

Trabajo fin de grado GESTION DE TALLER REYbOXES



Álvaro Gómez Tejada

1. Introducción y Planteamiento del Proyecto.

1.1. Presentación del proyecto y justificación en el contexto del sector productivo.

1.2. Objetivos generales y específicos del proyecto.

1.3. Recopilación de información sobre el contexto tecnológico y empresarial.

* Clasificación de las empresas del sector según su organización y servicios.
* Análisis de estructuras organizativas y funciones de los departamentos clave.
* Identificación y priorización de necesidades del sector.
* Evaluación de oportunidades de negocio y tendencias tecnológicas.

1.4. Identificación de necesidades y oportunidades de intervención.

* Necesidades más demandadas por las empresas
* Oportunidades de negocio previsibles en el sector
* Análisis de proyectos similares y benchmarking.

1.5. Factores facilitadores y barreras que pueden afectar al desarrollo del proyecto.

* Obligaciones fiscales, laborales y de prevención de riesgos
* Posibles ayudas o subvenciones para la incorporación de nuevas tecnologías.

1.6. Metodología de trabajo y planificación inicial.

2. Estudio de viabilidad y Análisis del Proyecto

2.1. Establecimiento del alcance del sistema.

2.2. Estudio de la situación actual (proyectos similares y diferencias)

2.3. Definición de los requisitos del sistema.

2.4. Estudio de alternativas de solución.

2.5. Valoración de las alternativas

2.6. Selección de la solución.

3. Plan de Intervención

3.1 Determinación del tipo de proyecto requerido para dar respuesta a las necesidades detectadas. (permisos y autorizaciones necesarias)

3.2. Caracterización de las especificaciones técnicas del proyecto.

3.3. Priorización y secuenciación de las fases del proyecto (Ciclo de vida y metodología).

3.4. Planificación detallada de la intervención (Diagrama de Gantt).

3.5. Identificación de los recursos materiales y humanos necesarios.

3.6. Estimación de costes y necesidades de financiación.

3.7. Gestión de riesgos y elaboración del plan de prevención de incidencias

3.8. Diseño de la documentación técnica del proyecto.

3.9. Definición del plan de atención al cliente o usuario final.

4. Diseño del Proyecto.

4.1. Diseño de casos de uso (diagramas de caso de uso).

4.2. Diseño de clases/arquitectura módulos de sistema (diagrama de clases/módulos).

4.3. Diseño físico de datos (Diagramas E/R)

4.4. Definición de interfaces de usuario.

4.5. Plan de pruebas y requisitos de aceptación.

4.6. Migración (en caso de ser necesaria) y requisitos de implantación (que se requiere).

5. Seguimiento, Evaluación y Control de Calidad

5.1. Definición de procedimientos de evaluación del proyecto.

5.2. Establecimiento de indicadores de calidad y métricas de éxito.

5.3. Registro y análisis de incidencias surgidas durante la ejecución.

5.4. Plan de resolución de incidencias y optimización de procesos.

5.5. Gestión de cambios en recursos y tareas según necesidades.

5.6. Evaluación de la satisfacción del usuario final y feedback recibido.

5.7. Aseguramiento del cumplimiento de los requisitos y condiciones del proyecto.

6. Conclusiones, Documentación y Presentación Final

6.1. Evaluación global del proyecto y comparación con los objetivos iniciales.

6.2. Dificultades encontradas y estrategias de resolución.

6.3. Propuestas de mejora y evolución del proyecto.

6.4. Documentación final y elaboración del informe de resultados.

6.5. Presentación y defensa del proyecto ante la comisión evaluadora.

1. Introducción y Planteamiento del Proyecto

1.1 Presentación del proyecto y justificación en el contexto del sector productivo

En el contexto actual de constante transformación digital y evolución tecnológica, los sectores tradicionales como el de la mecánica automotriz han comenzado a experimentar una necesaria transición hacia la implementación de soluciones informáticas que permitan una gestión más eficiente, centralizada y automatizada de los procesos internos. Dentro de este escenario, el presente proyecto —titulado Sistema Integral de Gestión para Talleres Mecánicos ReyBoxes— surge como una respuesta innovadora, funcional y adaptativa a las crecientes demandas del sector productivo vinculado a los servicios de reparación, mantenimiento y gestión vehicular. Asimismo, contempla la posibilidad de adaptarse a otras líneas de negocio relacionadas, como la compraventa de vehículos, ofreciendo una solución personalizada y ajustada a las particularidades de cada tipo de taller.

La necesidad de contar con un sistema integral que unifique y digitalice los procedimientos operativos de los talleres mecánicos es una demanda recurrente en el ámbito empresarial. Este tipo de establecimientos, tradicionalmente gestionados mediante métodos manuales o herramientas informáticas rudimentarias (como hojas de cálculo), enfrentan en la actualidad numerosos retos en términos de eficiencia, trazabilidad de intervenciones, gestión de clientes, historial de vehículos, control del personal, facturación y cumplimiento normativo. Es en este contexto donde se justifica el desarrollo del presente sistema, cuyo principal objetivo es facilitar una transformación digital accesible, robusta y escalable para su implementación en entornos reales como el del taller *ReyBoxes*, destinatario principal de esta solución.

A través de la implementación de este sistema, se busca no solo optimizar el rendimiento operativo de los talleres, sino también dotar al personal técnico y administrativo de herramientas modernas que fomenten la profesionalización del servicio, la fidelización de los clientes y la mejora continua de los procesos. Esta iniciativa se enmarca plenamente dentro de la estrategia de modernización de los servicios productivos, promoviendo la competitividad del sector y su adaptación a las nuevas realidades del mercado.

1.2 Objetivos generales y específicos del proyecto

Objetivo general

Desarrollar e implementar un sistema informático integral para la gestión completa del taller mecánico de la empresa cliente, que permita automatizar, controlar y optimizar todos los procesos operativos, administrativos y técnicos. El sistema debe facilitar una gestión eficiente de clientes, vehículos, intervenciones, personal y recursos, así como integrar funcionalidades específicas para la compraventa de vehículos, adaptándose a las particularidades de cada área del negocio.

Objetivos específicos

Diseñar una arquitectura modular que se adapte a diferentes perfiles de usuarios (administrador, mecánico, personal administrativo).

Implementar un sistema de autenticación seguro con control de roles y permisos diferenciados.

Permitir el registro, consulta y modificación de clientes y vehículos, incluyendo funcionalidades avanzadas de búsqueda y autocompletado.

Integrar un sistema de gestión de fichajes para el control horario del personal técnico.

Incorporar un historial completo de intervenciones, con seguimiento de fechas, costos, observaciones y estado de cada reparación o mantenimiento.

Facilitar la exportación de datos e informes en formatos estándar (CSV, PDF), incluyendo reportes visuales personalizados con el logotipo de la empresa.

Garantizar la seguridad y confidencialidad de la información almacenada, utilizando cifrado de contraseñas, validación de entradas y buenas prácticas de desarrollo.

1.3 Recopilación de información sobre el contexto tecnológico y empresarial

Clasificación de las empresas del sector según su organización y servicios

El sector de los talleres mecánicos puede clasificarse, a grandes rasgos, en tres tipos de organizaciones:

Talleres independientes o tradicionales: Generalmente gestionados por uno o pocos técnicos, ofrecen servicios generales de reparación sin una estructura empresarial formalizada. Suelen carecer de herramientas digitales avanzadas y dependen fuertemente del conocimiento técnico individual.

Talleres multimarca o franquiciados: Pertenecen a redes o cadenas más amplias que proporcionan estándares de calidad, imagen corporativa y protocolos de atención. Están en un proceso más avanzado de digitalización, aunque no siempre cuentan con soluciones integradas personalizadas.

Talleres oficiales o de concesionario: Vinculados a marcas específicas, operan con sistemas informáticos propietarios o muy específicos. Cuentan con mayor infraestructura tecnológica, pero a menudo presentan rigidez ante cambios o adaptaciones.

Cada una de estas organizaciones presenta retos y oportunidades distintos en cuanto a digitalización, lo que justifica el desarrollo de una solución adaptable como la que se propone.

Análisis de estructuras organizativas y funciones de los departamentos clave

Un taller moderno suele contar con los siguientes departamentos o roles funcionales:

Recepción y atención al cliente: Encargado de registrar las citas, explicar los servicios y atender las consultas. Es clave para la experiencia del cliente.

Técnico o mecánico: Eje central del taller, realiza diagnósticos, reparaciones, mantenimientos y reporta información técnica.

Gestión administrativa: Maneja presupuestos, facturación, control de pagos, compras de repuestos y seguimiento financiero.

Dirección técnica o gerencia: Toma decisiones estratégicas, coordina equipos, evalúa indicadores de rendimiento y asegura el cumplimiento normativo.

El sistema propuesto debe contemplar las interacciones y flujos de información entre todos estos roles, permitiendo una comunicación fluida y evitando la duplicidad de tareas o errores humanos.

Identificación y priorización de necesidades del sector

A partir de estudios de campo, entrevistas con profesionales del sector y análisis de flujos de trabajo, se han identificado las siguientes necesidades prioritarias:

Reducción del uso de papel y digitalización de formularios y reportes.

Registro automatizado de entradas y salidas de personal.

Consulta rápida del historial de vehículos y clientes.

Generación de presupuestos y facturas con control de descuentos y promociones.

Control de citas y órdenes de trabajo.

Análisis de rentabilidad por tipo de intervención o mecánico.

Almacenamiento seguro y backup periódico de la información.

Interfaz amigable y personalizable según el perfil del usuario.

Evaluación de oportunidades de negocio y tendencias tecnológicas

La transformación digital del sector mecánico representa una oportunidad estratégica poco explotada hasta el momento. La mayoría de los talleres aún no cuentan con sistemas propios integrados, y dependen de software genérico o procesos manuales. Esto abre una ventana de innovación para soluciones a medida como la aquí presentada.

Entre las tendencias tecnológicas más relevantes que sustentan esta propuesta se encuentran:

Digitalización de PYMEs mediante software específico de bajo coste y fácil implementación.

Computación en la nube y bases de datos distribuidas, que permiten la portabilidad y escalabilidad del sistema.

Integración de informes en tiempo real para toma de decisiones basada en datos.

Sistemas de seguridad reforzada, como cifrado de datos y gestión avanzada de permisos.

Automatización de tareas repetitivas, reduciendo errores humanos y tiempos de gestión.

1.4 Identificación de necesidades y oportunidades de intervención

En un entorno cada vez más competitivo y exigente, el sector de los talleres mecánicos se enfrenta a múltiples desafíos que requieren respuestas tecnológicas innovadoras. La identificación de necesidades recurrentes, junto con el análisis de oportunidades emergentes, permite establecer una base sólida para justificar la intervención mediante el desarrollo del presente proyecto.

• Necesidades más demandadas por las empresas

Las empresas del sector han manifestado, de forma reiterada, un conjunto de necesidades que reflejan la carencia de herramientas informáticas específicas que se ajusten a su realidad operativa. Entre las demandas más relevantes se destacan:

La necesidad de un sistema centralizado para gestionar clientes, vehículos, reparaciones, agendas y fichajes del personal.

La reducción del uso del papel, en favor de soluciones digitales que permitan registrar intervenciones, generar informes y consultar historiales de forma inmediata.

La automatización de tareas repetitivas, como la facturación, la generación de presupuestos o el control de stock de piezas.

La posibilidad de realizar consultas avanzadas y personalizadas, tanto por datos técnicos como por información comercial.

La incorporación de un sistema de informes y estadísticas que permita visualizar el rendimiento del negocio, detectar fallos y proponer mejoras continuas.

Estas necesidades, comunes a talleres de distinta envergadura, demuestran la urgencia de una solución tecnológica que pueda adaptarse a distintos escenarios y niveles de complejidad.

• Oportunidades de negocio previsibles en el sector

El sector automotriz, y particularmente el segmento de mantenimiento y reparación, está experimentando un proceso de modernización sin precedentes. Este cambio estructural se traduce en una serie de oportunidades de negocio:

La digitalización de procesos abre el camino a modelos de negocio basados en el Software as a Service (SaaS), con licencias adaptadas a talleres pequeños, medianos o grandes.

El auge de los vehículos eléctricos e híbridos conlleva nuevas necesidades de mantenimiento, documentación y formación, generando nichos de especialización.

La compraventa de vehículos de segunda mano, junto con la gestión del historial mecánico certificado, representa una oportunidad clave de expansión funcional del sistema.

La creciente concienciación sobre la sostenibilidad y trazabilidad impulsa el uso de software para el control responsable de residuos, emisiones, y materiales utilizados.

Estas oportunidades indican que el desarrollo de soluciones como ReyBoxes no solo satisface una demanda actual, sino que se anticipa a futuras exigencias del mercado.

• Análisis de proyectos similares y benchmarking

Se ha llevado a cabo un proceso de benchmarking con distintos productos y servicios informáticos del sector, evaluando tanto herramientas locales como soluciones de alcance internacional. Entre los sistemas analizados se encuentran:

Soluciones genéricas como Microsoft Excel o Access, ampliamente utilizadas pero limitadas en funcionalidades específicas.

Programas de gestión ERP adaptados parcialmente al sector, que presentan interfaces complejas y una curva de aprendizaje elevada.

Plataformas SaaS extranjeras, como Shopmonkey, AutoFluent o GaragePlug, que si bien ofrecen funcionalidades avanzadas, no se ajustan completamente a las normativas locales ni al idioma, y suponen un coste elevado en licencias.

Frente a este panorama, ReyBoxes se posiciona como una alternativa intermedia pero poderosa: una solución de desarrollo nacional, adaptada a las particularidades del sector mecánico español, con interfaces amigables, enfoque modular y funcionalidades diseñadas en diálogo directo con los talleres que integran esta franquicia en expansión.

1.5 Factores facilitadores y barreras que pueden afectar al desarrollo del proyecto

El desarrollo de este sistema se enmarca en una realidad técnica, legal y económica que puede influir positiva o negativamente en su ejecución. Identificar estos factores permite planificar mejor los recursos, reducir riesgos y aprovechar los elementos facilitadores.

• Obligaciones fiscales, laborales y de prevención de riesgos

Los talleres mecánicos están sujetos a una amplia normativa legal que afecta directamente a la gestión de su actividad. Algunos aspectos relevantes incluyen:

Obligaciones fiscales como la correcta emisión de facturas, control del IVA e IRPF, y registros de operaciones.

Normativas laborales, incluyendo el registro obligatorio de jornada laboral, cumplimiento del convenio colectivo, y contratos del personal técnico.

Prevención de riesgos laborales, que exige documentar protocolos de seguridad, mantenimiento de equipos, uso de EPIs, y formación del personal.

El sistema ReyBoxes contempla estos aspectos desde su diseño, facilitando el cumplimiento normativo mediante módulos de control de fichajes, almacenamiento de documentos, alertas de vencimiento y trazabilidad de operaciones.

• Posibles ayudas o subvenciones para la incorporación de nuevas tecnologías

La coyuntura actual, enmarcada en los planes de recuperación económica y modernización de las PYMEs, representa una oportunidad inmejorable para proyectos tecnológicos como este. Existen distintas líneas de ayuda:

Subvenciones europeas vinculadas a los fondos Next Generation EU, con programas específicos para digitalización empresarial (Kit Digital, Programa Acelera Pyme).

Ayudas autonómicas y municipales orientadas a la modernización de sectores tradicionales mediante incorporación de TIC.

Deducciones fiscales por inversión en I+D+i aplicables a proyectos de desarrollo informático innovador.

Aprovechar estas ayudas puede suponer una aceleración en la implementación del sistema, así como una importante reducción de costes para las empresas usuarias.

1.6 Metodología de trabajo y planificación inicial

El desarrollo del proyecto se ha estructurado siguiendo una metodología de trabajo ágil y adaptativa, orientada a la consecución de resultados funcionales desde etapas tempranas. Para ello, se ha optado por un enfoque iterativo e incremental, basado en los principios del modelo Scrum dentro de una filosofía DevOps, que permite mantener la calidad del código, documentar el avance de manera ordenada y validar continuamente con el cliente.

La planificación inicial contempla las siguientes fases:

Análisis de requerimientos y diseño conceptual

Reuniones con el cliente para identificar necesidades, prioridades y características del entorno real. Diseño de la arquitectura lógica del sistema y definición de módulos funcionales.

Desarrollo técnico por módulos

Implementación progresiva de los distintos componentes del sistema (login, gestión de clientes, intervenciones, fichajes, exportación de informes, etc.), con validación continua y pruebas unitarias.

Integración y pruebas completas del sistema

Verificación de la interoperabilidad entre módulos, simulación de casos reales, corrección de errores y mejora del rendimiento general.

Despliegue, documentación y formación

Instalación en el entorno final, generación de manuales técnicos y de usuario, y capacitación del personal para el uso autónomo de la herramienta.

Soporte y evolución futura

Seguimiento posterior a la entrega, resolución de incidencias, y análisis de nuevas funcionalidades según el feedback recibido.

Esta metodología garantiza un producto de calidad, alineado con las expectativas del cliente, y fácilmente adaptable a futuras evoluciones del sector.

2. Estudio de viabilidad y Análisis del Proyecto

2.1 Establecimiento del alcance del sistema

El sistema propuesto tiene como finalidad proporcionar una solución informática integral para la gestión de talleres mecánicos, capaz de adaptarse a distintos perfiles de usuario (administrador, personal técnico y administrativo) y abarcar todas las áreas operativas del negocio: desde la atención al cliente y el control de intervenciones, hasta la gestión de fichajes, la emisión de informes y la organización de los vehículos registrados. Asimismo, se contempla la integración de funcionalidades específicas para la compraventa de vehículos y la generación automatizada de documentación fiscal y operativa.

En su alcance inicial, el sistema debía ser accesible desde cualquier dispositivo, incluyendo ordenadores, tablets y smartphones, por medio de la ejecución en navegadores modernos mediante tecnologías web como React. Sin embargo, debido a restricciones impuestas en el entorno de implementación, se prohibió expresamente el uso de tecnologías web o soluciones basadas en navegadores. Este cambio forzó una reformulación total del planteamiento técnico, reconduciendo el desarrollo hacia una aplicación de escritorio nativa, ejecutable en entornos Windows, desarrollada íntegramente en Python utilizando la biblioteca PySide6, el binding oficial de Qt para Python.

A pesar de este cambio sustancial en la tecnología base, se mantuvo el alcance funcional original del sistema, garantizando que la solución continúe siendo modular, escalable, y visualmente profesional. El objetivo sigue siendo ofrecer una herramienta completa, personalizable, y fácilmente operable por parte de cualquier usuario del taller, sin requerir conocimientos técnicos avanzados.

2.2 Estudio de la situación actual (proyectos similares y diferencias)

El análisis del mercado ha evidenciado la existencia de soluciones informáticas dirigidas al sector de la mecánica automotriz, muchas de ellas con arquitectura web y orientadas a modelos SaaS. Ejemplos de este tipo incluyen plataformas como:

Shopmonkey (EE.UU.): solución basada en la nube, centrada en talleres de reparación, con interfaz moderna pero escasa flexibilidad frente a normativas locales.

GaragePlug (India): software multiplataforma con enfoque en digitalización de talleres, pero con barreras de idioma, localización fiscal y soporte técnico.

Autowork Online (Reino Unido): sistema basado en navegador, con alta dependencia de conexión a internet y modelo de suscripción mensual.

Frente a estas soluciones, el presente proyecto presenta diferencias clave:

El sistema se ejecuta de forma local en cada equipo, manteniendo únicamente la conexión remota con la base de datos, con el objetivo de centralizar la información y permitir el acceso simultáneo desde todos los talleres de la franquicia.

Adaptación total al idioma, cultura organizativa y normativas fiscales y laborales españolas.

Personalización completa, desarrollada en estrecha colaboración con la franquicia ReyBoxes, con posibilidad de ajustar módulos según las necesidades reales del negocio.

Independencia tecnológica, al no depender de plataformas cerradas ni licencias de terceros, lo cual reduce costes y aumenta la sostenibilidad a largo plazo.

Estas diferencias confirman la viabilidad y pertinencia de una solución propia frente a adoptar productos genéricos del mercado.

2.3 Definición de los requisitos del sistema

El sistema debe cumplir una serie de requisitos funcionales y no funcionales para satisfacer las necesidades del cliente:

Requisitos funcionales:

Registro, modificación y consulta de clientes y vehículos.

Gestión de intervenciones (reparaciones, mantenimientos, observaciones, estados, fechas, costes).

Control de fichajes del personal (entrada/salida) con reloj en tiempo real.

Panel de control personalizado según rol de usuario.

Exportación de datos a CSV y generación de informes en PDF.

Recuperación de cuenta con validación por correo y código OTP.

Módulo de usuarios con control de roles (Administrador, Mecánico, Administrativo).

Autocompletado inteligente y validación de datos (DNI, correos, nombres).

Requisitos no funcionales:

Sistema multiplataforma (al menos compatible con distintos PCs Windows).

Interfaz visual moderna y accesible.

Alta responsividad en ventanas y formularios.

Seguridad en el almacenamiento de contraseñas (cifrado).

Base de datos en la nube (Supabase/PostgreSQL) con conexión externa protegida por SSL.

Modularidad en el código fuente y separación estricta de lógica de presentación y control.

2.4 Estudio de alternativas de solución

Durante las primeras fases del proyecto, se contemplaron diversas alternativas tecnológicas:

Aplicación web con React + Node.js (Express)

✔️ Ventajas: ejecución en navegador, responsive, amplia comunidad.

❌ Desventajas: prohibición de uso web, requiere servidor backend y dominio externo.

Aplicación híbrida con Electron

✔️ Ventajas: interfaz web con empaquetado como escritorio.

❌ Desventajas: peso elevado, consumo de recursos, dificultad para cumplir con prohibición explícita de tecnologías web.

Aplicación de escritorio en Java (JavaFX)

✔️ Ventajas: fuerte tipado, multiplataforma.

❌ Desventajas: curva de aprendizaje más elevada, dificultades estéticas, menor fluidez visual.

Aplicación de escritorio en Python con PySide6

✔️ Ventajas: rapidez de desarrollo, estética profesional con Qt, integración sencilla con base de datos externa, compatibilidad con equipos modestos.

❌ Desventajas: requiere instalación de dependencias específicas, distribución requiere empaquetado adicional.

2.5 Valoración de las alternativas

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Alternativa | Adaptabilidad | Desempeño | Cumple con restricciones | Facilidad de desarrollo | Total |
| React + Node.js | Alta | Alta | No | Media | 3/5 |
| Electron | Alta | Media | No | Media | 3/5 |
| JavaFX | Media | Alta | Sí | Baja | 3.5/5 |
| Python + PySide6 | Alta | Alta | Sí | Alta | 5/5 |

La solución basada en PySide6 resultó ser la única que cumplía con todas las condiciones impuestas por el entorno del proyecto: la prohibición de tecnologías web, la necesidad de ejecución local con posibilidad de conectarse a una base de datos compartida entre múltiples instancias del programa, una estética profesional y una capacidad de iteración rápida.

2.6 Selección de la solución

Tras el análisis detallado de alternativas, se decidió optar por la implementación del sistema utilizando Python y PySide6, por tratarse de una tecnología flexible, potente y altamente productiva, que permite desarrollar interfaces gráficas avanzadas respetando la arquitectura MVC. Esta elección no solo garantiza el cumplimiento estricto de las condiciones impuestas, sino que además permite mantener un diseño elegante, moderno y altamente usable por parte del personal de taller.

La elección de PySide6 también asegura que el sistema pueda ser empaquetado fácilmente como ejecutable (.exe) y distribuido sin complicaciones técnicas, favoreciendo su adopción por parte del cliente final sin necesidad de conocimientos técnicos ni configuración adicional. La conexión con la base de datos PostgreSQL alojada en Supabase refuerza la capacidad del sistema para operar con datos centralizados, respaldos remotos y sincronización segura.

3. Plan de Intervención

3.1 Determinación del tipo de proyecto requerido para dar respuesta a las necesidades detectadas

El tipo de intervención planteada se enmarca en el desarrollo de una aplicación informática a medida, orientada a entornos de escritorio y destinada a cubrir las necesidades operativas, administrativas y técnicas de una red de talleres mecánicos en expansión.

Dado que el proyecto se basa en la implementación de software que se ejecuta localmente, no requiere licencias urbanísticas, sanitarias ni ambientales. Sin embargo, se deben tener en cuenta ciertas autorizaciones legales y normativas de cumplimiento:

Cumplimiento del Reglamento General de Protección de Datos (RGPD), en lo relativo al almacenamiento y tratamiento de datos personales de clientes, empleados y proveedores.

Incorporación de mecanismos de seguridad informática, incluyendo cifrado de contraseñas y validación de accesos.

Utilización de software con licencias compatibles, incluyendo PySide6 (licencia LGPL) y módulos auxiliares, lo que permite un uso legal y redistribución del sistema.

3.2 Caracterización de las especificaciones técnicas del proyecto

El sistema se desarrollará con las siguientes especificaciones técnicas principales:

Lenguaje principal: Python 3.11

Interfaz gráfica: PySide6 (Qt para Python)

Base de datos: PostgreSQL alojada en Supabase (entorno cloud seguro con conexión SSL)

Sistema operativo objetivo: Windows 10/11

Distribución: Paquete ejecutable (.exe) autoextraíble mediante herramientas como PyInstaller o cx\_Freeze

Seguridad: Cifrado de contraseñas con bcrypt, validación de formularios, control de errores y logs

Arquitectura del software: Modelo-Vista-Controlador (MVC)

Modularidad: Módulos separados para login, usuarios, fichajes, clientes, vehículos, intervenciones, etc.

El sistema se diseñará con escalabilidad en mente, de forma que en fases posteriores pueda ampliarse con nuevos módulos (como facturación avanzada o control de inventario) sin reestructurar el núcleo del programa.

3.3 Priorización y secuenciación de las fases del proyecto (Ciclo de vida y metodología)

Se ha adoptado una metodología ágil, basada en iteraciones cortas y validaciones continuas por parte del cliente (Scrum adaptado). Las fases principales del ciclo de vida del proyecto son:

Análisis inicial de necesidades

Diseño modular del sistema

Desarrollo de núcleo funcional (login, usuarios, conexión DB)

Desarrollo progresivo por módulos temáticos (clientes, vehículos, fichajes...)

Integración, pruebas de validación y corrección de errores

Generación de ejecutable, documentación y entrega

Soporte, mantenimiento y propuestas de ampliación futura

Cada fase tiene entregables parciales, lo cual permite un control constante del progreso y la calidad.

3.4 Planificación detallada de la intervención (adaptada a 40 horas lectivas)

El desarrollo del proyecto se ha estructurado cuidadosamente para poder ejecutarse dentro del plazo asignado por el centro educativo, que equivale a 40 horas lectivas. Este marco temporal obliga a una planificación rigurosa, donde cada fase del proyecto debe ser abordada de forma progresiva, eficaz y sin desviaciones innecesarias.

Dado que no se permite el uso de diagramas visuales, la distribución temporal se explica detalladamente a continuación en formato narrativo:

Fase 1: Análisis y diseño preliminar (5 horas)

Durante esta etapa inicial se realiza un análisis profundo de las necesidades funcionales del cliente, se definen los requisitos técnicos y se estructura la arquitectura del sistema. Se elabora el diseño de la base de datos, se planifica el modelo de interacción entre los módulos, y se establece la división entre lógica de negocio y presentación (modelo MVC).

Fase 2: Implementación del núcleo del sistema (8 horas)

Esta fase se centra en construir la estructura base del programa, incluyendo la ventana principal, la lógica de navegación entre vistas, y el sistema de autenticación (login) con conexión segura a la base de datos PostgreSQL alojada en Supabase. También se configura el control de errores y se establece la estructura modular del proyecto.

Fase 3: Desarrollo de funcionalidades clave (16 horas)

Este bloque representa el grueso del trabajo. Incluye el desarrollo de los módulos principales del sistema:

Gestión de usuarios y control de roles.

Registro y consulta de clientes.

Registro de vehículos vinculados a los clientes.

Módulo de fichajes con reloj en tiempo real.

Historial de intervenciones con su respectivo formulario de registro y estado.

Cada módulo se programa de forma independiente, pero siguiendo una lógica coherente que permita su integración posterior.

Fase 4: Pruebas funcionales e integración (5 horas)

Se realiza la integración progresiva de los módulos y la ejecución de pruebas funcionales para validar que el sistema funciona correctamente en distintos escenarios. Se corrigen errores detectados, se ajustan estilos visuales y se verifican conexiones y consistencia de datos.

Fase 5: Documentación y empaquetado (4 horas)

En esta fase final se prepara toda la documentación técnica y del usuario. Se redactan los manuales de uso, se documenta el código fuente con comentarios explicativos, y se genera el ejecutable final. También se realiza una revisión general para asegurar que el producto cumple con todos los objetivos planteados.

Fase 6: Evaluación final y soporte (2 horas)

Se realiza una revisión detallada con el cliente final (taller ReyBoxes), se ajustan detalles menores según su feedback, y se prepara la entrega oficial. Se deja preparada una propuesta de mantenimiento y evolución futura.

Resumen de distribución del tiempo (sin gráficos):

|  |  |
| --- | --- |
| Fase | Horas estimadas |
| Análisis y diseño preliminar | 5 h |
| Desarrollo del núcleo del sistema | 8 h |
| Funcionalidades clave | 18 h |
| Pruebas e integración | 5 h |
| Documentación | 2 h |
| Evaluación y soporte final | 2 h |
| TOTAL | 40 h |

Esta planificación permite cumplir con los objetivos del proyecto dentro del marco formativo establecido por el centro, asegurando al mismo tiempo un desarrollo profesional, ordenado y funcional del sistema propuesto. Cada bloque se ha diseñado para que sea autónomo y medible, permitiendo una correcta gestión del tiempo y un seguimiento efectivo del progreso, incluso sin el apoyo de diagramas visuales.

3.5 Identificación de los recursos materiales y humanos necesarios

Recursos humanos:

Desarrollador principal (autor del proyecto)

Usuario cliente de referencia (taller ReyBoxes)

Colaboradores externos puntuales para pruebas o validación

Recursos materiales:

Ordenador con al menos 8 GB de RAM

Conexión a internet para acceder a Supabase

Herramientas de desarrollo: PyCharm, Git, navegador web, terminal

Servicios en la nube: Supabase, correo SMTP para pruebas de recuperación

Herramientas de documentación: Word, PDF, LaTeX (según entrega final)

3.6 Estimación de costes y necesidades de financiación

Este proyecto, al ser desarrollado dentro del contexto educativo, no requiere inversión económica directa, salvo el tiempo de dedicación y los recursos personales del desarrollador. Sin embargo, se realiza la siguiente estimación teórica de costes en un entorno profesional:

Horas de desarrollo: 40 h x 65 €/h = 2.600 €

Servidor PostgreSQL profesional (cloud): ~25 €/mes

Licencias PyInstaller/entornos profesionales: 0 € (uso de herramientas libres)

Diseño gráfico, iconos, logotipos personalizados: ~200 €

Total, estimado: 2825 € (sin contar soporte postventa)

En caso de escalar el sistema a otros talleres, podría contemplarse financiación externa o modelo SaaS.

3.7 Gestión de riesgos y elaboración del plan de prevención de incidencias

Riesgos identificados:

Fallo en la conexión con la base de datos remota

Errores en el empaquetado del ejecutable

Retrasos por sobrecarga de tareas externas (exámenes, prácticas)

Pérdida de datos por fallo no gestionado

Medidas preventivas:

Implementación de logs detallados y mensajes de error amigables

Validación constante en cada módulo antes de pasar al siguiente

Uso de sistema de backups automáticos desde Supabase

Control de versiones del código con Git

Además, se mantiene una documentación técnica continua, lo que facilita la rápida detección y corrección de errores.

3.8 Diseño de la documentación técnica del proyecto

La documentación técnica incluirá:

Manual de instalación y ejecución del sistema

Esquema de la base de datos (modelo relacional con claves primarias y foráneas)

Diagramas de clases, estructura del proyecto y relación entre módulos

Explicación del flujo de navegación y del uso de cada ventana

Registro de pruebas funcionales y resultados

Manual de usuario para el cliente final, con capturas y explicaciones paso a paso

3.9 Definición del plan de atención al cliente o usuario final

El sistema se entregará acompañado de:

Manual de usuario en PDF, con explicaciones detalladas de cada sección

Formulario de contacto por correo para incidencias o mejoras

Sistema de mensajes de error interpretables para ayudar al usuario a entender cualquier fallo

Ventanas intuitivas, con tooltips, íconos representativos y botones accesibles

Posibilidad de desarrollar en el futuro una versión portable para llevar en USB y usar en varios talleres sin necesidad de instalación compleja

4. Diseño del Proyecto

El diseño del sistema constituye una de las fases más críticas del proceso de desarrollo, ya que en él se definen con precisión las estructuras, comportamientos e interfaces que darán forma al producto final.

4.1 Diseño de casos de uso

El sistema está estructurado en torno a roles diferenciados de usuario y los casos de uso principales que cada uno de ellos necesita cubrir. A continuación, se describe el conjunto de funcionalidades accesibles según el perfil de acceso:

ROL ADMINISTRADOR

El perfil de Administrador es el de mayor nivel jerárquico dentro del sistema. Tiene acceso total a todas las funcionalidades disponibles, lo que le permite no solo operar como usuario del sistema, sino también gestionar la configuración, supervisar al personal y mantener el control sobre las operaciones clave del taller. Sus funciones incluyen:

Fichar entrada/salida para control horario.

Consultar el historial completo de fichajes de todos los empleados.

Crear, modificar y eliminar usuarios, así como asignarles roles y restablecer contraseñas.

Gestionar los datos de clientes, incluyendo altas, bajas, modificaciones y búsquedas.

Gestionar vehículos asociados a clientes, incluyendo ficha técnica y tipo de vehículo.

Realizar recepcionamientos de vehículos (registro de entrada para reparación o evaluación).

Elaborar y guardar presupuestos detallados de reparaciones o mantenimientos.

Acceder al módulo de compraventa, para registrar vehículos en stock, ventas, compras y traspasos.

Generar reimpresiones de recepcionamientos, presupuestos, comprobantes de compra y venta.

Cerrar sesión de forma segura para finalizar su actividad.

Este perfil está pensado para la dirección técnica o el propietario del taller.

ROL MECÁNICO:

El perfil Mecánico tiene acceso únicamente a las funcionalidades operativas relacionadas con su labor técnica. El sistema está diseñado para que pueda trabajar con rapidez, sin distracciones ni accesos innecesarios a funciones administrativas o comerciales.

Fichar su entrada y salida laboral.

Consultar su propio historial de fichajes.

Acceder al módulo de vehículos para visualizar y consultar información técnica necesaria para las reparaciones.

Registrar nuevos recepcionamientos de vehículos asignados para intervención.

Elaborar presupuestos preliminares cuando detecta una necesidad técnica durante la inspección.

Acceder al módulo de reparaciones, donde puede registrar el avance, el tipo de intervención realizada, el estado de la tarea y observaciones técnicas.

Cerrar sesión al finalizar su jornada o cambiar de turno.

El sistema limita el acceso del mecánico a módulos sensibles como creación de usuarios, gestión de clientes o compraventa, para evitar modificaciones no autorizadas.

ROL ADMINISTRATIVO

El perfil Administrativo se enfoca en tareas de gestión documental, atención al cliente y soporte a los procesos operativos del taller. Su perfil tiene amplias capacidades, aunque sin acceso a la gestión de usuarios ni al módulo de compraventa.

Fichar entrada/salida y consultar su propio historial.

Gestionar la base de clientes, incluyendo el alta de nuevos clientes, edición de datos y seguimiento.

Gestionar los vehículos asociados a clientes, mantener sus datos actualizados.

Registrar recepcionamientos y derivarlos a los técnicos o responsables correspondientes.

Elaborar y revisar presupuestos propuestos por los técnicos.

Realizar reimpresiones de documentos operativos: recepcionamientos, presupuestos, comprobantes de compra o venta.

Cerrar sesión para proteger el acceso a sus funciones.

Su rol es esencial para mantener la fluidez administrativa y documental del taller, así como para asistir a la gerencia y al personal técnico.

ROL COMPRA/VENTA

El perfil de Compra/Venta está orientado al personal encargado de gestionar el stock de vehículos, las operaciones de compraventa y los reportes comerciales asociados.

Fichar entrada y salida y consultar historial de fichajes.

Consultar y actualizar el módulo de vehículos, especialmente aquellos en venta o evaluación.

Registrar recepcionamientos relacionados con procesos de tasación, compra o entrega.

Acceder al módulo de compraventa, donde puede gestionar entradas de stock, datos de transacción, valoraciones, observaciones comerciales y cierres de venta.

Acceder al módulo de reportes, para generar documentación vinculada a las operaciones realizadas (informes de tasación, comparativas, márgenes, etc.).

Cerrar sesión de forma segura.

Este perfil carece de acceso a módulos técnicos o de creación de usuarios, ya que su función se centra en el aspecto comercial.

Control de permisos y seguridad

El control de accesos está implementado desde el momento del inicio de sesión. Según el rol asociado al usuario autenticado, el sistema muestra u oculta automáticamente los botones y módulos no permitidos. Esto evita accesos no autorizados y proporciona una interfaz más limpia y adaptada a cada tipo de usuario.

Además, si un usuario intenta acceder a una función restringida mediante medios no autorizados (por ejemplo, forzando la ruta o manipulando la interfaz), el sistema valida el rol activo y bloquea el acceso mostrando un mensaje de advertencia.

4.2 Diseño de clases y arquitectura de módulos

Imagen que contiene Texto

El contenido generado por IA puede ser incorrecto.El sistema sigue una arquitectura modelo-vista-controlador (MVC) que garantiza la separación de responsabilidades. Esta organización facilita el mantenimiento, la escalabilidad y las pruebas unitarias.

Estructura de módulos:

Módulo: Main.py

Clase principal: Aplicacion

Tipo de clase: Clase central de inicialización y navegación de ventanas (controlador raíz)

Propósito general

El módulo Main.py contiene la clase principal del sistema: Aplicacion, que actúa como el núcleo de arranque, gestión de ventanas y navegación del flujo principal de la aplicación. Es el punto de entrada del programa y se encarga de:

Configurar el entorno gráfico.

Mostrar la pantalla de presentación al inicio.

Controlar el paso entre las ventanas: presentación → login → recuperación → panel principal.

Mantener las referencias activas a los controladores principales del sistema.

🧩 Estructura de la clase Aplicacion

🔹 Constructor \_\_init\_\_(self)

Inicializa la aplicación Qt (QApplication) usando los argumentos del sistema.

Establece el icono de la ventana principal mediante setWindowIcon, utilizando una ruta absoluta al archivo favicon.ico obtenida dinámicamente desde el módulo utilidades.rutas.

Crea y muestra la ventana de presentación (VentanaPresentacion), que funciona como pantalla inicial de carga/branding.

Define los atributos de instancia controlador\_login y controlador\_recuperar como None, preparándolos para su uso posterior.

Conecta la señal mostrar\_login desde la ventana de presentación a la función correspondiente que carga la ventana de login.

Esta fase de inicialización asegura que el sistema comience con una presentación visual y prepare los módulos necesarios sin abrir múltiples ventanas simultáneamente.

🔹 Método mostrar\_login(self)

Cierra la ventana de presentación si sigue abierta.

Crea una instancia del LoginControlador, que gestiona la lógica y eventos de la pantalla de inicio de sesión.

Conecta dos señales clave:

senal\_abrir\_recuperacion → llama a mostrar\_recuperar() para cambiar a la ventana de recuperación de cuenta.

senal\_login\_exitoso → llama a mostrar\_dashboard(usuario) cuando el login ha sido correcto.

Muestra la ventana de login al usuario.

Este método centraliza el paso del usuario a la pantalla principal del sistema tras la fase de bienvenida.

🔹 Método mostrar\_recuperar(self)

Cierra la ventana de login si está abierta (para evitar múltiples pantallas activas).

Crea una instancia del RecuperarControlador, que gestiona el proceso de recuperación de contraseña por correo electrónico.

Conecta la señal senal\_volver\_login a mostrar\_login() para permitir volver atrás desde la pantalla de recuperación.

Muestra la ventana de recuperación.

Este método se invoca solo cuando el usuario ha olvidado su contraseña y necesita iniciar el proceso de validación por código.

🔹 Método mostrar\_dashboard(self, usuario)

Recibe el diccionario usuario con los datos del login exitoso.

Crea una instancia del InicioControlador, pasando el nombre y rol del usuario autenticado.

Conecta la señal senal\_cerrar\_sesion al método mostrar\_login() para permitir cerrar sesión y volver a la pantalla de login.

Muestra la ventana principal del sistema (dashboard o menú de inicio) según el rol del usuario.

Este método adapta dinámicamente la pantalla de inicio a las funciones que puede realizar el usuario según su perfil.

🔹 Método ejecutar(self)

Inicia el bucle de eventos de Qt con self.app.exec().

Esta llamada bloquea el hilo principal hasta que la aplicación se cierre.

Este método debe ser invocado desde el if \_\_name\_\_ == "\_\_main\_\_" para lanzar toda la aplicación correctamente.

🧠 Conclusión técnica

La clase Aplicacion funciona como el controlador maestro del sistema, ya que:

Orquesta la apertura y cierre de todas las ventanas del programa.

Controla el flujo lógico de la navegación entre vistas.

Mantiene en memoria los controladores activos y sus señales.

Aísla la lógica de arranque del resto del sistema, lo que mejora la mantenibilidad.

Este enfoque garantiza una arquitectura limpia y modular, ya que cada ventana o controlador se encarga de su propia lógica interna, mientras Aplicacion únicamente gestiona cuándo y cómo se muestran.

🎨 Carpeta css/: Estilos personalizados del sistema

El sistema cuenta con una carpeta específica llamada css, en la cual se agrupan todos los archivos de estilos visuales personalizados, escritos en CSS, que definen la apariencia de cada una de las ventanas e interfaces del programa. Esta estrategia de organización busca garantizar la coherencia visual del sistema y facilitar el mantenimiento y modificación de estilos sin alterar la lógica funcional.

📁 Estructura general

Cada archivo .css dentro de esta carpeta corresponde directamente a una ventana o módulo del sistema, siguiendo una asociación uno a uno con los formularios definidos en vistas/. Esto permite aplicar estilos únicos y adaptados a la funcionalidad de cada pantalla, mejorando la experiencia del usuario y la claridad visual.

📄 Archivos destacados y su propósito

| Archivo CSS | Ventana asociada | Descripción funcional |
| --- | --- | --- |
| login.css | Pantalla de inicio de sesión | Estilos para campos de entrada, botón "Entrar", iconos e imagen de logo. |
| recuperar.css | Recuperación de cuenta | Estilo del formulario de correo, botón enviar código, fondo claro. |
| verificar\_codigo.css | Verificación de código OTP | Estilos para la cuenta atrás, campos de código y mensajes visuales. |
| restaurar.css | Restaurar contraseña | Estilo limpio, campos dobles, validación visual sin alertas intrusivas. |
| inicio.css | Panel principal (Dashboard) | Botones con iconos, disposición en cuadrícula, colores por rol. |
| clientes.css | Gestión de clientes | Formulario extenso, campos alineados, botones de acción con iconos. |
| crear\_cliente\_rapido.css | Alta rápida de cliente | Estilo compacto y visualmente limpio. |
| vehiculos.css | Gestión de vehículos | Tabla con estilos de fila alterna, botones de acción destacados. |
| presupuesto.css | Módulo de presupuestos | Estética formal, resaltado de totales, estilo de observaciones. |
| anadirTareaPresupuesto.css | Añadir tareas a presupuestos | Estilo de subtareas, resaltado por tipo de intervención. |
| recepcionamiento.css | Registro de recepción de vehículos | Diseño claro para firmas, campos técnicos y observaciones. |
| compraventa.css | Gestión de compraventa | Estilo de stock, tarjetas visuales, fondo blanco con detalles rojos. |
| reimpresionRecepcionamiento.css | Reimpresión de recepcionamientos | Ajustes de impresión y formato de campos. |
| reimpresionPresupuestos.css | Reimpresión de presupuestos | Estilo tipo factura, con márgenes y logotipo. |
| reimpresionCompras.css | Reimpresión de compras | Formato de tickets o comprobantes. |
| reimpresionVentas.css | Reimpresión de ventas | Estética limpia, énfasis en totales. |
| historial.css | Historial de fichajes | Tabla expandida, colores para diferenciar tipo de fichaje. |
| fichar.css | Fichaje de entrada/salida | Diseño con reloj visible, campos grandes y centrados. |
| usuarios.css | Gestión de usuarios | Estilo administrativo, botones agrupados, íconos por acción. |
| correo\_confirmacion.css | Estilo del correo HTML (confirmación) | Diseño con colores institucionales, título visual destacado. |

🧩 Importancia técnica

Separación de estilo y lógica: al mantener los estilos en archivos independientes, se sigue una buena práctica que mejora la mantenibilidad.

Modularidad visual: cada módulo tiene libertad para personalizar su interfaz sin interferir con otros.

Consistencia visual: gracias al uso de clases comunes (botones, inputs, iconos), se mantiene una línea estética uniforme.

Escalabilidad: si en el futuro se desea cambiar el diseño general, bastará con actualizar los archivos de estilos, sin tocar la lógica de cada ventana.

**📁 Carpeta documentacion/**

La carpeta documentacion/ contiene el archivo maestro del proyecto, correspondiente a la **memoria escrita del Trabajo Fin de Grado** (TFG), así como archivos temporales generados automáticamente por el editor de texto utilizado (Microsoft Word).

**Archivos incluidos:**

* **Documentacion\_TFG\_Taller\_ReyBoxes.docx**  
  Documento principal que recopila de forma estructurada todos los apartados requeridos por el centro educativo, incluyendo análisis, diseño, planificación, implementación, pruebas, anexos y conclusiones del sistema *ReyBoxes*. Contiene tanto el desarrollo técnico como la justificación empresarial del proyecto.

Carpeta vistas/: contiene todos los formularios e interfaces de usuario (login, inicio, clientes, fichajes, etc.).

Carpeta controladores/: alberga la lógica de interacción de cada ventana.

Carpeta consultas/: contiene los módulos con las consultas SQL organizadas por función (login, usuarios, fichajes, clientes...).

Carpeta utilidades/: funciones auxiliares, validaciones, generación de informes y herramientas compartidas.

Cada clase de controlador maneja eventos (botones, formularios), valida datos y delega las operaciones en las funciones SQL. Las vistas son visuales y están desacopladas del backend.

4.3 Diseño físico de datos

La base de datos utilizada es PostgreSQL, alojada en Supabase con conexión cifrada SSL. Se han definido las siguientes tablas principales:

usuarios: contiene nombre, contraseña cifrada, correo, estado y rol\_id.

roles: almacena los roles permitidos en el sistema (Administrador, Mecánico, Administrativo).

fichajes: registra fecha, hora, tipo (entrada/salida), y el usuario asociado.

clientes: datos personales, DNI, teléfono, dirección, observaciones.

vehiculos: vinculado a clientes, con marca, modelo, matrícula, tipo de vehículo.

tipos\_vehiculo: normaliza los tipos (turismo, moto, industrial...).

intervenciones: almacena reparaciones con fechas, estado, observaciones, costes, etc.

estados\_intervencion: tabla de referencia para el estado de cada reparación.

Todas las relaciones están definidas mediante claves foráneas que aseguran la integridad referencial. Se utilizan índices en campos clave para mejorar el rendimiento en búsquedas.

4.4 Definición de interfaces de usuario

Las interfaces se diseñaron con PySide6, imitando la estética de una aplicación de escritorio moderna, pero adaptada a resoluciones variadas. Se priorizó la claridad visual, el acceso rápido a funciones frecuentes, y la navegación intuitiva.

Características clave:

Ventanas personalizadas por tipo de usuario.

Botones grandes, iconos representativos y texto explicativo.

Tooltips en todos los controles importantes.

Pantallas centradas con estilo uniforme (colores institucionales, fuente Montserrat).

Ventanas modales para acciones críticas (fichajes, recuperación de contraseña).

Exportación y navegación guiada con mensajes de confirmación y errores amigables.

Cada vista fue construida manualmente, respetando un estilo coherente, sin plantillas automáticas ni generadores visuales.

4.5 Plan de pruebas y requisitos de aceptación

Para asegurar la calidad del sistema, se elaboró un plan de pruebas estructurado en tres niveles:

1. Pruebas unitarias

Validación de formularios (DNI, correos, contraseñas).

Comprobación de formatos de fecha y hora.

Ensayo de funciones auxiliares como generación de informes.

2. Pruebas funcionales

Simulación de inicio de sesión con distintos roles.

Registro de clientes y comprobación en base de datos.

Inserción y consulta de intervenciones.

Registro y exportación de fichajes.

3. Pruebas de integración

Pruebas completas con flujo de navegación entre vistas.

Verificación de que los cambios en la base de datos se reflejan en la interfaz.

Evaluación de rendimiento con múltiples registros.

Requisitos de aceptación:

El sistema debe ejecutarse sin errores en cualquier PC Windows moderno.

Debe poderse conectar correctamente a la base de datos en Supabase.

El usuario debe poder completar al menos un caso de uso completo por rol sin asistencia externa.

El ejecutable debe incluir todo lo necesario para ejecutarse sin instalación compleja.

4.6 Migración (en caso de ser necesaria) y requisitos de implantación

Dado que el sistema no sustituye a otro anterior, no se requiere migración de datos. Sin embargo, se han tenido en cuenta aspectos importantes para una futura implantación en entornos reales:

Requisitos mínimos de implantación:

Ordenador con Windows 10 o superior.

Conexión a internet para acceder a Supabase.

Usuario con cuenta registrada en el sistema.

Archivo ejecutable .exe proporcionado por el desarrollador.

Posibles migraciones futuras:

Migración de datos desde hojas de cálculo (CSV a PostgreSQL).

Conexión a otras bases de datos (MySQL, SQLite) si se decide cambiar el motor.

Integración con plataformas externas (correo corporativo, módulos contables, etc.).

Para facilitar cualquier migración futura, la base de datos se ha diseñado de forma normalizada y abierta, permitiendo exportación e importación sin dependencias complejas.