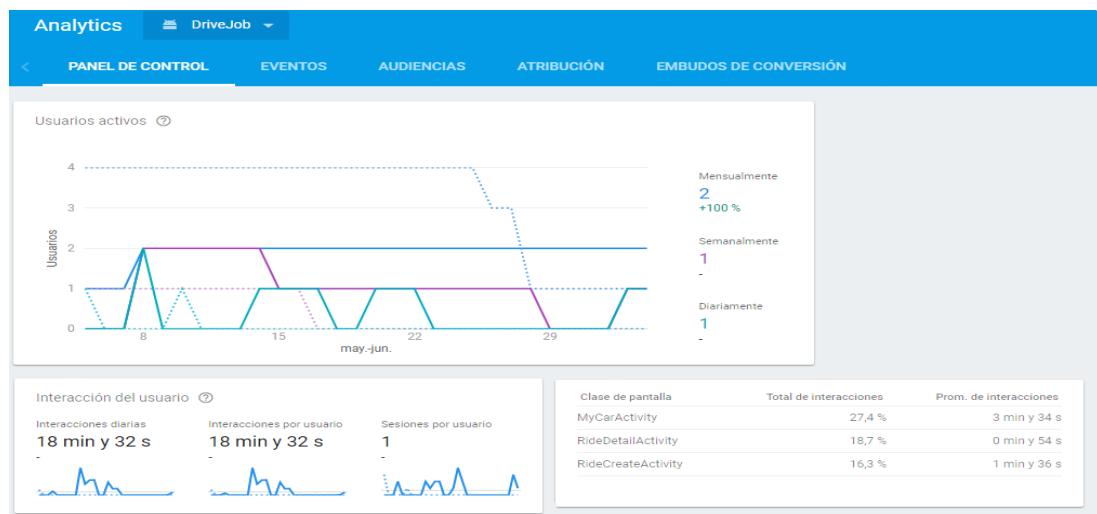


Ilustración 7.1: Analíticas de Firebase sobre nuestros usuarios

A partir del uso de los usuarios, también podremos conocer otros datos derivados. Entre ellos, podemos identificar de qué país provienen la mayor parte de nuestros usuarios, de modo que podamos centrarnos en adaptar la aplicación a aquellos países que ofrecen mayor rentabilidad

Otro dato de gran interés y que aporta un gran valor, ya que puede permitir gestionar mejor los esfuerzos en el desarrollo de la aplicación, es el modelo del teléfono y su versión del sistema operativo. Esto puede resultar de gran utilidad porque, por ejemplo, si es necesario aumentar la versión mínima requerida por la aplicación, podremos saber cuántos usuarios perderíamos debido a que no podrán volver a tener acceso a la aplicación.

Como se ha mostrado en la Ilustración 1.1, existen una gran variedad de versiones Android en el mercado y la mayoría de dispositivos cuentan con un sistema operativo de más de tres años de antigüedad. Sin embargo, la existencia de estas versiones antiguas puede deberse a la gran popularidad de terminales económicos para países subdesarrollados. Por ello, si nuestro mercado es principalmente europeo, puede no afectarnos ese problema y centrar nuestros esfuerzos en ofrecer mejores en la aplicación, en vez de ofrecer compatibilidad con versiones antiguas.

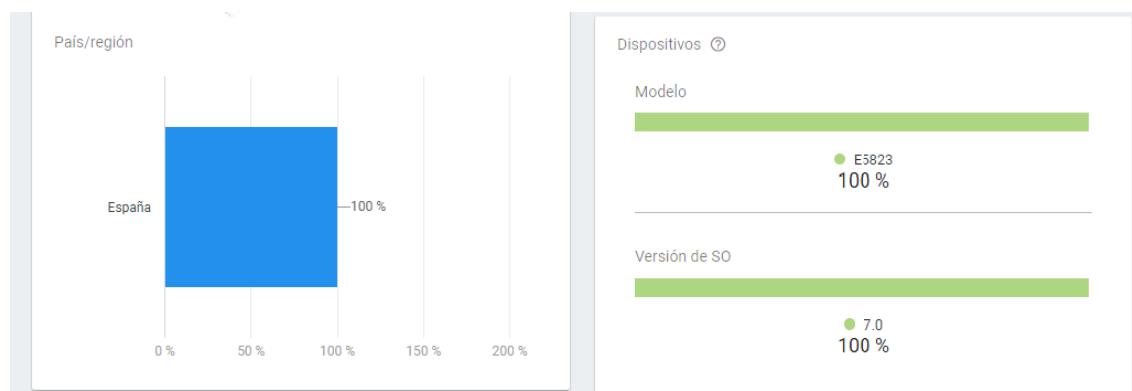


Ilustración 7.2: Análisis de Firebase sobre la procedencia de los usuarios y sus dispositivos

7.3 CRASH REPORTING

Firebase Crash Reporting permite conocer al momento los problemas que sufren nuestros usuarios mientras interactúan con nuestra aplicación. Simplemente instalando una dependencia, Firebase detecta automáticamente y sube al servidor aquellas excepciones no controladas que han sufrido los usuarios y aquellos eventos que decidimos añadir. Además, es capaz de agrupar los fallos que sean similares, lo que facilita su gestión permite detectar enseguida un fallo de la aplicación.

Esta herramienta puede resultar de gran utilidad, ya que ofrece un informe detallado de los problemas que sufre la aplicación y permite actuar de manera más rápida y adelantarse a las quejas de los usuarios. De esta manera, se puede evitar la publicidad negativa en las redes sociales de los usuarios debido al descontento por los errores de la aplicación, y valoraciones bajas en la tienda de aplicaciones de la plataforma debido a malas experiencias.

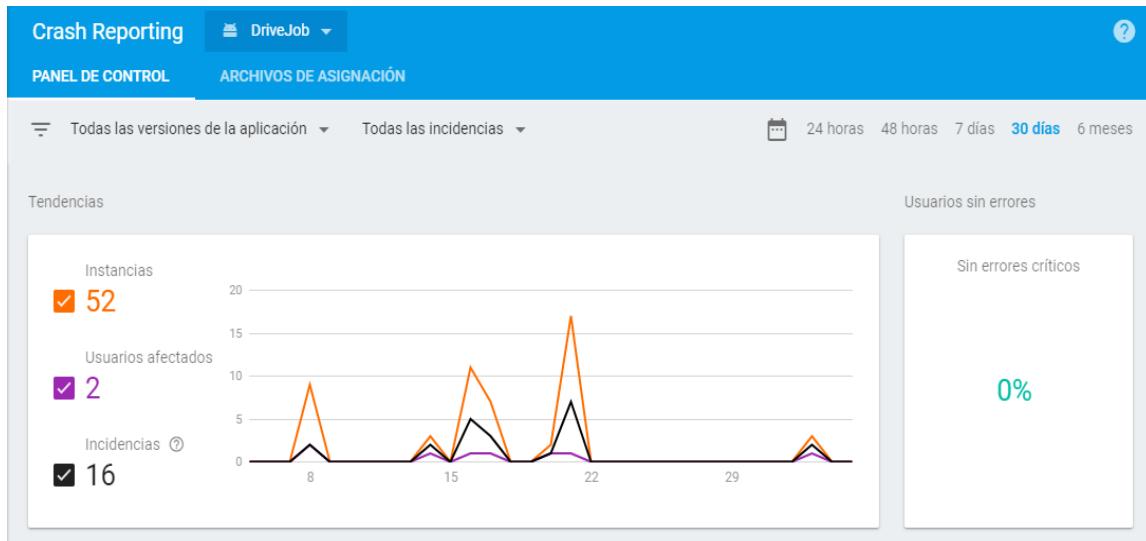


Ilustración 7.3: Informe sobre los fallos que se han producido

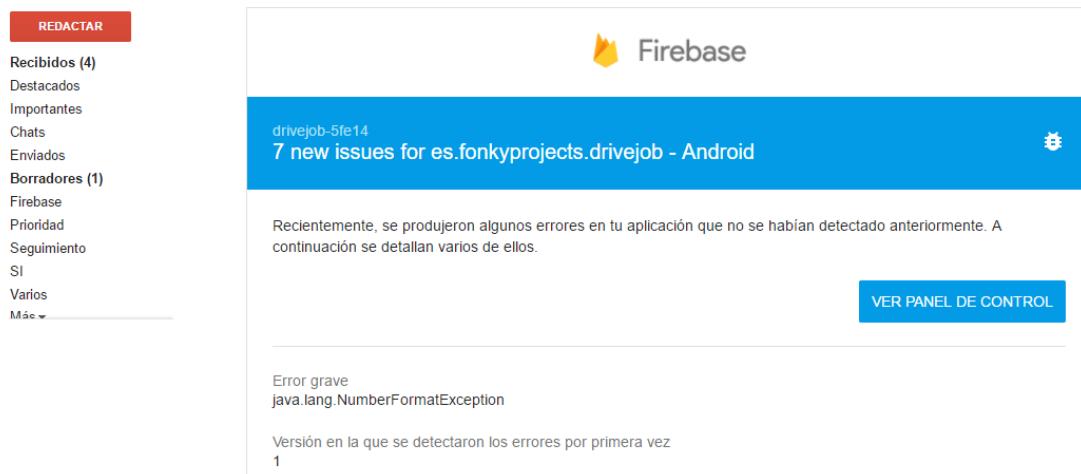


Ilustración 7.4: Email diario con los errores detectados

Se decidió utilizar esta funcionalidad por su facilidad de implementación y por su gran utilidad, ya que resolver los problemas de la aplicación cuando solo ha afectado a un pequeño grupo, puede decidir hoy en día que aplicación logra triunfar y cual se queda por el camino.

8. MANUAL DE USUARIO

8.1 VISIÓN GENERAL

En este apartado se muestran capturas de pantalla para ilustrar de manera sencilla las principales funcionalidades de la aplicación.

8.2 LOGIN

En esta pantalla, los usuarios pueden iniciar sesión en la aplicación rellenando los campos con el email y la contraseña. Si aún no se han registrado, el usuario puede acceder al formulario de registro pulsando *¿No tienes cuenta? Crea una*



Ilustración 8.1: Login

8.3 REGISTRO

Los usuarios que quieran acceder a la aplicación, primero deben registrarse. Para ello, basta con indicar su nombre, apellido, una dirección de email y su contraseña. La longitud de la contraseña deberá ser superior a 4 caracteres e inferior de 10.

Ilustración 8.2: Formulario de registro

8.4 MENU

Una vez el usuario inicie sesión en la aplicación, se le mostrará un menú con las acciones que puede realizar dentro de la aplicación. Además, arriba a la derecha, aparecerá un menú desplegable para acceder a su perfil, modificar los ajustes o desconectarse.

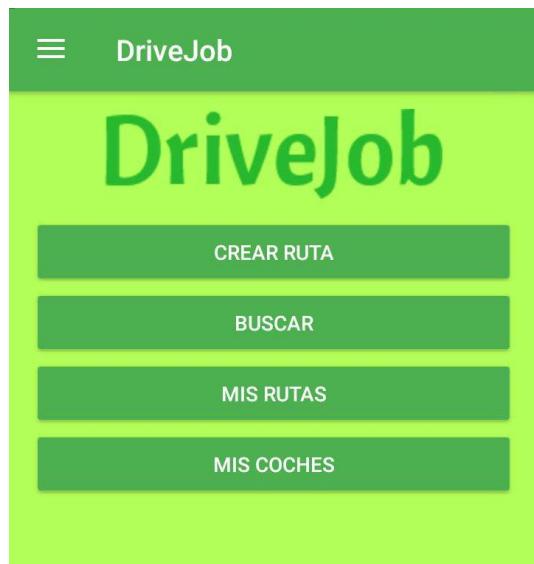


Ilustración 8.3: Menú de la aplicación.

Para navegar por la aplicación, se ha creado un menú lateral desplegable que permite acceder de manera rápida y sencilla a las pantallas principales de la aplicación.

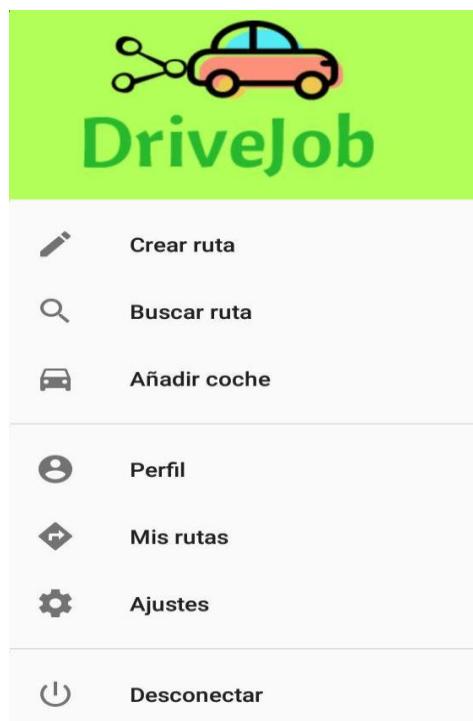


Ilustración 8.4: Menú lateral

8.5 COCHE

Antes de que el usuario puede crear una ruta, debe haber añadido al menos un coche. Para ello, deberá llenar un formulario indicando la marca y el modelo del coche, además de seleccionar el tipo de motor entre las opciones indicadas en el menú desplegable.

Marca

Modelo

- Gasolina
- Diesel
- Híbrido
- Eléctrico

Ilustración 8.5: Añadir coche

8.6 CREAR RUTA

Para crear una ruta, los usuarios deben indicar tanto los lugares como las horas para la ida y la vuelta, así como los pasajeros a los que desean llevar, los días que realizarán la ruta, el precio por mes que deberán pagar aquellos usuarios que deseen unirse y el coche que utilizarán, que deberán haber creado previamente.

Desde

Hacia

Ida Vuelta

Días

Precio

Pasajeros

Ferrari A4 (Eléctrico)

CREAR RUTA

Ilustración 8.6: Crear ruta

Además, cuándo el usuario pulse en *Ida* o *Vuelta*, aparecerá un reloj para indicar la hora. Asimismo, cuando pulsen en *Días*, se mostrará una lista para que escogen los días en los que realizarán la ruta.



Ilustración 8.7: Reloj para indicar la hora

Ilustración 8.8: Lista para indicar los días

8.7 LOCALIZACIÓN EN GOOGLE MAPS

Para obtener la mayor precisión posible, los usuarios cuando pulsan para indicar su localización tanto a la ida como a la vuelta, se abre una Actividad con Google Maps para indicar el punto exacto.



Ilustración 8.9: Google Maps

8.8 MI RUTA

Cuando el usuario pulse en Crear ruta, se le redirigirá al detalle de la ruta. Al ser el creador de la ruta, tendrá la opción de editarla o eliminarla. Cuándo algún usuario solicite unirse a la ruta, aparecerá en la lista de solicitudes para que puede decidir si aceptarlo o rechazarlo.



Ilustración 8.10: Vista de la ruta desde la perspectiva del conductor

8.9 RESULTADOS

Para encontrar las rutas que mejor se adapten a las necesidades del usuario, primero debe llenar un formulario, indicando el lugar y la hora de salida y llegada, y los días en los que quiere realizar el viaje. Con los datos introducidos, el algoritmo se encarga de buscar aquellas rutas que se encuentren dentro de los parámetros indicados por el usuario. Además, aquellas rutas cuyo coche sea eléctrico, aparecerá indicado con un símbolo.



Ilustración 8.11: Resultados de la búsqueda

8.10 DETALLE DE LA RUTA

Cuando el usuario pulsa sobre una ruta, se mostrará el detalle de la misma. En este caso, como el usuario no es el conductor, no tendrá la opción ni de editar ni de eliminar la ruta, pero aparecerá el botón de *Unirse*.



Ilustración 8.12: Ruta desde la perspectiva del pasajero

Cuando el usuario pulse el botón, aparecerá una pantalla con una lista para indicar que días, de los que esa ruta tiene asientos libres, quiere realizar el viaje. Una vez seleccionados, se le añadirá en la lista de solicitudes, indicando su nombre y los días que ha seleccionado, para que el conductor pueda decidir si le acepta o rechaza. Si el conductor decide aceptar al usuario, pasará a la lista de usuarios unidos, donde el conductor aún puede expulsar si lo considera oportuno.

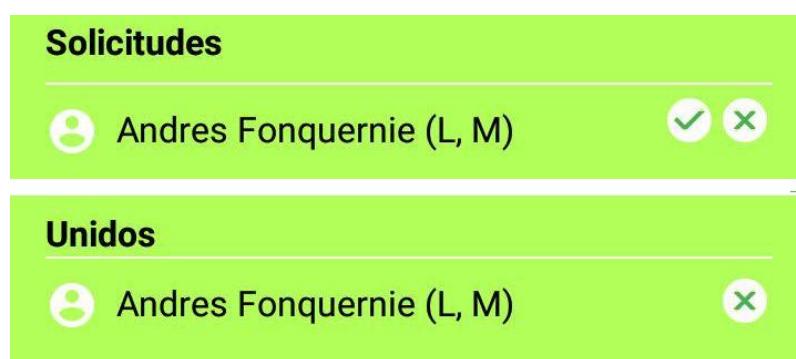


Ilustración 8.13: Lista de solicitudes y usuarios unidos

8.11 PERFIL USUARIO

Si un usuario desea conocer más información sobre otro usuario, para ponerse en contacto con él, puede acceder a su perfil de usuario pulsando sobre el nombre de dicho usuario. En esta pantalla, aparece el nombre, apellido y el email del usuario. Al lado del email, aparecerá un tick o una x, para indicar si el usuario ha verificado su email. Además, si el usuario visita su propio perfil, podrá editar la información o pedir que le envíen un email para verificar el suyo, en caso de que no lo haya realizado anteriormente.



Ilustración 8.14: Perfil de usuario

8.12 NOTIFICACIONES

Cuando se realizan ciertas acciones, se envían notificaciones a los usuarios implicados para que están informados sobre los eventos que suceden dentro de la aplicación. Por ejemplo, cuando un usuario solicita unirse a una ruta, se envía una notificación al conductor y, si pulsa sobre ella, se le abrirá la aplicación en el detalle de la ruta.

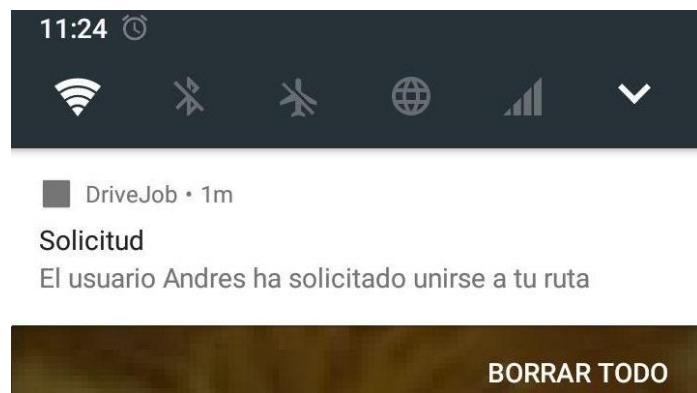


Ilustración 8.15: Notificación

8.13 AJUSTES

En la pantalla de ajustes, el usuario deberá indicar la distancia y el tiempo máximo, de modo que al buscar las rutas solo aparezcan aquellas que mejor se adaptan.

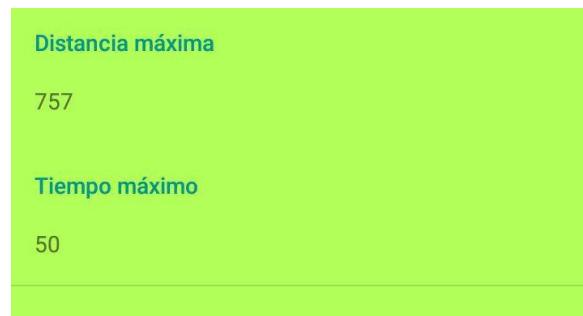


Ilustración 8.16: Distancia y tiempo máximo

Cuando el usuario pulse sobre uno de los valores, se mostrará una ventana con una barra deslizante para que indique sus preferencias.

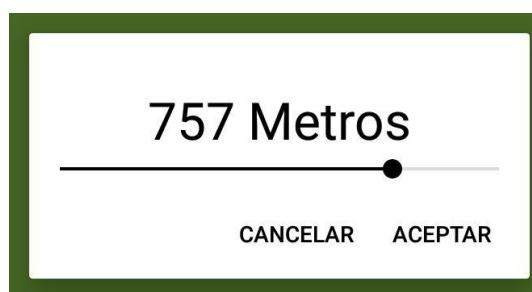


Ilustración 8.17: SeekBar para indicar la distancia máxima

9. CONCLUSIONES Y LÍNEAS FUTURAS

9.1 VISIÓN GENERAL

En este apartado se detallan las incidencias ocurridas durante el desarrollo del proyecto y las conclusiones obtenidas tras su realización, así como las posibles mejoras y expansiones de la aplicación móvil.

9.2 CONCLUSIONES

El desarrollo de este proyecto para la asignatura de Proyecto Fin de Master me ha permitido profundizar mis conocimientos sobre Android, lo cual considero de gran importancia debido al gran auge actual de las aplicaciones para móviles y su capacidad de generar nuevos negocios, cubriendo diferentes necesidades de los usuarios. Es por ello que, durante el desarrollo de la aplicación, siempre se ha tenido en cuenta el posible futuro comercial y se ha diseñado con la idea de poder ser explotada en un futuro.

Además, este proyecto me ha permitido conocer nuevas tecnologías que no había utilizado hasta el momento como las Bases de Datos no relacionales, más concretamente MongoDB, aunque conocía su existencia debido a que se han mencionado en alguna asignatura. De esta manera, he podido conocer más a fondo su funcionamiento y descubrir sus ventajas y desventajas frente a SQL.

También ha sido nuevo para mí, el desarrollo de un servidor web en NodeJS, ya que hasta ahora había trabajado con JavaScript en alguna pequeña página web solo para realizar alguna operación en el lado del cliente.

Por último, he podido otras herramientas útiles para el desarrollo de aplicaciones, como la API de Google Maps para añadir funcionalidades de localización al proyecto, Firebase como backend para crear aplicaciones funcionales con bastante rapidez o Google Cloud SQL, para generar una Base de Datos online.

9.3 INCIDENCIAS

Durante el desarrollo del proyecto han aparecido nuevas versiones Android, por lo que ha sido necesario actualizar en varias ocasiones el SDK de Android. Uno de los principales problemas fue que, tras actualizar Android, el gradle daba errores de compatibilidad de versiones cuando se compilaba, por lo que el proyecto no se podía ejecutar, ni seguir desarrollando. Basándome en el mensaje de error, busqué la última versión para las dependencias añadidas, pero esto no solucionó el error. Tras realizar varias pruebas, se detectó que el problema estaba provocado al compilar las dependencias relativas a Google Maps y que también se producía cuando se creaba un nuevo proyecto Android con una Activity de Google Maps. Al ser un problema reciente, no había ninguna solución en internet. Tras buscar en varias páginas webs, se descubrió que había habido un cambio en las condiciones de seguridad de Android respecto a Google Maps, y a partir de ese momento las librerías de Google Maps había que importarlas de una en una. Busqué las librerías necesarias y el error se subsanó.

9.4 LÍNEAS FUTURAS

9.4.1 Monetización

Debido al posible carácter comercial de la aplicación, es importante pensar cómo generar ingresos. Para ello, siguiendo la idea de una de las aplicaciones más exitosas en el ámbito de compartir coche, Blablacar, se ha decidido que una buena idea para rentabilizar el proyecto sería a través del cobro de una pequeña comisión a todos los usuarios que paguen por unirse a una ruta.

9.4.2 Solución en caso de ausencia del conductor

Los pasajeros que deciden unirse a una ruta utilizando esta aplicación están confiando en que el conductor asista diariamente para realizar trayecto. Debido a esto, en caso de falta, ya sea conocida de antemano o haya sucedido algún improvisto, se considera que sería importante que los pasajeros pudiesen obtener alguna compensación debido a las molestias ocasionadas. Para solucionar el problema, se han decidido dos posibles soluciones:

- Los pasajeros no pagan la parte proporcional correspondiente a ese día.
- Un pasajero se convierte en conductor, si así lo deciden entre todos los usuarios, y se le descontaría al conductor su parte proporcional para abonársela al pasajero-conductor.

9.4.3 Valoración entre usuarios

Para conocer la opinión de los usuarios sobre sus compañeros, tanto conductores como pasajeros, se considera que sería importante añadir una funcionalidad que permita que los usuarios se valoren entre ellos. De esta manera, se podrían reducir las malas experiencias de los usuarios con el uso de la aplicación debido a individuos poco deseables. Sin embargo, como la aplicación está destinada a cubrir los viajes rutinarios de los usuarios, se espera poco movimiento en cuanto a cambiar de compañeros de viaje, por lo que la funcionalidad no se considera importante.

10. BIBLIOGRAFÍA

- [1] “Autofacil”: <http://www.autofacil.es/tecnica/2016/01/29/contamina-diesel-o-gasolina/30341.html>, (consultado el 25/06/2017).
- [2] “Ecologistas en acción”: <http://www.ecologistasenaccion.org/article9846.html>, (consultado el 25/06/2017). “Organización mundial de la salud”: <http://www.who.int/>, (consultado el 25/06/2017).
- [3] “Te interesa”: http://www.teinteresa.es/motor/restriccion-trafico-tradicion-capitales-europeas_0_1216680156.html, (consultado el 25/06/2017).
- [4] “Observatorio de movilidad”:
http://www.observatoriomovilidad.es/images/stories/05_informes/informe_OMM2014.pdf, (consultado el 25/06/2017).
- [6] “Abc”: http://www.abc.es/motor/reportajes/abci-solo-11-por-ciento-comparte-coche-para-trabajo-201603311859_noticia.html, (consultado el 25/06/2017).
- [7] “Expansión”:
<http://www.expansion.com/empresas/transporte/2015/08/24/55db3e3f46163f27748b4589.html>, (consultado el 25/06/2017).
- [8] “El país”: http://ccaa.elpais.com/ccaa/2016/06/21/madrid/1466503998_606595.html
- [9] “Abc”: <http://www.abc.es/motor-reportajes/20150428/abci-conductores-estres-trabajo-coche-201504272008.html>, (consultado el 25/06/2017).
- [10] “Blablacar”: <https://www.blablacar.es>, (consultado el 25/06/2017).
- [11] “Amovens”: <https://amovens.com/>, (consultado el 25/06/2017).
- [12] “Poolmyride”: <https://poolmyride.com/>, (consultado el 25/06/2017).
- [13] “Liftshare”: <https://liftshare.com/uk>, (consultado el 25/06/2017).
- [14] “Abc”: http://www.abc.es/tecnologia/moviles/telefonía/abci-espana-pais-mas-smartphones-habitante-mundo-201611081019_noticia.html, (consultado el 25/06/2017).
- [15] “El mundo”:
<http://www.elmundo.es/sociedad/2016/04/04/57026219e2704e90048b465e.html>, (consultado el 25/06/2017).
- [16] “Expansión”: <http://www.expansion.com/economia-digital/companias/2015/12/09/56684be1ca474151018b4590.html>, (consultado el 25/06/2017).
- [17] “Xataka Android”: <https://www.xatakandroid.com/mercado/android-marshmallow-ya-es-la-tercera-version-mas-usada-por-detras-de-lollipop-y-kitkat>, (consultado el 25/06/2017).
- [18] “Gadgetos”: <http://www.gadgetos.com/noticias/usuarios-ios-mas-rentables-que-android/>, (consultado el 25/06/2017).
- [19] “mlab”: <https://mlab.com/>, (consultado el 25/06/2017).
- [20] “Desarrollo iterativo y creciente”:
https://es.wikipedia.org/wiki/Desarrollo_iterativo_y_creciente (consultado el 25/06/2017)
- [21] “Android”: https://www.android.com/intl/es_es/, (consultado el 25/06/2017).
- [22] “Componentes de una aplicación Android”: <http://www.sgoliver.net/blog/componentes-de-una-aplicacion-android/>, (consultado el 25/06/2017).
- [23] “Java”: [https://es.wikipedia.org/wiki/Java_\(lenguaje_de_programación\)](https://es.wikipedia.org/wiki/Java_(lenguaje_de_programación)), (consultado el 25/06/2017).
- [24] “Firebase”: <https://firebase.google.com/>, (consultado el 25/06/2017).
- [25] “NodeJS”: <https://www.ibm.com/developerworks/ssa/opensource/library/os-nodejs/> (consultado el 25/06/2017)
- [26] “Express”: <https://en.wikipedia.org/wiki/Express.js> (consultado el 25/06/2017)
- [27] “Mongoose”: <https://blog.xervo.io/getting-started-with-mongoose> (consultado el 25/06/2017)

- [28] “JavaScript”: <https://es.wikipedia.org/wiki/JavaScript> (consultado el 25/06/2017)
- [29] “MongoDB”: <https://www.mongodb.com/es> (consultado el 25/06/2017)
- [30] “NoSQL”: <https://es.wikipedia.org/wiki/NoSQL> (consultado el 25/06/2017)
- [31] “JSON”: <https://es.wikipedia.org/wiki/JSON> (consultado el 25/06/2017)
- [32] “Google Cloud SQL”: <https://cloud.google.com/sql> (consultado el 25/06/2017)
- [33] “Google Maps”: <https://www.google.es/maps> (consultado el 25/06/2017)
- [34] “Android Studio”: <https://developer.android.com/studio/features.html?hl=es-419> (consultado el 25/06/2017)
- [35] “Github”: <https://github.com/>
- [36] “Notepad++”: <https://notepad-plus-plus.org/> (consultado el 25/06/2017)
- [37] “Sublime Text”: <https://www.sublimetext.com/> (consultado el 25/06/2017)
- [38] “TortoiseGit”: <https://tortoisegit.org/>
- [39] “Visual Paradigms”: https://en.wikipedia.org/wiki/Visual_Paradigm_for_UML (consultado el 25/06/2017)
- [40] “Microsoft Project”: https://es.wikipedia.org/wiki/Microsoft_Project (consultado el 25/06/2017)