



C.P.R. Liceo “La Paz”

Proyecto Fin de Ciclo

# Desarrollo de Aplicaciones Multiplataforma

Autor: Ángel Fernández Blanco  
Tutor: Jesús Ángel Pérez-Roca Fernández

## Resumen

El presente proyecto, titulado “**Gearborn Motors**”, consiste en el desarrollo de una aplicación de gestión para un concesionario de vehículos.

El proyecto está dividido en dos módulos principales: un Backend desarrollado con Spring Boot, encargado de gestionar la lógica de negocio y persistencia de datos a la base de datos, y un Frontend construido con JavaFX, que proporciona una interfaz gráfica para usuarios empleados y comerciales.

Gearborn Motors permite gestionar clientes, empleados, ventas, gastos y vehículos, con funcionalidades CRUD completas, filtros, autenticación, y persistencia en base de datos.

Para el desarrollo de esta aplicación, se ha utilizado una arquitectura en capas y se han empleado buenas prácticas de desarrollo orientado a objetos y diseño limpio, diseño modular y control de acceso. El objetivo ha sido simular un entorno profesional de trabajo, replicando la operativa de una pequeña empresa real.

El objetivo principal ha sido replicar el funcionamiento de una pequeña empresa del sector automotor. Se ha priorizado la claridad del código, la reutilización de componentes y métodos, y la separación de responsabilidades, utilizando diferentes carpetas y clases. En conjunto, Gearborn Motors representa una solución profesional y completa para la gestión de un concesionario, y demuestra el desarrollo de software desde una perspectiva realista y aplicable a entornos empresariales.

La interfaz gráfica permite a los usuarios navegar fácilmente por los diferentes módulos del sistema, realizar búsquedas avanzadas, aplicar filtros y gestionar la información de manera eficiente. La experiencia de usuario ha sido un punto clave en el diseño, buscando siempre un equilibrio entre funcionalidad y facilidad de uso, especialmente pensando en perfiles no técnicos como vendedores o administrativos.

Finalmente, este proyecto no solo cumple una función académica y técnica, sino que también sirve como ejemplo de aplicación práctica de tecnologías modernas en el desarrollo de software empresarial. Su estructura modular y su enfoque en buenas prácticas lo convierten en una base sólida para futuras ampliaciones o adaptaciones a otros contextos comerciales similares.

## Abstract

The present project, titled “*Gearborn Motors*”, consists of the development of a management application for a car dealership.

The project is divided into two main modules: a Backend developed with Spring Boot, responsible for managing business logic and data persistence; and a Frontend built with JavaFX, which provides a graphical interface for employee and sales staff users.

Gearborn Motors allows for the management of customers, employees, sales, expenses, and vehicles, offering full CRUD functionalities, filters, authentication, and database persistence.

The application was developed following a layered architecture and applying best practices in object-oriented programming, clean design, modularity, and access control. The goal was to simulate a professional work environment, replicating the operations of a small real-world company.

The main objective was to recreate the functioning of a small automotive business. Code clarity, component and method reusability, and separation of responsibilities were prioritized, using a well-organized folder and class structure. Overall, Gearborn Motors represents a complete and professional solution for dealership management, demonstrating software development from a realistic and business-oriented perspective.

The graphical interface allows users to easily navigate through the system’s various modules, perform advanced searches, apply filters, and efficiently manage information. User experience was a key focus in the design, aiming for a balance between functionality and ease of use, especially for non-technical profiles such as sales or administrative staff.

Finally, this project not only fulfills an academic and technical purpose but also serves as a practical example of the application of modern technologies in enterprise software development. Its modular structure and focus on best practices make it a solid foundation for future extensions or adaptations to other similar commercial contexts.

## Palabras clave

**Spring Boot**: Framework de Java para desarrollo rápido de aplicaciones backend.

**JavaFX**: Biblioteca de Java para crear interfaces gráficas.

**CRUD**: Operaciones básicas de persistencia: Crear, Leer, Actualizar, Eliminar.

**DTO**: (Data Transfer Object): Objeto que transporta datos entre capas del sistema.

**Repository Pattern**: Patrón de diseño para abstraer el acceso a datos.

**Entity-Relationship (ER)**: Modelo para representar estructuras de base de datos.

**Maven**: Herramienta para gestión de proyectos y dependencias en Java.

**FXML**: Lenguaje XML usado para diseñar interfaces en JavaFX.

*Dedico este trabajo, que simboliza el cierre de una etapa importante en mi vida, a mi familia. Agradezco profundamente su apoyo incondicional a lo largo de todo este proceso, así como los valores que me han transmitido, especialmente el esfuerzo, la constancia y la perseverancia.*

*Quiero expresar también mi agradecimiento a los profesores y, en particular, a mi tutor durante estos dos años de formación. Su dedicación, claridad en las explicaciones y disposición para resolver dudas han sido fundamentales en mi desarrollo académico.*

*Asimismo, me gustaría destacar a mis compañeros, a todos y cada uno de ellos. Gracias por el compañerismo, el intercambio de ideas y, sobre todo, por los buenos momentos compartidos a lo largo de esta enriquecedora etapa.*

# ÍNDICE

<b>Resumen.....</b>	<b>2</b>
<b>Abstract.....</b>	<b>3</b>
<b>Palabras clave.....</b>	<b>4</b>
<b>Introducción y motivaciones.....</b>	<b>7</b>
<b>Objetivos.....</b>	<b>8</b>
<b>Estado del arte.....</b>	<b>9</b>
Fortalezas y debilidades de las soluciones existentes.....	9
Comparación con Gearborn Motors.....	9
Tecnologías utilizadas y alternativas.....	10
<b>Caso de estudio.....</b>	<b>11</b>
Casos de uso.....	12
<b>Diagramas.....</b>	<b>14</b>
Diagrama de clases.....	14
Diagrama de flujo.....	14
Diagrama E-R modelo entidad - relación.....	16
<b>Desarrollo del proyecto.....</b>	<b>17</b>
Backend: Arquitectura y Decisiones Técnicas.....	17
Frontend: JavaFX + FXML + Controladores.....	17
Caso prueba “Inicio de sesión”.....	19
Caso prueba “registro de un nuevo usuario”.....	21
Caso prueba “registro compra de un vehículo”.....	23
<b>Manual Administrador.....</b>	<b>26</b>
Requisitos previos.....	26
Clonación del repositorio.....	26
Configuración de la base de datos.....	27
Configuración del Frontend.....	27
<b>Manual Usuario.....</b>	<b>28</b>
<b>Viabilidad tecno-económica.....</b>	<b>32</b>
Costes técnicos e infraestructura.....	32
Costes de desarrollo.....	32
Rentabilidad.....	33
Conclusión.....	33
<b>Trabajo futuro.....</b>	<b>34</b>
1. Verificación de compras mediante validación por parte de empleados.....	34
2. Integración de servicios de correo electrónico.....	34
3. Panel de usuario para actualización de datos personales.....	35
4. Gestión documental.....	35
5. Panel de estadísticas e informes visuales.....	36
6. Versión web y despliegue en la nube.....	36

<b>Conclusiones.....</b>	<b>37</b>
<b>Biblioteca de recursos web y referencias.....</b>	<b>39</b>

## Introducción y motivaciones

La motivación principal para llevar a cabo este proyecto surge de la voluntad de aplicar de forma práctica los conocimientos adquiridos durante el ciclo formativo de Desarrollo de Aplicaciones Multiplataforma (DAM), mediante el desarrollo de una solución completa y funcional que represente la gestión operativa de una empresa real. En este caso, se ha optado por simular el funcionamiento de un concesionario de vehículos, ya que permite abarcar una amplia variedad de aspectos del desarrollo de software, como la gestión de usuarios, operaciones CRUD (crear, leer, actualizar, eliminar), persistencia de datos, diseño de interfaces gráficas de usuario (GUI) y aplicación de medidas básicas de seguridad y control de acceso.

Además del enfoque técnico y académico, hay una motivación personal que ha influido directamente en la elección del tema. Desde siempre he sentido una gran pasión por el mundo del motor, especialmente por los coches. Esta afición ha sido un estímulo constante a lo largo del tiempo, no sólo como interés personal, sino también como una fuente de inspiración para aplicar mis habilidades en programación en un contexto que realmente me entusiasma. Por ello, decidí unir ambas pasiones en este proyecto (el desarrollo de software y el mundo del automóvil), con el objetivo de crear una herramienta que represente de manera fiel y profesional la operativa de un concesionario.

El proyecto, titulado Gearborn Motors, busca simular un entorno empresarial donde se gestionan distintos elementos esenciales: clientes, vehículos, empleados, ventas y gastos. Esta elección permite integrar diversos componentes técnicos: desde la estructura de una base de datos relacional eficiente, pasando por la implementación de un backend sólido utilizando Spring Boot, hasta una interfaz gráfica amigable y funcional desarrollada con JavaFX. Además, se ha seguido una arquitectura en capas, respetando principios de diseño limpio y orientado a objetos, con el objetivo de construir una aplicación mantenible, escalable y alineada con las buenas prácticas del sector.

En resumen, Gearborn Motors representa una oportunidad para poner en práctica todo lo aprendido durante el ciclo y la estancia en las prácticas de la empresa, enfrentándome a un desarrollo realista, completo y motivador. Ha sido también un ejercicio de creatividad, disciplina y aprendizaje continuo, en el que he podido desarrollar una solución útil, coherente y vinculada con uno de mis intereses personales más importantes.

## Objetivos

El objetivo principal de este proyecto es demostrar lo aprendido a lo largo de la formación mediante el desarrollo completo de una aplicación funcional, estructurada y orientada a un caso de uso real. A través de esta experiencia, se pretende reflejar no solo los conocimientos técnicos adquiridos, sino también la capacidad de organización, autonomía y resolución de problemas.

Otro de los propósitos fundamentales ha sido aproximarse a la manera en que se desarrollaría un proyecto en un entorno profesional. Para ello, se han definido fases de trabajo claras —análisis, diseño, desarrollo, pruebas y documentación— que han permitido mantener una estructura ordenada y un control constante sobre el progreso. Esto ha contribuido al desarrollo de habilidades de gestión del tiempo, planificación y toma de decisiones.

Además, el proyecto busca fomentar la comprensión del ciclo de vida completo de una aplicación: desde su concepción inicial hasta su implementación final. Se ha trabajado con especial atención en aspectos como la separación de responsabilidades, la reutilización de componentes, el diseño modular y la integración de distintas tecnologías que actúan de forma conjunta (backend, frontend, base de datos y control de acceso).

Por último, al tratarse de una temática vinculada con una afición personal —el mundo del motor y especialmente los automóviles—, también se ha perseguido el objetivo de mantener la motivación alta durante el proceso, aplicando lo aprendido en un contexto que resulta cercano y estimulante. Esta conexión ha favorecido el compromiso con el proyecto y la voluntad de alcanzar un resultado lo más profesional posible.



## Estado del arte

Actualmente, existen numerosas soluciones comerciales orientadas a la gestión de concesionarios de vehículos, muchas de ellas integradas como plataformas web o aplicaciones de escritorio de pago. Entre las más destacadas se encuentran **Autoline**, **CDK Global**, **Infor Dealer Management**, y en el mercado nacional, opciones como **I'MOVE** o **GestFuturo Concesionarios**. Estas herramientas están diseñadas para cubrir todas las necesidades de un concesionario: gestión de inventario, seguimiento de clientes (CRM), generación de presupuestos y facturas, control de ventas, gestión de empleados, y en algunos casos, incluso integración con sistemas contables o de postventa.

### Fortalezas y debilidades de las soluciones existentes

Las aplicaciones profesionales mencionadas tienen como principal ventaja su amplitud funcional y su nivel de integración, permitiendo centralizar todas las operaciones del concesionario en un único entorno. Además, suelen incluir módulos personalizables, interfaces web adaptadas y soporte técnico.

Sin embargo, también presentan algunas desventajas, especialmente para pequeñas empresas o usuarios individuales que no requieren todas las funcionalidades avanzadas. En muchos casos, estas soluciones tienen un costo elevado de licencia o suscripción, requieren conexión a internet constante, y pueden resultar complejas de utilizar si no están bien adaptadas al tamaño y necesidades concretas del negocio. Además, al ser sistemas cerrados, ofrecen poca flexibilidad para ser modificados o ampliados por terceros.

### Comparación con Gearborn Motors

En comparación, Gearborn Motors es una aplicación desarrollada como proyecto académico, pero con un enfoque profesional. Aunque no incluye todos los módulos avanzados de las soluciones comerciales, sí ofrece una experiencia completa de gestión adaptada a las necesidades básicas de un concesionario: control de vehículos, clientes, empleados, ventas y gastos, todo con funcionalidades CRUD, filtros, autenticación y una interfaz clara e intuitiva.

Su principal ventaja es su ligereza, accesibilidad y personalización. Al ser una aplicación de escritorio desarrollada en JavaFX y con backend en Spring Boot, no depende de servidores externos ni requiere conexión constante a internet. Además, puede ser fácilmente modificada o

extendida por desarrolladores con conocimientos en Java, lo que la hace ideal como base de proyectos más grandes o como solución para negocios pequeños.

## Tecnologías utilizadas y alternativas

Para el desarrollo de Gearborn Motors se han utilizado tecnologías de propósito general y ampliamente aceptadas en el sector del desarrollo de software empresarial:

- **Java + JavaFX** para el frontend, lo que permite crear interfaces gráficas ricas y multiplataforma.
- **Spring Boot** como framework principal del backend, por su robustez, escalabilidad y capacidad para gestionar lógica de negocio de forma estructurada.
- **JPA/Hibernate** para la persistencia de datos.
- **Base de datos relacional (MySQL)** por su familiaridad y eficiencia para consultas estructuradas.

Como **alternativas**, podrían haberse utilizado otras tecnologías según el enfoque deseado. Por ejemplo:

- **Para el frontend: Electron**, si se quisiera una interfaz web empaquetada como escritorio, o **Angular/React** si se optara por una solución completamente web.
- **Para el backend: Node.js** o **Django**, que también permiten estructurar APIs de forma eficiente.
- **Para la base de datos: MongoDB**, si se prioriza flexibilidad sobre estructura relacional.

## Caso de estudio

Se identifican como problemáticas principales: la falta de centralización de datos, el riesgo de errores por gestión manual, la dificultad para acceder rápidamente a información relevante y la carencia de herramientas accesibles para pequeñas empresas del sector automotor.

Para abordar esta situación, se ha diseñado una aplicación que permite digitalizar y automatizar los principales procesos internos del concesionario, estructurados en los siguientes módulos:

- Gestión de vehículos: permite registrar, consultar, actualizar o eliminar vehículos del inventario, así como ver detalles técnicos y su estado (disponible, reservado, vendido).
- Gestión de clientes: facilita el mantenimiento de la base de datos de clientes, permitiendo búsquedas y filtros personalizados.
- Gestión de empleados: se registran los usuarios del sistema con control de acceso según rol (comercial o administrador).
- Gestión de ventas: relaciona clientes, vehículos y empleados en operaciones de compraventa, registrando fechas, precios e impuestos.
- Gestión de gastos: se almacena información sobre egresos (mantenimiento, logística, servicios, etc.) permitiendo consultas por fechas y categorías.
- Autenticación: sistema de inicio de sesión con control de permisos.

## Casos de uso

A continuación, se describen los casos de uso que sustentan el diseño de la solución:

### 1. Registrar nuevo vehículo

Actor: Administrador

Descripción: El sistema permite introducir los datos de un nuevo vehículo, como marca, modelo, precio, año, tipo de combustible, etc., y almacenarlos en la base de datos.

### 2. Consultar inventario

Actor: Comercial o Administrador

Descripción: Los usuarios pueden listar vehículos disponibles, aplicar filtros por marca, estado o rango de precios, y visualizar información detallada.

### 3. Registrar venta

Actor: Comercial

Descripción: Al concretar una venta, el usuario selecciona cliente, vehículo y vendedor, y el sistema registra la operación, calcula totales y actualiza el estado del vehículo.

### 4. Registrar gasto

Actor: Administrador

Descripción: Permite ingresar un nuevo gasto (tipo, fecha, importe y descripción) y almacenarlo para futuras consultas y control financiero.

### 5. Gestión de empleados

Actor: Administrador

Descripción: Alta, modificación o baja de empleados, asignación de roles, y control de accesos al sistema.

## **6. Autenticación y control de acceso**

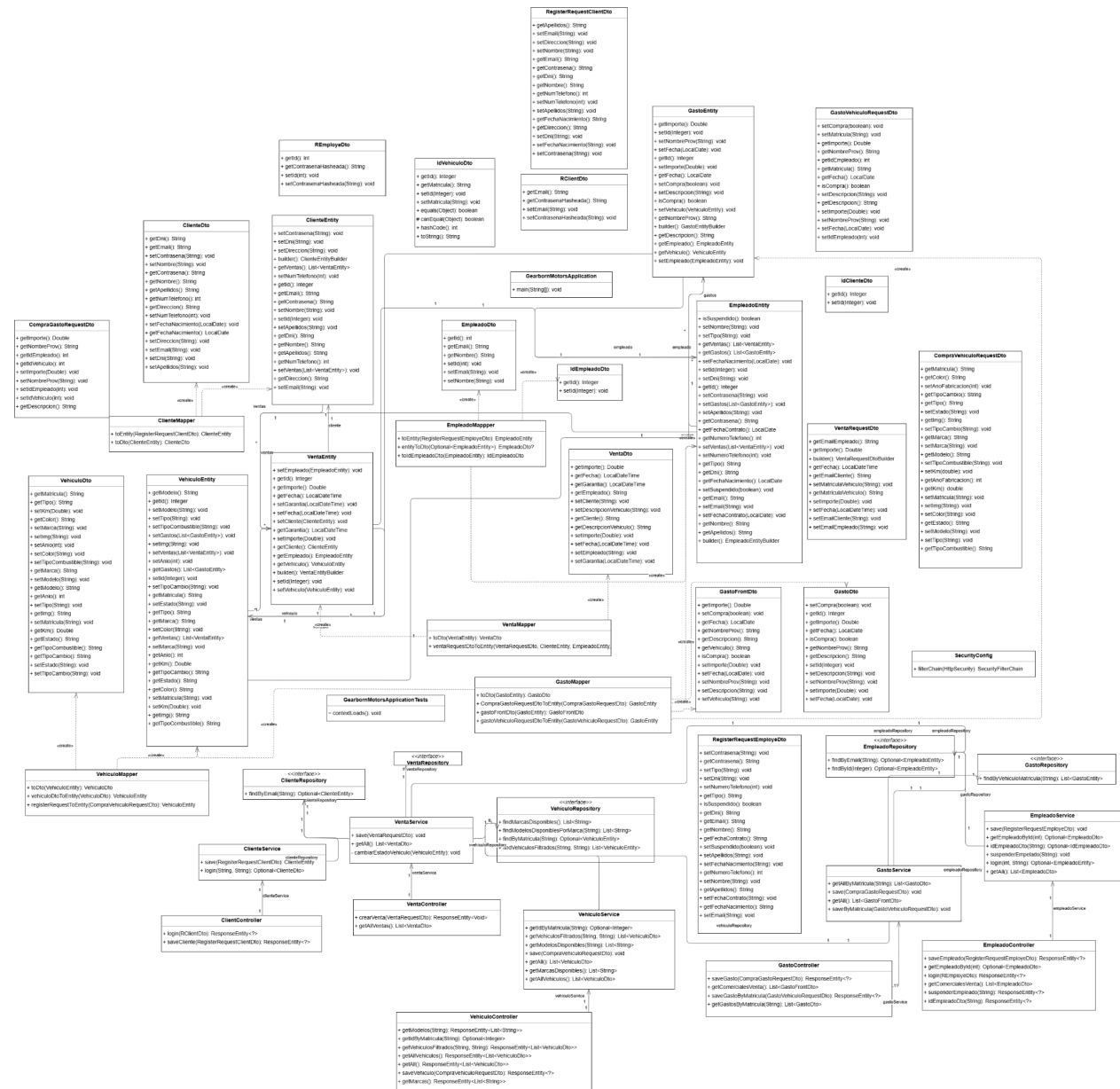
Actor: Usuario del sistema

Descripción: El sistema valida las credenciales en el inicio de sesión y otorga acceso según el rol asignado.

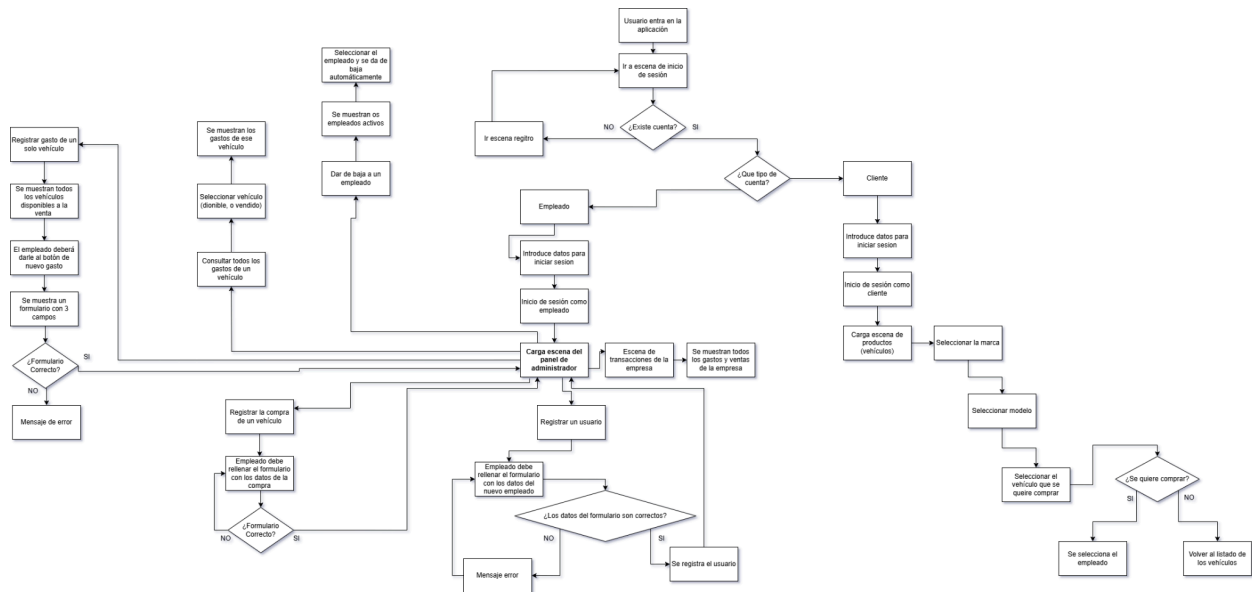
Estos casos de uso, junto con la estructura modular del sistema, permiten construir una aplicación robusta, intuitiva y adaptada a las necesidades concretas de un concesionario, mejorando la eficiencia y reduciendo errores humanos.

## Diagramas

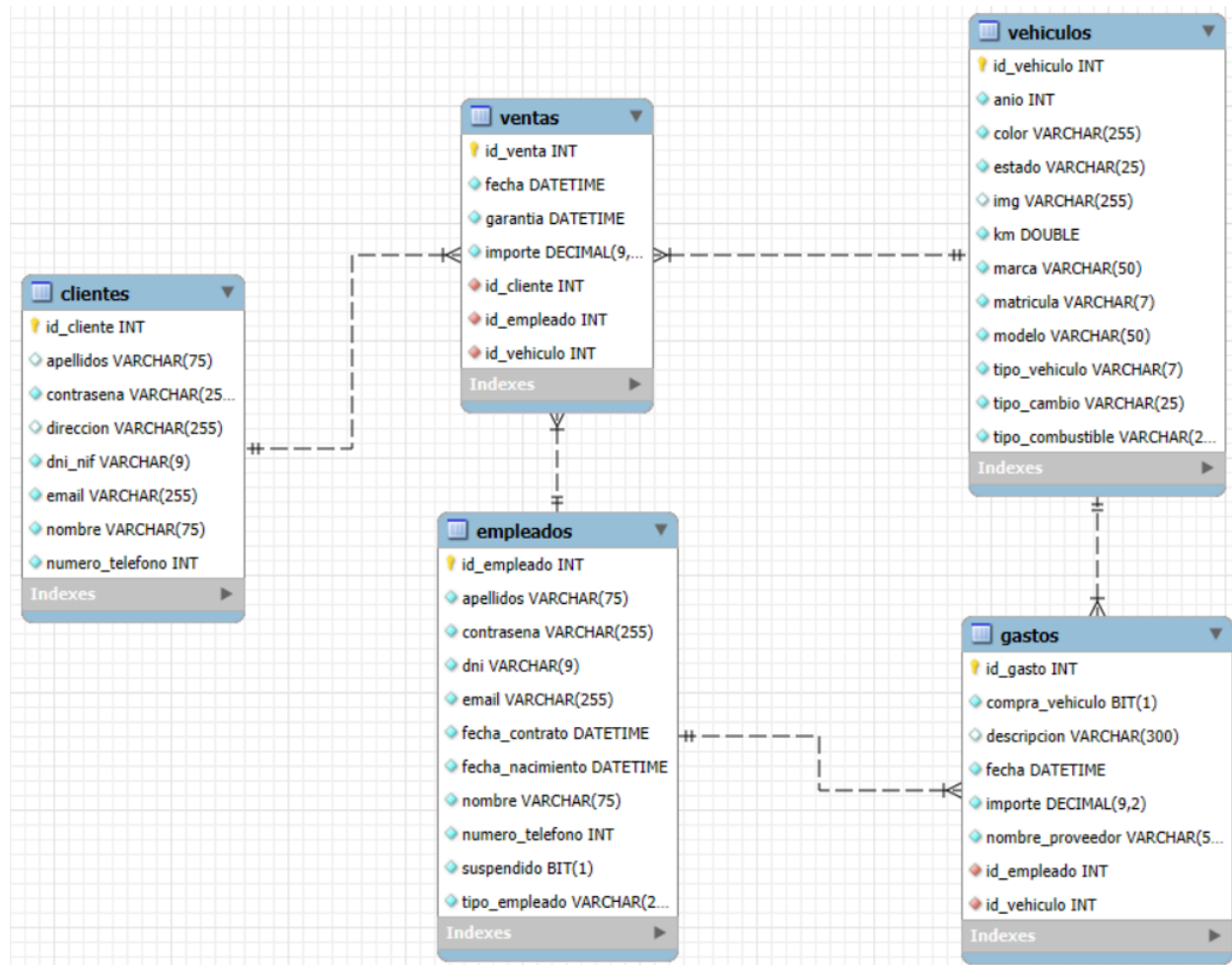
## Diagrama de clases



## Diagrama de flujo



## Diagrama E-R modelo entidad - relación





## Desarrollo del proyecto

Gearborn Motors se compone de dos grandes bloques: un Backend RESTful desarrollado con Spring Boot y un Frontend de escritorio desarrollado con JavaFX. El diseño ha seguido una arquitectura en capas, respetando el principio de separación de responsabilidades, y utilizando patrones como DTO, repositorio, y controladores desacoplados.

### Backend: Arquitectura y Decisiones Técnicas

#### Tecnología:

- Java 21
- Spring Boot
- Maven
- JPA (SpringJPA)
- Spring Security

#### Estructura de Capas:

- **Controller:** Recibe las peticiones HTTP. Ej.: ClienteController, VehiculoController.
- **Service:** Contiene la lógica de negocio. Ej.: ClienteService, VentaService.
- **Repository:** Extiende JpaRepository, maneja el acceso a la base de datos.
- **DTOs y Mappers:** Transforman entidades a objetos de transporte y viceversa.

### Frontend: JavaFX + FXML + Controladores

#### Tecnología:

- JavaFX
- FXML
- CSS para estilo visual

**Pantallas principales:**

- Login.fxml
- RegistroCliente.fxml
- Menu.fxml
- Concesionario.fxml
- PanelControl.fxml
- clienteCompraVehiculo.fxml

## Caso prueba “Inicio de sesión”

El sistema de autenticación de Gearborn Motors permite distinguir entre dos tipos de usuarios: clientes y empleados. Cada uno accede a la aplicación mediante una interfaz adaptada a su perfil, asegurando que solo puedan realizar las acciones que les corresponden dentro del sistema.

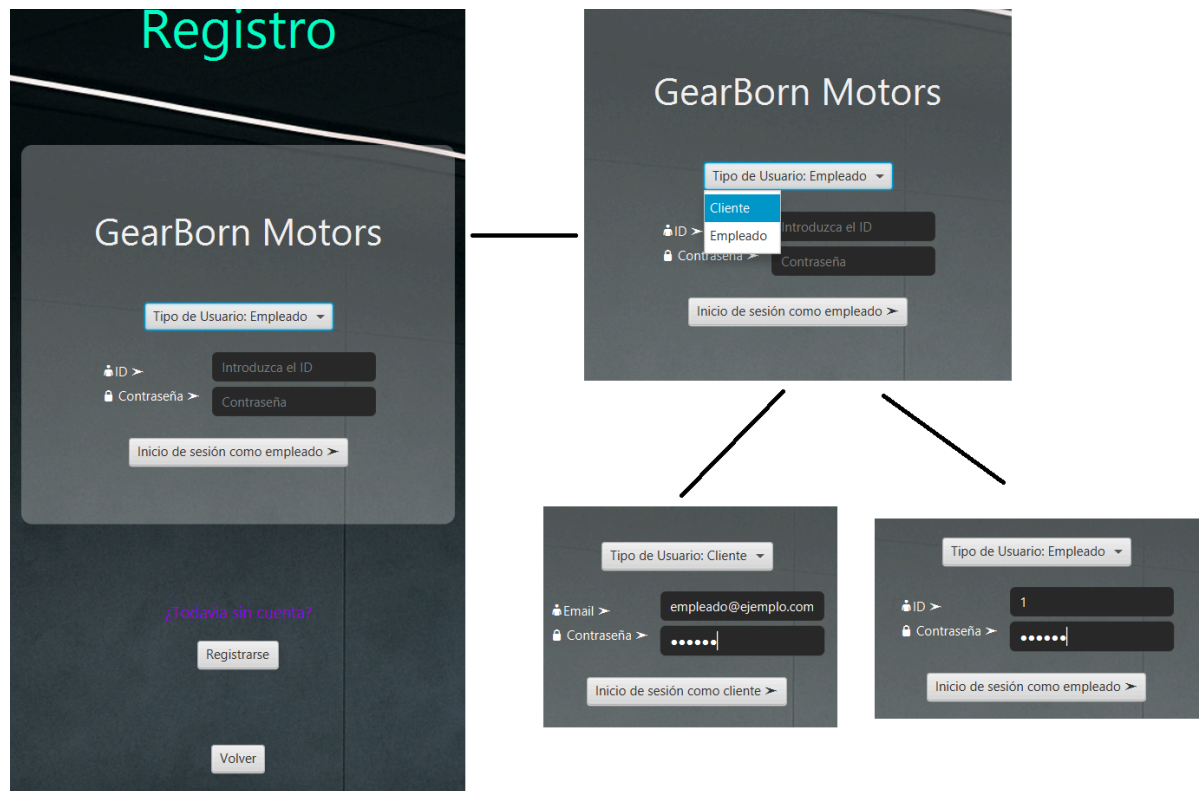
### Funcionamiento general

Al ejecutar la aplicación, por defecto se muestra la escena de inicio de sesión para clientes. En esta pantalla se identifica claramente el nombre de la aplicación ("Gearborn Motors") y el título de la escena. El cliente debe introducir su correo electrónico registrado y su contraseña, y posteriormente pulsar el botón “Iniciar sesión como cliente”. Si los datos son correctos y coinciden con los almacenados en la base de datos, el cliente accederá a su área correspondiente.

Para los empleados, el procedimiento es similar pero con un método de identificación distinto. En este caso, el usuario debe seleccionar una opción mediante un menú desplegable, indicando que desea iniciar sesión como empleado. A continuación, se le solicitará su ID numérico y su contraseña. Tras pulsar el botón “Iniciar sesión como empleado”, si las credenciales son válidas, accederá al panel de control de empleado, con funcionalidades específicas habilitadas.

Además, la interfaz incluye una opción adicional en la parte inferior de la escena de clientes que permite a un usuario no registrado crear una nueva cuenta. Al pulsar el botón de “Registrar nuevo cliente”, el sistema redirige al formulario de registro, donde se podrá ingresar la información requerida. Una vez registrado, el usuario podrá autenticarse con las credenciales recién creadas.

	<b>Inicio de sesión (clientes y empleados)</b>
Descripción	Verificar que el sistema distingue correctamente entre cliente y empleado, y permite acceder a cada rol solo si las credenciales son válidas.
Datos de entrada	Cliente: correo y contraseña válidos / Empleado: ID numérico y contraseña
Resultado esperado	Acceso al sistema correspondiente al tipo de usuario
Resultado obtenido	Acceso exitoso tras validación de credenciales
Estado	Superado



## Caso prueba “registro de un nuevo usuario”

El sistema permite a nuevos clientes registrarse directamente desde la interfaz de inicio de sesión. Esta funcionalidad está orientada a facilitar el alta de usuarios sin necesidad de intervención administrativa, brindando acceso inmediato a la aplicación una vez completado el proceso de registro.

### Funcionamiento general

En la escena de inicio de sesión, específicamente en la interfaz de acceso para clientes, se encuentra una opción en la parte inferior que permite acceder al formulario de registro de nuevo cliente. Al hacer clic en esta opción, el sistema redirige al usuario a una nueva escena donde podrá introducir sus datos personales para crear una cuenta.

Los campos requeridos para el registro incluyen: nombre completo, correo electrónico, contraseña, entre otros datos personales. Una vez completados los campos y enviada la solicitud, el sistema valida que los datos sean correctos (por ejemplo, que el correo no esté ya registrado y que la contraseña cumpla ciertos requisitos mínimos).

Si todo es válido, los datos se guardan en la base de datos y el sistema confirma el registro exitoso. A partir de ese momento, el usuario puede volver a la pantalla de inicio de sesión e ingresar con sus nuevas credenciales.

	Registro de nuevos clientes
Descripción	Verificar que un usuario no registrado puede crear una cuenta de cliente válida y luego iniciar sesión.
Datos de entrada	Nombre: “Laura García”, Email: “laura@example.com”, Contraseña: “1234laura”
Resultado esperado	El sistema almacena los datos, confirma el registro y permite el acceso con esas credenciales
Resultado obtenido	Registro exitoso, acceso posible tras iniciar sesión
Estado	Superado

The diagram illustrates a customer registration process. On the left, a form titled "Registro de Cliente" contains fields for Email, Nombre, DNI, Dirección, Teléfono, Apellidos, Fecha Nacimiento, and Contraseña. Below the form are two buttons: "Volver" and "Regístrate". An arrow points from the "Regístrate" button to a data output section on the right. This section displays the data entered in the form, with some fields truncated for brevity. Below the data output, a series of labels and values are shown, representing the data structure after registration.

**Registro de Cliente**

Email: Cliente@ejemplo.com

Nombre: ejemplo

DNI: 123456789

Dirección: calle ejemplo, 345

Teléfono: 111111111

Apellidos: apellido1 apellido2

Fecha Nacimiento: 01/01/2000

Contraseña: \*\*\*\*\*

Volver Regístrate

Email: ejemplo@gmail.com

Nombre: ej Pezito

DNI: ej 123456789A

Dirección: ej calleEjemplo

Teléfono: ej 123456789

Apellidos: ej Pérez Gómez

Fecha Nacimiento: ej 01/01/1990

Contraseña: ej Abc123R

apellido1 apellido2 d40dc261069d6ffe494cdd8e582243e6 Calle Ejemplo 1 111111111A ejemplo@ejemplo.com Ejemplo 111111111

## Validaciones incluidas

- Comprobación de que el email no exista ya en la base de datos.
- Contraseñas no vacías y con requisitos básicos de seguridad.
- Todos los campos obligatorios completados.

Al acabar el registro, si todo fue correcto, se carga la escena principal.

## Caso prueba “registro compra de un vehículo”

El módulo de registro de compras permite que un empleado dé de alta la adquisición de un vehículo nuevo para incorporarlo al inventario del concesionario. Esta funcionalidad es fundamental para mantener actualizada la lista de vehículos disponibles y controlar el origen y coste de los activos.

### Funcionamiento general

Desde la interfaz de gestión de vehículos, un empleado con permisos puede acceder a la opción de “Guardar compra de un vehículo”. En esta sección, se introducen los datos del nuevo vehículo adquirido, junto con la información relevante sobre la operación de compra.

Una vez introducida toda la información, el usuario pulsa el botón “Registrar compra”. El sistema valida los datos, guarda el vehículo en la base de datos y lo añade automáticamente al inventario como disponible.

	Registro de una compra de vehículo
Descripción	Verificar que un usuario no registrado puede crear una cuenta de cliente válida y luego iniciar sesión.
Datos de entrada	Marca: Peugeot, Modelo: 208, Año: 2022, Combustible: Gasolina, Precio: 9.200 €, Fecha: 03/06/2025, Proveedor: “Automóviles Gallego S.L.”
Resultado esperado	Vehículo almacenado correctamente, añadido al inventario con estado “disponible”
Resultado obtenido	Vehículo registrado, aparece en el listado de inventario
Estado	Superado

### Validaciones incluidas

- Todos los campos obligatorios deben estar completos.
- El precio debe ser un valor numérico positivo.
- El año debe estar dentro de un rango válido (por ejemplo, no futuro).
- El proveedor debe ser una cadena de texto no vacía.
- No se permiten duplicados exactos (opcional, si se ha implementado).

### Formulario de registro de vehículo (compra)

La imagen muestra el formulario de alta de un vehículo en el sistema, utilizado por los empleados para registrar compras realizadas al concesionario. Este formulario recoge toda la información relevante del vehículo, incluyendo:

- Tipo de vehículo
- Estado (disponible, reservado, vendido, etc.)
- Tipo de combustible
- Matrícula
- Año de fabricación
- Tipo de cambio (manual, automático...)
- Color
- Marca y modelo
- Precio de compra
- Total de kilómetros
- Nombre del proveedor (campo de texto libre)



Además, aunque no se visualiza en esta imagen, el formulario también permite al usuario seleccionar una imagen del vehículo desde el sistema de archivos local, mediante un selector tipo FileChooser. Esta imagen puede ser utilizada para mostrar el vehículo en el listado o en una vista detallada.

## Manual Administrador

Este manual tiene como objetivo proporcionar las instrucciones necesarias para instalar, configurar y ejecutar la aplicación Gearborn Motors en cualquier equipo de forma sencilla y autónoma. También se incluye la información sobre la carga de datos de prueba y el acceso con credenciales predefinidas.

### Requisitos previos

Para ejecutar correctamente la aplicación, es necesario tener instalado el siguiente software:

- Java 21
- Apache Maven
- MySQL
- Git
- IDE recomendado: IntelliJ IDEA, Eclipse o Visual Estudio;

### Clonación del repositorio

El código fuente de la aplicación se encuentra en GitHub. Para descargarlo, basta con ejecutar el siguiente comando desde la terminal:

```
git clone https://github.com/usuario/gearborn-motors.git
```

## Configuración de la base de datos

### 1. Crear una base de datos llamada:

```
CREATE DATABASE tfc_bd_gearborn;
```

### 2. Configurar las credenciales de acceso en el archivo application.properties del backend:

```
spring.datasource.url=jdbc:mysql://localhost:3306/gearborn_motors  
spring.datasource.username=usuario_mysql  
spring.datasource.password=contraseña_mysql  
spring.jpa.hibernate.ddl-auto=update  
spring.sql.init.mode=always
```

### 3. El sistema puede generar automáticamente las tablas y cargar datos de prueba desde un archivo data.sql que está presente en src/main/resources.

## Configuración del Frontend

1. Abrir el proyecto de frontend JavaFX en el IDE.
2. Verificar que las rutas y endpoints estén correctamente configurados.
3. Ejecutar la clase principal de la aplicación (LauncherApp).
4. La interfaz gráfica se iniciará como una aplicación de escritorio.

## Manual Usuario

La aplicación Gearborn Motors ha sido diseñada con especial atención a la experiencia de los usuarios finales, en particular los clientes que acceden al sistema para informarse sobre los vehículos disponibles, explorar opciones de compra y tomar decisiones con confianza. Desde el primer momento en que un cliente inicia sesión, la interfaz ofrece un entorno visualmente atractivo, funcional y orientado a la facilidad de uso, incluso para personas sin conocimientos técnicos.

La experiencia comienza con un mensaje de bienvenida personalizado que no solo refuerza el vínculo con el cliente, sino que también establece un tono cercano y profesional. Este tipo de detalles, aunque sutiles, contribuyen a crear una atmósfera positiva, donde el usuario se siente valorado. En paralelo, el diseño visual del entorno —con colores suaves, tipografías legibles y una organización clara de los elementos— asegura una navegación cómoda e intuitiva.

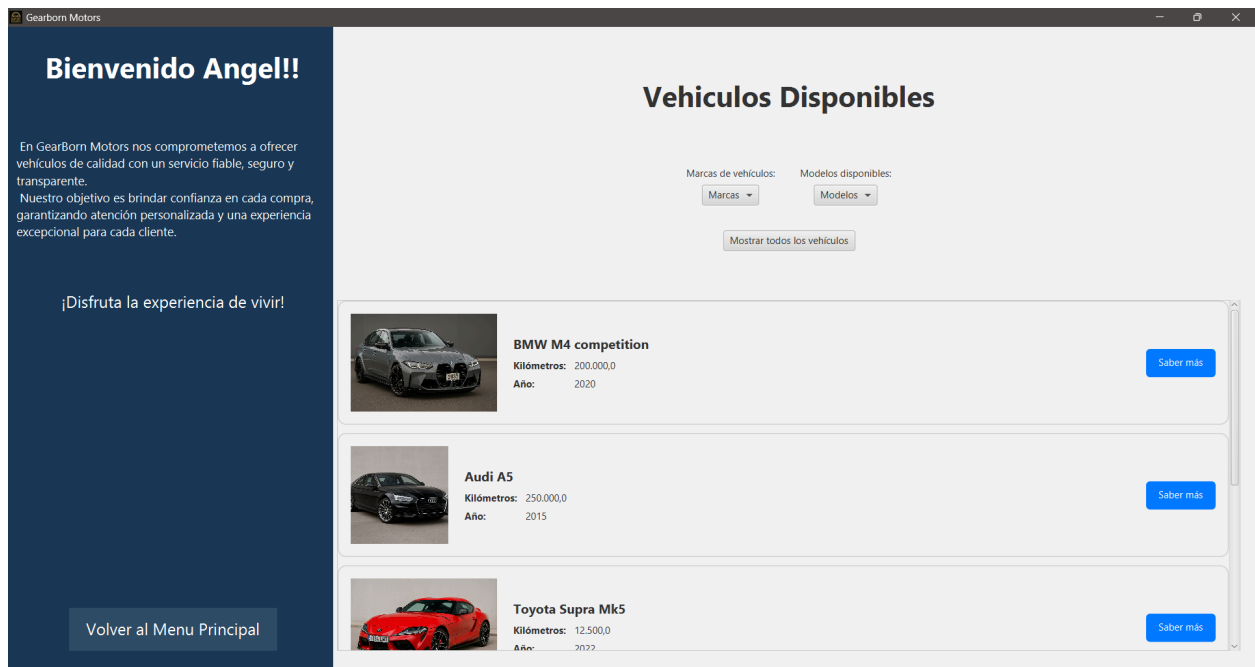
Cada funcionalidad ha sido desarrollada pensando en la simplicidad: los clientes pueden desplazarse por la aplicación, aplicar filtros para encontrar vehículos específicos, y acceder a la información relevante sin esfuerzo ni complicaciones. El sistema evita sobrecargar al usuario con datos innecesarios, mostrando únicamente lo más relevante y manteniendo la coherencia visual entre secciones. De esta forma, la interacción resulta fluida, directa y satisfactoria.

Uno de los objetivos principales del proyecto ha sido trasladar la confianza que se espera de un concesionario físico al entorno digital. Por ello, la aplicación ofrece un entorno transparente donde el cliente puede ver datos detallados de los vehículos, sin letra pequeña ni procesos confusos. Esta claridad permite que el usuario tome decisiones informadas, lo que refuerza la percepción de profesionalidad y seriedad por parte del sistema.

Además, se han minimizado las barreras de entrada: el formulario de registro es breve y claro, y el acceso posterior al sistema es inmediato. Una vez dentro, los clientes pueden navegar por el catálogo sin necesidad de instrucciones, ya que la interfaz se apoya en convenciones comunes y en una disposición lógica de las funciones.

En resumen, Gearborn Motors proporciona una experiencia digital centrada en el cliente, donde la accesibilidad, la confianza y la claridad son pilares fundamentales. La aplicación busca replicar el trato cercano y profesional de un concesionario real, utilizando la tecnología para facilitar el proceso de exploración y toma de decisiones, sin perder de vista la importancia de una interfaz amigable y bien diseñada.

## Escena de bienvenida y consulta de vehículos disponibles



En esta vista se da la bienvenida al cliente por su nombre, mostrando un panel informativo a la izquierda con un mensaje corporativo de confianza y compromiso. A la derecha, se presenta el listado de vehículos disponibles en el concesionario.

En el panel central y derecho de la vista se encuentra el núcleo funcional de esta pantalla: la consulta del inventario de vehículos disponibles. Los vehículos se muestran en forma de tarjetas o bloques visuales, cada uno con una imagen representativa, el nombre del modelo, los kilómetros recorridos y el año de fabricación. Estos datos permiten al usuario hacerse una idea rápida de las características básicas de cada vehículo.

En la parte superior, se han incorporado **filtros dinámicos** por marca y modelo, que permiten al usuario refinar su búsqueda según sus preferencias. El botón "Mostrar todos los vehículos" sirve para restablecer los filtros aplicados y mostrar nuevamente todo el inventario.

Cada tarjeta de vehículo incluye un botón "**Saber más**", que redirige al usuario a una vista detallada del vehículo seleccionado, donde podrá consultar más información antes de tomar una decisión.


## Elementos destacados de la interfaz:

- Filtros superiores: permiten seleccionar una marca y/o modelo de vehículo para refinar la búsqueda.
- Botón “Mostrar todos los vehículos”: elimina cualquier filtro aplicado y recarga el inventario completo.
- Tarjetas de vehículos: se muestran con imagen, nombre, kilómetros y año de fabricación.
- Botón “Saber más”: lleva a una vista detallada del vehículo seleccionado.

## Escena de compra de un vehículo

**BMW M4 competition** [Atrás](#)

### Datos del vehículo

Marca del vehículo: BMW		Kilómetro del vehículo: 200000.0
Modelo del vehículo: M4 competition		Tipo de combustible: Gasolina
Tipo de vehículo: Coche		Tipo de cambio: Manual
Año de fabricación: 2020		Color del vehículo: <input type="text"/>

### Datos de venta

Garantía hasta : 2027-06-05  
Importe total de la transacción: 0.0 €  
Fecha de compra: 2025-06-05

### Empleados disponibles

Seleccione el empleado que le atenderá

Nombre: **Adrian**  
Correo: adrian@gmail.com

Nombre: **Jacobo**  
Correo: Jacobo@gmail.com

[Realizar Compra](#)

Al pulsar “Saber más”, se accede a una vista completa del vehículo, donde se presentan todos sus atributos relevantes: marca, modelo, tipo de vehículo, tipo de combustible, tipo de cambio, color, año de fabricación y kilómetros recorridos. Además, se muestra una imagen ampliada del vehículo para una mejor visualización.

En esta misma pantalla, se incluye una sección dedicada a los **datos de la operación de compraventa**, como la fecha de compra, la fecha de garantía ofrecida y el importe total de la transacción. Si bien esta compra no se registra de forma definitiva para el cliente (en caso de ser una vista de demostración), sirve como ejemplo completo de cómo sería una interacción real con el sistema.

En el panel derecho, se muestra una lista de **empleados disponibles**, con su nombre y correo electrónico. El usuario puede seleccionar el empleado que le atenderá durante la compra. Finalmente, un botón “**Realizar Compra**” simula la ejecución de la operación, permitiendo registrar la transacción y actualizar el estado del vehículo en la base de datos.

## Viabilidad tecno-económica

El análisis de viabilidad tecno-económica tiene como objetivo valorar si el desarrollo e implantación del proyecto *Gearborn Motors* sería factible desde el punto de vista técnico y económico en un entorno real. Para ello, se han considerado los recursos necesarios para su funcionamiento, los costes derivados del desarrollo, y una estimación sobre su rentabilidad frente a soluciones comerciales existentes.

### Costes técnicos e infraestructura

Una de las principales ventajas del proyecto es que ha sido desarrollado íntegramente utilizando herramientas de código abierto y software libre. Esto implica que no se requiere ningún tipo de licencia de pago para ejecutar el sistema, lo que reduce significativamente el coste de implantación.

En cuanto al hardware necesario, el sistema podría funcionar en una infraestructura sencilla:

- Un equipo servidor para alojar el backend y la base de datos, con especificaciones medias (procesador i5 o Ryzen, 8 GB de RAM, disco SSD), tendría un coste estimado de entre 600 y 800 euros.
- Para los puestos de trabajo (comerciales, administrativos), serían suficientes entre 2 y 5 equipos cliente, con un coste aproximado de 400 a 600 euros por unidad, sumando entre 1.200 y 2.000 euros.

En lo referente al software, no se requiere ninguna inversión adicional, ya que todas las herramientas empleadas (Java 21, Apache Maven, MySQL, JavaFX) son gratuitas y multiplataforma.

### Costes de desarrollo

En un escenario real, el mayor coste vendría determinado por el tiempo de desarrollo. Si se estima una dedicación de entre 200 y 300 horas para la implementación completa del sistema, y se considera una tarifa de desarrollo media de 25 euros por hora, el coste aproximado ascendería a entre 5.000 y 7.500 euros.

A esto podría añadirse un pequeño coste adicional asociado a tareas de documentación, formación de usuarios, pruebas y mantenimiento inicial, que podría situarse entre 750 y 1.000 euros.



Por tanto, el coste total estimado de desarrollo y puesta en marcha rondaría entre los 6.000 y 8.500 euros.

## **Rentabilidad**

Frente a soluciones comerciales del sector, que en muchos casos requieren el pago de licencias anuales, cuotas por usuario o costes de mantenimiento elevados, *Gearborn Motors* se presenta como una alternativa económicamente viable, especialmente para pequeñas empresas que no disponen de grandes presupuestos para digitalización.

Además, al tratarse de un sistema modular y de código abierto, existe la posibilidad de ampliarlo o personalizarlo fácilmente sin costes adicionales por parte de terceros, lo que refuerza su valor a largo plazo.

## **Conclusión**

Teniendo en cuenta los factores expuestos, puede afirmarse que el proyecto *Gearborn Motors* es técnicamente factible y económicamente viable. Su bajo coste de implantación, el uso de tecnologías libres y su orientación a un entorno empresarial real lo convierten en una solución rentable, accesible y con gran potencial de escalabilidad. Resulta especialmente interesante para concesionarios de tamaño medio o pequeño que deseen informatizar su operativa sin recurrir a costosas plataformas comerciales.

## Trabajo futuro

El desarrollo de Gearborn Motors ha permitido sentar las bases de una aplicación robusta, funcional y coherente con la realidad operativa de un concesionario de vehículos. No obstante, como ocurre en cualquier solución tecnológica orientada al uso real, existe un amplio margen de mejora y evolución. En este apartado se detallan diversas funcionalidades que se consideran viables y estratégicas para ser incorporadas en futuras versiones del sistema, con el objetivo de ampliar sus capacidades, mejorar la experiencia de usuario, reforzar la seguridad y hacer del proyecto una herramienta aún más completa y profesional.

### 1. Verificación de compras mediante validación por parte de empleados

Una de las mejoras más significativas que se plantea implementar es la incorporación de un sistema de validación previa de las compras por parte de empleados. Actualmente, cuando un cliente desea adquirir un vehículo, la operación se ejecuta de forma directa, sin intervención humana. Sin embargo, en un entorno empresarial real, es imprescindible que un profesional revise y confirme los datos asociados a cada operación.

La propuesta consiste en que, cuando un cliente manifieste su interés en realizar una compra, se genere automáticamente una solicitud pendiente de validación que quedará visible en el panel del empleado. El empleado deberá entonces revisar la información, comprobar la disponibilidad del vehículo y confirmar la operación. Una vez validada, el sistema procedería a formalizar la compraventa y enviaría una notificación personalizada al cliente a través de correo electrónico, confirmando la operación, la fecha acordada y los detalles del vehículo.

Este flujo no solo añade una capa de control y seguridad, sino que también refuerza la transparencia y el seguimiento interno de cada operación comercial.

### 2. Integración de servicios de correo electrónico

Como complemento a la funcionalidad anterior, se prevé la integración de una API de envío de correos electrónicos, como puede ser el servicio de Gmail, SendGrid o Mailgun. Esta funcionalidad permitirá al sistema enviar automáticamente notificaciones importantes a los clientes y empleados, como:

- Confirmaciones de registro y cambios de contraseña.
- Comunicaciones comerciales (nuevas ofertas, vehículos disponibles, promociones).
- Confirmación de ventas o solicitudes pendientes.

Además, se contempla implementar un sistema de verificación de correo electrónico durante el proceso de registro, que obligue al usuario a validar su dirección mediante un enlace de activación enviado por email, garantizando así que las cuentas registradas sean auténticas y operativas.

### **3. Panel de usuario para actualización de datos personales**

Otra mejora clave prevista es la creación de un panel de usuario personalizable, tanto para clientes como para empleados. Actualmente, los datos de los usuarios deben ser gestionados por el administrador, lo cual puede resultar poco flexible. En futuras versiones, se habilitará una sección donde cada usuario pueda:

- Consultar y modificar sus datos personales (nombre, dirección, correo electrónico, número de teléfono, etc.).
- Cambiar su contraseña de acceso mediante un sistema seguro.
- Ver el historial de interacciones o compras (en el caso de los clientes).
- Actualizar su perfil o foto.

Para los empleados, además de lo anterior, se podrán editar los datos profesionales básicos, siempre dentro de los límites establecidos por su rol y permisos en el sistema.

Esta funcionalidad mejora la autonomía del usuario, reduce la carga de trabajo para los administradores y garantiza que la información esté siempre actualizada.

### **4. Gestión documental**

En una versión futura, se plantea añadir un módulo de gestión documental, que permita adjuntar archivos y documentos relevantes a cada operación o entidad del sistema. Por ejemplo:

- Contratos de compraventa vinculados a una venta.
- Copia del DNI o justificante de domicilio asociado a un cliente.
- Ficha técnica o certificado de inspección técnica del vehículo.

- Facturas y comprobantes de gasto.

Estos documentos quedarían almacenados en el sistema y accesibles desde la interfaz gráfica, facilitando la trazabilidad documental y la centralización de la información.

## 5. Panel de estadísticas e informes visuales

Otra línea de trabajo futura muy relevante es el desarrollo de estadísticas e informes, destinado a los roles administrativos y de dirección. A través de gráficos interactivos y resúmenes automáticos, este panel permitiría consultar:

- Ventas por mes, trimestre o año.
- Gasto total acumulado.
- Vehículos más vendidos.
- Ingresos generados por empleado.
- Comparativas entre vehículos por precio, kilometraje, antigüedad, etc.

Este tipo de herramienta aporta valor estratégico, ya que permite tomar decisiones basadas en datos, analizar el rendimiento comercial y detectar patrones de comportamiento.

## 6. Versión web y despliegue en la nube

Actualmente, *Gearborn Motors* funciona como una aplicación de escritorio, lo cual resulta apropiado para entornos internos. Sin embargo, una evolución natural del sistema consistiría en desarrollar una versión web accesible desde cualquier navegador. Esto implicaría construir un frontend con tecnologías como Angular, React o Vue.js, manteniendo el backend en Spring Boot como API REST.

Además, el sistema podría desplegarse en servidores en la nube (como AWS, Azure o Google Cloud), lo que permitiría su uso remoto, mayor escalabilidad y disponibilidad 24/7. Esta mejora abriría las puertas a su comercialización como un producto SaaS (Software as a Service).

## Conclusiones

A lo largo del desarrollo del presente proyecto, titulado *Gearborn Motors*, se han alcanzado de manera satisfactoria la mayoría de los objetivos propuestos en la fase inicial de planificación. Se ha logrado implementar una aplicación funcional, estructurada y completa que permite gestionar de manera integral las operaciones más importantes de un concesionario de vehículos, cumpliendo con los requisitos técnicos, funcionales y organizativos establecidos.

La aplicación cubre áreas clave como la gestión de clientes, vehículos, empleados, ventas y gastos, ofreciendo un entorno gráfico intuitivo y eficiente, así como una arquitectura backend sólida basada en buenas prácticas de desarrollo. Se ha respetado la estructura modular, la separación de responsabilidades y la reutilización del código, lo cual ha facilitado la organización del proyecto y su escalabilidad futura.

No obstante, como en cualquier proyecto de esta naturaleza, existen funcionalidades que, aunque estaban previstas desde el inicio, no pudieron implementarse en esta primera versión. Entre ellas, destacan:

- Verificación de compras mediante validación por parte de empleados: se planificó implementar un sistema en el que las operaciones de compra por parte de los clientes debieran ser validadas previamente por un empleado, introduciendo así una lógica de control que simulara mejor la realidad comercial. Esta funcionalidad, ya estructurada conceptualmente, ha sido postergada como mejora futura.
- Integración de una pasarela de pagos: se contempló la posibilidad de utilizar plataformas como PayPal o Stripe para simular pagos reales dentro de la aplicación. Esta funcionalidad implica retos importantes de seguridad, gestión de tokens y validación de transacciones, por lo que ha quedado fuera del alcance temporal de esta primera entrega.
- Despliegue en la nube mediante AWS: otra de las ideas iniciales era subir el backend y la base de datos a servicios cloud como Amazon EC2 y RDS, permitiendo así el acceso remoto y multiplataforma a la aplicación. Aunque el planteamiento técnico estaba bien definido, no se pudo llevar a cabo en esta fase.
- Integración de una API de correo electrónico: se quería incorporar un servicio externo para validar direcciones de email y enviar notificaciones automáticas a los usuarios desde la propia aplicación. Esta funcionalidad sigue en el plan de desarrollo futuro.

Cabe destacar que una parte significativa de estas funcionalidades se basan en tecnologías que no fueron cubiertas directamente durante el ciclo formativo. Por ejemplo, el uso de Spring Boot, la integración de APIs externas, o la configuración de entornos cloud en AWS, son conocimientos que he adquirido de forma autónoma, en gran parte gracias a mi experiencia durante las prácticas en empresa. Este aprendizaje autodidacta, fuera del temario oficial, ha

sido especialmente valioso y ha enriquecido tanto el enfoque como la calidad técnica del proyecto.

La razón principal por la que algunas funcionalidades avanzadas no pudieron materializarse no responde a una falta de compromiso, sino precisamente a lo contrario. Durante el periodo de prácticas, tuve la oportunidad de formar parte de un proyecto real dentro de la empresa, lo cual implicó un nivel de implicación muy superior al esperado. Asistí a reuniones fuera del horario establecido, colaboré directamente con compañeros de equipo en tareas de diseño y desarrollo, y contribuí activamente al avance del proyecto empresarial. Esta experiencia, aunque exigente, me permitió consolidar habilidades reales del entorno profesional, que sin duda también han influido positivamente en la ejecución del TFC.

A pesar de ello, nunca abandoné el trabajo en *Gearborn Motors*. Desde el inicio hasta la entrega final, mantuve una constancia de trabajo elevada, buscando siempre el equilibrio entre mis responsabilidades en la empresa y el desarrollo de este proyecto. Por todo ello, considero que el resultado final no solo cumple con los objetivos fundamentales, sino que también refleja un crecimiento personal y profesional muy significativo. La base técnica está bien asentada y el camino para futuras mejoras está claro, lo cual garantiza que este proyecto pueda seguir evolucionando y adaptándose a nuevas necesidades y contextos.

## Biblioteca de recursos web y referencias.

Durante la elaboración del presente proyecto se ha priorizado la producción original tanto en la implementación técnica como en la documentación escrita, diagramas y materiales visuales. No obstante, como es habitual en el desarrollo de software y en la elaboración de proyectos complejos, se ha recurrido a diversas fuentes de consulta y apoyo para resolver dudas puntuales, comprender mejor el funcionamiento de determinadas tecnologías y mejorar la presentación general del trabajo.

Entre las fuentes más relevantes utilizadas durante el proceso se incluyen:

- Plataformas especializadas en desarrollo como Stack Overflow, que fueron de gran utilidad para identificar y corregir errores comunes relacionados con Spring Boot, JavaFX y configuración de Maven.
- Videotutoriales y cursos formativos alojados en Platzi y YouTube, orientados a entender de manera más clara conceptos como la autenticación en aplicaciones Java, la conexión con bases de datos, y la estructuración de aplicaciones bajo una arquitectura en capas.
- Documentación oficial de tecnologías utilizadas, como la de Spring Boot, JavaFX, Java 21, y MySQL, para asegurar el uso correcto de las herramientas y seguir buenas prácticas de desarrollo.
- Uso ocasional de ChatGPT como asistente técnico, empleado principalmente para resolver pequeños errores de sintaxis o lógica en fragmentos de código, aclarar conceptos específicos o generar explicaciones para los casos de prueba o partes de la memoria. Además, se ha utilizado como apoyo para mejorar la redacción de algunos textos y descripciones técnicas, y en el tratamiento estilístico de imágenes generadas a partir de capturas propias.

Referencias consultadas:

Stack Overflow. (varios autores): *Foros de preguntas y respuestas sobre programación*, disponible en: <https://stackoverflow.com>

Platzi. (2023): *Curso de Spring Boot, Java Básico y Arquitectura de Software*, disponible en: <https://platzi.com>

YouTube. (varios autores, 2023–2024): *Tutoriales de desarrollo en Java, JavaFX y Spring Boot*, disponible en: <https://www.youtube.com>

OpenAI. (2024): *ChatGPT como asistente de desarrollo al solucionar pequeños errores del código y redacción técnica*, disponible en: <https://chat.openai.com>

Spring Boot Official Docs. (2024): *Spring Framework Documentation*, disponible en: <https://spring.io/projects/spring-boot>

JavaFX Official Guide. (2023): *JavaFX Documentation*, disponible en: <https://openjfx.io>

Todas las imágenes, capturas de pantalla y elementos gráficos incluidos en esta memoria son de elaboración propia, generados directamente desde el desarrollo funcional de la aplicación *Gearborn Motors*, con algunas modificaciones visuales asistidas mediante herramientas de edición o generación automática a partir de escenas reales.