# TFG - Greenst


- **Centro de estudios**: Instituto Tecnológico de Edix.
- **Grado**: DAW.
- **Fecha**: 20 de Mayo de 2022..
- **TFG realizado por**: Adrián Nuevo Pérez y Carlos Fuentes Vázquez.

# Introducción, justificación y objetivos

La idea de esta aplicación es sobre la creación de una empresa ("***Greenst***") de alquiler de coches interurbanos que cumplan las leyes medioambientales actuales. Es bien sabido que, el medioambiente es uno de los grandes problemas de la actualidad y, aunque cada vez se están tomando medidas que apoyen las leyes medioambientales que se están desarrollando, en la actualidad siguen existiendo residuos químicos generados por las industrias, emisiones vertidos a la atmósfera, y, uno de los mayores problemas: el procesamiento y refinamiento de combustibles fósiles, como el carbón o el petróleo.

Los residuos de las fábricas y las emisiones de los automóviles siempre han sido unos de los mayores vertientes de emisiones contaminantes en las ciudades. En los últimos años, se han implementado medidas para reducir el impacto de estos, como el decrecimiento del número de fábricas, zonas de bajas emisiones (como "Madrid Central"), la generación de energías renovables o, el transporte público ecológico, es sobre este último tema sobre el que nuestra empresa se apoya.

Greenst es una empresa de alquiler de coches interurbanos con un modelo ecológico. Todos sus vehículos son considerados ecológicos ya que, son vehículos eléctricos que no utilizan ningún derivado del petróleo para su funcionamiento.

Los clientes podrán alquilar un vehículo según sus necesidades y preferencias para el transporte por la ciudad. Los vehículos cumplen todas las medidas medioambientales, lo que hace que, puedan acceder a cualquier punto de la ciudad, incluyendo las zonas de bajas emisiones. El modelo de negocio es sencillo:

Un cliente alquila uno de los vehículos y va a recogerlo a una de las oficinas, el cliente puede disponer del vehículo el tiempo que necesite y, lo devolverá a una de las oficinas cuando ya no necesite de él. Hay oficinas rodeando prácticamente los mayores puntos de interés de la zona céntrica de la ciudad, por lo que, siempre dispondrá de una oficina cerca o incluso lo utilizará para trasladarse de un punto a otro de la ciudad pasando por las zonas con las que con un vehículo normal y corriente no podría acceder. Se dispone de varios tipos de vehículos según la necesidad del cliente, desde coches hasta motos y furgonetas (VAN). Así se aseguran de cumplir la necesidad de los clientes.

**Objetivos**:

- El presente TFG tiene como objetivo analizar y aplicar distintas herramientas y tecnologías que se presentan actualmente en el mercado.
- Enfocar lo máximo posible el trabajo a la metodología MERN-stack.
- Poner a prueba la capacidad de análisis de los participantes.
- Desarrollar la capacidad de autoaprendizaje.
- Descubrir nuevas metodologías de trabajo.
- Emplear los conocimientos en realizar una aplicación lo más cercana a la vida real posible.
- Aplicar las buenas prácticas del desarrollo de software.

# Agradecimientos

Tras dos intensos años en el Grado Superior de Desarrollo de Aplicaciones Web en el Instituto Tecnológico EDIX, llega el momento de finalizar esta gran etapa con el Proyecto de Fin de Grado. Durante el transcurso de estos dos años, hemos conocido a grandes personas que nos han ayudado a que el camino fuera más sencillo y llevadero, es por ello que les queremos dedicar unas palabras:

- Agradecer a todos los profesores del instituto, por su dedicación hacia la enseñanza y por despertar en nosotros el entusiasmo por aprender cosas nuevas, así como la dedicación que han demostrado hacia nosotros para que saliéramos adelante.
- Agradecer a todos los compañeros, que a pesar de la modalidad a distancia, hemos conseguido crear una piña en las buenas y las malas, donde nos apoyamos unos a otros cuando surgen problemas.
- Agradecer a nuestra familia y amigos, por apoyarnos siempre sin importar la situación, por creer en nosotros cuando nosotros mismos no lo hacemos y empujarnos a no perder nunca de vista el objetivo de sacar adelante cualquier cosa que nos propongamos.

## Palabras clave

- **Frontend:** Parte de la aplicación con la que interactúan directamente los usuarios. Destaca por el desarrollo en los apartados visuales, interactividad y usabilidad. No realiza operaciones directamente a bases de datos o servicios, para ello se tiene que comunicar con una aplicación de servidor (backend) a través de peticiones, normalmente HTTP.

- **Backend:** Aplicación de servidor encargada de orquestar diferentes servicios, acciones, Bases de Datos, y definir la lógica y reglas de negocio, la cual enviará y recibirá solicitudes del frontend.

- **CRUD:** Módulo de una aplicación, ya sea en el frontend o el backend, dedicada especialmente a operaciones de escritura, lectura, modificación o eliminación de una base de datos.

- **SOLID:** Principios o buenas prácticas para escribir y estructurar el código de las aplicaciones. S de Single Responsability, cada componente, clase, función o cualquier otra unidad de código debe tener una responsabilidad única. O de Open Closed, cada unidad de código debe ser diseñada de forma que no se pueda modificar en un futuro, pero si se pueda expandir su comportamiento. L de Liskov Substitution, principio que se basa en la herencia y el polimorfismo para poder reemplazar objetos de diferentes clases, pero de la misma naturaleza, normalmente a través del uso de interfaces. I de Interface Segregation, principio que aplica Single Responsability a las interfaces, dónde se establece que las interfaces deberían de describir sólo elementos útiles para las clases que las implementan, y no obligar a las clases a utilizar métodos o propiedades que no necesitan. Interface Segregation propone dividir una interfaz, para que quede sólo con los elementos necesarios para las clases que las implementan. D de Dependency Injection, principio que establece que el código debe ser diseñado de forma que los elementos presenten bajo acoplamiento entre sí, de forma que cada elemento que interactúa con otro, pueda ser fácilmente reemplazable por otro elemento que pueda reproducir las mismas entradas y salidas, pero con distinto comportamiento.

- **DDD:** Domain Driven Design, patrón de diseño que establece un estándar en la estructura de un sistema informático, dividiendo el sistema en contexts, que son áreas macro del sistema, que pueden funcionar perfectamente como subsistemas independientes, y a su vez cada context en entidades llamadas AggregateRoot, las cuales son cada uno de los elementos que participan en los procesos del sistema, a su vez cada AggregateRoot se divide en aplicación, infraestructura y dominio. El dominio son los elementos escritos específicamente para el sistema, aplicación son los servicios que se comunicaran al exterior, e infraestructura son todos los elementos que no dependen del sistema sino de terceros.

- **JWT:** (JSON Web Token), es un estándar que define una manera segura de transmitir información a través de JSON. Lo que hace a JWT seguro es su firma digital, la cual permite que quien lo recibe pueda verificar su autenticidad y saber si ha sido modificado o no.

# Índice


| | |
|---|---|
| Introducción, justificación y objetivos | 1 |
| Agradecimientos | 3 |
| Palabras clave | 4 |
| Índice | 6 |
| Resumen | 6 |
| Módulos formativos aplicados en el trabajo | 7 |
| Herramientas / Lenguajes utilizados | 9 |
| Componentes del equipo y aportación realizada por cada estudiante | 14 |
| Fases del proyecto | 17 |
| Desarrollo de la idea | 17 |
| Modelo de datos utilizado. | 18 |
| Conclusiones | 24 |
| Puntos a mejorar | 24 |
| Bibliografía | 25 |
| Anexos (Optativos) | 26 |

# Resumen

Este Trabajo de Fin de Grado del ciclo superior formativo de “Desarrollo de Aplicaciones Web”, expone la formación y conocimientos que se han adquirido durante el curso.

En la actualidad, existen diversas tecnologías; un grupo de tecnologías reducidas para realizar funciones específicas o un gran abanico para realizar la misma función. Es cierto que, hay veces que algunos grupos no están preparados para tener una compatibilidad entre sí y, se puede dificultar el hecho de realizar una aplicación completa. Este no es el caso, el concepto de desarrollo en tecnologías “MEAN” o “MERN”, brinda la posibilidad de desarrollar aplicaciones “full-stack” de código abierto en JavaScript.

Y qué mejor forma de adquirir y afianzar conocimientos con este tipo de tecnologías que realizando una aplicación que simule a la vida real, en este caso *“**Greenst**”*. La aplicación incluye un sistema de gestión de Bases de Datos estructurada por BSON, que es un formato de intercambio de datos “No-SQL” en MongoDB, incluye un sistema de gestión de back-end, como NodeJS que es un sistema para la utilización de JavaScript en el servidor, a la par que Express que facilita la realización de APIS. O React, que es una librería para la formación de la estructura del front-end (entorno cliente).

## Módulos formativos aplicados en el trabajo

Durante la realización del proyecto, hemos intentado implementar el mayor número posible de asignaturas realizadas durante el curso. Para así aprovechar al máximo la formación obtenida durante estos dos años. Las asignaturas usadas son las siguientes:

- **Programación**:

Fundamentos teóricos de programación básica, estructura de proyectos, generación de buenas prácticas, estructura y declaración de datos e interacción con métodos.

- **Bases de Datos**:

En este caso no se han utilizado las tecnologías cursadas durante el curso, ya que la aplicación utiliza MongoDB, una base de datos "no relacional". Lo cual difiere de la tecnología usada durante el curso, SQL.

Pero los conocimientos adquiridos son indispensables para la estructuración del proyecto, ya que ayuda a diferenciar los distintos componentes de la aplicación (Vehículos, Usuarios, Reservas…) y la información que debe de llevar cada uno para poder realizar una aplicación completa y coherente.

- **Entornos de desarrollo**:

Se han utilizado los conocimientos adquiridos en la asignatura en todo el proyecto, desde el inicio cuando se estructuró el proyecto, mediante los diagramas de "clases" y de "casos de uso". Hasta el final en el cual se aplicó la depuración de código para que este quede legible y bien estructurado.

- **Lenguaje de Marcas y Sistemas de Gestión de Información**:

Se han aplicado los conocimientos obtenidos en HTML CSS de esa asignatura para desarrollar la parte "Front" del proyecto para que sean estéticas, accesibles y concisas.

- **Desarrollo Web Entorno Servidor**:

Aunque el proyecto no se ha desarrollado con Java, los conocimientos obtenidos en esta tecnología han sido potencialmente útiles para la realización de este proyecto. En esta asignatura, se adquirieron las habilidades para desarrollar software utilizando herramientas como la programación “Orientada a Objetos”, también se motivó a la investigación de principios “SOLID”, patrones de diseño “DDD” y “Clean-Architecture”.

- **Desarrollo Web Entorno Cliente**:

Fundamental la base obtenida en JavaScript para la realización del proyecto y la utilización de frameworks o librerías como NodeJS, ExpressJS o ReactJS, las cuales han sido utilizadas en el proyecto.

- **Inglés Técnico:**

Muy útil a la hora de entender terminologías o generar un vocabulario específico para el proyecto. También, muy útil para la búsqueda de información sobre las tecnologías, ya que es el idioma estándar de este mercado laboral.

# Herramientas / Lenguajes utilizados

**Visual Studio Code**:

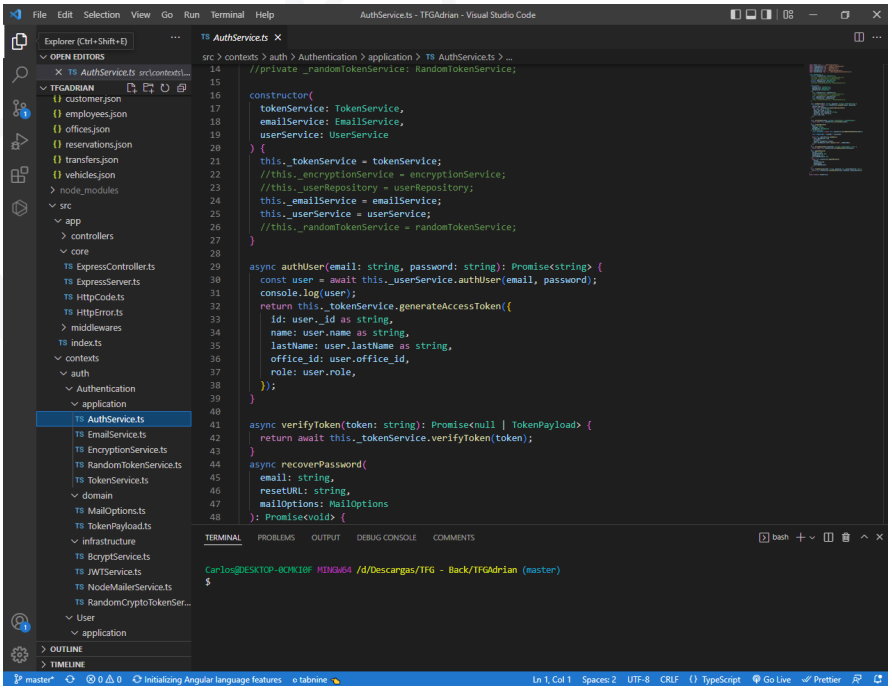
Visual Studio Code ha sido el principal editor de código. En este entorno se ha realizado todo el proyecto, y gracias a la gran cantidad de opciones que ofrece, cada integrante ha podido personalizarlo a su gusto para poder desarrollar de una forma cómoda. Existen customizaciones desde atajos de teclado para la edición más rápida del código, extensiones de terceros para agilizar el desarrollo de la aplicación, etc… Algunas de estas extensiones son:

- **Prettier**:

  Es una extensión básica para el formateo del código y que nuestro código quede legible y bien organizado. Ofrece soporte para multitud de lenguajes y además ofrece una gran personalización.

- **TabNine AI**:

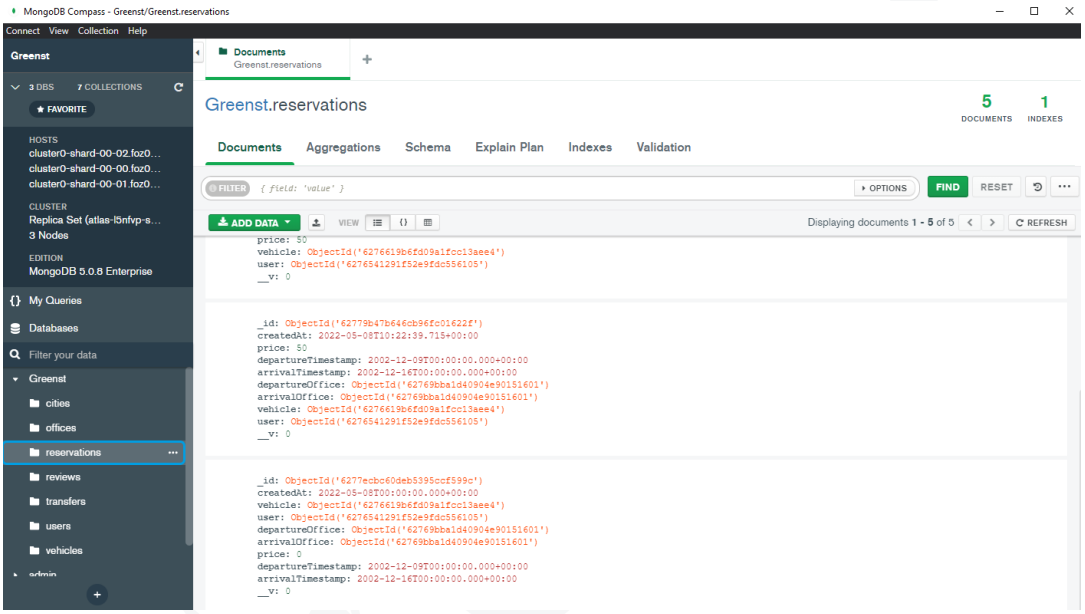
  Extensión que agiliza la creación de código mediante sugerencias producidas por Inteligencia Artificial.



**MongoDB Compass**:

MongoDB Compass es una herramienta interactiva para consultar, optimizar y analizar los datos en MongoDB.

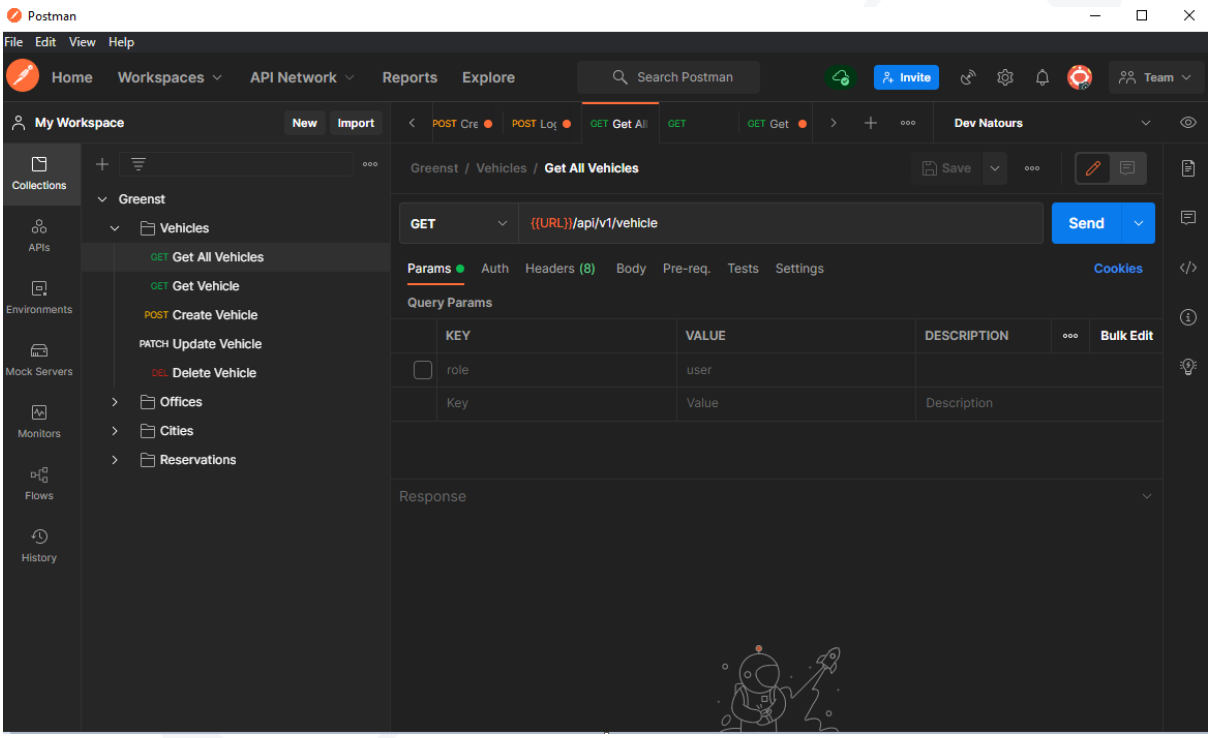
Ha sido otra de las bases en este proceso de desarrollo ya que la base de datos está en MongoDB y gracias a esta herramienta se pueden consultar los datos introducidos, así como editarlos y ver estadísticas sobre el cluster de datos.



## Postman

Postman ha sido un pilar fundamental para el desarrollo. Es una aplicación que permite realizar pruebas API. Es un cliente HTTP que ofrece la posibilidad de testear "HTTP Request" a través de una interfaz gráfica de usuario, por medio de la cual se obtienen diferentes tipos de respuesta que posteriormente serán validadas.

Ofrece los métodos necesarios para interactuar con los "endpoints": **GET**, **POST**, **PUT**, **PATCH** y **DELETE**, así como la visualización de los códigos de respuesta. También permite crear distintos ambientes para la creación de variables.



**MERN-Stack:**

Lo componen las siguientes tecnologías:

- **MongoDB:**

  Es un sistema de base de datos "noSQL" (Not Only-SQL) orientado a documentos y de código abierto.

  Al ser una base de datos noSQL o no relacional, los datos no se guardan en tablas, si no que se guardan en documentos con estructura BSON (parecido a JSON), con un esquema dinámico haciendo que la integración de datos sea más fácil.

- **ExpressJS:**

  Es un framework minimalista para Node JS que permite estructurar la aplicación de forma ágil y proporcionar utilidades como el enrutamiento y la gestión de las sesiones.

- **ReactJS:**

  Creado por Facebook. Es una biblioteca de JS de código abierto muy efectiva utilizada para crear interfaces de usuario intuitivas.

  Es más ligera que Angular, y gracias a su poco tamaño de registro, adaptabilidad, similitud con numerosas bibliotecas tiene una curva de aprendizaje moderada.

- **NodeJS:**

  Desarrollado sobre la base del motor Chrome V8. Es un entorno de código abierto pensado para el desarrollo de aplicaciones JavaScript en la parte del servidor. Esto hace que estas aplicaciones puedan ejecutarse dentro de diferentes SO como Microsoft Windows, Linux o OS X.

  Además implementa una amplia biblioteca de módulos JS que simplifica el desarrollo de aplicaciones web.

**Otras tecnologías:**

- **TypeScript:**

  Es un lenguaje de programación de código abierto desarrollado y mantenido por Microsoft.

  Es un superconjunto de JavaScript que añade tipos estáticos y objetos basados en clases. Esto es una gran ventaja ya que al extender de JS se puede hacer uso del mismo en cualquier momento del código y este seguirá funcionando.

  Otra de las grandes ventajas es que se trata de un lenguaje compilado por lo que detecta los errores mientras escribimos el código.

- **Bootstrap:**

  Bootstrap es un framework de código abierto que utiliza diseños basados en CSS y HTML. Se trata de un marco popular que se ocupa sólo de las aplicaciones front-end (formularios, componentes de la interfaz y complementos de JavaScript).

## Componentes del equipo y aportación realizada por cada estudiante

Una vez surgió la idea del proyecto, los componentes del equipo tras varias sesiones de trabajo llegaron a una idea general de cómo iba a ser el proyecto. Primero se estructuró el Back para así poder gestionar de qué elementos iba a disponer la aplicación.
Una vez creada la idea general del Back , se dispusieron a formar la estructura de Front, para decidir que se iba a mostrar al usuario y cómo hacer uso del Back anteriormente diseñado.

La metodología de trabajo ha consistido en dividir el proyecto en dos partes, para que cada uno de los componentes del equipo realice su parte. Una vez avanzado individualmente se exponen los avances para realizar correcciones y aportar ideas o mejoras a la parte del otro compañero, así como unificar el código para que este disponga de una cohesión general.

**BackEnd**:

- **Adrián:** Adrián se ha encargado del desarrollo del context de "*backoffice*", el cual es el apartado del backend para la administración de catálogos de la empresa, como empleados, clientes, vehículos, marcas, modelos, etc. Además construyó los respectivos controladores del backoffice.

- **Carlos:** Carlos se ha encargado de desarrollar el context de "*website*". Es la parte del backend dedicada al website que visitan los clientes para reservar vehículos. En esta parte del backend, se gestiona el registro de clientes, consulta de vehículos disponibles y filtros de marcas, modelos, oficinas y ciudades.

  Además desarrolló el context de auth, el cual es el módulo del backend especializado en la autenticación y autorización de usuarios, mediante JWT.

**<u>FrontEnd</u>:**

- **Adrián**:

  Desarrolló la aplicación en ReactJS, que se comunica con todos los módulos del backend. Utiliza el módulo de auth para realizar todas las operaciones de autenticación y autorización a través de Bearer Token y el módulo de "*backoffice*" para realizar los respectivos CRUDS de la base de datos.

- **Carlos**:

  Desarrolló el módulo de website para la página principal dónde los clientes consultan y reservan vehículos.

# Fases del proyecto

## Desarrollo de la idea

Al final del curso, es necesario realizar un proyecto de fin de grado, y ante tal idea, los dos estudiantes del grupo se reunieron a pensar que podría formar un buen proyecto.

Surgieron varias ideas iniciales, como la realización de una tienda web, o la página de un restaurante, pero finalmente se fijaron en el panorama actual de movilidad y vieron que en los últimos años había crecido mucho la movilidad sostenible con coches eléctricos y el carsharing. Fue por ello que finalmente decidieron llevar a cabo el proyecto presentado en este documento, en el cual desarrollan una aplicación de alquiler de vehículos sostenibles llamada Greenst.

Una vez tuvieron la idea del proyecto, se pararon a pensar que fases iban a requerir para llevarlo adelante. Primero marcaron un esquema básico de las funcionalidades de su aplicación como son la visualización de los coches y su reserva, y a continuación la estructura de sus usuarios, con dos grandes tipos, el cliente que va a disfrutar de la aplicación y el empleado que gestiona que dicha aplicación funcione correctamente.

Una vez realizada la estructura básica, pensaron en que modelo de datos iban a usar y realizaron los casos de uso a continuación descritos.

**Diagramas de clases y casos de uso:**

**Diagrama de casos de uso general**

En nuestra aplicación va a haber dos grandes tipos de usuarios, clientes y empleados.

Los clientes pueden ver y reservar los coches y gestionar dichas reservas realizadas por los mismos.

Dentro de los empleados hay dos categorías, el gestor que se encarga de administrar los vehículos, las reservas y los clientes y el CEO (encargado) que se encarga de gestionar las oficinas y los empleados.

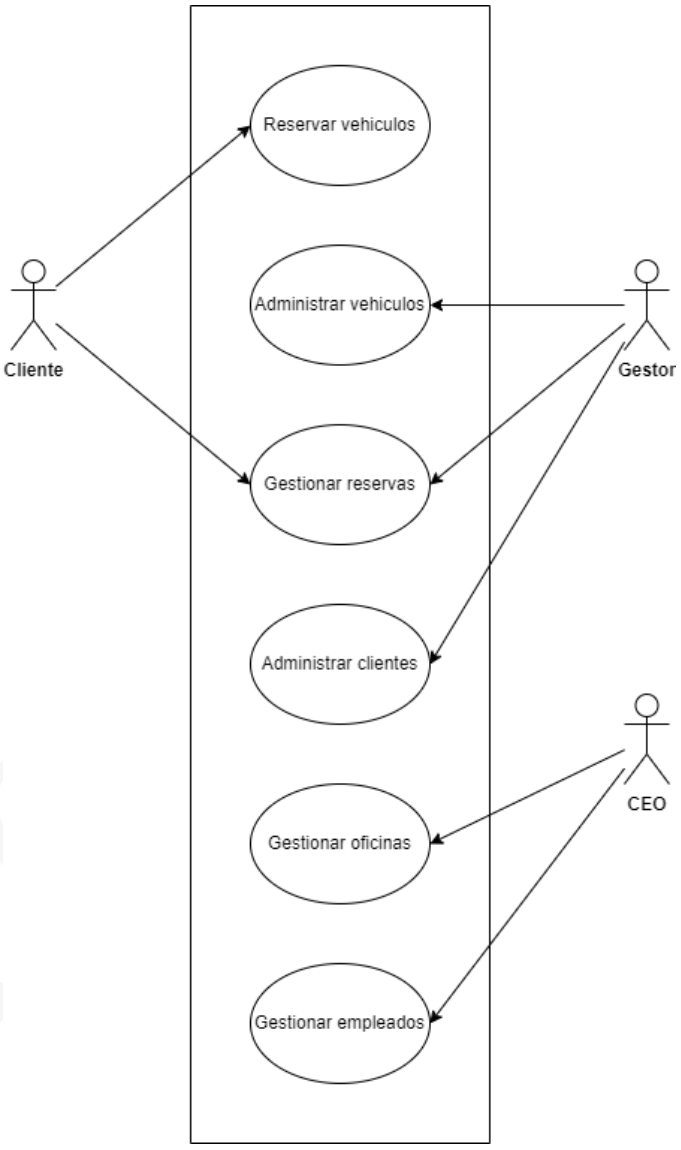


**Diagrama de casos de uso - Caso cliente**

De forma más detallada, los clientes pueden consultar los vehículos para ver su disponibilidad y reservarlos.

También pueden gestionar sus reservas para actualizarlas o cancelarlas.

Para poder realizar todas las acciones anteriores es necesario tener una cuenta de cliente, por lo que se deben registrar y logear primero.

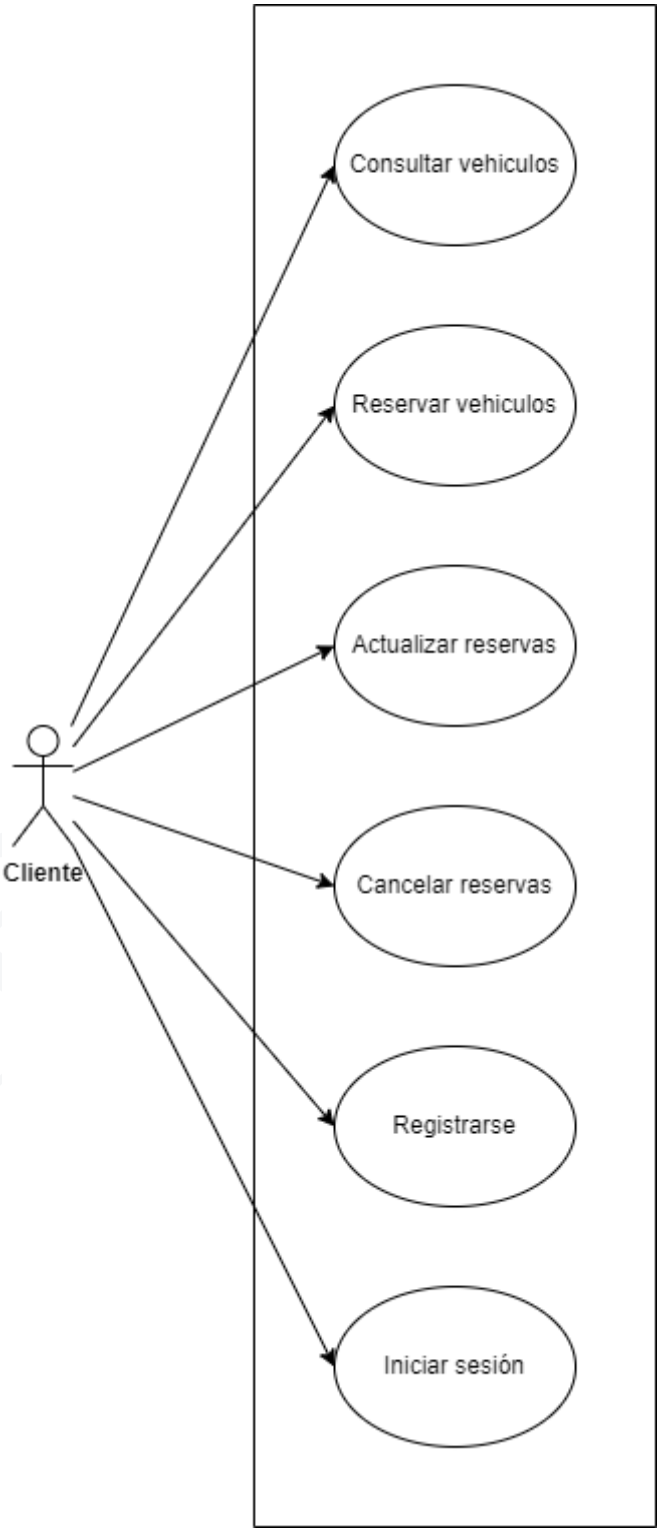


**Diagrama de casos de uso - Caso gestor**

El gestor se encarga de administrar los clientes y sus reservas, además puede realizar las funciones de cliente como consultar y reservar vehículos.

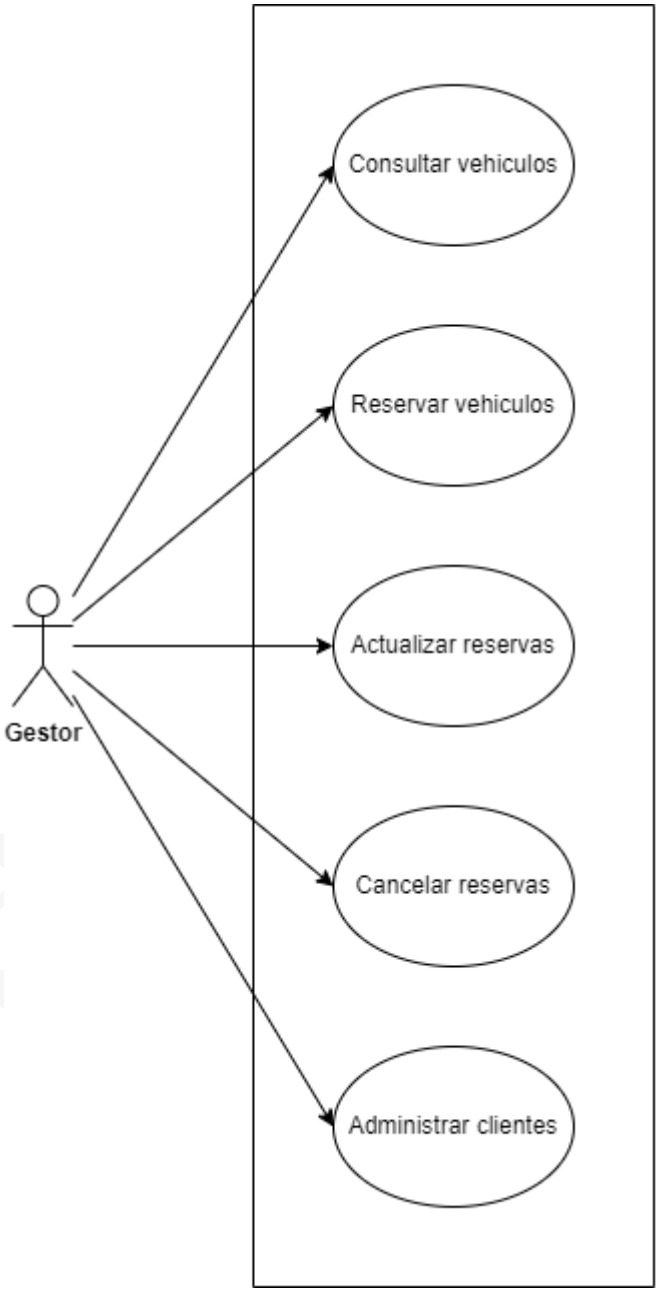
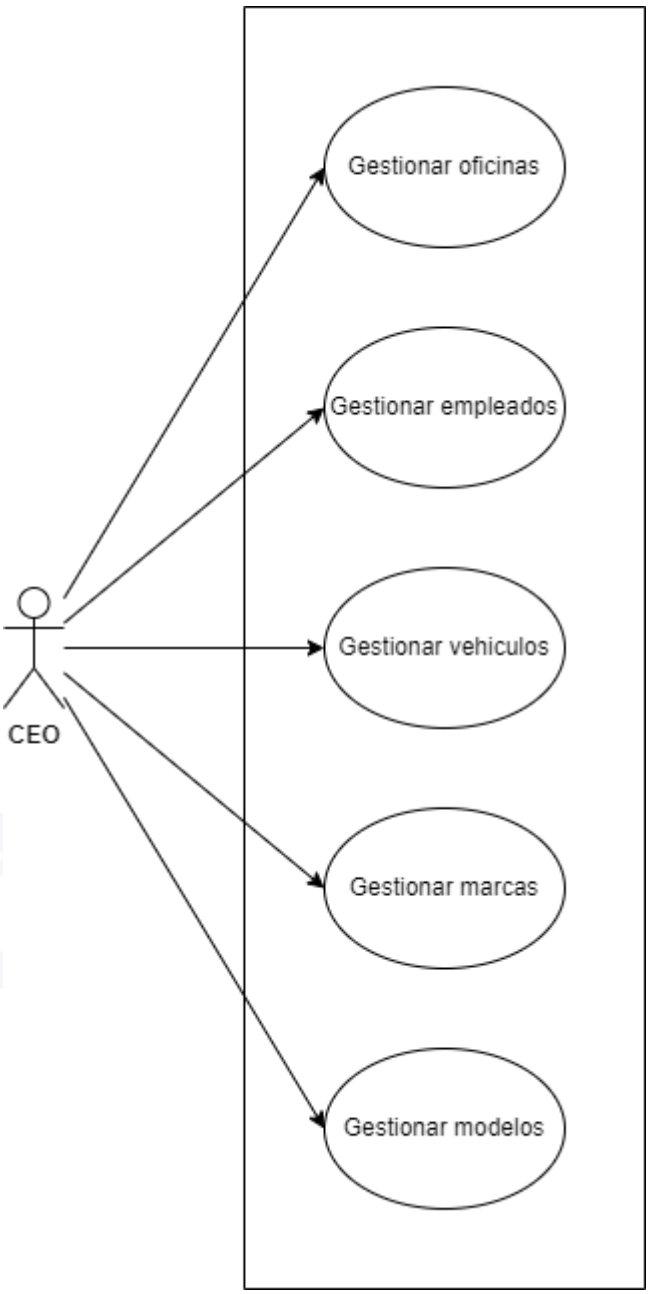


**Diagrama de casos de uso - Caso CEO (Encargado)**

El encargado puede realizar todas las tareas públicas del cliente, como ver y gestionar los vehículos.

Además también puede realizar las funciones del gestor de administrar los clientes y sus reservas.

Como encargado , tiene acceso exclusivo a gestionar las oficinas, los empleados y los vehículos.



**Partes específicas de la aplicación (renombrables).**

- **Auth context:** Módulo de la aplicación encargada de la autenticación y autorización de los usuarios, a través de JWT y encriptado de contraseñas. Además este módulo se encarga de la modificación y recuperación de contraseñas

- **Backoffice context:** Módulo de la aplicación para gestionar los catálogos de la empresa, los cuales son vehículos, modelos, marcas, clientes, empleados, ciudades y oficinas.

- **Website context:** Módulo de la aplicación que consiste en el sitio web dónde los clientes se registran, consultan y reservan vehículos.

# Conclusiones

## Puntos a mejorar

- La estructura y la formalidad del front-end.
- La interacción con los usuarios.
- El diseño.
- Finalizar la estructura de la "Base de Datos" y el "Diagrama de Clases".

## Resumen de los objetivos conseguidos

Generalmente, se ha conseguido cumplir la mayoría de los objetivos que se habían establecido de forma inicial para el proyecto. El proyecto comenzó con una estructura empresarial y unas reglas de negocio ya establecidas pero, según ha ido avanzando el proyecto ha evolucionado de otra forma. Definiendo reglas nuevas que han permitido mantener la integridad y el uso lógico de la empresa. Han desaparecido algunos roles de empleados que se ha observado que no tenía sentido lógico en la empresa y se ha creado un nuevo rol más globalizado. El rol de "cliente" ha mantenido la misma estructura y los "gestores" siguen manteniendo las mismas funcionalidades.

## Resultados obtenidos

Algunos puntos destacables que al final han quedado fuera de la aplicación por el tiempo de desarrollo, ha sido la parte de los "Coordinadores de operaciones" y los "Transportistas". En resumen, la aplicación por cuestiones de tiempo y desarrollo no se ha extendido al punto deseado pero el resultado final es más que satisfactorio.

Es importante destacar, que el desarrollo del back-end cumple con las expectativas que estaban programadas. El front-end, sin embargo, el resultado ha sido menos satisfactorio debido a la falta de experiencia con estas tecnologías.

# Bibliografía


Campus Edix: https://campus.edix.com/

Stack Overflow Docs: https://stackoverflow.com/

W3 Schools: https://www.w3schools.com/

Udemy: https://www.udemy.com/

Node Documentation: https://nodejs.org/es/docs/

Express Documentation: https://expressjs.com/es/guide/routing.html

MongoDB Documentation: https://www.mongodb.com/docs/

React Documentation: https://es.reactjs.org/docs/getting-started.html