MEMORIA ESCRITA DEL PROYECTO

CFGS Desarrollo de Aplicaciones Multiplataforma

**VehicleGest**

**Autor:** Carlos Francisco Caruncho Serrano

**Tutor:** Mario Gago

**Fecha de entrega:** dd/mm/aaaa

**Convocatoria:** 2º Semestre - 2022

**Documentos del proyecto:**Enlace a la carpeta del Drive

Índice de contenidos

[1. Introducción 3](#_Toc118376213)

[1.1. Motivación 3](#_Toc118376214)

[1.2. Abstract 4](#_Toc118376215)

[1.3. Objetivos propuestos (generales y específicos) 5](#_Toc118376216)

[2. Metodología usada 6](#_Toc118376217)

[2.1. Fases del ciclo de vida del proyecto 8](#_Toc118376218)

[3. Tecnologías y herramientas utilizadas en el proyecto 10](#_Toc118376219)

[4. Estimación de recursos y planificación 13](#_Toc118376220)

[5. Análisis del proyecto 14](#_Toc118376221)

[6. Diseño del proyecto 16](#_Toc118376222)

[7. Despliegue y pruebas 17](#_Toc118376223)

[8. Conclusiones 18](#_Toc118376224)

[9. Vías futuras 19](#_Toc118376225)

[10. Bibliografía 20](#_Toc118376226)

# Introducción

**Vehiclegest** será una aplicación móvil Android, dedicada a gestionar el control y mantenimiento de una flota de vehículos de transporte de carga y personas. 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.

**VehicleGest** le dará a la empresa un plus de productividad al eliminar los procesos basados ​​en papel, pizarras y documentos de Office, y permitirá la creación de informes y avisos móviles, ahorrando tiempo, y 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, mejorará la productividad, y al mejorar la productividad, aumentarán los ingresos.

## Motivación

La motivación principal, surgió debido a mi profesión y al trabajo que realizo en ella. Soy responsable de una flota de vehículos de transporte, tanto de carga como de personas. Normalmente todos estos datos se almacenan en hojas de Excel, Word, o pizarras y no son actualizados en tiempo real, dando lugar a errores de datos o duplicidades.

Además, se ha dado el caso de no estar bien anotadas las fechas de inspección o no recordar que día son, así como pasarse la fecha por estar anotado en una hoja de Excel o pizarra que no nos avisa de ella, dando lugar a fallos en la logística que hacen perder tiempo y recursos.

De estas vicisitudes y teniendo en cuenta la gran cantidad de vehículos de que disponemos, surge la necesidad de tener un control de datos centralizado y en tiempo real, desde cualquier dispositivo móvil.

También es necesario tener un control centralizado de los inventarios de los vehículos. Dichos inventarios están también anotados en hojas de Excel, dificultando su control y auditorias.

Este proyecto puede ser una solución para estos problemas tanto en mi ámbito cómo en ámbito civil, para flotas de autobuses, camiones, vehículos de transporte de personas, etc. Los empresarios y trabajadores tendrán todos los datos de la empresa en tiempo real y con datos centralizados en sus terminales móviles Android.

## Abstract

VehicleGest is a complete Android solution to improve the control and maintenance of a logistics company's vehicle fleet, with a centralized database of vehicles, technicians, materials and personel, taking all this data wherever the company's personnel are.

In the past, where computers and telephones did not exist, all information was stored on paper. In recent decades, technology has evolved very quickly, as well as the way of storing data in businesses, companies and corporations. First of all, the information was stored in digital documents, but it is not productive for many reasons: low availability, inefficient teamwork, low consistency, low security, low reliability, among others.

VehicleGest will bring some features to improve it. Firstly, it presents a warning screen, bringing user technical incidents and ITV (Technical Inspection of Vehicles) dates, to avoid errors and improve scheduling control. On the one hand, it notifies when the ITV dates are close or expired. On the other hand, it warns about vehicle technical deficiencies or whatever that affects to each vehicle. VehicleGest have a main menu on this screen bottom that let users navigate to the other sections.

The second section is a list of services provided by our company. It has a search bar with filters to search a service. Each service is associated by a vehicle or vehicle group, and personel.

The next section is a vehicle list, that has a vehicle search engine and filters. Also users can navigate to each vehicle datasheet, and know what ITVs and personel has assigned to.

Finally, it has a personel list section that has a search engine too. It be able to search single personel datasheet filtering by some criterias, and assign a person in charge of each vehicle.

In conclusion, VehicleGest will improve the productivity of the company deleting the paper-based processes ,saving time and ensuring data is correct and easily accessible. It helps work more efficiently and reduces operating costs.

## Objetivos propuestos (generales y específicos)

Los **objetivos generales** son:

* Digitalizar y centralizar en una base de datos todos los datos de los vehículos, ITV, material, personal y servicios de la empresa.
* Disponer de toda la información en cualquier lugar en los **terminales móviles Android** de los trabajadores y los responsables de la empresa.
* Poder actualizar y controlar los datos de mantenimiento, ITV, servicios y personal asociados de los vehículos.

Los **objetivos específicos** son:

* Crear una base datos en tiempo real online, evitándonos invertir en una infraestructura propia para la base de datos. Crearemos con la estructura completa de datos necesarios, y la conectaremos a la aplicación instalada en los smartphones del personal de la empresa.
* Diseñar una interfaz de usuario con una usabilidad lo elevada posible. Debe ser clara y sencilla, con un esquema de colores agradable a la vista. Los formularios y las listas deben ser responsivas y adecuadas al tamaño de las pantallas de los terminales móviles, evitando agrupar demasiados en una misma actividad.
* Los usuarios se autenticarán mediante un sistema de usuario y contraseña conectado a la base de datos, lo que permitirá mejorar la seguridad, privacidad y la integridad de los datos de la empresa.
* Las partes de la aplicación serán accesibles mediante un menú general, con iconos y nombres.
* Como primera pantalla mostrará un panel de avisos y alertas general tras la autenticación correcta.
* Avisos y control en el panel de avisos, por parte de los responsables generales y de los responsables directos de cada vehículo, de su estado de mantenimiento: niveles, estado de ruedas, limpieza, etc.
* Avisos y control en el panel de avisos, de las fechas de vencimiento de las inspecciones oficiales ITV, así como las fechas programadas para estas, y el resultado de cada inspección.
* Disponer de un listado servicios realizados por cada uno de los trabajadores y los vehículos, pudiendo buscar un registro específico mediante un buscador con filtros.
* Disponer de un listado vehículos ha usado cada trabajador, en que fechas, así como saber también que servicios y que conductores ha tenido cada vehículo, pudiendo buscar un registro específico mediante un buscador con filtros.
* Disponer de un listado de inventario de herramientas general e individual en cada vehículo, pudiendo detectar que herramientas faltan o están deterioradas, pudiendo buscar un registro específico mediante un buscador con filtros.
* 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.

# Metodología usada

Cómo **metodología de flujo de trabajo** se ha utilizado la metodología ágil **Kanban,** combinada con la **incremental tradicional**. Es una metodología ágil que se utiliza para definir, administrar y mejorar el trabajo de una manera visual. Aumenta la calidad, eficiencia y la productividad en el desarrollo.

Es un proceso **iterativo e incremental**. Con Kanban se puede subdividir el proyecto en pequeñas tareas individuales. Las tareas se representan en una pizarra con fichas o tarjetas, y por fases. (APD, 2021)

El más básico puede presentar columnas como “Trabajo pendiente, En progreso y Terminado” como vemos en la **Figura 1**, pero podemos poner las columnas y filas que sean necesarias. Las tareas avanzan a través de las diferentes fases hasta que estén finalizadas.

**Figura 1**

*Ejemplo de tablero Kanban básico*



Nota: Ilustración del modelo Kanban básico. (Tecnosoluciones, s.f.)

Sus ventajas específicas para este proyecto son:

* Facilidad para arrastrar y soltar las tareas de una columna a otra, visualizando cuales están pendientes y activas, permitiendo visualizar el flujo de trabajo.
* Permite la planificación y seguimiento rápido de las tareas.
* Detallado en el seguimiento, permitiendo añadir detalles con notas, subtareas, fechas de inicio y vencimiento, etc.
* Sin límite de información, pudiendo agregar tantas columnas y tareas como se necesiten

(Martins, 2022)

El lema **“Stop starting, start finishing”** es el lema principal de esta metodología. El trabajo en curso debe ser limitado, para garantizar la calidad. (APD, 2021), por lo que se debe definir cuantas tareas como máximo puede realizarse en cada fase del ciclo de trabajo (ejemplo, como máximo 3 tareas en curso), a ese número de tareas se le llama límite del “work in progress” o WIP. Para poder iniciar otra tarea se deberá haber terminado otra previamente. En este proyecto se tendrán como máximo dos funcionalidades al mismo tiempo al ser solo una persona. (Garzas, 2011)

Los dos conceptos de métrica temporal que se usan en Kanban para medir el tiempo de desarrollo son el **“Lead Time”** y **“Cycle Time”**. El “Cycle Time” mide desde que el trabajo sobre una tarea comienza hasta que termina. El “Lead Time” se mide lo que ven los clientes, lo que esperan. (Garzas, 2011)

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.)

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

* **Pendiente:** Esta será la fase inicial, ya que he descartado poner una fase de “Backlog”. Los “Backlog” se suelen usar en Kanban cuando se van a introducir ideas y funcionalidades nuevas a mitad de proyecto. En este proyecto no se van a introducir ideas nuevas, las funcionalidades ya están predefinidas y no se introducirán nuevas. Estarán ya en la fase de pendiente desde el inicio del proyecto. En la fase de pendiente estarán en tarjetas individuales cada una de las funcionalidades de las que constará la aplicación.
* **En curso:** En la parte superior de esta fase pondremos 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 entrará automáticamente a la subfase de “Codificación”. Cuando se considere que esté terminada la codificación, se moverá a la subfase de “Probando”.
* **Finalizado:** En la fase de “Probando” se le harán las pruebas pertinentes y una vez pasadas se dará por terminada y pasará a la fase de “Finalizado”.

## Fases del ciclo de vida del proyecto

* **Iniciación**

Se escoge la idea principal para la realización del proyecto, es decir mejorar la gestión de los vehículos de una empresa logística. La empresa tiene problemas en la gestión de la flota, todo se apunta en papel y pizarras, lo que llevaba a errores y fallos en ITVs, servicios, etc.

* **Planificación**

Se hace un plan de gestión de tiempo con el calendario y se transcribe a un **diagrama de Gantt**, para dividir las tareas en el tiempo disponible. La información de esta fase se desarrolla en el **apartado 4** de esta memoria.

**Necesidad a cubrir:** mejorará la gestión y la eficiencia, y a su vez la productividad de la empresa. Los empleados con acceso a la aplicación tendrán la información de todos los vehículos de la flota, de sus deficiencias técnicas, ITVs y servicios.

**Viabilidad:** el proyecto es perfectamente viable en tiempo, ya que es un proyecto pequeño, que un programador puede realizar en un mes y medio.

**Costes:** estimamos los costes en función del tiempo utilizado para la realización del proyecto, desde que se aprueba en este caso la propuesta del mismo. Un programador junior cobra de media 1500 euros al mes, por lo que podríamos fijar un presupuesto inventado de 2000 euros.

* **Ejecución**

En esta fase se crean las tareas, y se pasan en el tablero Kanban a “En Curso”, a la par se empiezan a ejecutar las fases de ciclo de vida del software, que se subdividen en pequeñas tareas y se pasan a un diagrama de Gantt y al tablero Kanban. Se han ido realizando las tareas de la memoria y las funcionalidades que se identifican a partir de los requisitos. A medida que se van terminando las tareas o funcionalidades, se va comprobando la calidad de los mismos antes de pasarlas a Finalizado.

Dentro de la fase de ejecución del proyecto empezamos a ejecutar las fases del con el ciclo de vida del software:

* + **Fases del ciclo de vida del software**

Para la vida del software usaremos el modelo en cascada tradicional, en el que antes de pasar a la siguiente fase, tendremos que terminar la anterior. Permite iteraciones para corregir problemas o agregar funcionalidades. Podemos ver las fases del modelo en cascada tradicional en la **Figura 2**, y las iremos definiendo una por una.



**Figura 2:** Ciclos de vida del software. Modelo en cascada. (Ilerna S.L., 2021)

* **Análisis:** En esta fase definimos los objetivos del software a partir de la idea principal, y lo desglosamos en tareas más pequeñas, definiendo los requisitos funcionales y no funcionales.
* **Diseño:** Se decide como implementar el software y su estructura. Pasaremos los requisitos funcionales y no funcionales a un diseño de software. Se diseñan las estructuras de datos, la estructura de los componentes, la interfaz gráfica de la aplicación, y los componentes en detalle.
* **Codificación:** Primero elegimos las herramientas adecuadas para el desarrollo, entre ellas el IDE. Con el IDE y la interfaz ya diseñada, procederemos a escribir el código fuente. Seguimos las convenciones y normas para escribir un código claro y legible. Además, documentamos el código a medida que lo vamos escribiendo, para su posterior mantenimiento de una forma más eficiente.
* **Pruebas:** Mediante las pruebas, detectamos los errores en la codificación, para corregirlos. Se dice que una prueba es un éxito si encuentra algún error.
* **Mantenimiento:** Está fase no existe al ser un proyecto académico**.** En esta fase iríamos corrigiendo los errores que el cliente encuentre, adaptaremos el software a las nuevas necesidades de la empresa, y se le añadirían nuevas funcionalidades si fuese necesario. Esta fase es la última y no se realiza en la ejecución, si que se deja para después del cierre del proyecto.
* **Supervisión**

En la supervisión revisamos si estamos cumpliendo los plazos y los requisitos para la consecución de los objetivos. Si no es así, podemos corregir el rumbo y hacer ajustes en el plan original.

* **Cierre**

Aquí se analiza si se han conseguido los objetivos del proyecto. Se debe hacer un juicio crítico para ver que ha ido bien y que se hubiera podido mejorar, y documentarlo todo. Se suben los archivos a la nube para su acceso y entrega. A partir de aquí entraríamos en mantenimiento si fuese un proyecto para un cliente, corrigiendo los bugs, errores y programando nuevas funcionalidades si fuera necesario, pero no es el caso, porque es un proyecto académico.

# Tecnologías y herramientas utilizadas en el proyecto

* **Software de diseño vectorial - Microsoft Visio**

Microsoft Visio es una herramienta de creación de diagramas y gráficos vectoriales. 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.

* **Entorno de desarrollo integrado – IDE - Android Studio**

Un IDE no es más que una aplicación informática que facilita a los desarrolladores de software, el desarrollo del mismo, con lo que disminuye el tiempo de desarrollo, la complejidad, con lo que aumenta la productividad. Agrupa un conjunto de herramientas, para codificar, editar, testear y empaquetar. Posee un entorno gráfico para facilitar todas estas tareas. (Ilerna Online S.L.)

Para este proyecto, y tratándose una aplicación nativa para el sistema operativo Android, la mejor elección es **Android Studio**. Está desarrollado por JetBrains basado en su IDE estrella IntelliJ Idea. Se puede programar tanto en Kotlin como en Java. (Wikipedia, 2022). Es el mejor IDE ya que está desarrollado expresamente para el desarrollo de aplicaciones Android.

* **Lenguaje de programación - Kotlin**

Con Android Studio podemos programar en los lenguajes **Java o Kotlin** indistintamente, el IDE soporta ambos lenguajes, y es capaz de traducir ficheros de uno a otro, o trabajar con ficheros de ambos lenguajes en un mismo proyecto. El lenguaje de programación seleccionado en este proyecto es Kotlin, ya que posee una serie de ventajas para la programación de aplicaciones Android. (Ilerna Online S.L.)

* **Formato de archivos JSON**

JSON (Javascript object notation) es un formato de archivo sencillo de interpretar y escribir, para almacenar datos estructurados, que sirve para intercambiar datos entre sistemas. La sintaxis viene derivada de lenguajes de programación y se almacena en archivos de texto plano. (NextU, s.f.)

* **Sistema gestor de base de datos – Firebase Realtime DB**

Una base de datos es un conjunto de datos estructurados que pertenecen a un mismo contexto y se utiliza para administrar de forma electrónica grandes cantidades de información. Gracias al avance de la tecnología, hoy en día las bases de datos están en formato digital, evitando tener que guardar como se hacía antaño todos los datos en papel. (Wikipedia, 2022)

El sistema de bases de datos que vamos a utilizar es **Firebase Realtime Database**. Es un conjunto de herramientas que Google pone a disposición de los programadores para facilitar el desarrollo de aplicaciones móviles. (Presta)

* **Ventajas**

Las ventajas que han llevado a utilizar Firebase RB para el proyecto son: (Presta)

1. **Es un servicio gratuito en la nube para proyectos pequeños**

Firebase RB ofrece un plan gratuito para proyectos pequeños y de pago con precio adaptable a las necesidades de la aplicación.

1. **Velocidad de desarrollo**

Aumenta la productividad al poder centrase solo en la programación de la interfaz, ahorrando tiempo en la programación de la base de datos y el acceso a esta, es decir nos centraremos en la programación del “Frontend” y reduciremos coste y complejidad a la programación del “Backend” de la aplicación.

1. **No es necesaria infraestructura de servidores**

Para alojar una base de datos es necesario un servidor o servidores en “cluster”. Podemos tener nuestra propia infraestructura o contratar una. En ambos casos necesitamos dedicación y conocimientos para su administración, esto se traduce en tiempo y menos productividad, a medida que los datos aumentan y el servidor escala.

1. **Monitoreo de errores**

Firebase utiliza una función denominada Crashlytics, para la detección de errores de manera rápida, que puede monitorear errores fatales y no fatales, y los informes se generan en función de cómo los errores afectan la experiencia de los usuarios.

1. **Seguridad**

Firebase garantiza la integridad y disponibilidad de los datos por medio de una copia de seguridad periódica.

* **Desventajas**

Firebase también posee **desventajas,** pero se ha considerado que las ventajas prevalecen sobre estas para este proyecto. Las desventajas más significativas que nos afectan son (Presta):

1. **No es de código abierto**

Esta puede ser la mayor limitación de Firebase y evita que la comunidad mejore el producto, aumentando los niveles de flexibilidad y las opciones de auto alojamiento para los desarrolladores que no pueden pagar los precios de Firebase.

1. **Dependencia del proveedor**

Es un problema importante ya que, si quisiéramos migrar a otro proveedor, deberíamos reescribir todo el “backend”, por falta de acceso al código fuente.

1. **Usa estructuras de datos NoSql**

Por lo que no se pueden realizar consultas complejas. Migrar a una base de datos Sql no será sencillo.

1. **Consultas lentas**

Firebase es lenta y tiene limitaciones, en varios tipos de transacciones, consultas y otras características que afectan al rendimiento.

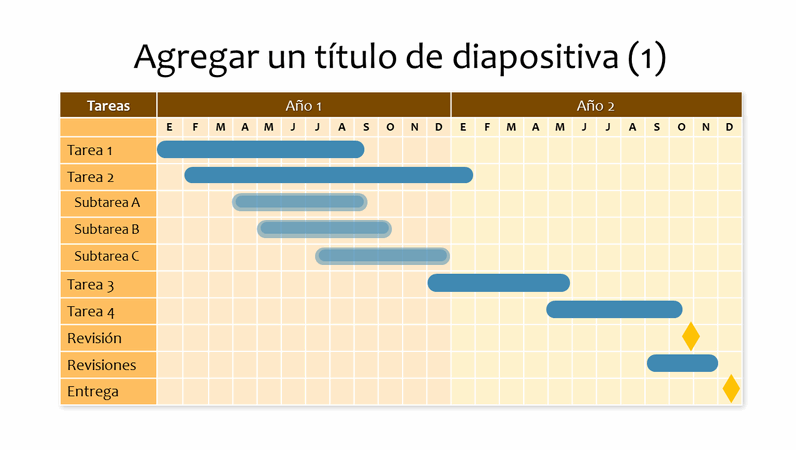
Existen otras desventajas, pero no afectan directamente a nuestro proyecto, debido a su entidad, alcance y tamaño.

# Estimación de recursos y planificación

En este apartado vamos a plasmar una estimación del tiempo previsto y el real que se ha empleado en realizar el proyecto en un **diagrama de Gantt**, plasmado en el **anexo 1.**

Un diagrama de Gantt es una herramienta de análisis que sirve para medir y planificar el tiempo de realización de un proyecto, asignar las tareas del proyecto a realizar a unos periodos de tiempo concretos, quien los debe realizar, y la relación entre ellas. (Teamleader, 2021)

El diagrama de Gantt suele está compuesto de una lista de tareas a la izquierda y un cronograma de barras a la derecha, como vemos en la plantilla de Excel de la **Figura 2**.



**Figura 2:** Diagrama de Gantt para proyecto de dos años (Microsoft, s.f.)

Al combinar el método ágil Kanban con un diagrama de Gantt para la planificación podremos tener las ventajas de las dos herramientas, es decir, tener una planificación visual en un tablero con los estados de las tareas y por otro lado un cronograma con la temporización para cada una de las tareas.

# Análisis del proyecto

* **Funcionamiento de la aplicación**

En esta sección se describe la descripción del funcionamiento de la aplicación.

Al iniciar el usuario la aplicación se encuentra una pantalla con un logo, y un formulario para iniciar sesión en la aplicación. No es posible registrarse ya que los usuarios los da de alta un administrador de la empresa.

Al iniciar sesión nos aparecerá la pantalla inicial de la aplicación. En esta ventana la que aparecerán listados de alertas, e información general de numerales de vehículos, servicios, ITVs y personal activo. Tendrá un menú de navegación inferior para desplazarse por las distintas actividades principales de la aplicación. En la parte superior derecha habrá un icono de una campana para navegar hasta la actividad de alertas de la aplicación. Al presionarla navegaremos hasta una actividad que tendrá un listado de alertas. Al presionar una de estas alertas se abrirá una ventana emergente con el detalle de esta alerta, que tendrá un botón con icono de basura para borrarla en la parte superior y una cruceta para cerrarla.

El menú constará de 5 iconos, una casa para volver a la pantalla de inicio, un coche para navegar al listado de vehículos, una llave inglesa con un martillo para las ITVs, una hoja con un lápiz para gestión de los servicios, y un muñeco para gestión de personal.

En la actividad de listado de vehículos aparecerán los datos de alta en la base de datos, en la parte superior habrá un menú para abrir una pantalla con un formulario de creación de nuevos vehículos. Presionando un icono de lupa en la parte superior derecha se podrán buscar vehículos discriminando por datos. Al presionar cada vehículo se nos abrirá una pantalla con la ficha del vehículo más detallado. En la actividad del vehículo aparecerán una foto, con un menú superior derecha para editar los datos o eliminar el registro. Tendrá otra pestaña de inventario para ver las herramientas asignadas a este vehículo. Aparecerá un listado de herramientas con una foto y se podrá editar la información y la foto.

En la actividad de listado de ITVs aparecerán los datos de alta en la base de datos. En la parte superior habrá un menú para abrir una pantalla con un formulario de creación de nuevas ITVs. Presionando un icono de lupa en la parte superior derecha se podrán buscar ITVs discriminando por datos. Al presionar cada ITV se nos abrirá una pantalla con la ficha más detallada. En la actividad de la ITV aparecerán datos ampliados, con un menú superior derecha para editar los datos o eliminar el registro. Estos registros aparecerán en un inventario general.

En la actividad de listado de servicios aparecerán los datos de alta en la base de datos. En la parte superior habrá un menú para abrir una pantalla con un formulario de creación de nuevos servicios. Presionando un icono de lupa en la parte superior derecha se podrán buscar servicios discriminando por datos. Al presionar cada servicio se nos abrirá una pantalla con la ficha más detallada. En la actividad del servicio aparecerán datos ampliados, con un menú superior derecha para editar los datos o eliminar el registro.

En la actividad de listado de personal aparecerán los datos de alta en la base de datos. En la parte superior habrá un menú para abrir una pantalla con un formulario de creación de nuevo empleados. Presionando un icono de lupa en la parte superior derecha se podrán buscar empleados discriminando por datos. Al presionar cada empleado se nos abrirá una pantalla con la ficha más detallada con su foto, con un menú superior derecha para editar los datos o eliminar el registro.

**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.

* **Requisitos funcionales**

Un **requisito funcional** es una declaración de cómo debe comportarse un sistema. Define lo que el sistema debe hacer para satisfacer las necesidades o expectativas del usuario. Se pueden considerar como características que el usuario detecta.

|  |  |
| --- | --- |
| ID | DESCRIPCIÓN |
|  | Iniciar sesión individual |
|  | Mostrar información general en la pantalla principal |
|  | Navegación sencilla entre las secciones |
|  | Listado de alertas |
|  | Visualizar detalles de alertas |
|  | Eliminar alertas |
|  | Listado de vehículos |
|  | Buscar vehículos |
|  | Visualizar detalle de vehículos |
|  | Editar vehículos |
|  | Eliminar vehículos |
|  | Listado de inventario de cada vehículo |
|  | Listado de servicios de cada vehículo |
|  | Buscar items inventario |
|  | Editar items de inventario |
|  | Eliminar items de inventario |
|  | Listado de ITVs |
|  | Buscar ITVs |
|  | Visualizar detalle de ITVs |
|  | Editar ITVs |
|  | Eliminar ITVs |
|  | Listado de averías de cada ITV |
|  | Listado de vehículos de cada ITV |
|  | Listado de servicios |
|  | Buscar servicios |
|  | Editar servicios |
|  | Eliminar servicios |
|  | Listado de personal |
|  | Buscar personal |
|  | Editar personal |
|  | Eliminar personal |
|  | Listado de vehículos asignados a cada persona |
|  | Listado de inventario general |
|  | Buscar en inventario general |
|  | Editar registro de inventario general |
|  | Eliminar registro de inventario general |

* **Requisitos no funcionales**

Los **requisitos no funcionales** definen cómo debe funcionar internamente el sistema.

(VIsure Solutions, s.f.)

|  |  |
| --- | --- |
| ID | DESCRIPCIÓN |
|  | La interfaz debe tener los controles bien distribuidos |
|  | La interfaz debe tener una paleta de colores agradable a la vista con colores suaves |
|  | El menú de navegación debe estar en la parte inferior con iconos distintivos |
|  | El sistema debe asegurar que los datos estén protegidos del acceso no autorizado |
|  | La aplicación debe tardar menos de 2 segundos en responder |
|  | Las alertas serán accesibles en todas las secciones de la aplicación |
|  | Los accesos a base dan datos no bloquearan el hilo principal de la aplicación |
|  | La aplicación seguirá los patrones de diseño de controles de Google M3[[1]](#footnote-1) |
|  | La aplicación podrá ser ejecutada en la mínima versión de Android que permita el diseño M3. |
|  | La aplicación debe señalar al usuario que está realizando una operación asíncrona. |
|  | El dispositivo debe tener conexión a internet para acceder a la base de datos Firebase. |

# Diseño del proyecto

* **Bocetos**
  + **Wireframes**
* **Diagrama de entidad-relación**

Ver **anexo 1.**

* **Diagrama relacional**

Ver anexo 2.

* **Diagrama de clases**

Ver anexo 3.

# Despliegue y pruebas

# Conclusiones

# Vías futuras

# Bibliografía

APD. (2021, Junio 8). *¿En qué consiste la metodología Kanban y cómo utilizarla?* From APD: https://www.apd.es/metodologia-kanban/

Garzas, J. (2011, Noviembre 22). *Kanban*. From Javier Garzas: https://www.javiergarzas.com/2011/11/kanban.html#:~:text=Las%20principales%20reglas%20de%20Kanban,como%20%E2%80%9Clead%20time%E2%80%9D).

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.

Martins, J. (2022, Julio 22). *¿Qué es la metodología Kanban y cómo funciona?* From Asana: https://asana.com/es/resources/what-is-kanban

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

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.

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/

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/

VIsure Solutions. (n.d.). *Qué son los requisitos funcionales: ejemplos, definición, guía completa*. From VIsureSolutions: https://visuresolutions.com/es/blog/functional-requirements/

Wikipedia. (2022, Octubre 4). *Android Studio*. From Wikipedia: https://en.wikipedia.org/wiki/Android\_Studio

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

ANEXOS

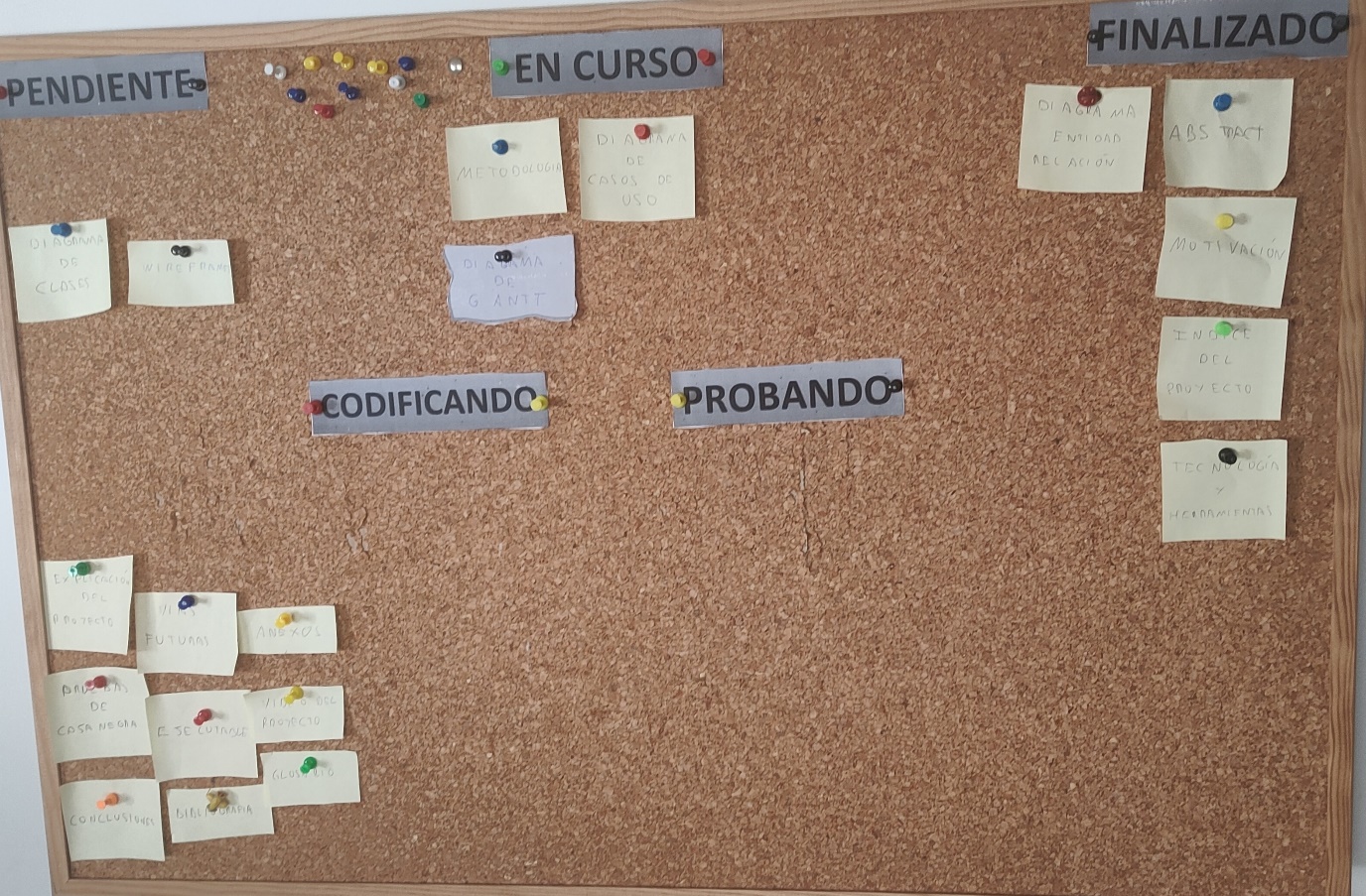
****

DIAGRAMA ENTIDAD-RELACIÓN





1. Google Material Design 3 es un sistema de diseño de código abierto para aplicaciones Android. [↑](#footnote-ref-1)