

# MEMORIA ESCRITA DEL PROYECTO

CFGS Desarrollo de Aplicaciones Multiplataforma

## VehicleGest

**Autor:** Carlos Francisco Caruncho Serrano

**Tutor:** Mario Gago

**Fecha de entrega:** 05/12/2022

**Convocatoria:** 2º Semestre - 2022

**Documentos del proyecto:** [Enlace a la carpeta del Drive](#)



## Contenido

<b>1. INTRODUCCIÓN .....</b>	<b>3</b>
1.1. Motivación .....	3
1.2. Abstract .....	4
1.3. Objetivos propuestos (generales y específicos) .....	5
<b>2. METODOLOGÍA USADA .....</b>	<b>6</b>
<b>3. TECNOLOGÍAS Y HERRAMIENTAS UTILIZADAS EN EL PROYECTO .....</b>	<b>9</b>
<b>4. ESTIMACIÓN DE RECURSOS Y PLANIFICACIÓN .....</b>	<b>10</b>
<b>5. ANÁLISIS DEL PROYECTO .....</b>	<b>12</b>
<b>6. DISEÑO DEL PROYECTO .....</b>	<b>23</b>
<b>7. DESPLIEGUE Y PRUEBAS .....</b>	<b>33</b>
<b>8. CONCLUSIONES .....</b>	<b>34</b>
<b>9. VÍAS FUTURAS .....</b>	<b>35</b>
<b>10. BIBLIOGRAFÍA .....</b>	<b>37</b>

## 1. Introducción

**Vehiclegest** es una aplicación móvil desarrollada para el sistema operativo Android, dedicada a gestionar el control y mantenimiento de una flota de vehículos. Sus principales características serán: control y gestión de los datos e incidencias de una flota de vehículos, control de avisos importantes y fechas de vencimiento, consulta y gestión de las Inspecciones Técnicas de Vehículo, listado de servicios prestados por la empresa y control de asignaciones de personal a vehículos e ITV<sup>1</sup>.

**VehicleGest** está diseñado para darle a la empresa un plus de productividad al eliminar los procesos basados en papel, pizarras, asegurando que los datos sean correctos y de fácil acceso desde cualquier parte del mundo, y en tiempo real. Ayudará a funcionar de manera más eficiente y reducirá los costes operativos. En resumen, mejorar la productividad aumentará los ingresos.

### 1.1. Motivación

La motivación principal, surgió debido a la ocupación que tengo en mi puesto de trabajo. Soy responsable de una flota de vehículos de transporte, tanto de carga como de personas. Normalmente todos los datos relacionados con los vehículos se almacenan en hojas de Excel, Word, o pizarras y no son actualizados en tiempo real.

Almacenar datos en hojas de cálculo y de texto nos produce las siguientes desventajas:

- **Incoherencias:** produjeron errores humanos, como la introducción de datos incorrectos.
- **Dificultad para compartir:** Es difícil compartir y colaborar con otras personas en tiempo real a menos que todas las personas tengan acceso a la misma hoja de cálculo o documento de texto en todo momento.
- **No disponible en tiempo real:** Los datos en hojas de cálculo y de texto no están disponibles en tiempo real, por lo que la gente responsable no puede actualizar los datos en el momento.
- **Falta de seguridad:** Las hojas de cálculo y los documentos de texto no tienen las mismas medidas de seguridad que las bases de datos, lo que puede hacer que los datos sensibles sean vulnerables a la pérdida o al robo.

---

<sup>1</sup> La ITV es una revisión periódica y obligatoria a los vehículos de motor, para mantener unas condiciones de seguridad mínimas y contribuir a mantener el medio ambiente. (¿QUÉ ES ITV? NORMATIVA Y TIPOS)

- **Duplicidades de datos:** se producen duplicidades que no se detectan fácilmente, lo que lleva a una falta de precisión en los informes y análisis.

Este proyecto puede ser una solución para estos problemas tanto en mi ámbito como en ámbito civil, para flotas de autobuses, camiones, vehículos de transporte de personas, etc. Los directivos y empleados tendrán un acceso en tiempo real a los datos centralizados de la empresa en sus dispositivos Android.

## 1.2. Abstract

VehicleGest is a complete Android app designed to improve the control and maintenance of a logistics company's vehicle fleet. It features a centralized database of vehicles, services, materials and personnel, which can be accessed by the company's personnel wherever they are.

Store fleet information in non-centralized digital or paper documents is not productive for several reason, such as low availability, inefficient teamwork, low consistency, low security, low reliability, among others. VehicleGest aims to address these issues with its features. It also has security access, ensuring that only fleet employees can view the information.

The app begins with a login screen, you can register but only employees can access to fleet information. The next screen displays a list of vehicles, and there is a main menu at the bottom of this screen for navigation to the other sections. There are five additional sections, each with a search bar with filters to search them.

The second section is a general list of ITVs passed by the company, the third section is a list of services provided by our company. The fourth section is an inventory of all the company's tools and where their assignments, and the fifth section is a list of employees. There is also a section that displays a general list of alerts associated with each vehicle, for upcoming or expired ITV dates and other any information that affects each vehicle.

In conclusion, VehicleGest will improve the productivity of the logistics company, by reducing operating costs and helping employees work more efficiently.

### 1.3. Objetivos propuestos (generales y específicos)

Los **objetivos generales** son:

- **Digitalizar, centralizar, actualizar y controlar** en una base de datos **escalable y coherente** todos los datos de los vehículos, ITV, material, personal y servicios de la empresa.
- **Movilidad y disponibilidad:** Disponer de toda la información en cualquier lugar en los terminales móviles Android de los empleados y directivos de la empresa.
- **Seguridad de acceso.** Evitar fuga de datos personales y empresariales.

Los **objetivos específicos** son:

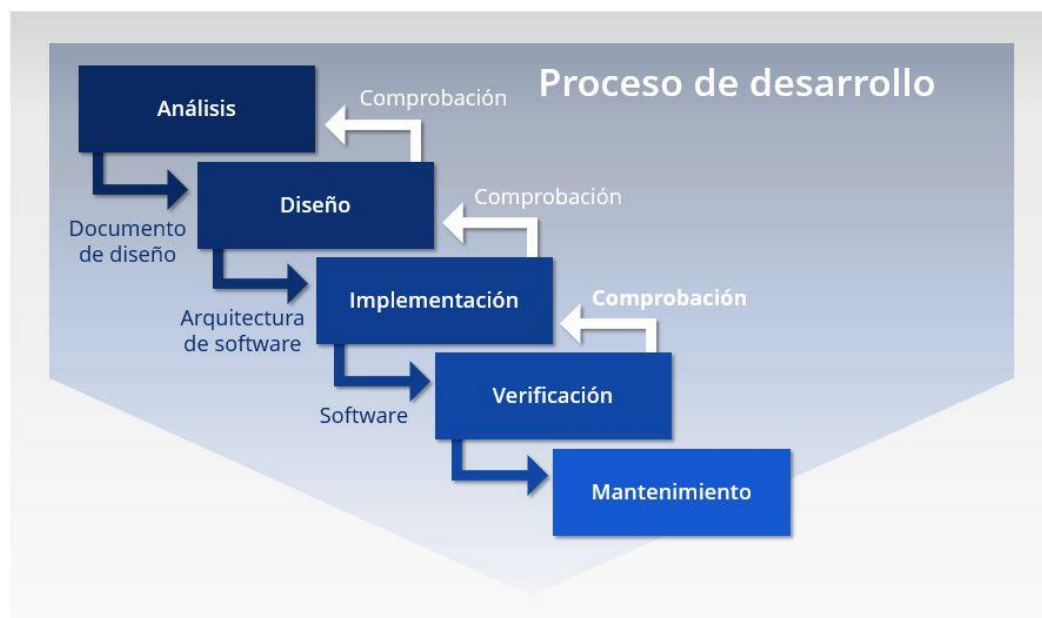
- Crear una **base datos en tiempo real online**, evitándonos invertir en una infraestructura propia para la base de datos.
- Diseñar una interfaz de usuario con una **usabilidad**<sup>2</sup> lo más elevada posible. Debe ser clara y sencilla, con un esquema de colores agradable a la vista.
- Los usuarios deben autenticarse mediante un **sistema de usuario y contraseña**, para mejorar la seguridad, privacidad y la integridad de los datos de la empresa.
- Las partes de la aplicación deben ser **accesibles mediante un menú general**, con iconos y nombres.
- **Listado de vehículos** y su estado de mantenimiento: niveles, estado de ruedas, limpieza, etc.
- **Listado de las ITV**, así como las fechas programadas para estas.
- Debe mostrar un **listado de servicios realizados** por cada uno de los empleados y los vehículos, pudiendo buscar un registro específico mediante un buscador con filtros.
- Debe mostrar un **listado de inventario de herramientas general** en la aplicación y otro individual en cada vehículo, pudiendo detectar que herramientas faltan o están deterioradas, pudiendo buscar un registro específico mediante un buscador con filtros.
- Debe disponer de un **listado de personal de la empresa**, pudiendo acceder a la ficha individual y los listados de servicios y vehículos asociados. Buscar un registro específico mediante un buscador con filtros.

---

<sup>2</sup> La usabilidad es una medida de la facilidad con la que el usuario puede interactuar con una interfaz web o aplicación

## 2. Metodología usada

Cómo **metodología de flujo de trabajo** se ha utilizado la metodología ágil **Kanban**<sup>3</sup>, combinada con la de **cascada con retroalimentación**. (Ilerna Online SL, 2022)



*Ilustración 1: Ciclos de vida del software. Modelo en cascada con retroalimentación. (IONOS, s.f.)*

Las metodologías se impartieron en la asignatura de **Entornos de Desarrollo**. Combinando Kanban y cascada con retroalimentación surge un proceso **iterativo e incremental**. Con Kanban subdividimos el proyecto en pequeñas tareas individuales. Las tareas se representan en una pizarra con fichas o tarjetas, y por fases. (APD, 2021)

Las etapas de codificación y pruebas del ciclo de software se representan en el tablero kanban cómo "codificando" y "probando", pudiendo comprobar la calidad del proyecto final a medida que se va desarrollando. (viewnext, s.f.) En el [Anexo I](#) se muestra una fotografía del tablero de corcho usado para el proyecto.

Las fases en las que se encontrarán las tareas y funcionalidades son:

- ❖ **Pendiente:** En este proyecto no se introducen ideas nuevas, las funcionalidades ya están predefinidas y no se introducirán nuevas. Estarán ya en la fase de pendiente

<sup>3</sup> Sistema ágil de gestión de proyectos en un tablero físico o digital, escribiendo las tareas en tarjetas dividiendo el tablero en fases, pasando cada una de las tareas por todas las fases.

desde el inicio del proyecto. En la fase de pendiente estarán en tarjetas individuales cada una de las tareas del proyecto y funcionalidades de las que consta la aplicación.

- ❖ **En curso:** En la parte superior de esta fase se ponen las tareas que no sean de codificación. La parte inferior está subdividida en dos columnas, “**Codificando**” y “**Probando**”. Cuando una funcionalidad vaya a empezar a ser codificada entra automáticamente a la subfase de “Codificación”. Cuando se considera que está terminada la codificación, se mueve a la subfase de “Probando”.
- ❖ **Finalizado:** En la fase de “Probando” se le hacen las pruebas pertinentes y una vez pasadas se dará por terminada y se pasa a la fase de “Finalizado”.

## 2.1. Ciclo de vida del proyecto. Fases

### 2.1.1. *Iniciación*

Se ha elegido la **temática** para el proyecto, que es mejorar la gestión de los vehículos de una empresa logística. La necesidad es mejorar la gestión, la eficiencia y la productividad de la empresa. Los empleados con acceso a la aplicación tendrán información sobre todos los vehículos de la flota, incluyendo sus defectos técnicos, ITVs, material y servicios. Se ha realizado un **estudio de mercado** y se ha determinado que la aplicación es viable, ya que no hay aplicaciones similares en el mercado.

### 2.1.2. *Planificación*

Se definen las **tecnologías** a utilizar en el [apartado 3](#).. La **viabilidad del proyecto** es viable en términos de tiempo, ya que es un proyecto pequeño que un programador junior puede realizar en ese periodo. Se estiman los **costes** basados en el tiempo utilizado para el proyecto, con un presupuesto de 6000 euros, asumiendo que un programador junior cobra en promedio 1500 euros al mes, y el proyecto se ha realizado en unos 4 meses más o menos. Se ha aplicado un plan de gestión de tiempo con un calendario, y se ha dividido en tareas utilizando un **diagrama de Gantt**, y un **tablero Kanban** para dividir las tareas. La información de planificación se desarrolla en el [apartado 4](#).

### 2.1.3. *Ejecución*

- En esta fase se desarrollan las funcionalidades de la aplicación, se utiliza el método de cascada con retroalimentación mencionadas en el [apartado 2](#). Esta parte se aprende en la asignatura **de Entornos de Desarrollo**.

- **Análisis:** En esta fase se definen los objetivos del software a partir de la idea principal, se desglosan en tareas más pequeñas y se definen los requisitos funcionales y no funcionales. Se desarrolla en el [apartado 5](#).
- **Diseño:** Se decide cómo implementar el software y se diseñan las estructuras de datos, la estructura de los componentes, la interfaz gráfica y los componentes en detalle. Se desarrolla en el [apartado 6.1](#).
- **Implementación:** Se **desarrolla el código fuente** de la interfaz con XML y de la lógica con Kotlin, siguiendo convenciones y normas para escribir un código claro y legible. Se documentó el código mientras se escribía para su posterior mantenimiento. Se desarrolla en el [apartado 6.2](#).
- **Pruebas:** En esta fase se analiza la **calidad** del software. Se detectaron errores en la codificación mediante las pruebas y se corrigieron. Se considera que una prueba es un éxito si encuentra algún error. Se desarrolla en el [apartado 7](#).
- **Mantenimiento:** Podemos considerar mantenimiento el periodo de recuperación del proyecto, donde se ha **reescrito y refactorizado** el código, reparados los bugs y desarrolladas más funcionalidades. Se sacarán nuevas versiones corregidas. En la supervisión hemos revisado a si estamos **cumpliendo los plazos** y de **la consecución de los objetivos**.

#### 2.1.4. Cierre

En este apartado, se ha llevado a cabo una **evaluación exhaustiva del proyecto**, con el fin de determinar su éxito en términos de **cumplimiento de objetivos, calidad de la aplicación y corrección de errores**. Para ello, se han analizado los resultados obtenidos durante el desarrollo del proyecto y se han recopilado lecciones aprendidas.

**Se ha probado su correcto funcionamiento y usabilidad.** Además, se ha completado la documentación correspondiente, asegurándose de que esta sea clara y explique de manera detallada cómo utilizar la aplicación.

Por último, se ha completado la **memoria del proyecto**, incluyendo toda la información relevante obtenida durante el desarrollo del mismo. Con esta evaluación exhaustiva y las correcciones necesarias realizadas, se garantiza una entrega exitosa del proyecto. Desarrollado en el [apartado 8](#).



### 3. Tecnologías y herramientas utilizadas en el proyecto

#### 3.1. Software de diseño vectorial - Microsoft Visio

Microsoft Visio es una herramienta de creación de **diagramas y gráficos vectoriales**<sup>4</sup>. Se ha usado para crear los diagramas de entidad-relación, diagrama relacional, diagrama de casos de uso, y diagrama de clases. He elegido esta herramienta por estar ya familiarizado con ella para hacer planos vectoriales, además es bastante sencilla de utilizar. (Lucidchart)

#### 3.2. Microsoft Office

Excel, PowerPoint y Word son programas de Microsoft que se utilizaron para diferentes propósitos en el proyecto. Excel se utilizó para crear un diagrama de Gantt, PowerPoint para la presentación incluida en el video de defensa y Word para redactar la memoria. Todos estos programas son fáciles de usar y brindan funciones útiles para sus respectivos propósitos.

#### 3.3. Formato PDF

PDF es el acrónimo de Portable Document Format (Formato de Documento Portátil) y es un formato de archivo utilizado para representar documentos de manera uniforme y independiente del hardware o software del que se disponga. Con un simple lector se puede abrir en cualquier máquina. Se ha usado para presentar la memoria.

#### 3.4. Entorno de desarrollo integrado – IDE - Android Studio

Para este proyecto, y tratándose una aplicación nativa para el sistema operativo Android, la mejor elección es **Android Studio**. Se puede programar tanto en Kotlin como en Java. Es el mejor IDE ya que está desarrollado expresamente para el desarrollo de aplicaciones Android, tiene emulador, herramientas de depuración, compilación, y herramientas **Firebase** integradas. Se ha utilizado para pasar los bocetos a mano a layouts en lenguaje **XML**<sup>5</sup>. Esta parte se impartió en las asignaturas de **Entornos de Desarrollo y en Desarrollo de Aplicaciones Multimedia y Dispositivos Móviles..**

---

<sup>4</sup> Son gráficos digitales que no se componen por mapas bits, si no por líneas, círculos y curvas.

<sup>5</sup> XML es un lenguaje de marcado similar a HTML. Significa Extensible Markup Language (Lenguaje de Marcado Extensible) y es una especificación de W3C como lenguaje de marcado de propósito general.

### 3.5. Lenguaje de programación – Kotlin-XML

El lenguaje de programación elegido para la lógica en este proyecto es **Kotlin**, un lenguaje de programación **orientado a objetos, de alto nivel y de código abierto** desarrollado por JetBrains<sup>6</sup>, ya que está totalmente integrado con el IDE Android Studio, y posee una serie de ventajas respecto a JAVA. Se ha tenido que aprender desde cero la sintaxis para poder desarrollar el proyecto. Para las interfaces de usuario, se utiliza **XML** en Android Studio, que cuenta con un editor visual intuitivo. Los conceptos básicos de programación y a desarrollar se aprendieron en las asignaturas de **Programación** y el lenguaje Kotlin en concreto en **Desarrollo de Aplicaciones Multimedia y Dispositivos Móviles**.

### 3.6. Herramientas Firebase para desarrolladores

Se decidió utilizar las herramientas de Google Firebase para el desarrollo del proyecto. Se ha usado **Firebase Auth** para la autenticación de usuarios para el backend de software. (Google). **Firebase Firestore Cloud** como sistema de base de datos, NoSql, escalable y flexible, con soporte sin conexión. (Google). Se adquirieron los conocimientos básicos sobre esta tecnología en **Bases de Datos y en Sistemas Gestores de Bases de Datos**.

### 3.7. Control de versiones con Git y repositorio en Github

Se ha utilizado Git<sup>7</sup> para controlar las versiones del código y se ha almacenado en un repositorio en Github. Se han estado versionando los cambios diarios tanto en código fuente como en la memoria y la documentación. Esta parte se impartió en la asignatura de **Entornos de Desarrollo**.

## 4. Estimación de recursos y planificación

### 4.1. Diagrama de Gantt previsto

En este apartado se presentará una estimación del tiempo previsto y el real que se ha empleado para realizar el proyecto a través de un **diagrama de Gantt**. Este diagrama suele estar compuesto de una lista de tareas a la izquierda y un cronograma de barras a la derecha, como se puede ver en la [ilustración 2](#), que muestra un diagrama de Gantt simplificado. El cronograma ampliado se puede encontrar en el **diagrama previsto** del [Anexo II](#).

---

<sup>6</sup> Empresa checa de desarrollo de herramientas para desarrolladores de software. Ha desarrollado Android Studio basándose en su IDE IntelliJ Idea.

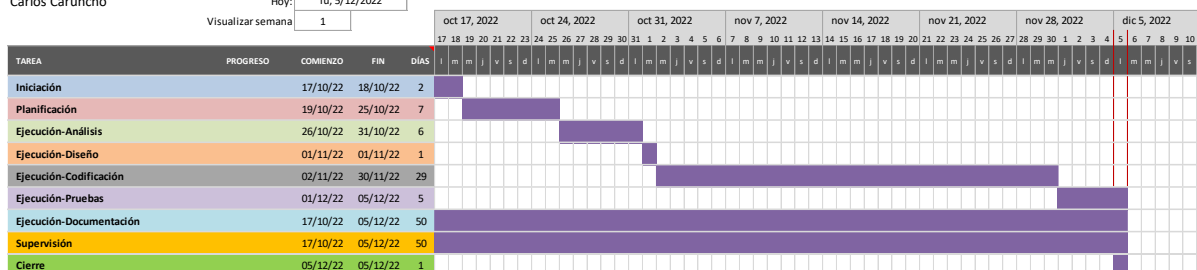
<sup>7</sup> Sistema de control de versiones de código fuente.

La planificación del proyecto se ha realizado utilizando una plantilla de Excel descargada de internet (Vertex42). Se estimaba que la iniciación del proyecto sería rápida, con solo unos pocos días para las tareas cortas. La planificación, por su parte, requeriría un poco más de tiempo, debido a la necesidad de elaborar el diagrama de Gantt y otras tareas relacionadas. La fase de ejecución se subdividió en fases en cascada, con la fase de codificación como la más prolongada. Se planeó realizar las pruebas en dos días y se preveía una cantidad limitada de pruebas. Por último, la documentación se realizaría a lo largo de todo el proyecto.

#### VEHICLEGEST

ILERNA ONLINE  
Carlos Caruncho

Inicio del proyecto: Ju, 20/10/2022  
Hoy: Lu, 5/12/2022  
Visualizar semana 1



SIMPLE GANTT CHART by Vertex42.com  
<https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html>  
© 2018-2019 Vertex42 LLC

Ilustración 2 - Diagrama de Gantt previsto

## 4.2. Diagrama de Gantt – Real

No se pudo completar el proyecto en tiempo y se plasmó el diagrama de Gantt real. Este resultó ser muy diferente al diagrama previsto inicialmente, con la mayor parte de las funcionalidades y la documentación sin completar, lo que muestra que la estimación inicial del tiempo requerido para cada fase no fue precisa. Durante el proceso de ejecución, se requirió un ajuste constante debido a dificultades, falta de tiempo, falta de familiarización con las tecnologías, imprevistos y otras vicisitudes. Esto demuestra la importancia de una planificación cuidadosa, de tener en cuenta las posibles dificultades y obstáculos que puedan surgir durante el desarrollo del proyecto. Se detalla en el **diagrama real** del [anexo II](#).

## 4.3. Diagrama de Gantt – Ampliación

Durante la fase de recuperación del proyecto se completaron las funcionalidades faltantes, se ha refactorizado, reestructurado, depurado y realizado una recodificación completa del proyecto siguiendo los principios del libro Código Limpio (Martin, 2012). Esto ha hecho que el proyecto sea más escalable y legible. La [Ilustración 3](#) muestra el diagrama de Gantt real, que refleja el tiempo invertido en los cambios realizados en el proyecto.

## VEHICLEGEST

ILERNA ONLINE  
Carlos Caruncho

Inicio del proyecto: lu, 9/1/2023  
Hoy: do, 12/2/2023  
Visualizar semana: 1

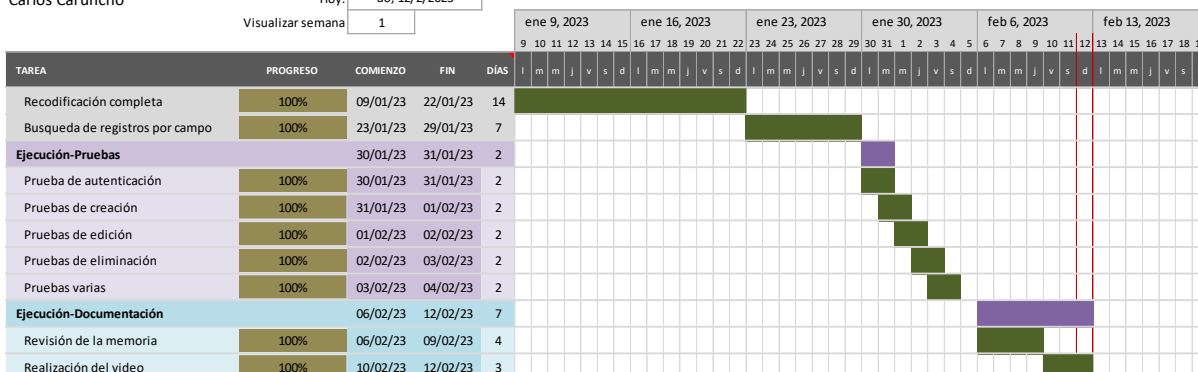


Ilustración 3 - Diagrama de Gantt ampliado

## 5. Análisis del proyecto

### 5.1. Funcionamiento de la aplicación

En esta sección, se describirá en detalle el análisis del funcionamiento esperado de la aplicación. Al iniciar la aplicación, el usuario se encontrará con una pantalla con un logo y un formulario para iniciar sesión. Además, será posible registrarse, pero el permiso solo podrá ser otorgados por un usuario administrador de sistema para acceder a la aplicación.

Una vez iniciada la sesión y que el sistema detecte que el usuario autenticado es administrador, accederá a la pantalla principal de la aplicación. En esta ventana, se mostrará el fragmento del listado general de vehículos. La aplicación contará con un menú de navegación inferior para desplazarse por los distintos fragmentos principales de la aplicación, que son los listados generales de vehículos, servicios, inventario, ITVs y personal. Además, en la parte superior derecha, habrá un icono de una campana para navegar al fragmento de alertas de la aplicación. Al presionar una de estas alertas, se abrirá una ventana emergente con el detalle de esta alerta, que contará con un botón con un icono de basura para borrarla y una cruz para cerrarla.

En la parte inferior de cada fragmento de listado, habrá un botón flotante para abrir una pantalla con un formulario de creación de nuevo registro y un menú superior derecho para editar o eliminar los datos.

En la parte superior, habrá un icono de lupa para buscar registros específicos mediante cadenas de caracteres introducidos, mediante consulta a la base de datos. Al presionar cualquier registro en los listados, se abrirá una pantalla con una ficha más detallada,

que incluirá la foto del registro, si es necesario, y contará con un menú superior de dos botones para borrar el registro o editarlo.

Al presionar el botón de edición se activarán los campos de texto para su edición y se podrán modificar los valores. A su vez se activará un botón para poder salvar estos cambios en la base de datos.

## 5.2. Requisitos funcionales y no funcionales

En esta fase vamos a analizar y definir los **requisitos funcionales** y los **no funcionales** a partir de los objetivos propuestos, es decir las características operativas del software, cual es la interfaz que desarrollamos y sus restricciones.

### 5.2.1. *Requisitos funcionales*

ID	DESCRIPCIÓN
RF001	Iniciar sesión individual
RF002	Ver listado de alertas
RF003	Ver detalle de alerta
RF004	Eliminar alerta
RF005	Buscar alertas
RF006	Listado de vehículos
RF007	Ver detalle de vehículo
RF008	Añadir vehículo
RF009	Editar vehículo
RF0010	Eliminar vehículo
RF0011	Buscar vehículos
RF0012	Ver listado de servicios
RF0013	Ver listado de alertas
RF0014	Ver listado de ITVs de cada vehículo
RF0015	Ver listado de servicios
RF0016	Buscar servicios
RF0017	Ver detalle de servicio
RF0018	Añadir servicio
RF0019	Editar servicio
RF0020	Eliminar servicio

<b>RF0021</b>	Buscar servicios
<b>RF0022</b>	Listado de inventario
<b>RF0023</b>	Ver detalle de ítem de inventario
<b>RF0024</b>	Añadir ítem de inventario
<b>RF0025</b>	Editar ítem de inventario
<b>RF0026</b>	Eliminar ítem de inventario
<b>RF0027</b>	Buscar ítems
<b>RF0028</b>	Ver listado de ITVs
<b>RF0029</b>	Buscar ITV
<b>RF0030</b>	Visualizar detalle de ITV
<b>RF0031</b>	Añadir ITV
<b>RF0032</b>	Editar ITV
<b>RF0033</b>	Eliminar ITV
<b>RF0034</b>	Ver listado de empleados
<b>RF0035</b>	Buscar empleado
<b>RF0036</b>	Añadir empleado
<b>RF0037</b>	Editar empleado
<b>RF0038</b>	Eliminar empleado
<b>RF0039</b>	Restringir usuarios no administradores

### 5.2.2. *Requisitos no funcionales*

<b>ID</b>	<b>DESCRIPCIÓN</b>
<b>RNF001</b>	La interfaz debe tener los controles bien distribuidos
<b>RNF002</b>	La interfaz debe tener una paleta de colores agradable a la vista con colores suaves
<b>RNF003</b>	El menú de navegación debe estar en la parte inferior con iconos distintivos
<b>RNF004</b>	El sistema debe asegurar que los datos estén protegidos del acceso no autorizado
<b>RNF005</b>	La aplicación debe tardar menos de 2 segundos en responder
<b>RNF006</b>	Las alertas serán accesibles en todas las secciones de la aplicación
<b>RNF007</b>	Los accesos a base de datos no bloquearán el hilo principal de la aplicación

<b>RNF008</b>	La aplicación seguirá los patrones de diseño de controles de Google M3 <sup>8</sup>
<b>RNF009</b>	La aplicación podrá ser ejecutada en la mínima versión de Android que permita el diseño M3.
<b>RNF0010</b>	La aplicación debe señalar al empleado que está realizando una operación asíncrona.
<b>RNF0011</b>	El dispositivo debe tener conexión a internet para acceder a la base de datos Firebase.

### 5.3. Diagrama de entidad-relación.

Se ha realizado el diagrama con todas las entidades relacionadas con unas normas llamadas formas normales, con esto nos aseguramos que nuestra base de datos es escalable y bien estructurada. Se puede ver el diagrama en el [anexo III](#).

- **Reglas Primera forma normal (1FN)**
  - Todos los atributos son atómicos, es decir no tienen múltiples valores.
  - Cada entidad tiene una clave primaria única.
- **Reglas Segunda Forma Normal (2FN)**
  - Cumple la 1FN
  - Los atributos deben depender completamente de la clave primaria de la entidad. Suele pasar que no dependen completamente cuando hay clave primaria compuesta.
- **Reglas Segunda Forma Normal (3FN)**
  - Cumple la 2FN
  - No hay relaciones transitivas, es decir ningún atributo depende de otro elemento que no sea la clave primaria.

(Ortega)

### 5.4. Diagrama relacional.

Se puede ver el diagrama completo en el [anexo IV](#).

### 5.5. Diagrama de clases

Se puede ver el diagrama completo en el [anexo V](#).

---

<sup>8</sup> Google Material Design 3 es un sistema de diseño de código abierto para aplicaciones Android.

## 5.6. Diagrama de casos de uso

Se puede ver el diagrama de clases en el [anexo VI](#)

## 5.7. Diseño de Firestore Database

Nos registramos en Firebase y creamos la base de datos Firestore Database. Modelamos los datos pasando del modelo relacional al modelo NoSQL de Firestore. Traducimos al inglés los campos ya que es la lengua universal en programación. (Hernández, 2021) Se pueden consultar en el [anexo VIII](#).

## 5.8. Especificaciones de casos de uso

A continuación, explicamos los casos de uso más representativos de la aplicación. El resto de casos de uso se pueden consultar en el [anexo VII](#).

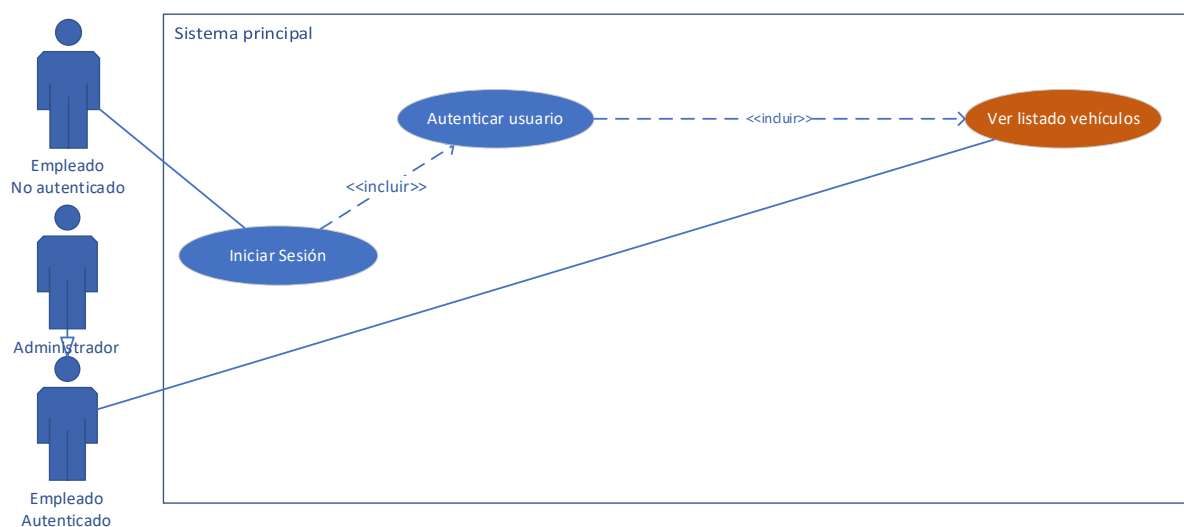


Ilustración 3 : Diagrama casos de uso de sistema principal

Identificador	CU_01	
Nombre	Iniciar sesión	
Descripción	Habrá dos campos, usuario y contraseña al <i>empleado</i> o al <i>empleado administrador</i> , y un botón para aceptar.	
Actores	Empleado, administrador, sistema	
Precondición	Al ejecutar por primera vez solo existe el usuario "admin@admin.com" con contraseña admin123" en la base de datos.	
	Paso	Acción



Secuencia normal	1	El <i>empleado</i> rellena los campos usuario y contraseña.
	2	El <i>sistema</i> comprueba si las credenciales existen en la base de datos.
Postcondición	El empleado queda autenticado y tiene acceso a toda la aplicación.	
Excepciones	Paso	Acción
	2	Si hay error de credenciales el sistema manda un mensaje de error.
Comentarios		

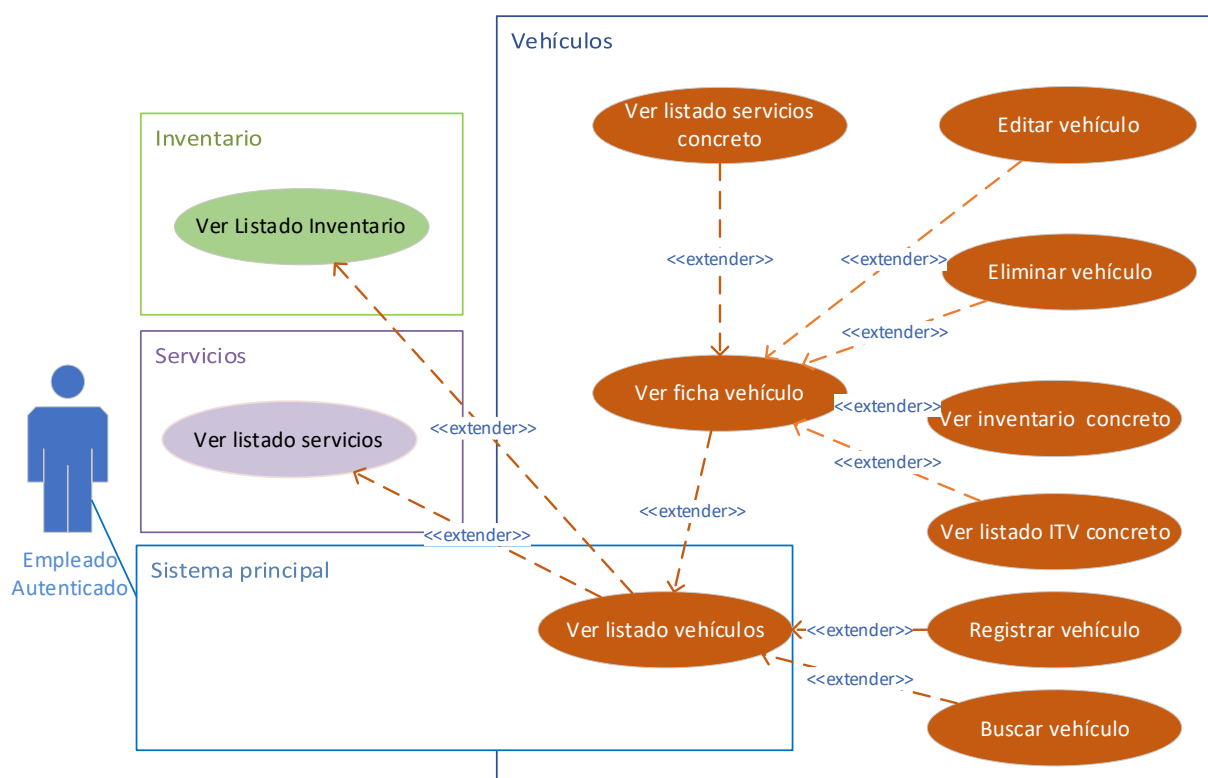


Ilustración 4 - Diagrama de casos de uso de vehículos

Identificador	CU_02
Nombre	Ver listado de vehículos
Descripción	El empleado ve un listado de los vehículos datos de alta en la base de datos. El listado es de fichas de los vehículos con una pequeña foto del vehículo y los datos más importantes: matrícula, marca, modelo y color.

	Se puede presionar un icono de lupa para buscar vehículos por varios criterios.	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	Empleado autenticado.	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El <i>empleado</i> presiona la ficha de un vehículo. Se ejecuta CU_10.
	2	El <i>empleado</i> presiona el icono de lupa arriba derecha. Se ejecuta CU_09
<b>Postcondición</b>		
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
<b>Comentarios</b>		

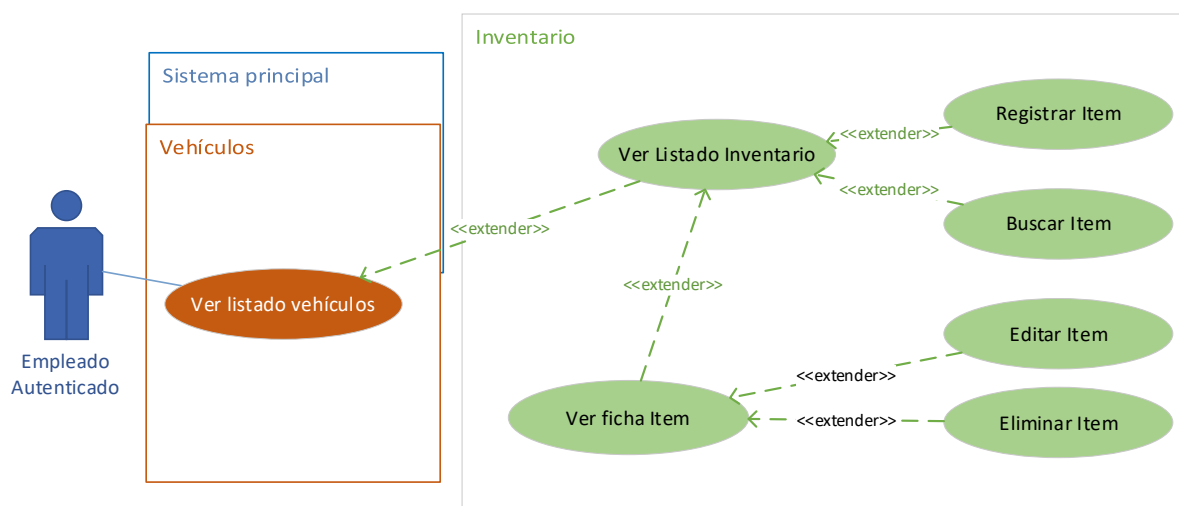


Ilustración 5: Diagrama de casos de uso de inventario

<b>Identificador</b>	<b>CU_03</b>	
<b>Nombre</b>	Ver listado de inventario general	
<b>Descripción</b>	Se ve un listado de herramientas general total que existe en la base de datos. Se puede presionar un icono de lupa para buscar ítems por varios criterios.	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	<i>Empleado autenticado.</i>	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El <i>empleado</i> presiona sobre un registro para abrir el detalle. Ejecuta CU_020.
	2	

<b>Postcondición</b>	Solo se ve el listado de Inventario de ese vehículo en concreto	
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
<b>Comentarios</b>		

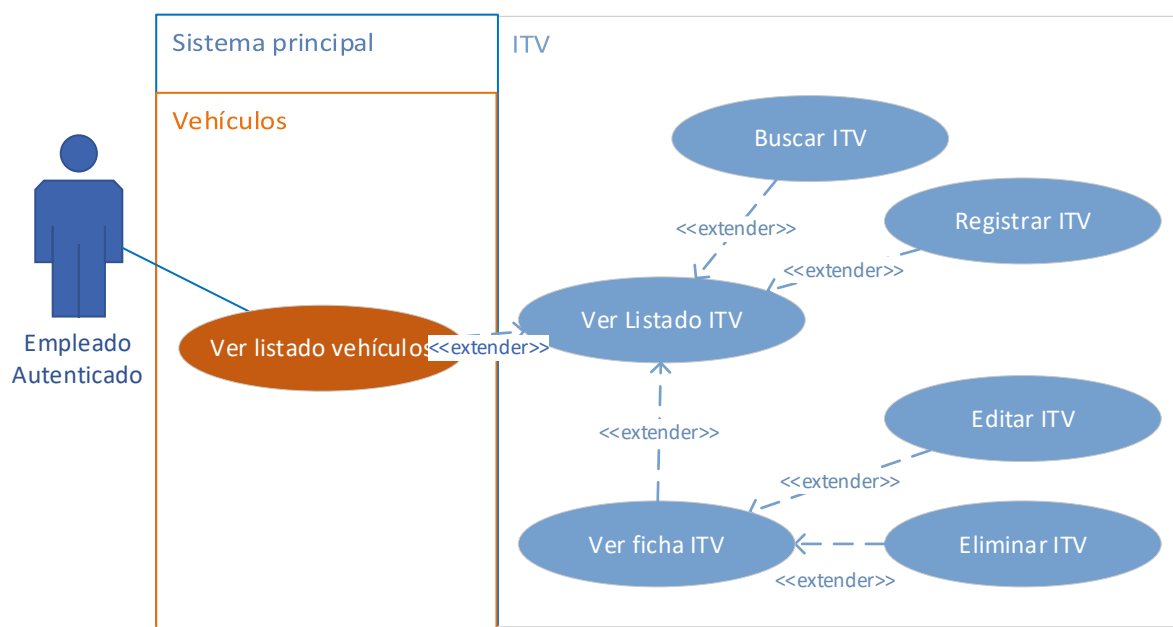


Ilustración 6: Diagrama de casos de uso de ITV

<b>Identificador</b>	<b>CU_04</b>	
<b>Nombre</b>	Ver listado de ITVs	
<b>Descripción</b>	Se ve un listado de ITVs general.	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	<i>Empleado</i> autenticado.	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Secuencia normal</b>
	1	El <i>empleado</i> presiona sobre cada registro para abrir el detalle. Ejecuta CU_22.
	2	
<b>Postcondición</b>		
<b>Excepciones</b>	<b>Paso</b>	<b>Excepciones</b>

## Comentarios

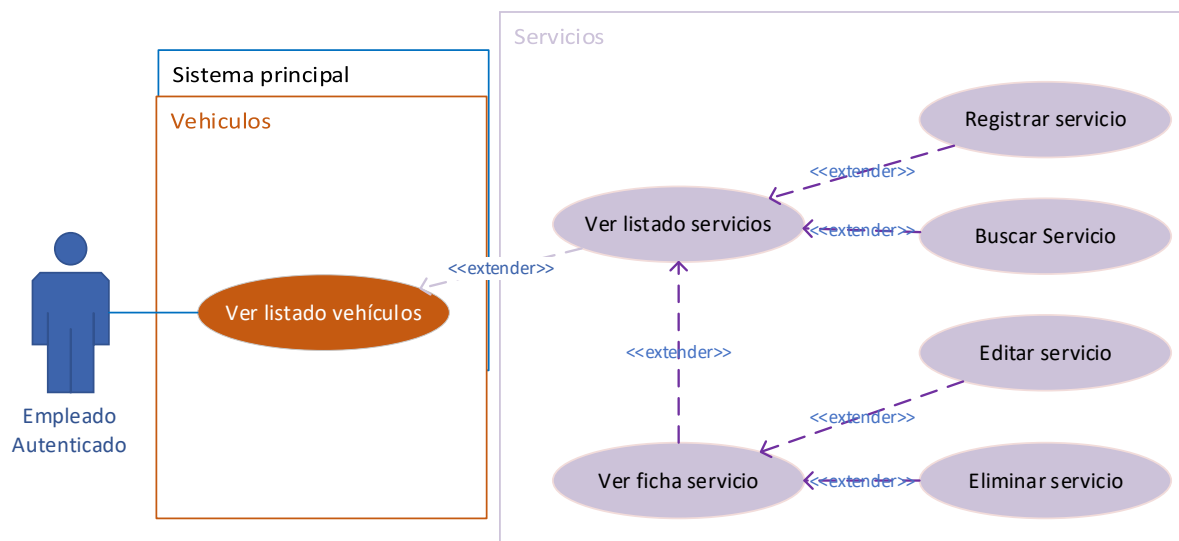


Ilustración 7: Diagrama de casos de uso de servicios

Identificador	CU_05	
Nombre	Ver listado de servicios	
Descripción	Se ve un listado de servicios general.	
Actores	Empleado, administrador.	
Precondición	Empleado autenticado.	
Secuencia normal	Paso	Secuencia normal
	1	El <i>empleado</i> presiona sobre un registro para abrir la ficha de detalle. Se ejecuta CU_026.
	2	
Postcondición		
Excepciones	Paso	Excepciones
Comentarios		

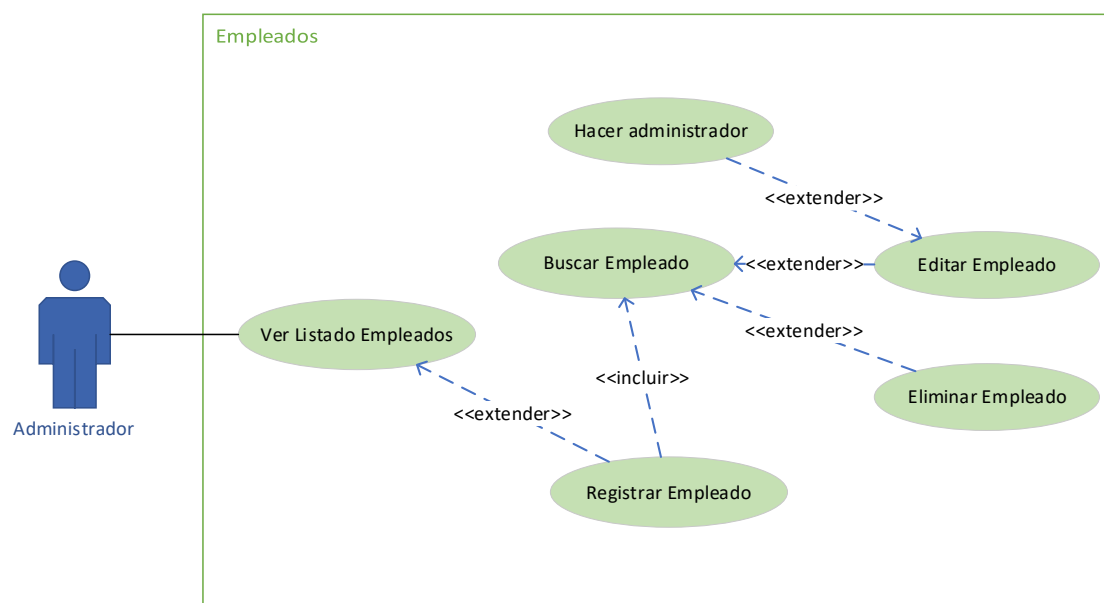


Ilustración 8 - Diagrama de casos de uso de gestión de empleados

Identificador	<b>CU_06</b>	
Nombre	Ver listado de <i>empleados</i>	
Descripción	Se ve un listado de <i>empleados</i>	
Actores	<i>Empleado</i> , administrador, sistema.	
Precondición	Administrador o <i>empleado</i> autenticado.	
Secuencia normal	Paso	Secuencia normal
	1	El <i>empleado</i> presiona sobre un registro de <i>empleado</i> para abrir el detalle. Se ejecuta CU_031.
	2	
Postcondición		
Excepciones	Paso	Excepciones
Comentarios		

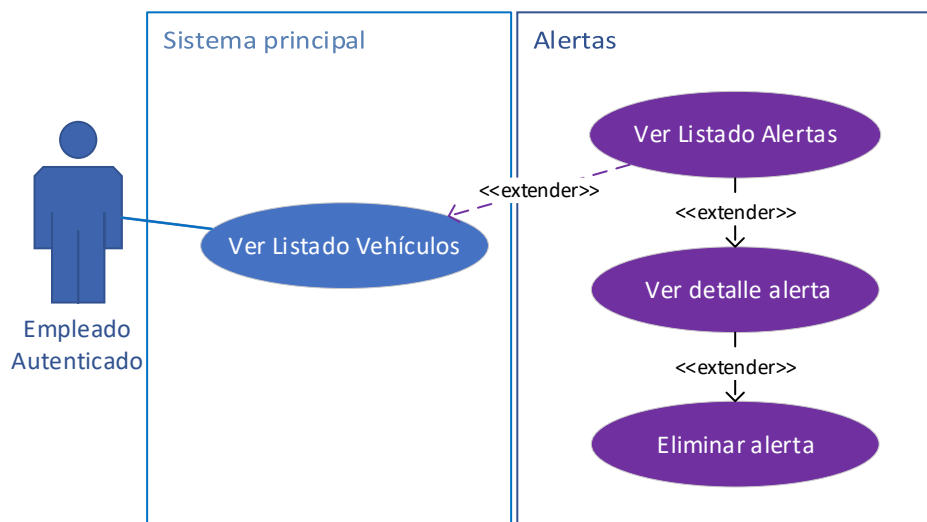


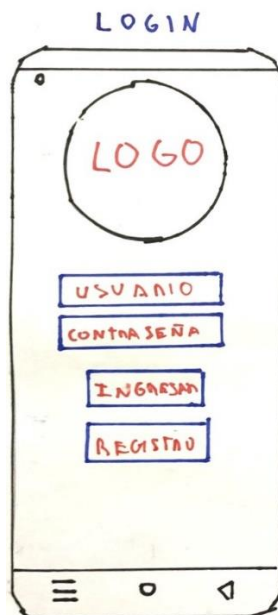
Ilustración 9: Diagrama de casos de uso de alertas

<b>Identificador</b>	<b>CU_07</b>	
<b>Nombre</b>	Ver listado alertas	
<b>Descripción</b>	Se abre al presionar el icono “Alertas” de arriba derecha de la pantalla principal.	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	Empleado autenticado.	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El <i>empleado</i> presiona sobre la alerta para abrir el detalle de la alerta. Se ejecuta CU_08
<b>Postcondición</b>		
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
<b>Comentarios</b>		

## 6. Diseño del proyecto

### 6.1. Bocetos de la aplicación

#### 6.1.1. Pantalla de autenticación de usuario



*Ilustración 10 - Boceto de pantalla de autenticación*

En la ilustración 8 podemos “**activity**”<sup>9</sup> o **actividad** que ve el usuario al entrar a la aplicación después de ejecutarla, en ella aparece el logotipo de la aplicación en grande, y un formulario con los campos de email y contraseña, con un botón para aceptarlo. Esta vista envía los datos introducidos en el formulario a la base de datos de Firestore, si las credenciales son correctas navegará al listado de vehículos. Los usuarios volverán a esta vista cuando se deslogueen desde cualquier vista. En el caso de que no tengan usuario registrado podran crear uno presionando el botón “Registro”, lo que les lleva a la siguiente actividad.

#### 6.1.2. Pantalla de registro de usuario

---

<sup>9</sup> Una activity o actividad es un punto de entrada a la aplicación, se puede decir que es una pantalla y se puede cargar en cualquier momento.

En la ilustración 9 vemos la pantalla de registro, en esta pantalla el usuario deberá ingresar correo y la contraseña dos veces, para asegurarse que la escribe bien, se realiza una comparativa entre los dos campos, si esta es correcta, el sistema le creará un usuario y navegará hacia la pantalla de **actividad principal**.

### 6.1.3. Pantalla de listado de vehículos

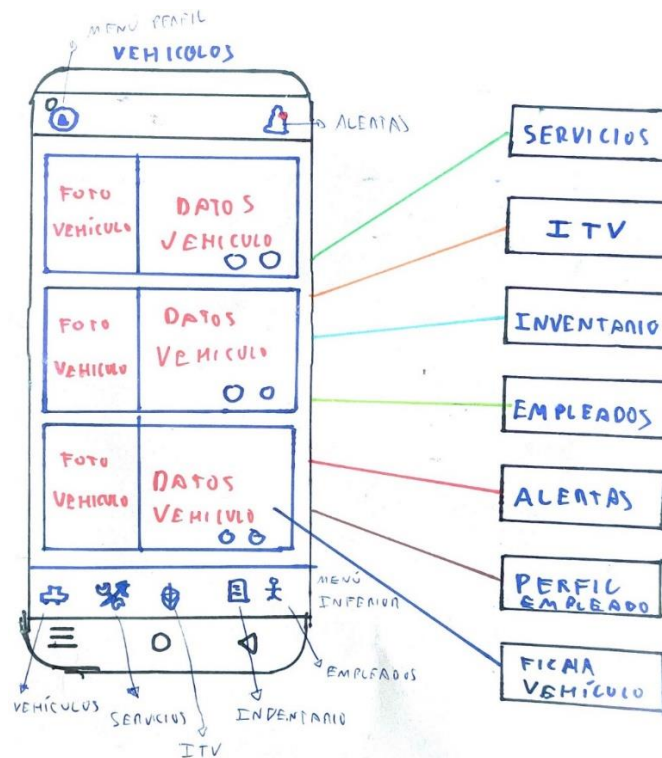


Ilustración 11 - Boceto de pantalla principal

El siguiente boceto es de la actividad principal con el fragmento central cargado de **listado de vehículos**, empezando por la parte de arriba tendremos una barra de herramientas. En la parte derecha de la barra un icono con una campana de notificaciones, podremos navegar al listado de notificaciones y avisos, y otro icono de “deslogueo”, que nos llevará de nuevo a la actividad de login. En la parte central hay un “**recyclerview**<sup>10</sup>” que muestra el listado de ítems, cada ítem tendrá la imagen del vehículo. Esta parte se puede desplazar hacia arriba y hacia abajo con un “**scrollview**<sup>11</sup>” vertical. Esta parte central será también dinámica para la navegación y cambiará de “**fragment**<sup>12</sup>” o **fragmento** cuando

<sup>10</sup> Es una lista dinámica de ítems almacenados en la base de datos con imágenes y posibilidad de presionarlas y seleccionadas. Mejora el rendimiento al reciclar las vistas de los ítems.

<sup>11</sup> Un scrollview es un control que permite mostrar una vista más amplia de lo que ocupa la pantalla desplazando hacia arriba o abajo o a los lados.

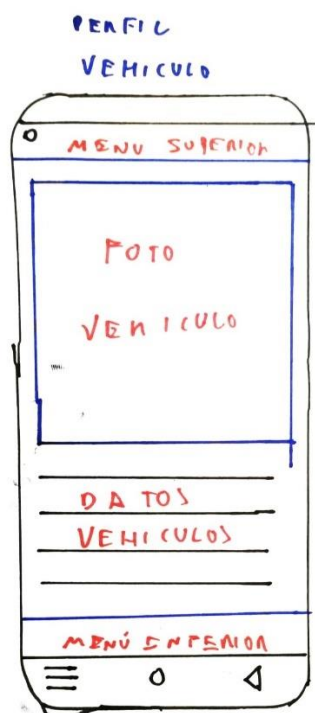
<sup>12</sup> Un fragmento es una parte reutilizable de la vista, no puede existir por si sola, esta alojada en otra actividad principal.



presionemos sobre cada uno de los iconos de avisos, o del menú inferior de navegación. El menú inferior de navegación está compuesto de 5 iconos. Para navegar hacia los apartados de vehículos, servicios, ITV, inventario y personal. También habrá icono “+” flotante para agregar registros en cada apartado.

Presionando la ficha de vehículos se abrirá una pantalla emergente donde se ampliará la foto del vehículo, y se podrán consultar en detalle la información del vehículo, borrarlo y editarlo.

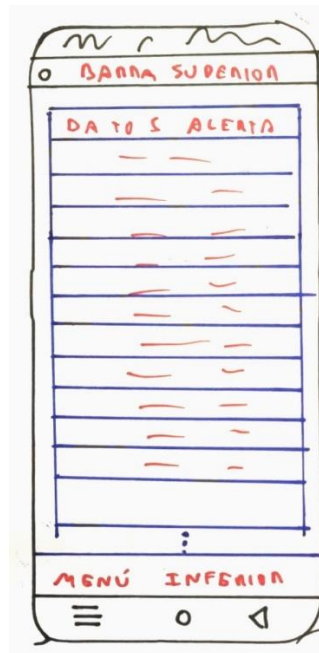
#### **6.1.4. Dialogo de vista de detalle de vehículo**



*Ilustración 12 - Boceto de detalle de vehículo*

En este dialogo podemos ver la foto del vehículo que ocupa media pantalla, debajo de la foto estarán todos los datos del vehículo pudiendo bajar también con un “scrollview” para ver más datos. Habrá tres botones superiores donde se puede volver atrás, borrar el registro y otro para editar el vehículo

### 6.1.5. Vista de listado de alertas

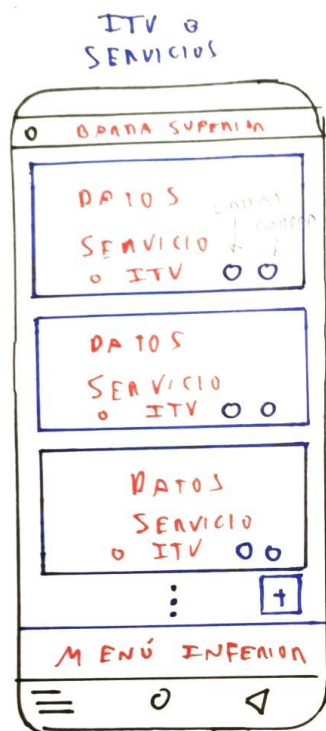


*Ilustración 13 - Boceto de listado de alarmas*

En este fragmento podremos ver una lista de todas las alertas generadas en una lista, esta lista será una “recyclerview” simple, con la fecha las alertas, el motivo y la fecha de su generación.

Se podrán borrar desde un botón superior y agregar más desde un botón de + inferior.

### **6.1.6. Vista de listado de servicios**



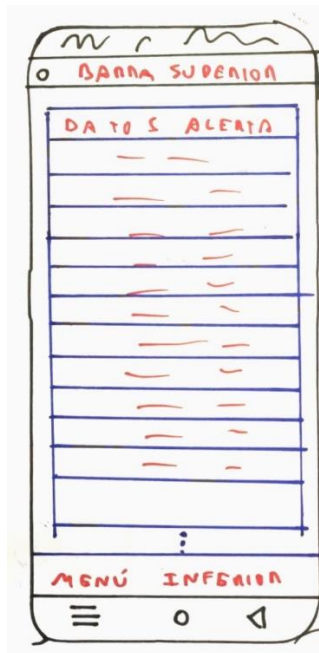
*Ilustración 14 - Boceto de listado de servicios*

En la ilustración 12 podemos ver el fragmento de listado de servicios, es similar al de ITV, pero variando los datos. Se podrán agregar y abrir en detalle cada uno de los servicios.

### **6.1.7. Vista de detalle de servicios**

En esta vista se podrá ver un fragmento con el detalle del servicio que hemos presionado. Se podrán borrar o editar los datos. Será como la de vehículo pero sin la fotografía.

### **6.1.8. Vista de listado de inventario**



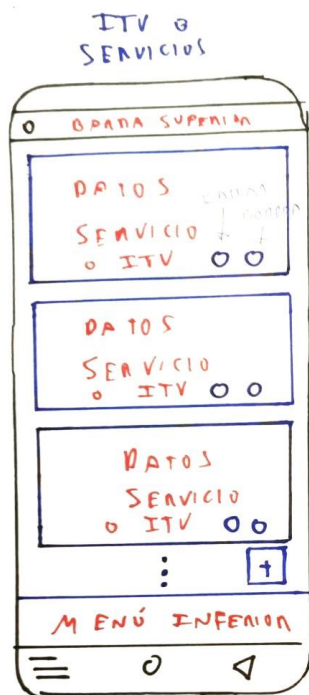
En la ilustración 12 podemos ver el fragmento de listado de items, es similar a las alertas, pero variando los datos. Se podrán agregar y abrir en detalle cada uno de los

Se podrán borrar desde un botón superior y agregar más desde un botón de + inferior.

### **6.1.9. Vista de detalle de ítem**

En esta vista se podrá ver un fragmento con el detalle del ítem que hemos presionado. Se podrán borrar o editar los datos. Es igual que la de vehículo.

### **6.1.10. Vista de listado de ITV**



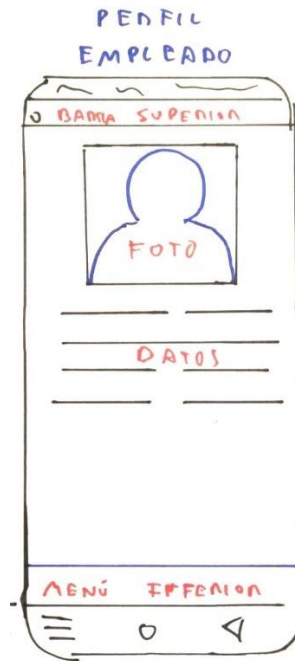
*Ilustración 15 - Boceto de listado de ITV*

En la ilustración 12 podemos ver el fragmento de listado de servicios, es similar al de servicios, pero variando los datos. Se podrán agregar y abrir en detalle cada uno de las ITV.

### **6.1.11. Vista de listado de personal**

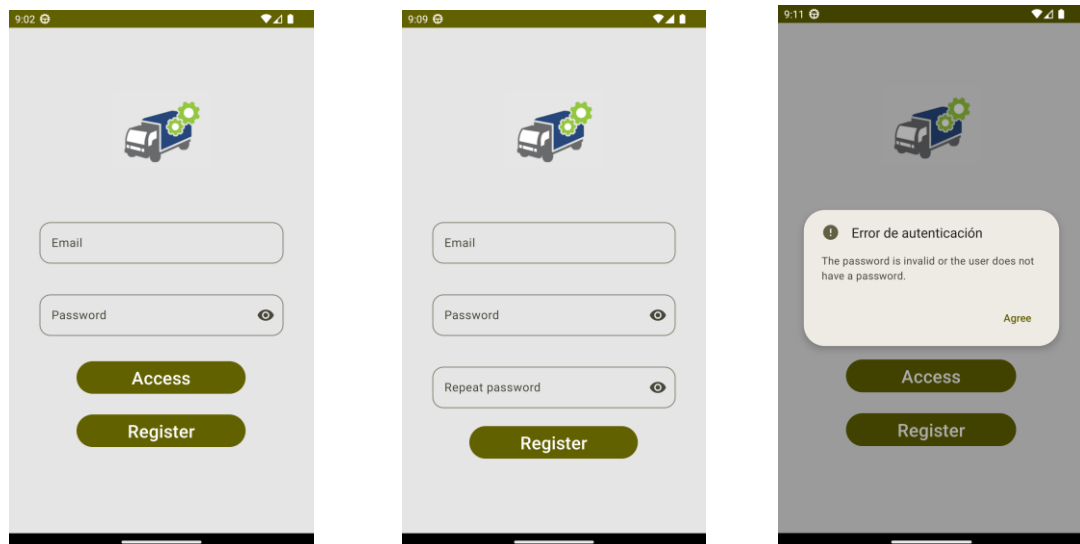
Esta vista es similar a la de listado de vehículos, con los datos esenciales en cada registro y una foto en minatura del empleado, se podrá presionar sobre cada empleado para ver los detalles.

### 6.1.12. Vista de detalle de empleado



## 6.2. Implementación en detalle

### 6.2.1. Implementación de la autenticación y registro

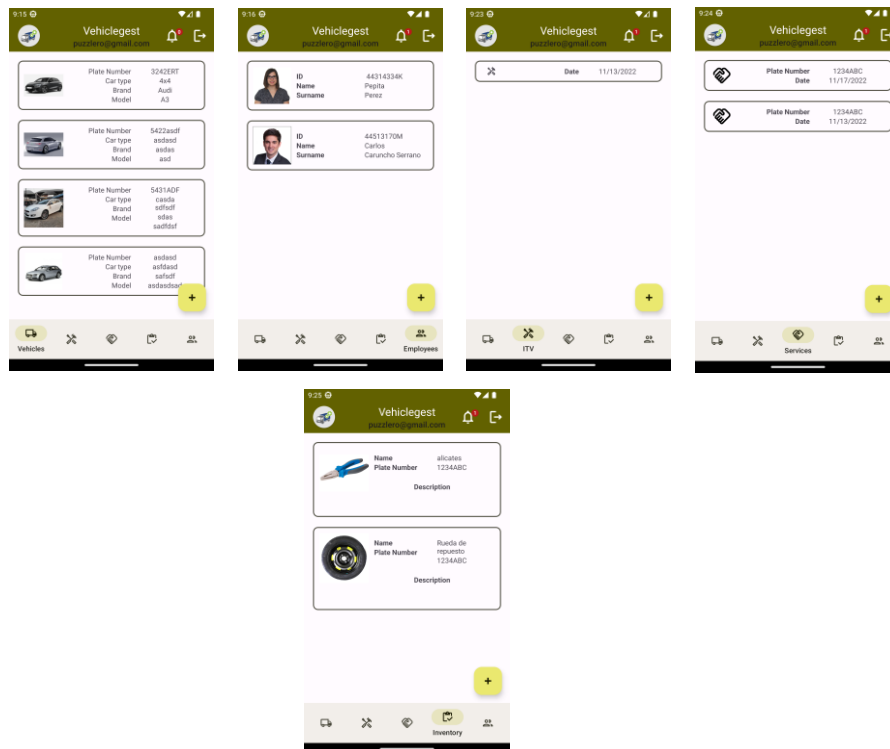


Se implementa la conexión a la base de datos de Firestore auth. Se ha utilizado la documentación de Firebase para implementarlo. <https://firebase.google.com/docs/auth>.

La aplicación puede estar en castellano e inglés, ya que se han traducido cada una de las cadenas de texto con el sistema de android studio, almacenadas en los archivos strings.xml

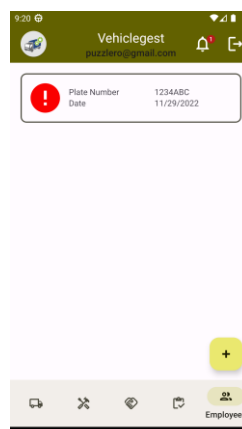
Se ha implementado dialogo de información de errores de usuario y contraseña.

### 6.3. Menú de navegación de fragmentos



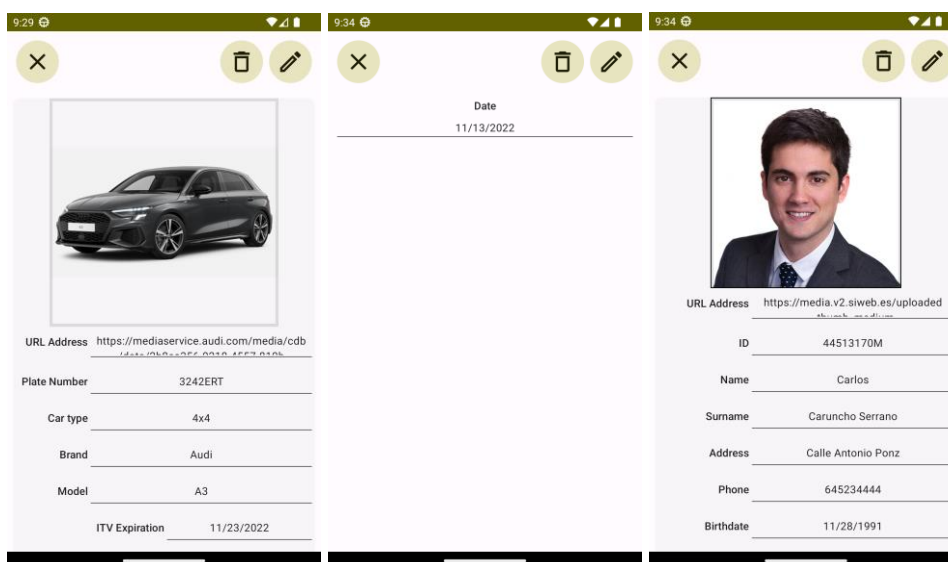
Mediante la implementación del menú inferior podemos navegar entre los fragmentos, que aparecen en un frame central manteniendo los menús superior e inferior, y el botón flotante para agregar registros.

### 6.4. Barra superior de información y navegación



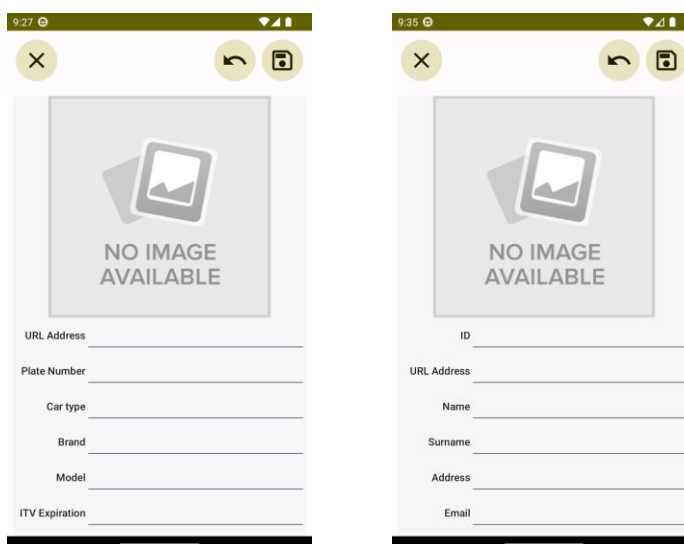
En la barra superior podemos encontrarnos un botón para desloguear y volver a la actividad de autenticación. El icono de avisos con un contador de avisos implementado. También aparece el título de la aplicación y la dirección de correo electrónico del usuario que está logueado.

### **6.2.2. Pantallas de detalle de registros**



En estas pantallas nos encontramos un icono para cerrar la ventana, uno para borrar el registro y otro para editar la información del vehículo. La fotografía se carga mediante la URL almacenada en la base de datos. Los menús se esconden al entrar a los fragmentos.

### **6.2.3. Pantallas de formularios de registro**





En los formularios de registro nos encontramos un botón de cerrar el fragmento, otro de borrar los datos ingresados en el formulario y otro para guardar los datos ingresados en el formulario. Los menús se esconden al entrar a los fragmentos.

## 7. Despliegue y pruebas

Nº	ESPECIFICACIÓN DE PRUEBAS
1	<p><b><u>Objetivo probado:</u></b> Apertura de la aplicación</p> <p><b><u>Requisitos probados:</u></b> Debe aparecer la actividad de login para solicitar los datos de ingreso</p> <p><b><u>Pruebas que realizar:</u></b> Se abre la aplicación y se ingresan los datos de usuario.</p>
2	<p><b><u>Objetivo probado:</u></b> Registro de usuario</p> <p><b><u>Requisitos probados:</u></b> Debe aparecer la actividad de login para solicitar los datos de ingreso</p> <p><b><u>Pruebas que realizar:</u></b> Se abre la aplicación y se ingresan los datos de usuario.</p>
3	<p><b><u>Objetivo probado:</u></b> Ver actividad principal</p> <p><b><u>Requisitos probados:</u></b> Debe aparecer la actividad principal con el fragmento central de vehículos</p> <p><b><u>Pruebas que realizar:</u></b> Se abre la actividad y se puede presionar sobre los iconos de los menús.</p>
4	<p><b><u>Objetivo probado:</u></b> Ver listados de registros</p> <p><b><u>Requisitos probados:</u></b> Deben aparecer los registros de vehiculos, servicios, personal, inventario, ITV y alertas.</p> <p><b><u>Pruebas que realizar:</u></b> Se conecta a la base de datos y muestra los registros de cada una de las listas,</p>
5	<p><b><u>Objetivo probado:</u></b> Crear registros</p> <p><b><u>Requisitos probados:</u></b> Crear registros en la base de datos.</p> <p><b><u>Pruebas que realizar:</u></b> Se conecta a la base de datos y registra los datos dados ingresados en los formularios de datos de cada una de las secciones</p>
5	<p><b><u>Objetivo probado:</u></b> Borrar registros</p> <p><b><u>Requisitos probados:</u></b> Borrar registros en la base de datos.</p> <p><b><u>Pruebas que realizar:</u></b> Se conecta a la base de datos y borra los datos dados ingresados en los formularios de datos de cada una de las secciones</p>

## 8. Conclusiones

Tras un análisis exhaustivo, podemos afirmar que el proyecto ha sido un **éxito** en términos de cumplimiento de objetivos y calidad de la aplicación final. La elección de **herramientas adecuadas, gratuitas y fáciles de usar**, ha sido clave en el éxito del proyecto. Se adquirieron habilidades valiosas en Firestore, un sistema de datos basado en JSON, el IDE Android Studio, la programación en Kotlin y nuevos conceptos de programación en el sistema operativo Android.

A mitad del desarrollo del proyecto, se empezaron a seguir los principios del libro "**Código Limpio**" de **Robert C. Martin** (Martin, 2012), lo que permitirá una organización eficiente y una facilidad de mantenimiento a largo plazo. Estos principios son la base de un código mantenible, confiable y eficiente, y son una guía para desarrolladores para escribir código de alta calidad. Esto requirió tiempo extra para reestructurar y reescribir gran parte del código. Se ha aprendido que hay muchas maneras de escribir código para hacer una misma tarea, pero la diferencia entre una forma u otra puede determinar la escalabilidad de la aplicación.

Además de los aspectos positivos mencionados anteriormente, también debemos mencionar las **dificultades** que se han enfrentado durante el desarrollo del proyecto. Como alumno de programación sin experiencia previa, la tarea de desarrollar una aplicación completa ha sido un reto importante. La falta de conocimientos técnicos y la falta de experiencia en el uso de herramientas y tecnologías específicas han requerido una **dedicación extra en la investigación y la documentación**.

El **tiempo limitado** ha sido un factor crucial para el desarrollo del proyecto, debido a tener que compaginarlo con su trabajo. La planificación y la estimación de tiempos han sido un reto en sí mismos, y se han requerido ajustes y cambios a lo largo del proceso para cumplir con los objetivos y plazos establecidos. La falta de tiempo ha limitado la cantidad de funcionalidades que se podrían haber agregado y ha obligado a priorizar y seleccionar cuidadosamente las funciones más importantes.

En resumen, a pesar de las dificultades mencionadas, el estudiante ha logrado desarrollar una **aplicación funcional y de calidad**. Este proyecto ha sido una oportunidad valiosa para adquirir nuevas habilidades y conocimientos, y ha demostrado la importancia de perseverar y aprender a lo largo del camino.

## 9. Vías futuras

Como vías futuras para mejorar la aplicación se consideran:

1. Mejorar la interfaz general de la aplicación aplicando el estándar de diseño Material3 en las partes que lo requieran.
2. Optimizar la colocación de los menús para que sea más intuitivo para el usuario.
3. Reestructurar la base de datos para hacerla más eficiente y adaptada al tipo de aplicación.
4. Agregar la posibilidad de imprimir informes de las deficiencias de cada vehículo.
5. Implementar las funcionalidades no completadas como los listados de herramientas, ITVs en cada vehículo.
6. Incluir en la base de datos la información de los clientes, para poder marcar en cada servicio a quién se le ha realizado.
7. Agregar listado de deficiencias de cada vehículo, en un submenú en el apartado de vehículos.
8. Crear un tablón central de avisos e información general que aparezca al entrar el usuario a la aplicación.
9. Desarrollar un sistema de roles para dar permisos de acceso según se requiera. Estas mejoras permitirán a la aplicación ofrecer una experiencia más satisfactoria al usuario, aumentar la eficiencia en la gestión de datos y garantizar un buen funcionamiento en general.
10. Incluir más datos en cada una de las secciones.
11. Integración con sistemas externos, como por ejemplo una aplicación de seguimiento GPS para el rastreo de los vehículos en tiempo real.
12. Implementación de un sistema de notificaciones(push) para alertar a los usuarios de nuevos avisos o información importante relacionada con su vehículo.
13. Implementar un sistema de generación de informes, para sacar documentación de la base de datos.
14. Seguir usando el libro "Código Limpio" de Robert C. Martin, mantener un código legible, documentado y optimizado, mientras se desarrollan nuevas funcionalidades y se mejoran las existentes

En general, hay muchas posibilidades para mejorar y ampliar la aplicación, con el objetivo de brindar una solución más completa y satisfactoria a los clientes.

VehicleGest

Carlos Fco. Caruncho Serrano



## 10. Bibliografía

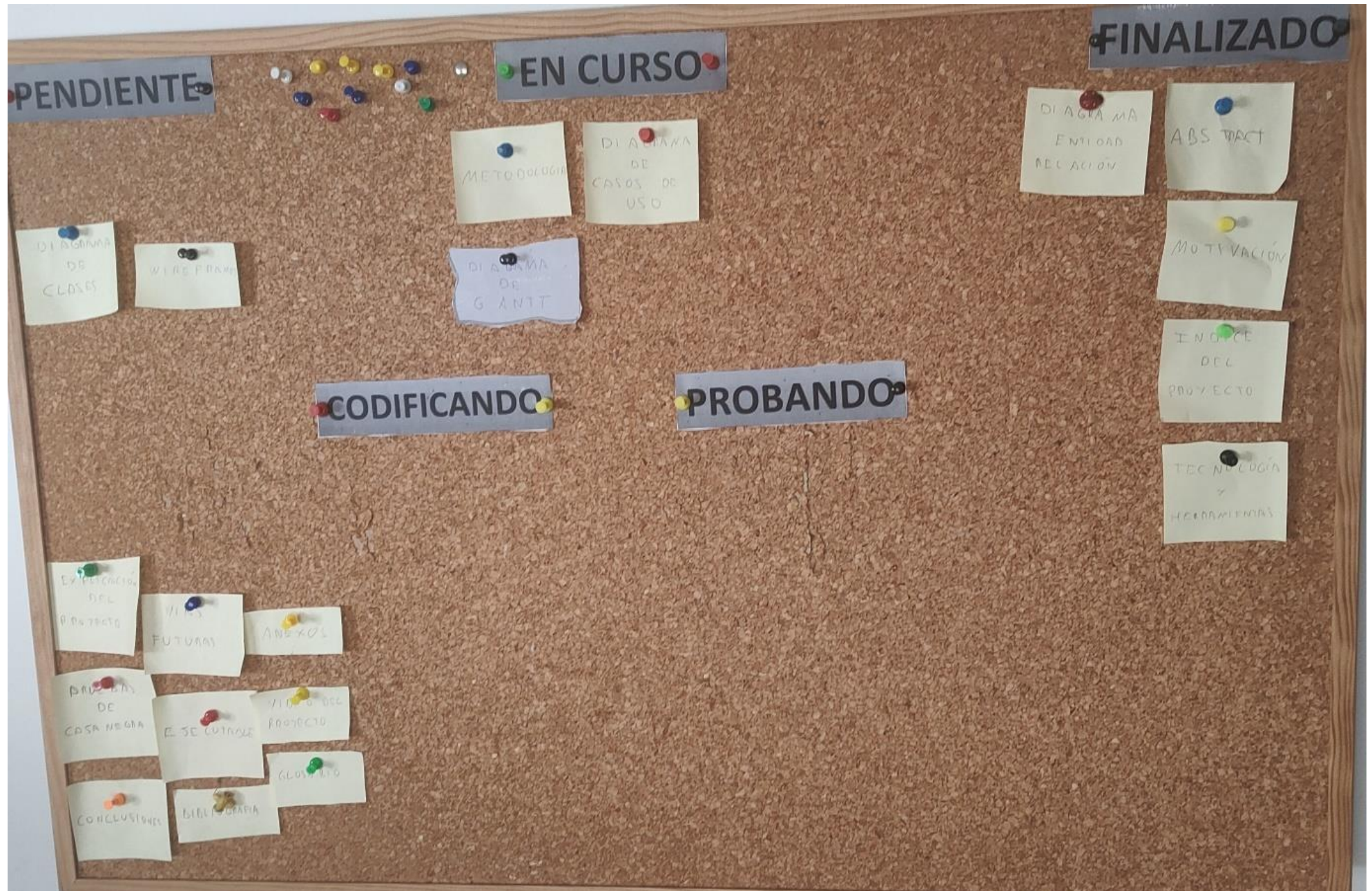
- ¿QUÉ ES ITV? NORMATIVA Y TIPOS. (n.d.). From Servicios ITV:  
<https://www.serviciositv.es/que-es-itv>
- APD. (2021, Junio 8). *¿En qué consiste la metodología Kanban y cómo utilizarla?* From APD:  
<https://www.apd.es/metodologia-kanban/>
- Curso Kotlin. (n.d.). *Curso Kotlin*. From Capítulo 26 – DatePicker en Kotlin:  
<https://cursokotlin.com/capitulo-26-datepicker-en-kotlin/>
- Fox Android. (n.d.). *Bottom Navigation Bar - Android Studio*. From Bottom Navigation Bar - Android Studio: <https://www.youtube.com/watch?v=YIIHxIAoHzU&t=119s>
- Google. (n.d.). *Firebase Auth*. From Firebase Auth: <https://firebase.google.com/docs/auth>
- Google. (n.d.). *Firestore Cloud Plataform*. From Firestore Cloud Plataform:  
<https://firebase.google.com/docs/firestore>
- Hernández, C. (2021, Febrero 24). *Modelado NoSQL con Firebase Firestore*. From GDG Marbella Youtube: <https://www.youtube.com/watch?v=UHIhOKctxD8>
- Ilerna Online S.L. (n.d.). Programación Multimedia y Dispositivos Móviles. In I. O. S.L., *Programación Multimedia y Dispositivos Móviles*.
- Ilerna Online SL. (2022). Metodología Kanban. In I. O. SL, *Entornos de desarrollo*. Lleida: Ilerna Online SL.
- Ilerna S.L. (2021). *Entornos de Desarrollo*. Ilerna S.L.
- Invarato, R. (n.d.). *Context de Android*. From Jarroba: <https://jarroba.com/context-de-android/>
- IONOS. (n.d.). *El modelo en cascada: desarrollo secuencial de software*. From IONOS:  
<https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/el-modelo-en-cascada/>
- Lucidchart. (n.d.). *Que es Visio*. From <https://www.lucidchart.com:https://www.lucidchart.com/pages/es/que-es-microsoft-visio>
- Martin, R. C. (2012). *Código Limpio*. ANAYA.
- Meardon, E. (n.d.). *About Gantt Charts*. From Altassian:  
<https://www.atlassian.com/es/agile/project-management/gantt-chart>

- Microsoft. (n.d.). *Diagrama de Gantt de dos años*. From Office.com: <https://templates.office.com/es-es/diagrama-de-gantt-de-dos-a%C3%B1os-tm56599548>
- Moure, B. (n.d.). *FIREBASE Authentication Android*. From FIREBASE Authentication Android : <https://www.youtube.com/watch?v=dpURgJ4HkMk>
- NextU. (n.d.). *¿Qué es Json? ¿Por qué es importante conocerlo?* From NextU: <https://www.nextu.com/blog/que-es-json-por-que-es-importante-conocerlo-rc22/#:~:text=En%20resumen%2C%20JSON%20no%20es,para%20transferir%20informaci%C3%B3n%20entre%20sistemas>.
- Ortega, Á. (n.d.). *Cursos de programación*. From Código Online: [www.codigoonline.es](http://www.codigoonline.es)
- Presta, M. (n.d.). *¿Qué es Firebase? Todos los secretos desbloqueados*. From back4app: <https://blog.back4app.com/es/que-es-firebase/>
- Teamleader. (2021, Agosto 18). *¿Qué es y para qué sirve un diagrama de Gantt?* From Teamleader: <https://www.teamleader.es/blog/diagrama-de-gantt>
- Tecnosoluciones. (n.d.). *Te damos 10 razones para usar la metodología Kanban en tu organización*. From Tecnosoluciones: <https://tecnosoluciones.com/10-razones-para-usar-la-metodologia-kanban-en-tu-organizacion/>
- Trello. (n.d.). *Aprende los aspectos básicos del tablero de Trello*. From Trello: <https://trello.com/guide/>
- Vertex42. (n.d.). *Simple Gantt Chart*. From Vertex42: [https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html?utm\\_source=v42&utm\\_medium=file&utm\\_campaign=templates&utm\\_term=simple-gantt-chart\\_ms&utm\\_content=url](https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html?utm_source=v42&utm_medium=file&utm_campaign=templates&utm_term=simple-gantt-chart_ms&utm_content=url)
- viewnext. (n.d.). *El ciclo de vida de las metodologías ágiles de desarrollo*. From viewnext: <https://www.viewnext.com/el-ciclo-de-vida-de-las-metodologias-agiles-de-desarrollo/>
- Vlsure Solutions. (n.d.). *Qué son los requisitos funcionales: ejemplos, definición, guía completa*. From VlsureSolutions: <https://visuresolutions.com/es/blog/functional-requirements/>
- Wikipedia. (2022, Octubre 4). *Android Studio*. From Wikipedia: [https://en.wikipedia.org/wiki/Android\\_Studio](https://en.wikipedia.org/wiki/Android_Studio)

Wikipedia. (2022, Octubre 14). *Bases de datos*. From Wikipedia:  
[https://es.wikipedia.org/wiki/Base\\_de\\_datos](https://es.wikipedia.org/wiki/Base_de_datos)



## ANEXO I – Tablero Kanban

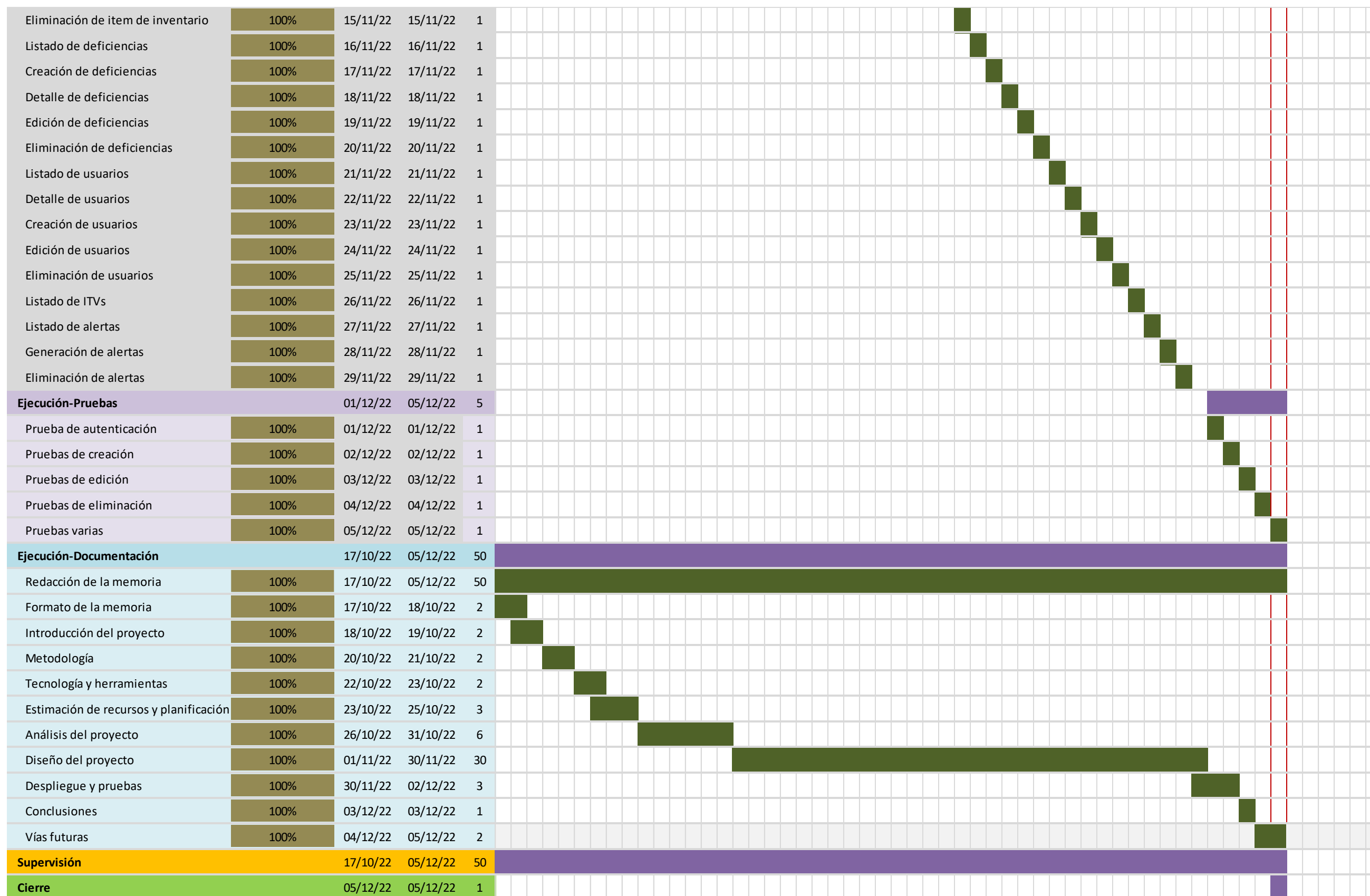




## DIAGRAMA PREVISTO

Inicio del proyecto:	ju, 20/10/2022	
Hoy:	lu, 5/12/2022	
Visualizar semana	1	

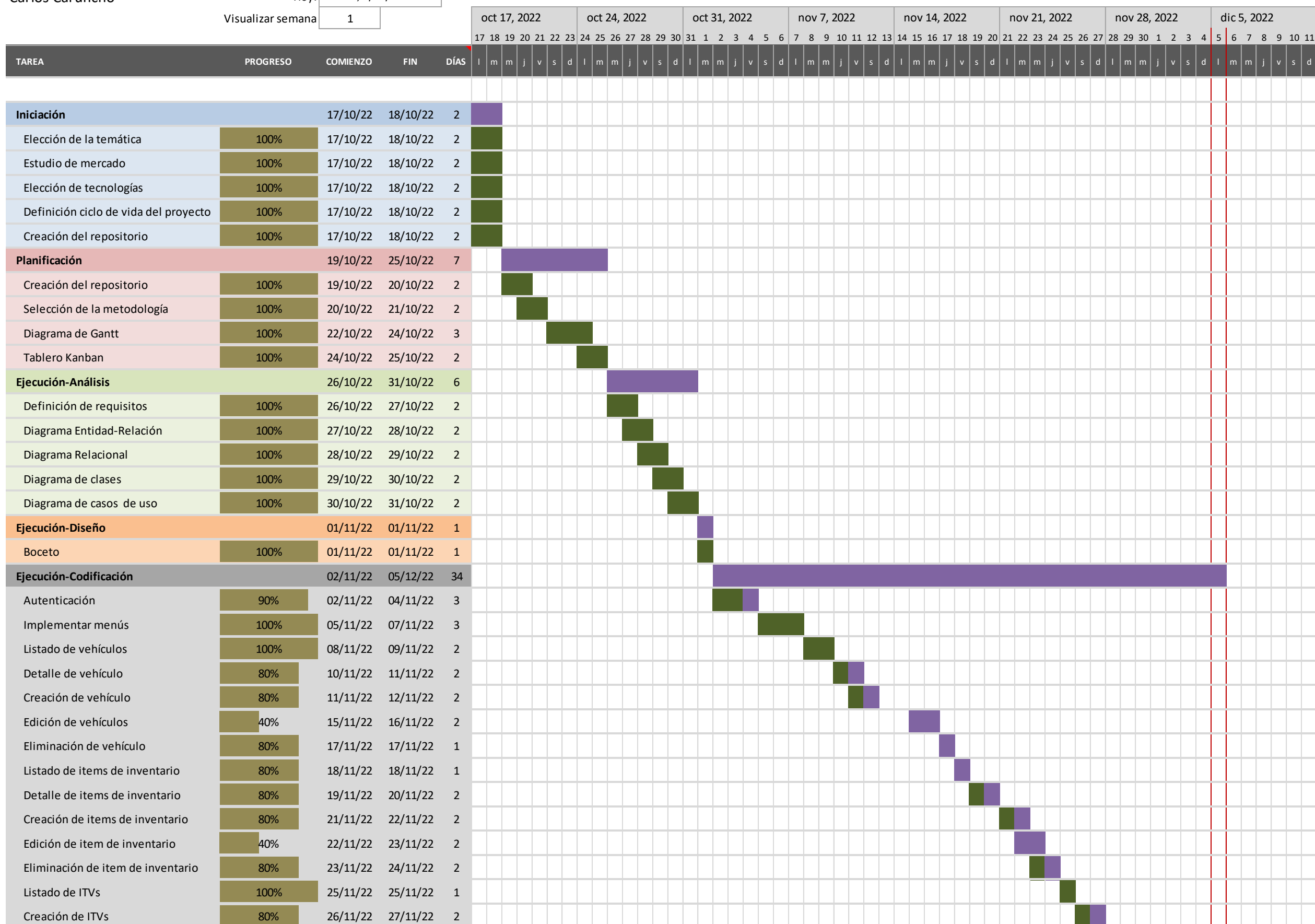
[illegible]



## DIAGRAMA REAL

---

1



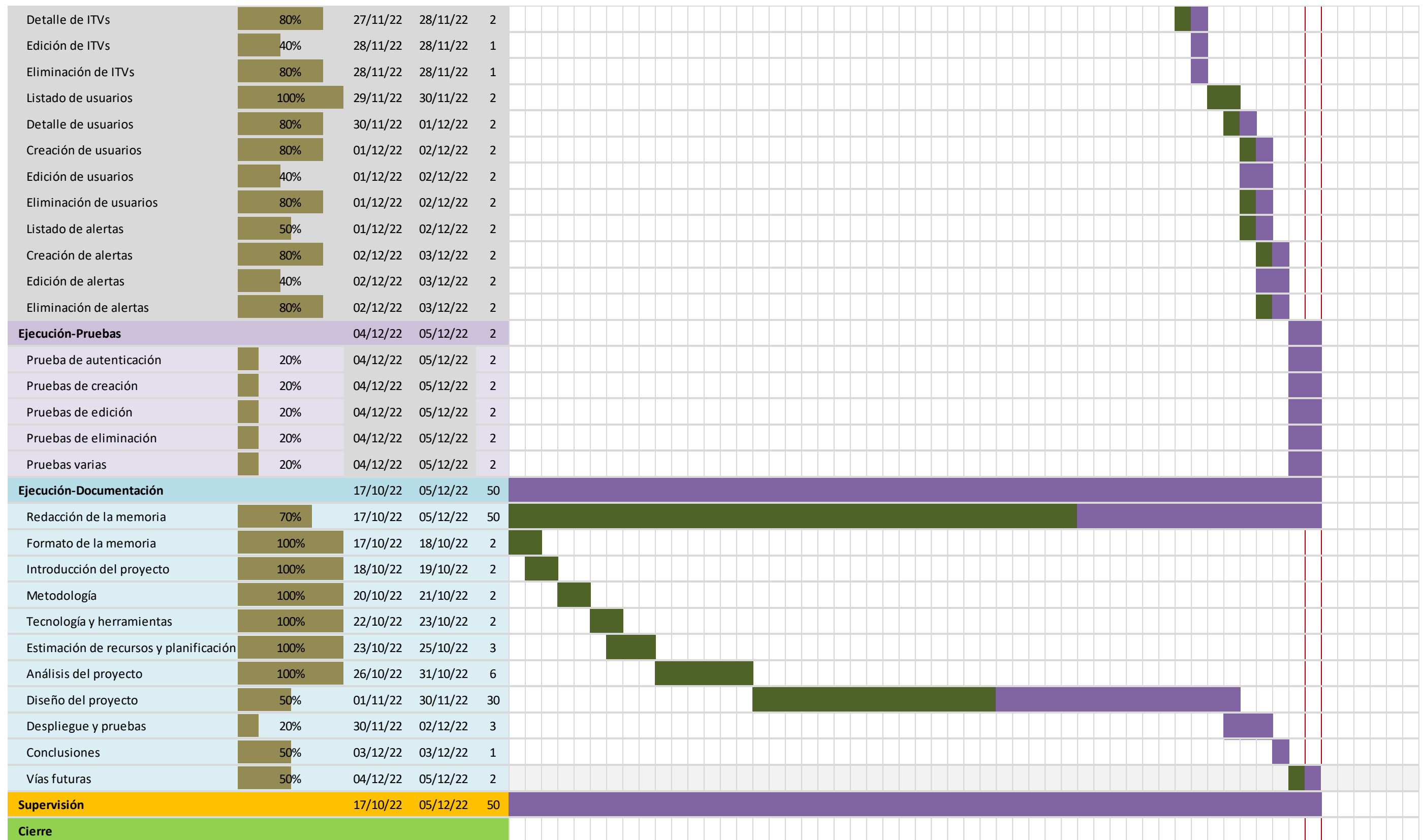


DIAGRAMA AMPLIACIÓN

VEHICLEGEST

ILERNA ONLINE

Carlos Caruncho

ienzo del proyecto:

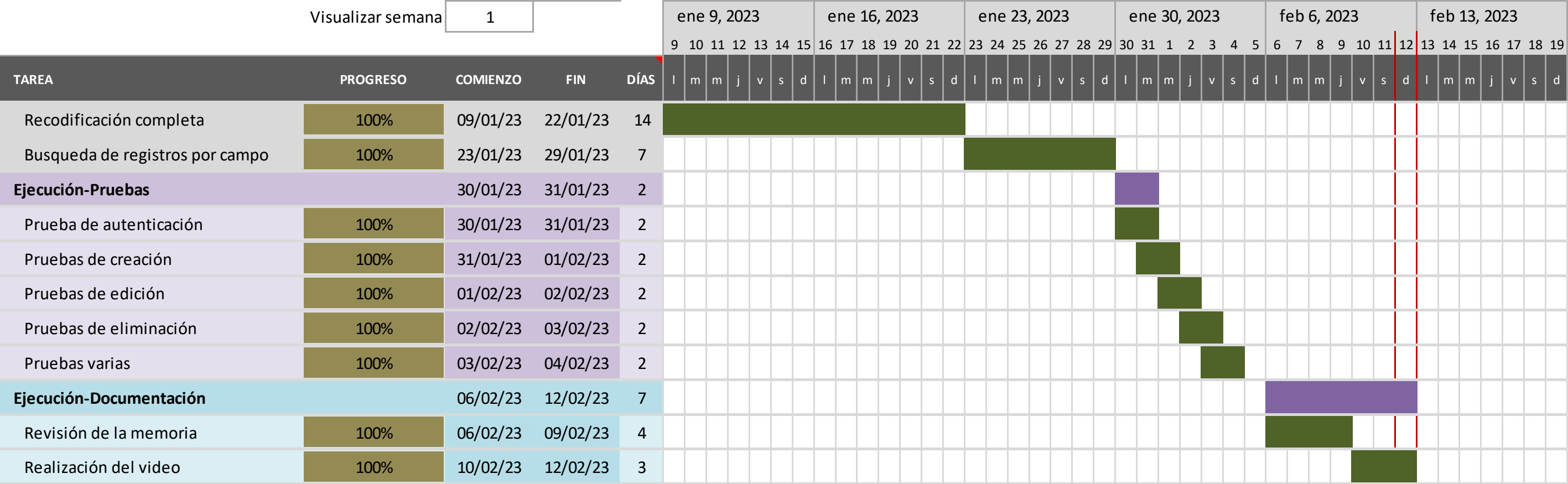
lu, 9/1/2023

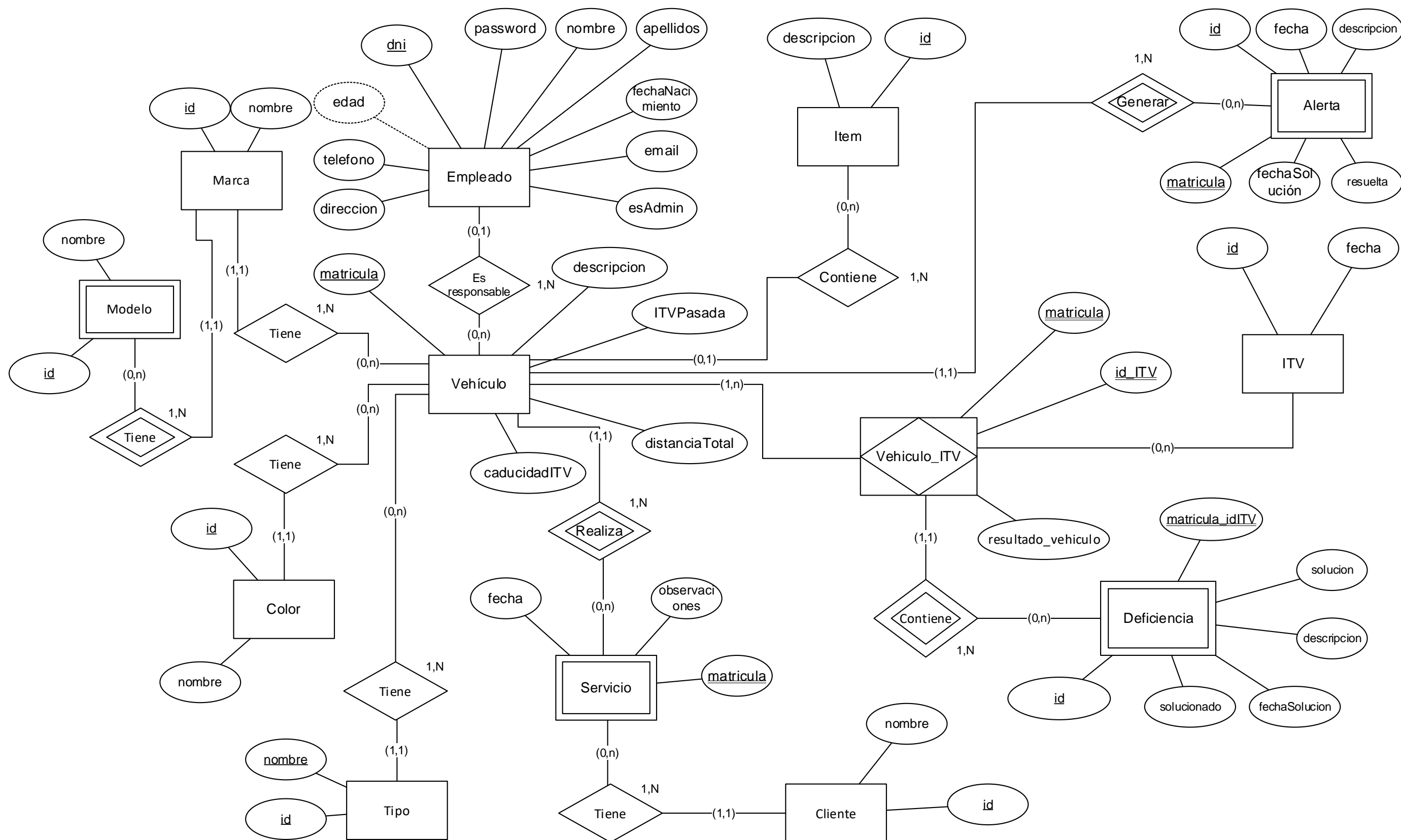
Hoy:

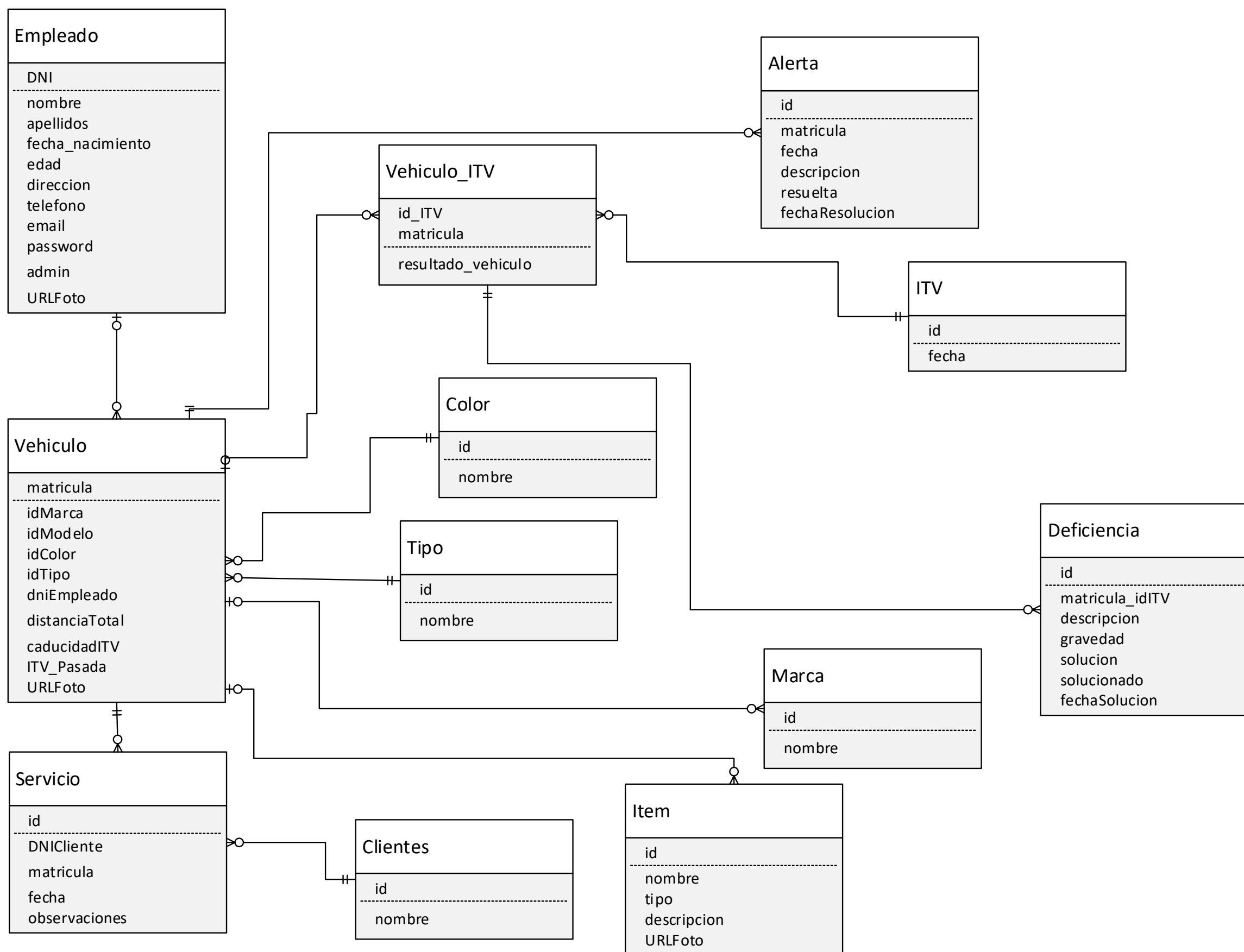
do, 12/2/2023

Visualizar semana

1









```

LoginActivity
  (x) 🔒 activityLoginBinding: ActivityLoginBinding
  (x) 🔒 firebaseAuth: FirebaseAuth
  🔒 onCreate(Bundle?): Unit
  🔒 userAuthentication(String, String): Unit
  🔒 navigateMain(FirebaseUser?): Unit
  🔒 navigateRegister(): Unit
  
```

```

MainActivity
  (x) 🔒 activityMainBinding: ActivityMainBinding
  (x) 🔒 firebaseAuthReference: FirebaseAuth
  (x) 🔒 firestoreDatabaseReference: FirebaseFirestore
  (x) 🔒 badgeAlert: BadgeDrawable
  🔒 topToolBar: MaterialToolBar
  🔒 navBarBot: BottomNavigationView
  (x) 🔒 adminCheck: Boolean
  (x) 🔒 userReference: DocumentReference
  🔒 onCreate(Bundle?): Unit
  🔒 onStart(): Unit
  🔒 setTopBarSettings(): Unit
  🔒 setNavBarListeners(): Unit
  🔒 initializeAlertsBadgeCounter(): Unit
  🔒 getUserData() -> Unit: Unit
  🔒 checksAdmin(): Unit
  🔒 onCreateOptionsMenu(Menu?): Boolean
  🔒 onOptionsItemSelected(MenuItem): Boolean
  🔒 checksLoggedInUser(): Unit
  
```

```

RegisterActivity
  (x) 🔒 activityRegisterBinding: ActivityRegisterBinding
  (x) 🔒 firebaseAuth: FirebaseAuth
  (x) 🔒 dbFirestoreReference: CollectionReference
  🔒 onCreate(Bundle?): Unit
  🔒 userRegistration(String, String): Unit
  🔒 addUserToDataBase(): Unit
  🔒 newEmployee(): Employee
  🔒 navigateMain(FirebaseUser?): Unit
  
```

```

Controller
  🔒 getBitmapFromUrlAsync(String): Deferred<Bitmap?>
  🔒 companion object of Controller
    🔒 dateToStringFormat(Date?): String
    🔒 fragmentReplacer(Fragment, FragmentManager): Unit
    🔒 setDefaultImage(ImageView): Unit
    🔒 stringToDateFormat(String): Date
    🔒 showShortToast(String): Unit
    🔒 showLongToast(String): Unit
    🔒 isInspectionExpired(Date, Date): Boolean
    🔒 checkInspectionExpiration(String): Boolean
  
```

```

DatePickerFragment
  🔒 constructor DatePickerFragment((day: Int, month: Int, year: Int) -> Unit)
  🔒 listener: (day: Int, month: Int, year: Int) -> Unit
  🔒 onCreateDialog(Bundle?): Dialog
  🔒 onDateSet(DatePicker, Int, Int, Int): Unit
  
```

```

Vehiclegest
  🔒 onCreate(): Unit
  🔒 companion object of Vehiclegest
    (x) 🔒 instance: Vehiclegest
  
```

```

Constants
  🔒 passwordPattern: String
  
```



```

ITV
  constructor ITV(Date? = ..., String? = ...)
  (x) date: Date?
  (x) remarks: String?
  constructor ITV(Parcel)
  describeContents(): Int
  writeToParcel(Parcel, Int): Unit
  companion object of ITVCREATOR
    createFromParcel(Parcel): ITV
    newArray(Int): Array<ITV?>

```

```

Item
  constructor Item(String? = ..., String? = ..., String? = ..., String? = ...)
  (x) plateNumber: String?
  (x) name: String?
  (x) description: String?
  (x) photoURL: String?
  constructor Item(Parcel)
  describeContents(): Int
  writeToParcel(Parcel, Int): Unit
  companion object of ItemCREATOR
    createFromParcel(Parcel): Item
    newArray(Int): Array<Item?>

```

```

Vehicle
  constructor Vehicle(String? = ..., String? = ..., String? = ...,
  (x) plateNumber: String?
  (x) type: String?
  (x) brand: String?
  (x) model: String?
  (x) expiryDateITV: Date?
  (x) totalDistance: Int?
  (x) itvPassed: Boolean?
  (x) description: String?
  (x) photoURL: String?
  constructor Vehicle(Parcel)
  describeContents(): Int
  writeToParcel(Parcel, Int): Unit
  companion object of VehicleCREATOR
    createFromParcel(Parcel): Vehicle
    newArray(Int): Array<Vehicle?>

```

```

Service
  constructor Service(String? = ..., Date? = ..., String? = ..., String? = ...)
  (x) plateNumber: String?
  (x) date: Date?
  (x) remarks: String?
  (x) costumer: String?
  constructor Service(Parcel)
  describeContents(): Int
  writeToParcel(Parcel, Int): Unit
  companion object of ServiceCREATOR
    createFromParcel(Parcel): Service
    newArray(Int): Array<Service?>

```

```

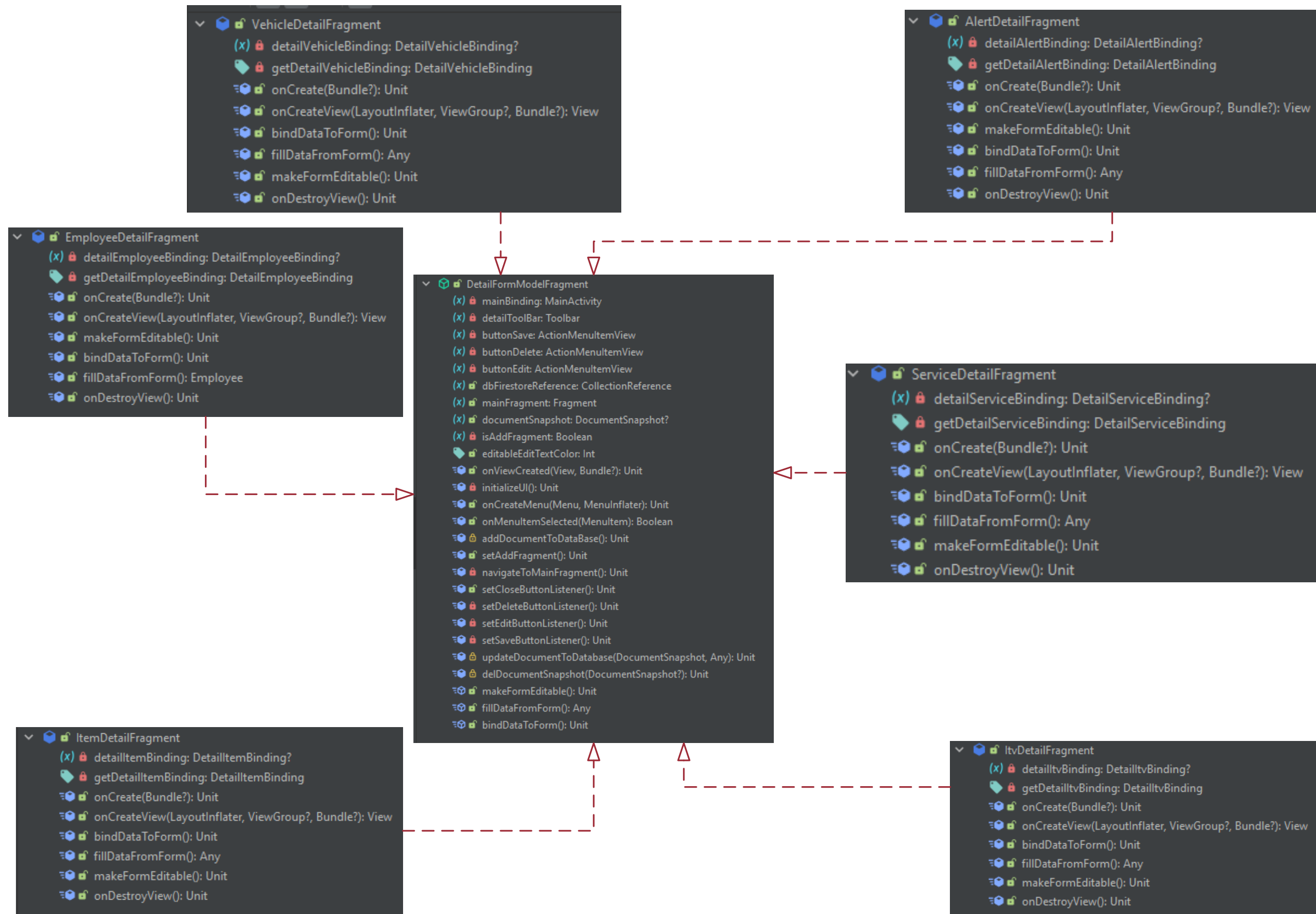
Alert
  constructor Alert(String? = ..., Date? = ..., String? = ..., Boolean? = ..., Date? = ..., String? = ...)
  (x) plateNumber: String?
  (x) date: Date?
  (x) description: String?
  (x) solved: Boolean?
  (x) solveddate: Date?
  (x) solution: String?
  constructor Alert(Parcel)
  describeContents(): Int
  writeToParcel(Parcel, Int): Unit
  companion object of AlertCREATOR
    createFromParcel(Parcel): Alert
    newArray(Int): Array<Alert?>

```

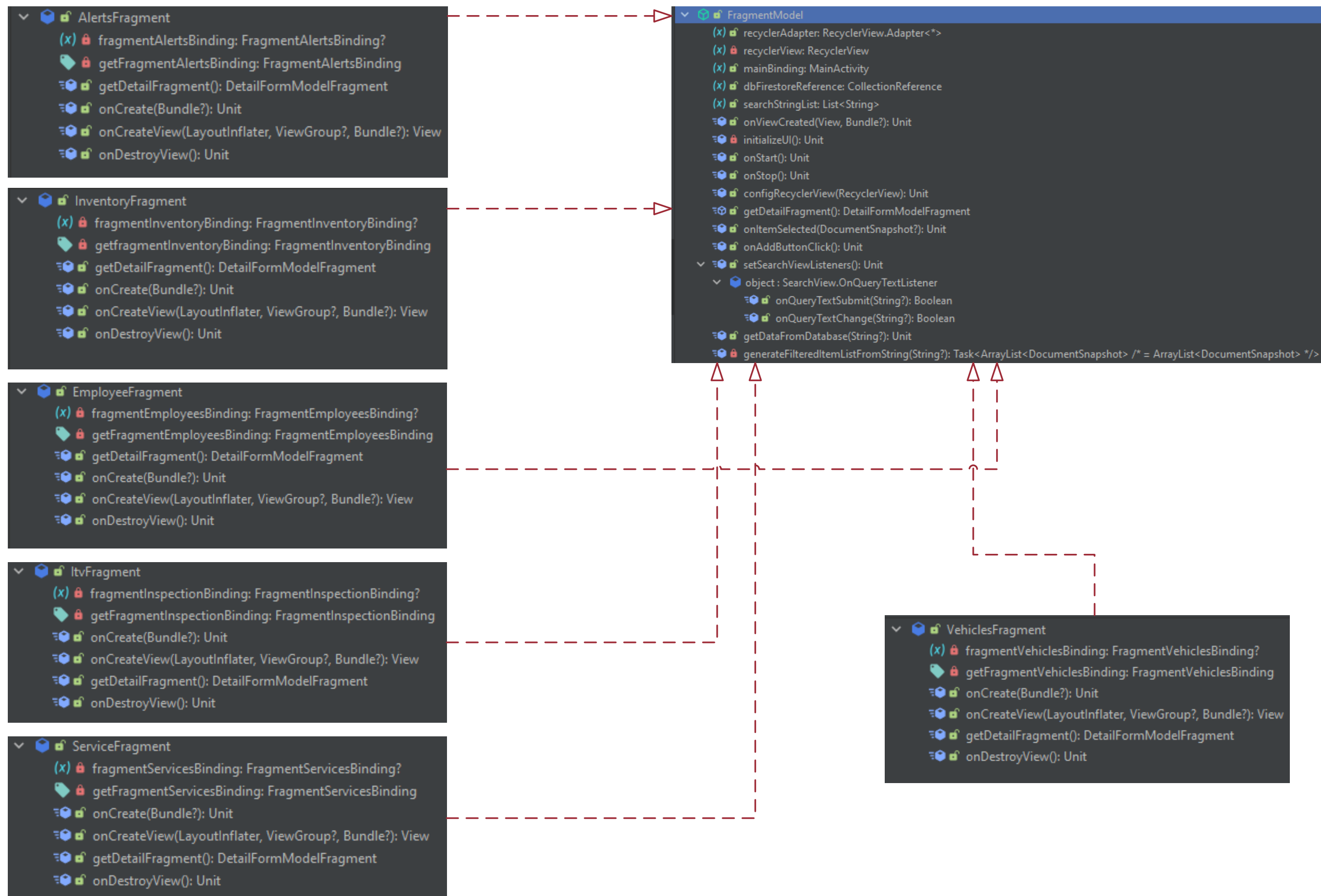
```

Employee
  constructor Employee(String? = ..., String? = ..., String? = ..., S
  (x) dni: String?
  (x) name: String?
  (x) surname: String?
  (x) address: String?
  (x) email: String?
  (x) phone: String?
  (x) birthdate: Date?
  (x) photoURL: String?
  (x) admin: Boolean?
  constructor Employee(Parcel)
  describeContents(): Int
  writeToParcel(Parcel, Int): Unit
  companion object of EmployeeCREATOR
    createFromParcel(Parcel): Employee
    newArray(Int): Array<Employee?>

```

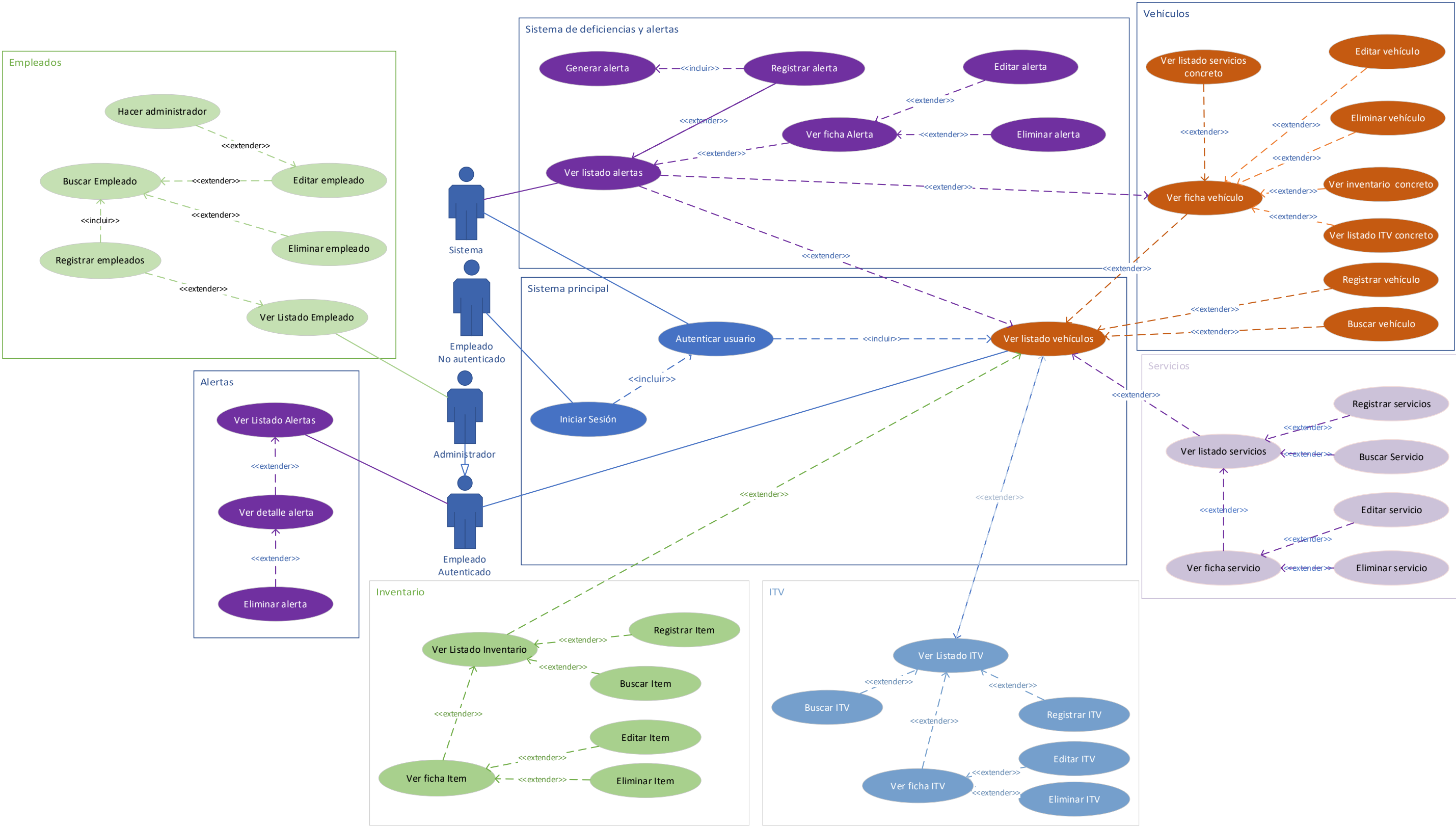








ANEXO VI – Diagrama de Casos de Uso



## **ANEXO VI – Diagrama de Casos de Uso**

## **ANEXO VII – Especificaciones de Casos de Uso**

<b>Identificador</b>	<b>CU_08</b>	
<b>Nombre</b>	Ver detalle alerta	
<b>Descripción</b>	Al presionar sobre una alerta del listado de alertas se abre una ventana emergente con su ficha detallada. En la alerta se detalla si es por fechas próximas o vencidas de ITV, o bien de deficiencias no corregidas.	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	Empleado autenticado. Listado de alertas abierto.	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El <i>empleado</i> presiona el botón “Borrar alerta”. Se ejecuta CU_05.
	2	El <i>empleado</i> puede cerrar la ventana. Vuelve a CU_03.
<b>Postcondición</b>		
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
<b>Comentarios</b>		

<b>Identificador</b>	<b>CU_09</b>	
<b>Nombre</b>	Eliminar alerta	
<b>Descripción</b>	Borra la alerta permanentemente de la base de datos.	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	Empleado autenticado. Detalle de alertas abierto.	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El <i>empleado</i> ha borrado del sistema la alerta de la base de datos.
<b>Postcondición</b>	La alerta desaparece del listado de alertas	
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	1	Si la deficiencia no está corregida el sistema no permitirá borrar la alerta.
<b>Comentarios</b>		

<b>Identificador</b>	<b>CU_010</b>
----------------------	---------------

## **ANEXO VII – Especificaciones de Casos de Uso**

<b>Nombre</b>	Buscar vehículo	
<b>Descripción</b>	Se abre una ventana modal de búsqueda con una línea de texto, y filtros de búsqueda por matrícula, marca o modelo.	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	Empleado autenticado. Listado de vehículos abierto	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El <i>empleado</i> introduce el texto en el campo de texto. El listado de vehículos se actualiza automáticamente según los filtros y/o el texto. Solo aparecen los vehículos que coinciden con los criterios. Volvemos a CU_06
	2	El <i>empleado</i> cierra el diálogo mediante una flecha de la parte arriba izquierda. Volvemos a CU_06
<b>Postcondición</b>	El listado de vehículos se actualiza según criterios de búsqueda	
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	1	Si no aparecen registros coincidentes aparecerá un mensaje de error de búsqueda.

<b>Identificador</b>	<b>CU_011</b>	
<b>Nombre</b>	Ver detalle de vehículo	
<b>Descripción</b>	Se abre una ficha con toda la información detallada del vehículo, incluida su foto.	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	Empleado autenticado. Que existan registros de vehículos en la base de datos. Listado de vehículos abierto	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El empleado presiona el botón "Editar". Se ejecuta CU_09.
	2	El empleado presiona el botón "Eliminar". Se ejecuta CU_10
	3	El empleado presiona el botón "Inventario". Se ejecuta CU_11.
	4	El empleado presiona el botón "Servicios". Se ejecuta CU_12.
	5	El empleado presiona el botón "ITVs". Se ejecuta CU_13.
	6	El empleado puede cerrar la ventana. Vuelve a CU_06



## **ANEXO VII – Especificaciones de Casos de Uso**

<b>Postcondición</b>		
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
<b>Comentarios</b>		

<b>Identificador</b>	<b>CU_012</b>	
<b>Nombre</b>	Editar Vehículo	
<b>Descripción</b>	Hace editable la ficha del vehículo para cambiarlos en la base de datos.	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	Empleado autenticado. Que el registro de vehículo exista en la base de datos. Detalle de vehículo abierto.	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El <i>sistema</i> actualiza el registro en la base de datos.
	2	El <i>empleado</i> puede cierra la ventana. Vuelve a CU_08
<b>Postcondición</b>	El registro se actualiza en la base de datos y en el listado de vehículos.	
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	1	Si los datos son erróneos se dará un mensaje de error.
<b>Comentarios</b>		

<b>Identificador</b>	<b>CU_013</b>	
<b>Nombre</b>	Eliminar vehículo	
<b>Descripción</b>	Borra el registro de vehículo y sus datos de la base de datos.	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	Empleado autenticado. Que el registro de vehículo exista en la base de datos. Detalle de vehículo abierto.	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El <i>empleado</i> le da a aceptar en un modal. El registro es eliminado de la base de datos. Vuelve a CU_06
	2	El <i>empleado</i> le da a cancelar en un modal. El registro no varía. Vuelve a CU_08

## **ANEXO VII – Especificaciones de Casos de Uso**

<b>Postcondición</b>	El registro desaparece de la base de datos.	
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
<b>Comentarios</b>		

<b>Identificador</b>	<b>CU_014</b>	
<b>Nombre</b>	Ver listado de inventario concreto	
<b>Descripción</b>	Se ve un listado de inventario del vehículo filtrado	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	Empleado autenticado. Que haya registros de inventario de vehículo en la base de datos. Vehículo abierto.	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El <i>empleado</i> presiona sobre uno de los items. Ejecuta CU_016
	2	El <i>empleado</i> puede cierra la ventana. Vuelve a CU_08
<b>Postcondición</b>	Solo se ve el listado de Inventario de ese vehículo en concreto	
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	1	Si el vehículo no tiene asignado nada en el inventario aparecerá en un mensaje de aviso.
<b>Comentarios</b>		

<b>Identificador</b>	<b>CU_015</b>	
<b>Nombre</b>	Ver listado de servicios concreto	
<b>Descripción</b>	Se ve un listado de servicios de cada vehículo filtrado	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	Empleado autenticado. Que exista el registro de vehículo en la base de datos. Vehículo abierto. <sup>3</sup>	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El <i>empleado</i> presiona sobre una de servicios. Se ejecuta CU_022.
	2	El <i>empleado</i> puede cierra la ventana. Vuelve a CU_08.
<b>Postcondición</b>	Solo se ve el listado de servicios de ese vehículo en concreto	
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>

## **ANEXO VII – Especificaciones de Casos de Uso**

	1	Si el vehículo no tiene asignado servicios aparecerá en un mensaje de aviso.
<b>Comentarios</b>		

<b>Identificador</b>	<b>CU_016</b>	
<b>Nombre</b>	Ver listado de ITV concreto	
<b>Descripción</b>	Se ve un listado de ITV de cada vehículo filtrado	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	Empleado autenticado. Que el registro de vehículo exista en la base de datos. Vehículo abierto.	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Secuencia normal</b>
	1	El <i>empleado</i> presiona sobre una de las ITV. Se ejecuta CU_022
	2	El <i>empleado</i> puede cierra la ventana. Vuelve a CU_08
<b>Postcondición</b>	Solo se ve el listado de ITV de ese vehículo en concreto	
<b>Excepciones</b>	<b>Paso</b>	<b>Excepciones</b>
	1	Si el vehículo no tiene ITVs aparecerá en un mensaje de aviso.
<b>Comentarios</b>		

<b>Identificador</b>	<b>CU_017</b>	
<b>Nombre</b>	Ver listado de deficiencias concreto	
<b>Descripción</b>	Se ve un listado de deficiencias de cada vehículo filtrado	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	Empleado autenticado. Que el registro de vehículo exista en la base de datos. Vehículo abierto.	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Secuencia normal</b>
	1	El empleado presiona sobre una de las deficiencias. Se ejecuta CU_
	2	El empleado puede cierra la ventana. Vuelve a CU_
<b>Postcondición</b>	Solo se ve el listado de deficiencias de ese vehículo en concreto	

## **ANEXO VII – Especificaciones de Casos de Uso**

Excepciones	Paso	Excepciones
	1	Si el vehículo no tiene deficiencias aparecerá en un mensaje de aviso.
Comentarios		

Identificador	<b>CU_018</b>	
Nombre	Buscar item	
Descripción	Se abre una ventana modal de búsqueda con una línea de texto, y filtros de búsqueda por fecha	
Actores	Empleado, administrador.	
Precondición	<i>Empleado</i> autenticado. Listado de inventario general abierto.	
Secuencia normal	Paso	Acción
	1	El <i>empleado</i> introduce el texto en el campo de texto. El listado de registros se actualiza automáticamente según los filtros y/o el texto. Solo aparecen registros que coinciden con los criterios. Volvemos a CU_015.
	2	El <i>empleado</i> cierra el diálogo mediante una flecha de la parte arriba izquierda. Volvemos a CU_15.
Postcondición	El listado de registros se actualiza según criterios de búsqueda	
Excepciones	Paso	Acción
	1	Si no aparecen registros coincidentes aparecerá un mensaje de error de búsqueda.
Comentarios		

Identificador	<b>CU_019</b>	
Nombre	Editar item	
Descripción	Hace editable la ficha del item de inventario para cambiarlos en la base de datos.	
Actores	Empleado, administrador.	
Precondición	<i>Empleado</i> autenticado. Que el registro de item exista en la base de datos. Detalle de item de inventario abierto.	
	Paso	Acción

## **ANEXO VII – Especificaciones de Casos de Uso**

Secuencia normal	1	El <i>empleado</i> le da a aceptar en un modal. El registro es eliminado de la base de datos. Vuelve a CU_016
	2	El <i>empleado</i> le da a cancelar en un modal. El registro no varía. El <i>empleado</i> puede cerrar la ventana. Vuelve a CU_16.
Postcondición	El registro se actualiza en la base de datos y en el listado de inventario.	
Excepciones	Paso	Acción
	1	Si los datos son erróneos se dará un mensaje de error.
Comentarios		

Identificador	<b>CU_020</b>	
Nombre	Eliminar item	
Descripción	Borra el registro del inventario y sus datos de la base de datos.	
Actores	Empleado, administrador.	
Precondición	<i>Empleado</i> autenticado. Que el registro de item exista en la base de datos. Detalle de item de inventario abierto.	
Secuencia normal	Paso	Acción
	1	El <i>empleado</i> le da a aceptar en un modal. El registro es eliminado de la base de datos. Vuelve a CU_13
	2	El <i>empleado</i> le da a cancelar en un modal. El registro no varía. El <i>empleado</i> puede cerrar la ventana. Vuelve a CU_15
Postcondición	El registro desaparece de la base de datos.	
Excepciones	Paso	Acción
Comentarios		

Identificador	<b>CU_021</b>	
Nombre	Ver detalle de item	
Descripción	Se abre una ficha con toda la información detallada del vehículo, incluida su foto.	
Actores	Empleado, administrador.	

## **ANEXO VII – Especificaciones de Casos de Uso**

<b>Precondición</b>	<i>Empleado</i> autenticado. Que existan registros de inventario en la base de datos. Listado de inventario abierto	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El <i>empleado</i> presiona el botón “Editar”. Se ejecuta CU_18.
	2	El <i>empleado</i> presiona el botón “Eliminar”. Se ejecuta CU_19.
	3	El <i>empleado</i> presiona el icono de volver. Volvemos a CU_15.
<b>Postcondición</b>	El listado de registros se actualiza según criterios de búsqueda	
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	1	Si no aparecen registros coincidentes aparecerá un mensaje de error de búsqueda.
<b>Comentarios</b>		

<b>Identificador</b>	<b>CU_022</b>	
<b>Nombre</b>	Buscar ITV	
<b>Descripción</b>	Se abre una ventana modal de búsqueda con una línea de texto, y filtros de búsqueda por fecha	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	<i>Empleado</i> autenticado. Listado de ITVs abierto.	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El <i>empleado</i> introduce el texto en el campo de texto. El listado de ITVs se actualiza automáticamente según los filtros y/o el texto. Solo aparecen las ITV que coinciden con los criterios. Volvemos a CU_020
	2	El <i>empleado</i> cierra el diálogo mediante una flecha de la parte arriba izquierda. Volvemos a CU_020
<b>Postcondición</b>	El listado de ITVs se actualiza según criterios de búsqueda	
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	1	Si no aparecen registros coincidentes aparecerá un mensaje de error de búsqueda.
<b>Comentarios</b>		

## **ANEXO VII – Especificaciones de Casos de Uso**

<b>Identificador</b>	<b>CU_023</b>	
<b>Nombre</b>	Ver detalle de ITV	
<b>Descripción</b>	Se abre una ficha con toda la información detallada de la ITV.	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	<i>Empleado</i> autenticado. Que exista el registro de ITV en la base de datos. Listado de ITVs abierto	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El <i>empleado</i> presiona el botón “Editar”. Se ejecuta CU_23.
	2	El <i>empleado</i> presiona el botón “Eliminar”. Se ejecuta CU_24
	3	El <i>empleado</i> presiona el icono de volver. Volvemos a CU_20.
<b>Postcondición</b>		
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
<b>Comentarios</b>		

<b>Identificador</b>	<b>CU_024</b>	
<b>Nombre</b>	Editar ITV	
<b>Descripción</b>	Hace editable la ficha del vehículo para cambiarlos en la base de datos.	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	<i>Empleado</i> autenticado. Que el registro de vehículo exista en la base de datos. Detalle de ITV abierto.	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El sistema actualiza el registro en la base de datos.
	2	El <i>empleado</i> cierra el diálogo mediante una flecha de la parte arriba izquierda. Volvemos a CU_022.
<b>Postcondición</b>	El registro se actualiza en la base de datos y en el listado de vehículos.	
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
	1	Si los datos son erróneos se dará un mensaje de error.
<b>Comentarios</b>		

## **ANEXO VII – Especificaciones de Casos de Uso**

<b>Identificador</b>	<b>CU_025</b>	
<b>Nombre</b>	Eliminar ITV	
<b>Descripción</b>	Borra el registro de vehículo y sus datos de la base de datos.	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	<i>Empleado</i> autenticado. Que el registro de vehículo exista en la base de datos.	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El <i>empleado</i> le da a aceptar en un modal. El registro es eliminado de la base de datos. Vuelve a CU_14
	2	El <i>empleado</i> le da a cancelar en un modal. El registro no varía. El <i>empleado</i> puede cerrar la ventana. Vuelve a CU_16
<b>Postcondición</b>	El registro desaparece de la base de datos.	
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
<b>Comentarios</b>		

<b>Identificador</b>	<b>CU_026</b>	
<b>Nombre</b>	Buscar servicio	
<b>Descripción</b>	Se abre una ventana modal de búsqueda con una línea de texto, y filtros de búsqueda por fecha	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	<i>Empleado</i> autenticado. Listado de servicios abierto.	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El <i>empleado</i> introduce el texto en el campo de texto. El listado de servicios se actualiza automáticamente según los filtros y/o el texto. Solo aparecen los servicios que coinciden con los criterios. Volvemos a CU_025
	2	El <i>empleado</i> cierra el diálogo mediante una flecha de la parte arriba izquierda. Volvemos a CU_25
<b>Postcondición</b>	El listado de servicios se actualiza según criterios de búsqueda	
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>



## **ANEXO VII – Especificaciones de Casos de Uso**

	1	Si no aparecen registros coincidentes aparecerá un mensaje de error de búsqueda.
<b>Comentarios</b>		

<b>Identificador</b>	<b>CU_027</b>	
<b>Nombre</b>	Ver detalle de servicio	
<b>Descripción</b>	Se abre una ficha con toda la información detallada de la ITV.	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	<i>Empleado</i> autenticado. Que exista el registro de ITV en la base de datos. Listado de ITVs abierto	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El <i>empleado</i> presiona el botón “Editar”. Se ejecuta CU_28.
	2	El <i>empleado</i> presiona el botón “Eliminar”. Se ejecuta CU_29.
	3	El <i>empleado</i> presiona el icono de volver. Volvemos a CU_25.
<b>Postcondición</b>		
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>
<b>Comentarios</b>		

<b>Identificador</b>	<b>CU_028</b>	
<b>Nombre</b>	Editar servicio	
<b>Descripción</b>	Hace editable la ficha del servicio para cambiarlo en la base de datos.	
<b>Actores</b>	Empleado, administrador.	
<b>Precondición</b>	<i>Empleado</i> autenticado. Que el registro de servicio exista en la base de datos. Detalle de servicio abierto.	
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>
	1	El sistema actualiza el registro en la base de datos.
	3	El <i>empleado</i> presiona el icono de volver. Volvemos a CU_27.
<b>Postcondición</b>	El registro se actualiza en la base de datos y en el listado de servicios.	

## **ANEXO VII – Especificaciones de Casos de Uso**

Excepciones	Paso	Acción
	1	Si los datos son erróneos se dará un mensaje de error.
Comentarios		

Identificador	<b>CU_029</b>	
Nombre	Eliminar servicio	
Descripción	Borra el registro de servicio y sus datos de la base de datos.	
Actores	Empleado, administrador.	
Precondición	<i>Empleado</i> autenticado. Que el registro de vehículo exista en la base de datos.	
Secuencia normal	Paso	Acción
	1	El <i>empleado</i> le da a aceptar en un modal. El registro es eliminado de la base de datos. Vuelve a CU_25
	2	El <i>empleado</i> le da a cancelar en un modal. El registro no varía. El <i>empleado</i> puede cerrar la ventana. Vuelve a CU_27
Postcondición	El registro desaparece de la base de datos.	
Excepciones	Paso	Acción
Comentarios		

Identificador	<b>CU_030</b>	
Nombre	Buscar <i>empleado</i>	
Descripción	Se abre una ventana modal de búsqueda con una línea de texto, y filtros de búsqueda por fecha	
Actores	Administrador o <i>empleado</i>	
Precondición	Administrador o <i>Empleado</i> autenticado. Listado de <i>empleados</i> abierto.	
Secuencia normal	Paso	Acción
	1	El <i>empleado</i> introduce el texto en el campo de texto. El listado de <i>empleados</i> se actualiza automáticamente según los filtros y/o el texto. Solo

## ANEXO VII – Especificaciones de Casos de Uso

		aparecen los <i>empleados</i> que coinciden con los criterios. Volvemos a CU_025
	2	El <i>empleado</i> cierra el diálogo mediante una flecha de la parte arriba izquierda. Volvemos a CU_25
Postcondición	El listado de servicios se actualiza según criterios de búsqueda	
Excepciones	Paso	Acción
	1	Si no aparecen registros coincidentes aparecerá un mensaje de error de búsqueda.
Comentarios		

Identificador	<b>CU_031</b>	
Nombre	Ver detalle de <i>empleado</i>	
Descripción	Se abre una ficha con toda la información detallada del <i>empleado</i> .	
Actores	Administrador o <i>empleado</i>	
Precondición	Administrador o <i>empleado</i> autenticado. Listado de <i>empleados</i> abierto.	
Secuencia normal	Paso	Acción
	1	El administrador presiona el botón "Editar". Se ejecuta CU_33.
	2	El administrador presiona el botón "Eliminar". Se ejecuta CU_34.
	3	El administrador marca la casilla "Hacer administrador". Se ejecuta CU_35.
	4	El administrador presiona el icono de volver. Volvemos a CU_30.
Postcondición		
Excepciones	Paso	Acción
	1,2,3	Solo el <i>empleado</i> administrador puede editar o eliminar <i>empleados</i> .
Comentarios		

Identificador	<b>CU_032</b>	
Nombre	Editar <i>empleado</i>	
Descripción	Hace editable la ficha del <i>empleado</i> para cambiarlo en la base de datos.	

## **ANEXO VII – Especificaciones de Casos de Uso**

Actores	Administrador.	
Precondición	Administrador autenticado. Que el registro de <i>empleado</i> exista en la base de datos. Detalle de <i>empleado</i> abierto.	
Secuencia normal	Paso	Acción
	1	El sistema actualiza el registro en la base de datos.
	3	El administrador presiona el icono de volver. Volvemos a CU_32.
Postcondición	El registro se actualiza en la base de datos y en el listado de <i>empleados</i> .	
Excepciones	Paso	Acción
	1	Si los datos son erróneos se dará un mensaje de error.
Comentarios		

Identificador	<b>CU_033</b>	
Nombre	Eliminar <i>empleado</i>	
Descripción	Borra el registro de servicio y sus datos de la base de datos.	
Actores	Administrador.	
Precondición	Administrador autenticado. Que el registro de <i>empleado</i> exista en la base de datos. Detalle de <i>empleado</i> abierto.	
Secuencia normal	Paso	Acción
	1	El administrador le da a aceptar en un modal. El registro es eliminado de la base de datos. Vuelve a CU_30
	2	El administrador le da a cancelar en un modal. El registro no varía. El administrador puede cerrar la ventana. Vuelve a CU_32
Postcondición	El registro desaparece de la base de datos.	
Excepciones	Paso	Acción
Comentarios		

Identificador	<b>CU_034</b>	
Nombre	Hacer administrador	
Descripción	El administrador nombra administrador a otros <i>empleados</i> marcando una casilla de verificación.	

## **ANEXO VII – Especificaciones de Casos de Uso**

Actores	Administrador.	
Precondición	Administrador autenticado. Que el registro de <i>empleado</i> exista en la base de datos. El administrador maestro no puede ser degradado. <i>Empleado</i> abierto.	
Secuencia normal	Paso	Acción
	1	El <i>empleado</i> se convierte en administrador marcando una casilla de la ficha de detalle. Vuelve a CU_32
	2	
Postcondición	El <i>empleado</i> nombrado administrador puede acceder a las opciones avanzadas de edición de <i>empleados</i> .	
Excepciones	Paso	Acción
	1	El administrador maestro tiene la casilla marcada permanentemente y no puede ser desmarcada.
Comentarios		

Identificador	<b>CU_035</b>	
Nombre	Listado de vehículos asignados a cada <i>empleado</i>	
Descripción	En la ficha de <i>empleado</i> se puede ver un listado de vehículos asignados a cada <i>empleado</i> .	
Actores	<i>Empleado</i> o administrador.	
Precondición	<i>Empleado</i> autenticado. Que el registro de <i>empleado</i> exista en la base de datos. <i>Empleado</i> abierto.	
Secuencia normal	Paso	Acción
	1	El <i>empleado</i> presiona sobre uno de los vehículos. Ejecuta CU_016
	2	El <i>empleado</i> puede cierra la ventana. Vuelve a CU_32
Postcondición		
Excepciones	Paso	Acción
	1	
Comentarios		

Identificador	<b>CU_036</b>	
Nombre	Listado de servicios asignados a cada <i>empleado</i>	
Descripción	En la ficha de <i>empleado</i> se puede ver un listado de servicios asignados a cada <i>empleado</i> .	

## **ANEXO VII – Especificaciones de Casos de Uso**

Actores	<i>Empleado</i> o administrador.	
Precondición	<i>Empleado</i> autenticado. Que el registro de servicio exista en la base de datos. <i>Empleado</i> abierto.	
Secuencia normal	Paso	Acción
	1	El <i>empleado</i> presiona sobre uno de los servicios. Ejecuta CU_027
	2	El <i>empleado</i> puede cierra la ventana. Vuelve a CU_32
Postcondición		
Excepciones	Paso	Acción
	1	
Comentarios		

## **ANEXO VIII – Estructura de datos de Firestore Database**

- **Colección:** vehicle
  - **Documento:** 1234ABC (matrícula)
    - **Campos**
      - brand:string
      - expiryDateITV:timestamp
      - model:string
      - type:string
      - totalDistance:number
  
- **Colección:** service
  - **Documento:** Id aleatoria
    - **Campos**
      - plateNumber:string (Clave foránea)
      - date:timestamp
      - costumer:string
      - remarks:string
  
- **Colección:** ITV
  - **Documento:** Id aleatoria
    - **Campos**
      - date:timestamp
      - remarks
      - **Subcolecciones**
        - **Vehicle:**
          - **Documento:** 1234ABC (Vehicle plateNumber)
            - **Campos**
              - brand:string
              - model:string
              - color:string
              - type:string
        - **Deficiency:**
          - **Documento:** Id deficiency

## **ANEXO VII – Especificaciones de Casos de Uso**

- **Campos**
  - description:string
  - fixed:boolean
  - severity:string
  
- **Colección: deficiency**
  - **Documento:** Id aleatoria
    - **Campos**
      - plateNumber:string(Clave foránea)
      - description:string
      - severity:string
      - solution:string
      - fixed:boolean
      - fixedDate:timestamp
      - date:string
  
- **Colección: alert**
  - **Documento:** Id aleatoria
    - **Campos**
      - plateNumber:string(Clave foránea)
      - description:string
      - solution:string
      - solved:boolean
      - solveddate:timestamp
      - date:string
  
- **Colección: inventory**
  - **Documento:** Id aleatoria
    - **Campos**
      - plateNumber:string(Clave foránea)
      - name:string



## **ANEXO VII – Especificaciones de Casos de Uso**

- description:string
  - photoURL
- **Colección:** employees
  - **Documento:** Id aleatoria
    - **Campos**
      - dni: string
      - name:string
      - surname:string
      - admin: boolean
      - birthdate: timestamp
      - address: string
      - email: string
      - phone: string
      - photoURL

VehicleGest

Carlos Fco. Caruncho Serrano

---

