Formación Profesional

coreNetworks

Proyecto Fin de Grado
Grado Superior en Desarrollo de
Aplicaciones Web/Multiplataforma

Cochemania
Memoria

Fecha: 2024-2025
Curso académico: 2024-2025
Autor: Raúl Jiménez García
DNI: 47349212D
Tutor del PFG: Francisco Javier Luque Terriza

Índice

1. Resumen **3**
1.1. Objetivo principal 3
1.2. Motivación 3
2. Introducción **4**
2.1. Contextualización de problemas 4
2.2. Objetivos 4
2.3. Metodología empleada 5
3. Análisis del problema y requisitos **6**
3.1. Descripción de problemas 6
3.2. Requisitos funcionales 6
3.3. Requisitos no funcionales 6
3.4. Casos de uso 7
4. Arquitectura del sistema **8**
4.1. Frameworks y lenguajes usados 8
4.2. Base de datos 8
4.3. Justificación 9
5. Diseño del sistema **10**
5.1. Diseño funcional 10-11-12
5.2. Diseño de la base de datos 13-14-15
5.3. Gestión de seguridad 15
6. Desarrollo e implementación **16**
6.1. Planificación 16
6.1.1. Fase 1 16
6.1.2. Fase 2 16
6.1.3. Fase 3 16
6.1.4. Fase 4 16
6.2. Herramientas y tecnologías empleadas 17
7. Descripción funcional del sistema implementado **18**
7.1. Funcionalidades 18-19
7.2. Estructura del subsistema 19
7.3. Funcionalidades no implementadas 20
8. Pruebas y validación **21-22**
9. Despliegue **23**
10. Conclusiones **23**
11. Posibles mejoras y líneas de evolución futura **24**

1. Resumen

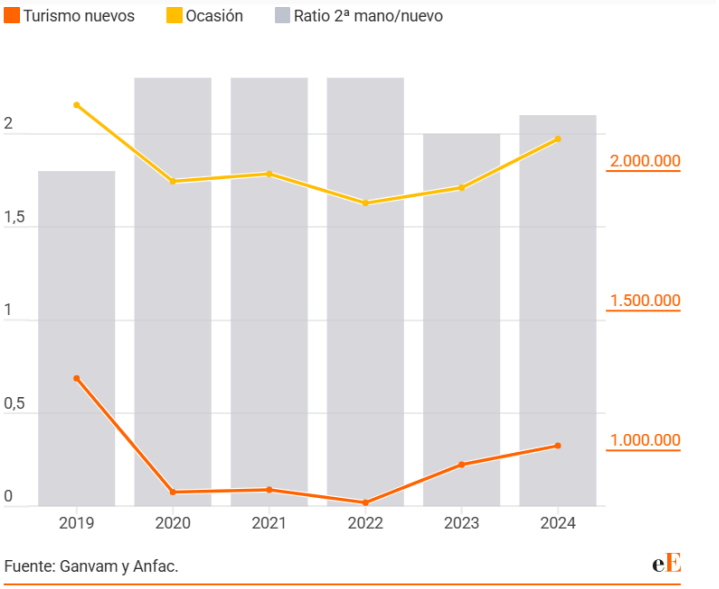
Mi proyecto es una aplicación móvil que también tendrá su plataforma web, y está diseñado para la compra, venta y alquiler de vehículos. El sistema permite que cualquiera, por ejemplo particulares, concesionarios, empresas de alquiler entre otros puedan publicar, buscar y gestionar vehículos de forma segura y eficiente. Cualquier usuario, sin necesidad de registro, puede explorar el catálogo completo y aplicar filtros de búsqueda para una búsqueda más exhaustiva. Para publicar anuncios, realizar reservas, contactar con los vendedores, añadir vehículos a favoritos, dejar valoraciones o configurar alertas personalizadas, es necesario crear una cuenta.

1.1. Objetivo principal

El objetivo principal de mi aplicación es el de conectar a compradores, vendedores, arrendadores y arrendatarios mediante un sistema que les permita gestionar todo el proceso de compraventa y alquiler de sus vehículos de manera rápida, segura, eficaz y transparente. Manteniendo en todo momento a los usuarios al tanto de las actividades relacionadas con sus vehículos como por ejemplo notificarlos cuando les hagan una petición de reserva.

1.2. Motivación

Mi principal motivación es que el mundo de la compraventa de coches de segunda mano es algo que está siempre en continuo auge, cada vez hay más gente que opta por comprar coches de segunda mano o kilómetro 0, y el mercado de los vehículos de alquiler también es bastante amplio, sobre todo el de los vehículos para el transporte de mercancías como furgonetas para el transporte de mercancías y los vehículos particulares en ciudades vacacionales.Además no hay muchas aplicaciones que se dediquen a ambas cosas el alquiler y la venta de vehículos.
Enlace información datos venta coches segunda mano españa.



2. Introducción

2.1. Contextualización de problemas

Actualmente hay muchos sitios de compraventa o alquiler de vehículos de segunda mano por lo que entrar en ese mercado de una forma rápida sería bastante complicado, por eso para entrar en el mercado lo mejor sería buscar de primeras asociaciones con algunos concesionarios o empresas que no tengan web propia donde publicar los vehículos. Por eso he decidido hacer una aplicación en la que se puede tanto vender como comprar y alquilar.

En muchas de las aplicaciones en las que los vendedores no son parte de la app sino que son vendedores particulares, por lo que existe el riesgo de la estafa ya que en muchas aplicaciones publicas el vehículo pero no requieren comprobación de titularidad del vehículo ni nada así.

También está el riesgo de que la descripción del estado y mantenimiento no sea real. Por ejemplo que se indique que el estado del vehículo sea perfecto pero tenga cualquier tipo de problema como

La aplicación carece de un sistema de chat en vivo entre usuario, lo que limita mucho el funcionamiento de la app haciendo que la puesta en contacto entre usuario y propietario para la venta de vehículos tenga que realizarse fuera de la aplicación. Tengo puesto como solución provisional que te mande al correo electrónico y en el apartado de ‘Para’ esté ya rellenado con el email del propietario, lo cuál no está bien porque muestra datos del usuario propietario pero lo tengo como solución provisional.

No tengo un sistema de verificación de propiedad del vehículo, por lo que no puedo comprobar que los usuarios que publiquen un vehículo sean los dueños legítimos para evitar estafas.

Ahora mismo todo lo que conlleva verificación (publicación de vehículos por parte de usuarios y aceptación de reservas) son gestionados de manera manual lo que hace el proceso más lento.

2.2. Objetivos

El objetivo principal del proyecto es crear una aplicación que conecte eficientemente a los diferentes actores del mercado de vehículos: compradores, vendedores, arrendadores y arrendatarios.
Otros objetivos son:
- Permitir la publicación y gestión de anuncios de vehículos.
- Integrar un sistema de búsqueda avanzada y filtros personalizados.
- Sistema de reservas para vehículos en alquiler.
- Ofrecer valoraciones, sistema de alertas y gestión de favoritos.
- Establecer mecanismos de autenticación y control de acceso seguros.

2.3. Metodología empleada

He usado la metodología de desarrollo en cascada para estructurar el desarrollo del proyecto.

- Primero para el proceso de investigación, entré a distintas apps y webs de alquiler y venta de vehículos para ver su diseño y funcionamiento para intentar ver las principales carencias y necesidades que hay en estas aplicaciones.

- Después para el proceso de diseño, primero hice un esquema con las tablas que viera necesarias para la app a partir de las funcionalidades que quería implementar, tras tener esto más o menos decidido hice desde Figma un diseño de como quería que fuera la app visualmente para móvil.

- En la fase de desarrollo empecé con el back añadiendo las dependencias necesarias y haciendo los modelos con sus respectivos repositorios, servicios y controladores y conectando la base de datos.

- Una vez conectada y con los primeros controladores hechos empecé a hacer pruebas con Postman para comprobar su funcionamiento con la base de datos.

- Tras las primeras pruebas en Postman y sabiendo que funcionaban comencé la creación del front y su conexión con el back.

- Cuando ya tuve el back y el front conectado y con las primeras implementaciones comencé con las pruebas de funcionamiento en móvil, la implementación del funcionamiento para web no la implementé hasta que tuve todo funcionando para móvil. Entonces empecé a adaptar las vistas del front para web.

3. Análisis del problema y requisitos

3.1. Descripción de problemas

Actualmente, las plataformas de compraventa de vehículos permiten a los usuarios publicar anuncios sin verificación técnica previa. Esto implica que los compradores interesados deben desplazarse hasta la ubicación del vendedor para inspeccionar personalmente el vehículo, lo cual no siempre garantiza una revisión adecuada ni ofrece seguridad o confianza al comprador.
Por eso quiero integrar un sistema donde cada vehículo publicado en la plataforma debe ser inspeccionado previamente en un taller autorizado asociado a la plataforma.

3.2. Requisitos funcionales

- Cualquiera puede registrarse en la app poniendo algunos datos personales como son su nombre, apellido, correo y estableciendo una contraseña, y con esos datos ya puede acceder a la app con la posibilidad de hacer reservas, comentarios…

- Los usuarios podrán hacer solicitudes para publicar vehículos, y estas solicitudes deberán ser aprobadas por un administrador después de comprobar la veracidad de la información.

- Los usuarios que tengan un vehículo aprobado y esté disponible en la app puede gestionar todo lo relacionado con su vehículo desde su perfil (ver las reservas que le han hecho, aceptar o rechazar las reservas, ver los comentarios que le han puesto).

- Los usuarios registrados podrán realizar la reserva de un vehículo de alquiler, para ello tienen que seleccionar entre los días disponibles del vehículo, tras eso rellenan sus datos completos y ponen un método de pago.

- Los usuarios podrán gestionar sus favoritos, reseñas, reservas, vehículos…

- Los administradores gestionarán las reservas y reseñas de todos los vehículos. Gestionarán los vehículos y los usuarios.

3.3. Requisitos no funcionales

- La aplicación deberá tener soporte para su funcionamiento en web y en móvil. Ahora mismo la parte de la publicación de vehículos desde web da error. Pero el resto de funcionalidades no dan fallo.

- En la ceración de vehículos, todos los campos deberán de ser rellenados.

- Interfaz fácil e intuitiva en todos los procesos de la aplicación. Para que la interacción del usuario sea lo más cómoda posible.

- La arquitectura debe permitir el crecimiento en el número de usuarios y vehículos sin afectar el rendimiento.

3.4. Casos de uso

- Un usuario nuevo se registra, inicia sesión, accede a un vehículo que quiera alquilar y realiza los trámites de alquiler rellenando la información necesaria.

- Un propietario acepta o rechaza la reserva que haya solicitado otro usuario sobre su vehículo.

- Un usuario hace una petición para publicar un vehículo, esta es aceptada y ya puede ser reservado por otros usuarios.

- Después de haber alquilado un vehículo el usuario que haya realizado la reserva, puede dejar una valoración al vehículo para que el resto de usuarios tengan una idea de la experiencia con ese coche.

- Un usuario busca un vehículo con unas características específicas, para eso usa los filtros de búsqueda que hay dentro de la app para encontrarlo.

4. Arquitectura del sistema

4.1. Frameworks y lenguajes usados

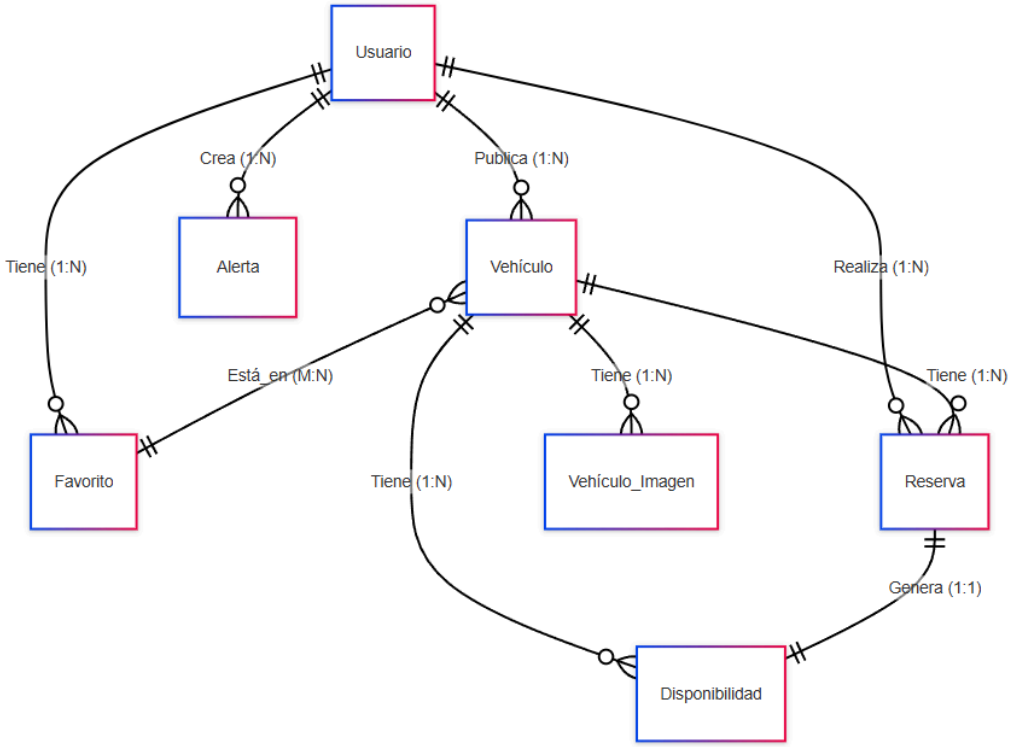
El backend lo he desarrollado en Spring Boot usando Java como lenguaje, Spring Security para la autenticación y autorización, Spring Data JPA para la persistencia y el acceso a datos, Lombok para la reducción y limpieza de código.

El frontend lo he hecho con React Native en Visual Studio usando JavaScript y CSS, implementando librerías como la de lucide para por ejemplo el uso de iconos dentro de la app. He usado únicamente React Native para el diseño móvil y web, simplemente adaptando los estilos para ambos.

Para el control de versiones he usado Github, haciendo una subida tanto del backend como del frontend cada vez que hacía cualquier avance.

4.2. Base de datos

Mi base de datos está creada en MySQL con las siguientes tablas:
- Usuarios: guarda la información personal del usuario registrado.
- Vehiculos: guarda la información del vehículo.
- Reservas: guarda las reservas hecha por los usuarios.
- Reseñas: guarda la reseña de un usuario a un vehículo.
- Disponibilidades: guarda las fechas que están reservadas y esto luego lo consulta la app para mostrar en la reserva los días que están y no están disponibles.
- Alertas: guarda los datos de vehículo que quiera encontrar el usuario para que le notifique.
- Vehiculo_imagen: guarda la ruta a la que accede el front para mostrar las imagenes.



4.3. Justificación

He usado React Native para el front porque creo que es el mejor framework para desarrollar aplicaciones móviles de una manera simple y también te permite configurarlo para su funcionamiento en web.

He usado MySQL para mi base de datos porque es muy fácil de utilizar y se integra muy fácil a Spring añadiendo Spring Data JPA.

He utilizado Spring Boot porque ofrece módulos como Spring Security y Spring Data que te hacen el trabajo más fácil.

He usado git para el control de versiones porque es fácil de usar e intuitivo a la hora de publicar contenido en él.

5. Diseño del sistema

5.1. Diseño funcional

El sistema está diseñado con tres roles diferentes el de administrador, el de usuario y el de propietario.
Usuario no registrado:

- Buscar vehículos
- Ver detalles de vehículos
- Registrarse en el sistema

Usuario registrado:

- Gestionar perfil
- Gestionar favoritos
- Crear reservas
- Dejar reseñas
- Configurar alertas
- Solicitar publicación de vehículo

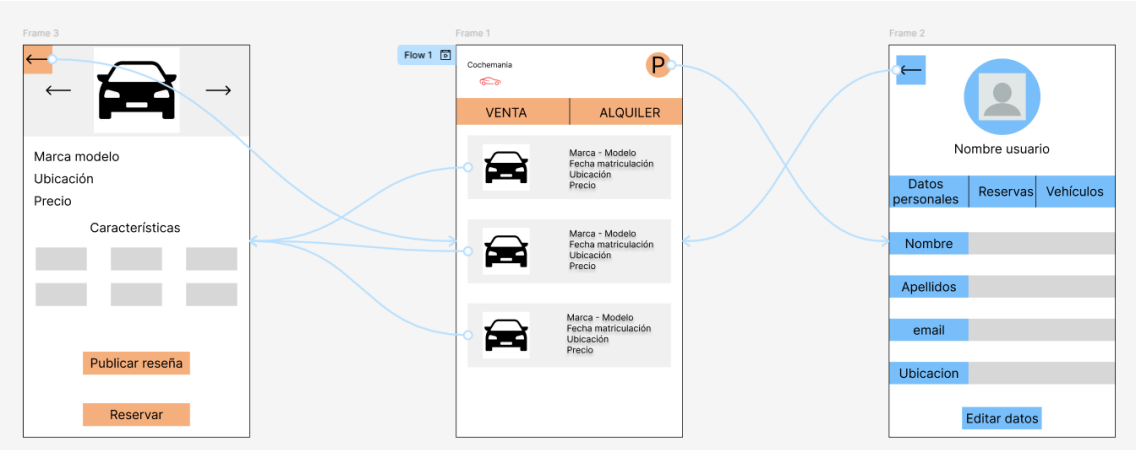
Propietario:

- Gestionar vehículos propios
- Confirmar/rechazar reservas
- Responder reseñas

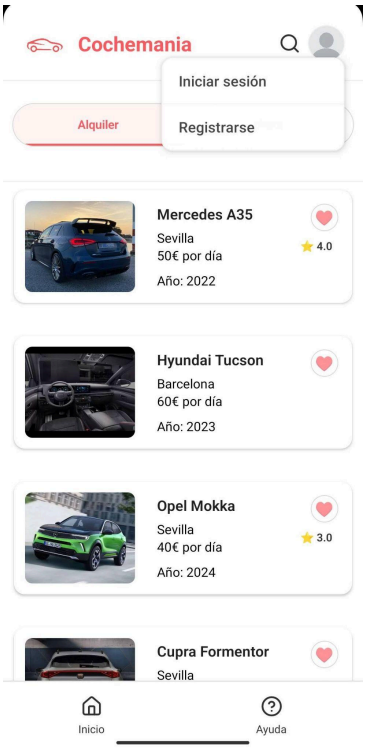
Administrador:

- Gestionar todos los vehículos
- Gestionar todos los usuarios
- Controlar reseñas
- Validar publicaciones

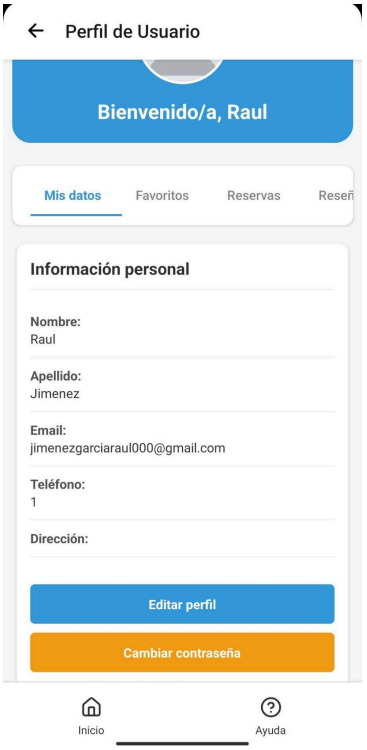
Wireframes
Diseño antes de creación de las vistas de la pantalla principal con acceso al perfil y acceso a la vista detallada de los vehículos:



Vista general de la app.

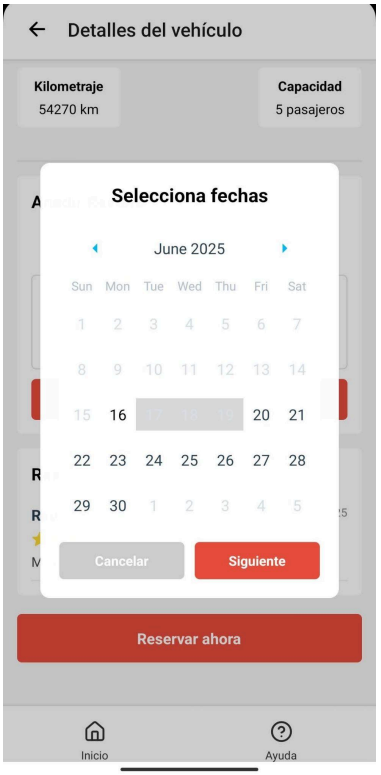


Perfil.



Vista detallada vehículo.

Proceso reserva.



Flujo de Reserva de un Vehículo:

- Un usuario busca un vehículo
- Selecciona las fechas del alquiler
- El sistema verifica la disponibilidad
- Si está disponible, se crea una reserva pendiente de ser aprobada
- El propietario recibe una notificación y confirma o rechaza
- Si se confirma, el vehículo se marca como no disponible para las fechas elegidas y notifica a propietario y user de que se ha hecho la reserva

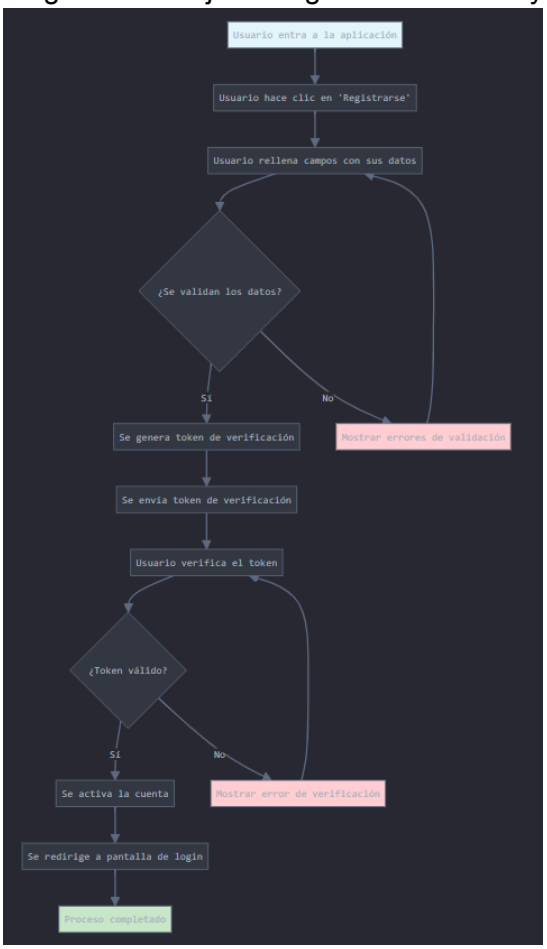
Flujo de Registro de un Usuario:

- Un usuario entra a la app y hace clic en Registrarse
- Es redirigido a un vista donde ingresa sus datos
- Se validan los datos para comprobar que los campos obligatorios estén rellenos y con el formato correcto

Los siguientes pasos son incluyendo el servidor de correo:

- Se genera un token de verificación que es enviado al correo puesto en el registro, este token es un código de 6 digitos que debe poner en la pantalla que le salta al darle a Registrar
- Se verifica que el token sea correcto
- Se crea la cuenta

Diagrama de flujo de registro de usuario y de reserva de vehículo.



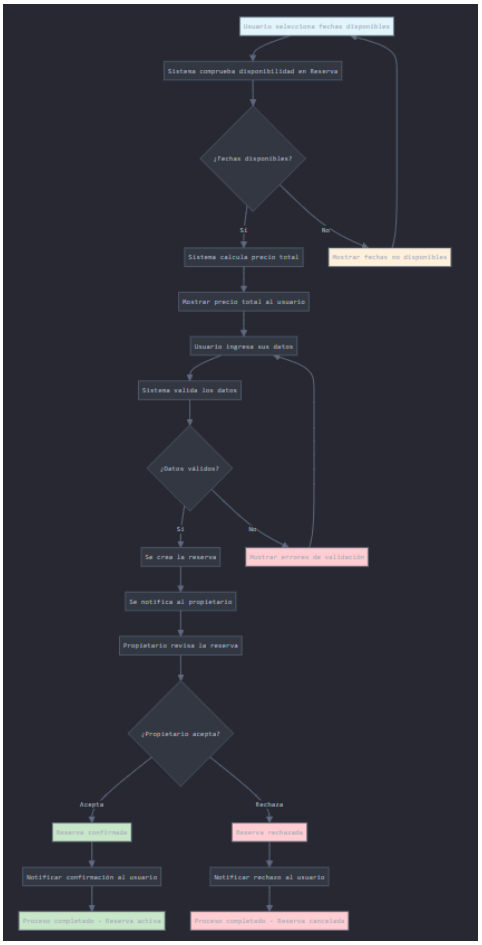
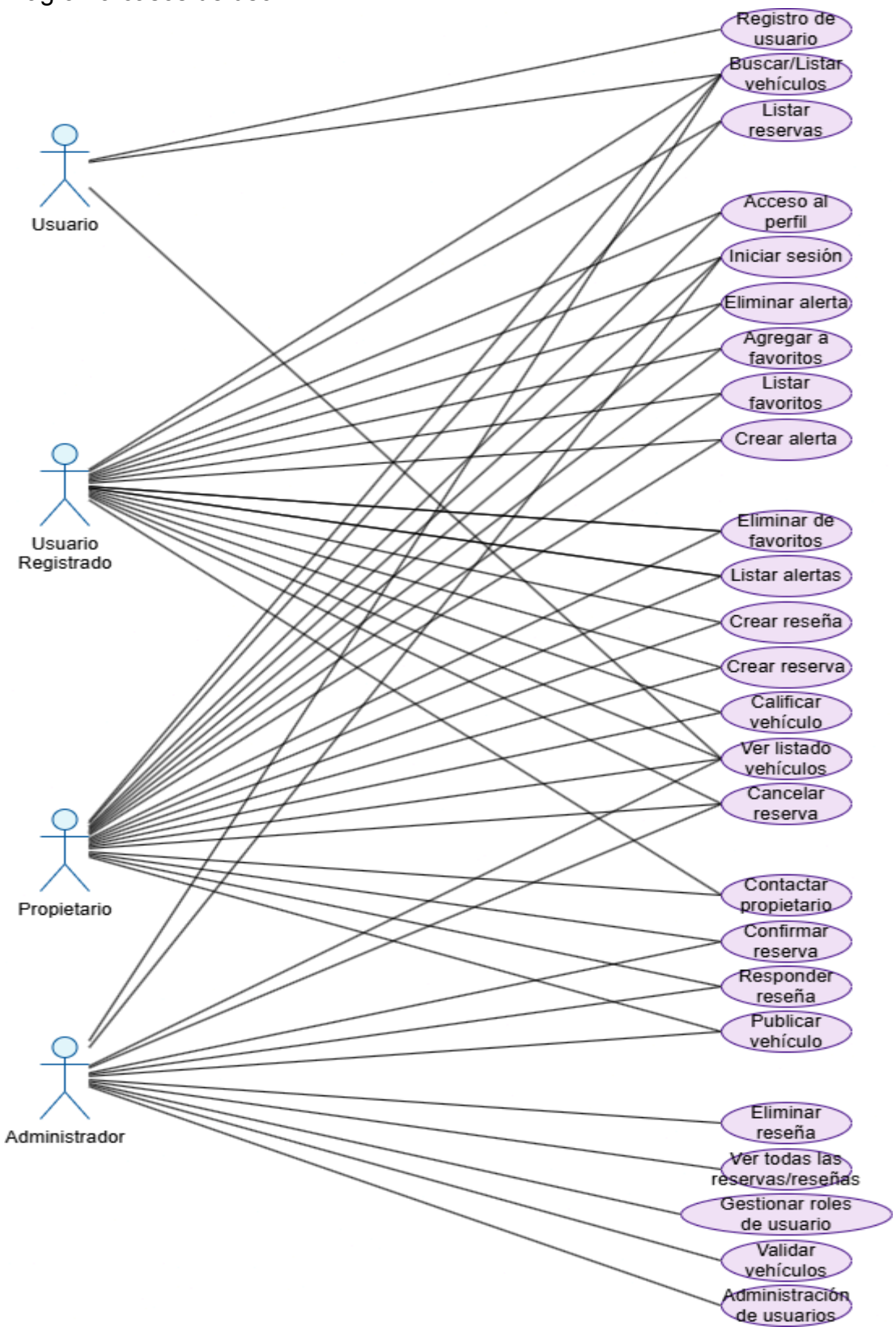


Diagrama casos de uso



5.2. Diseño de la base de datos

5.2.1. Modelo lógico

El sistema de gestión de vehículos consta de las siguientes tablas:

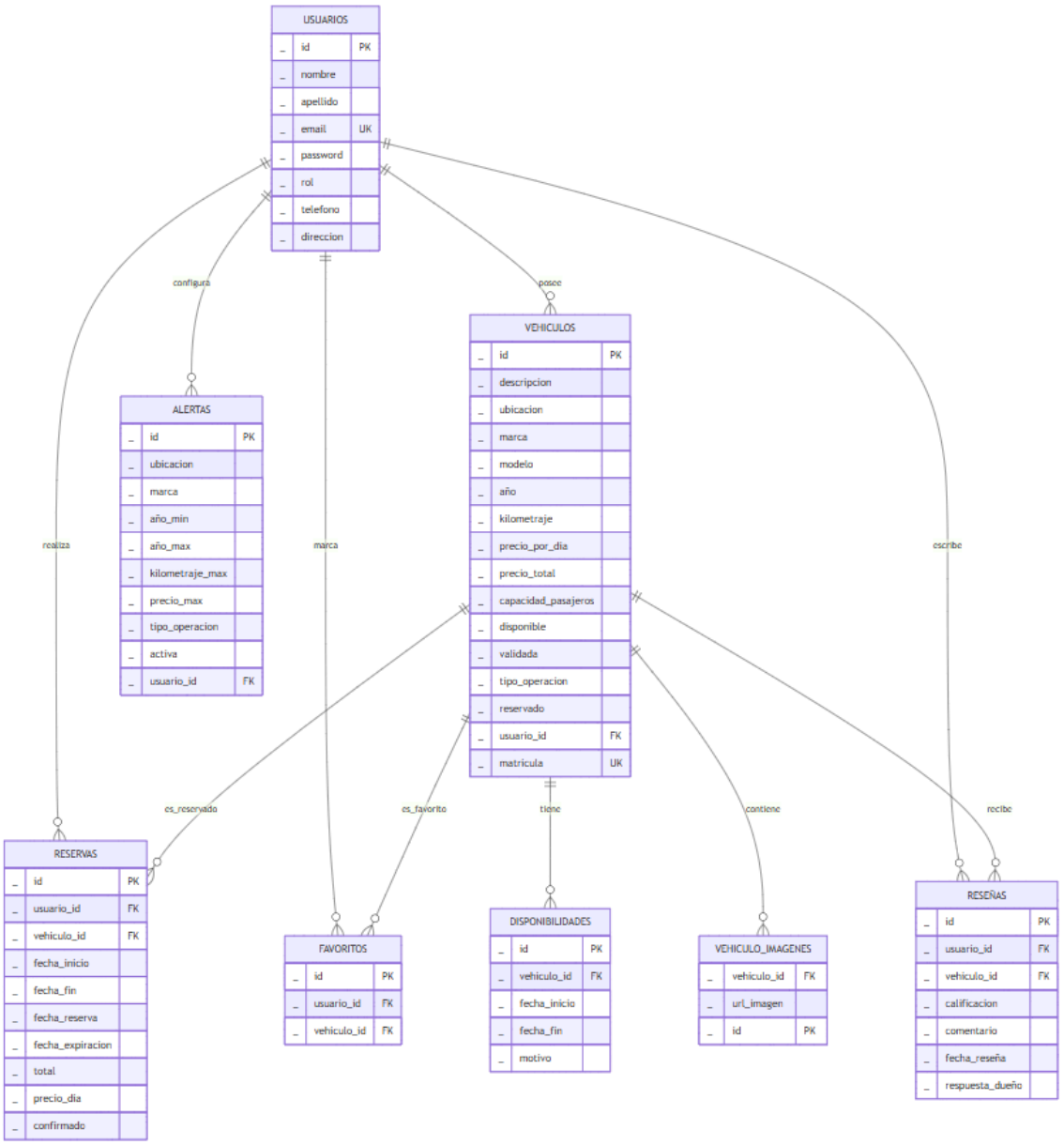
- Usuarios: almacena información de los usuarios del sistema.
- Vehículos: contiene los datos de los vehículos disponibles para venta y alquiler.
- Alertas: registra las preferencias de búsqueda de los usuarios para notificaciones.
- Favoritos: relación entre usuarios y vehículos marcados como favoritos.
- Reservas: gestiona las reservas de vehículos para alquiler.
- Reseñas: almacena las valoraciones y comentarios que los usuarios realizan sobre los vehículos, así como las posibles respuestas de los propietarios.
- Disponibilidades: controla los períodos de no disponibilidad de los vehículos.
- Vehículo_Imágenes: almacena las URLs de las imágenes asociadas a cada vehículo.

Hay implementada eliminación en cascada para:
- Si se elimina un vehículo, se cancelan las reservas sobre ese vehículo, se elimina de favoritos de los usuarios que lo tengan añadido, se eliminan las reseñas que tenga y se eliminan las rutas de las imágenes.

- Si un usuario es eliminado, se cancelen sus reservas, se eliminen sus favoritos, se eliminen sus reseñas y si su reseña tiene respuesta del dueño también será eliminada, se eliminen sus alertas, si tiene vehículos ejecuta la eliminación en cascada de vehículo.

- Si una reserva es cancelada o rechazada, se eliminan de disponibilidades las fechas que estuvieran elegidas para que vuelvan a estar disponibles.

5.2.2. Modelo físico



5.2.3. Claves foráneas

- Vehiculos:
usuario_id (FK a usuarios)

- Alertas:
usuario_id (FK a usuarios)

- Favoritos:
usuario_id (FK a usuarios)
Vehiculo_id (FK a vehiculos)

- Reservas:
usuario_id (FK a usuarios)
vehiculo_id

- Disponibilidades:
vehiculo_id (FK a vehiculos)

- Vehículo_imagenes:
vehiculo_id (FK a vehiculos)

- Reseñas:
usuario_id (FK a usuarios)
vehiculo_id (FK a vehiculos)

5.3. Gestión de seguridad

Autenticación: basada en JWT generados tras iniciar sesión, con expiración configurada.
Autorización: Spring Security permite definir roles (user, propietario, admin) y aplicar filtros según permisos.
Cifrado: las contraseñas son encriptadas con BCrypt antes de almacenarse.
Caducidad a los tokens para que se cierre sesión tras un plazo de tiempo automáticamente.

6. Desarrollo e implementación

6.1. Planificación

6.1.1. Fase 1: Diseño y arquitectura

Definición de requisitos funcionales y no funcionales.
Diseño de la base de datos y relaciones entre entidades.
Creación de diagramas de flujo para los principales casos de uso.

6.1.2. Fase 2: Desarrollo del backend

Implementación de la autenticación JWT con Spring Security.
Creación de los controladores para cada entidad principal.
Desarrollo de servicios y repositorios.
Implementación de validaciones y manejo de excepciones.
Configuración de CORS y seguridad básica.

6.1.3. Fase 3: Desarrollo del frontend

Creación de las pantallas principales.
Implementación de navegación entre pantallas.
Conexión con la API del backend mediante Axios.

6.1.4. Fase 4: Integración completa

Pruebas de integración completa entre frontend y backend.
Pruebas de usuario con casos de uso reales.
Corrección de errores.

6.2. Herramientas y tecnologías empleadas

- Desarrollo backend

Lenguaje: Java
Framework: Spring Boot
Seguridad: Spring Security + JWT
Persistencia: Spring Data JPA
Base de datos: MySQL
Herramientas:

- Lombok: para reducir código repetitivo
- Postman: pruebas de funcionamiento

- Desarrollo frontend

Framework: React Native
Lenguaje: JavaScript
Librerías principales:
React Navigation para gestión de rutas
Axios para llamadas HTTP
Lucide para iconos
Herramientas:

- Visual Studio Code: IDE principal
- Expo: para desarrollo y pruebas en dispositivos

- Otras herramientas

Control de versiones: GitHub
Diseño: Figma

7. Descripción funcional del sistema implementado

7.1. Funcionalidades

- Registro de usuarios con los campos nombre, email y contraseña obligatorios, el resto no son obligatorios. Aunque no está implementado el servidor de correo, quiero que para registrarte, tengas que confirmar el registro mediante el envío de un código de verificación que será enviado al correo puesto, ese código lo tienes que poner en una pantalla que sale al registrarte.

- Inicio de sesión usando el email y la contraseña establecidos al crear la cuenta.

- Cambio de contraseña, desde el perfil le das a cambiar contraseña, al igual que al registrarte te envía un código y una vez puesto el código puedes poner la nueva contraseña.
Si quieres cambiarla porque la has olvidado, en las vistas de iniciar sesión y registrarse hay un botón que dice '¿Olvidaste tu contraseña?' que al darle te aparece para poner el email de tu cuenta, esto hace que se te envíe un correo con un código y cuando pones el código ya puedes establecer tu nueva contraseña.

- Puedes marcar vehículos como favoritos tanto desde la vista general de la app dándole al icono del corazón que está en el cuadro del vehículo como si has accedido a la información del vehículo desde el mismo icono del corazón.

- Si haces clic sobre un vehículo se abre la información detallada del vehículo donde puedes ver todas las imágenes del vehículo.
También puedes desplazandote hacia abajo en los vehículos de alquiler hacer una reserva dandole al botón reservar, tras darle al botón tendrás primero que elegir entre las fechas disponibles las que tú quieras, luego tienes que rellenar toda tu información y hacer la reserva.
En los vehículos en venta en vez de hacer reserva pone 'Contactar con el vendedor' que lo que hace es redirigirte al correo con el email del usuario que vende el coche puesto.
En esta vista también puedes poner reseñas, para esto lo único que tienes que poner es una calificación de 1 a 5 y un comentario. También se muestran las otras reseñas que tenga ese vehículo y una media entre las reseñas que tenga.

- Desde el perfil puedes modificar toda tu información exceptuando el email. También tienes acceso al apartado 'Favoritos' donde se muestran los vehículos que tengas marcados como favoritos desde aquí si quieres puedes quitar los vehículos de favoritos.
Las reseñas que hayas escrito se mostrarán en el apartado 'Reseñas' desde donde podrás eliminarlas si quieres.
En 'Mis reservas' se muestrán las reservas que hayas hecho, se muestra si está pendiente de que la acepten o si ya está confirmada, también podrás cancelarlas desde aquí.

Desde el apartado ‘Mis vehículos’ los usuarios pueden hacer las peticiones para publicar sus vehículos. Lo único que tienen que hacer es rellenar la información del vehículo y una vez enviada la solicitud aparece como pendiente de aprobación y esto cambia cuando un admin lo revisa y aprueba y hasta que esta no sea aprobada el vehículo no se muestra en la app para los otros usuarios.
En el apartado ‘Alertas’ puedes crear una alerta que te notificará si se publica algún vehículo con las características que tu hayas puesto al crear la alerta, al crearla te enviará automáticamente un correo con los vehículos que ya cumplan con las características especificadas. En cualquier momento estas alertas podrán ser eliminadas por el usuario.

- Si un usuario tiene un vehículo publicado, su rol cambia a ‘Propietario’, esto hace que en el perfil se muestren dos apartados más que son para gestionar las reseñas en sus vehículos y las reservas. Pudiendo contestar a las reseñas y tambien pudiendo aceptar o rechazar las reservas.

- El icono de la lupa que hay en la parte superior abre un pequeño desplegable con filtros para hacer una búsqueda exhaustiva entre los vehículos.

- Desde el panel de administrador puedes gestionar las reservas de todos los vehículos.
Gestionar los usuarios, puedes cambiar sus datos, cambiarles el rol y eliminarlos.
Gestionan todos los vehículos pueden crear nuevo vehículos, aceptar o rechazar las peticiones de publicación de los usuarios y eliminar vehículos.
Gestionan las reseñas que haya en los vehículos pudiendo responder a ellas y eliminarlas.

7.2. Estructura del subsistema

Los usuarios que se registren en la aplicación serán identificados automáticamente con el rol USER. Este rol puede ser modificado por un administrador o si este usuario publica un vehículo y este es aprobado se actualiza su rol a PROPIETARIO.

La seguridad tiene:
Autenticación basada en JWT
Generación de tokens con expiración
Endpoint protegido para validación de tokens
Configuración basada en roles con Spring Security
Cifrado BCrypt para contraseñas
No almacena información sensible en JWT
Protección CORS configurado

7.3. Funcionalidades no implementadas

- Todo lo que tiene que ver con las notificaciones y alertas no está implementado porque no tengo servidor de correo SMTP que usar.

- Tampoco tiene un sistema de chat en vivo para que los usuarios puedan hablar y contactar con los dueños.

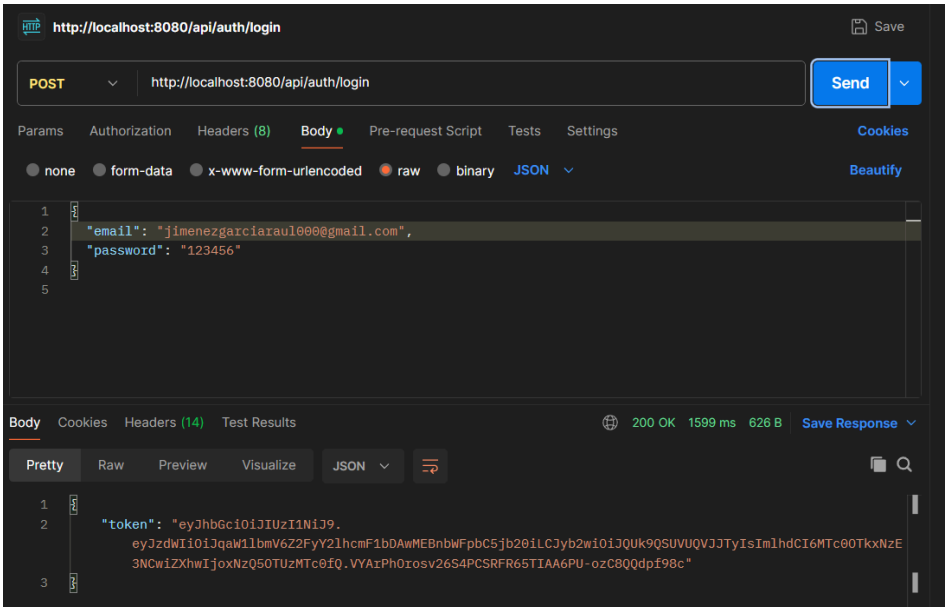
- Un filtro para realizar búsquedas según las fechas disponibles para los vehículos de alquiler.

- Implementar un filtro para ordenar los vehículos de manera ascendente y descendente por precio.

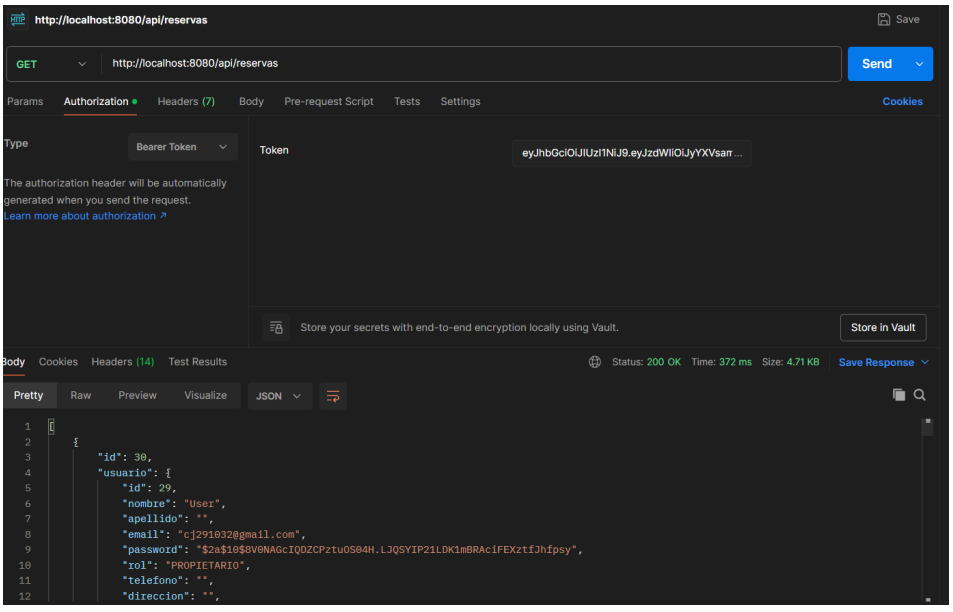
- Implementar un sistema de verificación de vehículos y propietarios.

8. Pruebas y validación

Para las pruebas funcionales he estado usando postman para comprobar que todo lo que iba implementando funcionara.
Por ejemplo:
Inicio de sesión de usuarios.



Obtener reservas de un usuario.



Una vez ya tuve lo principal del back hecho, empecé a conectarlo con el front. Y cada vez que hacía algo en el back, hacía una prueba de integración en el front para comprobar que funcionara.

Una vez ya estaba la aplicación más o menos funcional, realicé pruebas de usabilidad con mi hermano para comprobar que los procesos de la aplicación eran más o menos intuitivos, estaban explicados con claridad y era fácil navegar por ella.

9. Despliegue

Clonar el repositorio de git.
Abrir el back en Spring Boot e iniciarlo.
Abrir el front en Visual Studio, hacer los siguientes comandos:

- yarn install
- yarn add expo
- npm install axios
- yarn start

Con este último iniciamos el front. Una vez hecho todo esto, ahora accedemos desde el móvil a la aplicación de Expo Go y escaneamos el código QR que se genera en Visual Studio.

10. Conclusiones

Mi aplicación de compraventa y alquiler de vehículos funciona para web y móvil y creo que puede ser útil debido a la falta de aplicaciones que incluyen compraventa y alquiler de vehículos en una misma. Aunque mi aplicación tiene limitaciones y cosas que mejorar, creo que con el auge del mercado de vehículos de segunda mano puede ser útil.

11. Posibles mejoras y líneas de evolución futura

- Más adelante me gustaría que para los vehículos en venta se necesite una comprobación en taller del estado del vehículo para comprobar que esté todo correctamente y no se ponga un vehículo como que esté en perfecto estado cuando no lo está.

- También me gustaría implementar un sistema de verificación para comprobar que quien publica un vehículo sea el dueño de él pidiendo documentación del vehículo y del usuario.

- Una de las funcionalidades más importantes que me gustaría añadir sería un sistema de chat para poder comunicarse entre los propietarios de los vehículos y los usuarios. Tanto para alquiler como para compra.

- Un filtro para buscar por fechas disponibles para los vehículos de alquiler.

- Teniendo en cuanta que los ingresos de la aplicación creo que serían por comisiones en las ventas y alquileres, me gustaría implementar que para los vehículos de alquiler los usuarios tengan que contratar un seguro. Así trabajando con alguna empresa de seguros tendría más beneficios.

- La implementación de un servidor de correo SMTP para que funcionen todo lo que tiene que ver con notificaciones por correo en la app (reestablecimiento de contraseñas, notificación de solicitudes de reserva, respuesta a las solicitudes…).

- Implementar edición de vehículos y funcione de la siguiente manera. Un usuario edita su vehículo y antes de cambiarse la información esta es enviada para revisión de un administrador, mientras se aceptan los cambios o no este vehículo dejará de estar visible en la app y no se podrán hacer reservas ni reseñas ni añadirlo a favoritos mientras esté pendiente,pero no se cancelarán sus reservas ni se modificarán los datos de la reserva como el precio si es actualizado.