

Trabajo fin de grado - Sistema Integral de Gestión para Talleres Mecánicos ReyBoxes -



Álvaro Gómez Tejada

*A mi madre, por ser el ejemplo más puro de sacrificio, fortaleza y amor incondicional. Por enseñarme desde pequeño que el conocimiento abre puertas, que los tropiezos fortalecen, y que rendirse nunca es una opción.*

*A mi mujer, por su compañía incansable, por comprender mis silencios frente al teclado, mis desvelos entre líneas de código, y por convertir cada jornada complicada en un motivo más para seguir. Tu apoyo ha sido mi equilibrio durante todo este camino.*

*Y a todos mis profesores, que a lo largo de este proceso formativo no solo me han guiado con su conocimiento, sino que han sabido despertar mi curiosidad, exigirme con rigor y acompañarme con cercanía.*

*Gracias por vuestra entrega diaria, por mantener la vocación viva a pesar de las dificultades, y por recordar con cada clase que formar personas va mucho más allá de enseñar contenidos. Este proyecto también es fruto de vuestra labor silenciosa, paciente y comprometida.*

Índice:

[1. Introducción y Planteamiento del Proyecto 3](#_Toc199285571)

[1.1 Presentación del proyecto y justificación en el contexto del sector productivo 3](#_Toc199285572)

[1.2 Objetivos generales y específicos del proyecto 4](#_Toc199285573)

[1.3 Recopilación de información sobre el contexto tecnológico y empresarial 5](#_Toc199285574)

[1.3.1.- Clasificación de las empresas del sector según su organización y servicios 5](#_Toc199285575)

[1.3.2.- Análisis de estructuras organizativas y funciones de los departamentos clave 5](#_Toc199285576)

[1.3.3.- Identificación y priorización de necesidades del sector 6](#_Toc199285577)

[1.3.4.- Evaluación de oportunidades de negocio y tendencias tecnológicas 6](#_Toc199285578)

[1.4 Identificación de necesidades y oportunidades de intervención 7](#_Toc199285579)

[1.4.1.- Necesidades más demandadas por las empresas 7](#_Toc199285580)

[1.4.2.- Oportunidades de negocio previsibles en el sector 7](#_Toc199285581)

[1.4.3.- Análisis de proyectos similares y benchmarking 8](#_Toc199285582)

[1.5 Factores facilitadores y barreras que pueden afectar al desarrollo del proyecto 9](#_Toc199285583)

[1.5.1.- Obligaciones fiscales, laborales y de prevención de riesgos 9](#_Toc199285584)

[1.5.2.- Posibles ayudas o subvenciones para la incorporación de nuevas tecnologías 9](#_Toc199285585)

[1.6 Metodología de trabajo y planificación inicial 10](#_Toc199285586)

[2. Estudio de viabilidad y Análisis del Proyecto 11](#_Toc199285587)

[2.1 Establecimiento del alcance del sistema 11](#_Toc199285588)

[2.2 Estudio de la situación actual (proyectos similares y diferencias) 11](#_Toc199285589)

[2.3 Definición de los requisitos del sistema 12](#_Toc199285590)

[2.3.1.- Requisitos funcionales: 12](#_Toc199285591)

[2.3.2.- Requisitos no funcionales: 13](#_Toc199285592)

[2.4 Estudio de alternativas de solución 13](#_Toc199285593)

[2.5 Valoración de las alternativas 14](#_Toc199285594)

[2.6 Selección de la solución 14](#_Toc199285595)

[3. Plan de Intervención 15](#_Toc199285596)

[3.1 Determinación del tipo de proyecto requerido para dar respuesta a las necesidades detectadas 15](#_Toc199285597)

[3.2 Caracterización de las especificaciones técnicas del proyecto 15](#_Toc199285598)

[3.3 Priorización y secuenciación de las fases del proyecto (Ciclo de vida y metodología) 16](#_Toc199285599)

[3.4 Planificación detallada de la intervención 17](#_Toc199285600)

[3.5 Identificación de los recursos materiales y humanos necesarios 19](#_Toc199285601)

[3.6 Estimación de costes y necesidades de financiación 19](#_Toc199285602)

[3.6.1 Acuerdo comercial real 20](#_Toc199285603)

[3.7 Gestión de riesgos y elaboración del plan de prevención de incidencias 20](#_Toc199285604)

[3.8 Diseño de la documentación técnica del proyecto 21](#_Toc199285605)

[3.9 Definición del plan de atención al cliente o usuario final 21](#_Toc199285606)

[3.10 Planificación detallada de la intervención (REAL) 22](#_Toc199285607)

[4. Diseño del Proyecto 23](#_Toc199285608)

[4.1 Diseño de Casos de Uso 23](#_Toc199285609)

[4.1.1.- ROL ADMINISTRADOR 23](#_Toc199285610)

[4.1.2.- ROL MECÁNICO 25](#_Toc199285611)

[4.1.3.- ROL ADMINISTRATIVO 25](#_Toc199285612)

[4.1.4.- ROL COMPRA/VENTA 26](#_Toc199285613)

[4.1.5.- Control de permisos y seguridad del sistema 27](#_Toc199285614)

[4.2 Diseño de clases y arquitectura de módulos 28](#_Toc199285615)

[4.3 Estructura de módulos 29](#_Toc199285616)

[4.3.1.- Módulo: Main.py 29](#_Toc199285617)

[4.3.2 Carpeta css/: 31](#_Toc199285618)

[4.3.3 Carpeta documentacion/ 31](#_Toc199285619)

[4.3.4 Carpeta documentos/: 32](#_Toc199285620)

[4.3.5 Carpeta font/: 33](#_Toc199285621)

[4.3.6 Carpeta img/ 34](#_Toc199285622)

[4.3.7 Carpeta lib/: 34](#_Toc199285623)

[4.3.8.- Carpeta modelos 36](#_Toc199285624)

[4.3.9 Carpeta plantillas/: 37](#_Toc199285625)

[4.3.10 Carpeta tests/ 37](#_Toc199285626)

[4.3.11 Carpeta Scripts/ 38](#_Toc199285627)

[4.3.12 Carpeta utilidades/ 39](#_Toc199285628)

[4.3.13.1 Subdirectorio hilos/ 42](#_Toc199285629)

[4.3.14 Directorio vistas/ 43](#_Toc199285630)

[4.3.15 Carpeta controladores/ 45](#_Toc199285631)

[4.4 Diseño físico de dato 48](#_Toc199285632)

[4.4.1 Tabla: fichajes 49](#_Toc199285633)

[4.4.2 Tabla: usuarios 51](#_Toc199285634)

[4.4.3 Tabla: roles 53](#_Toc199285635)

[4.4.4 Tabla: clientes 55](#_Toc199285636)

[4.4.5 Tabla: intervenciones 58](#_Toc199285637)

[4.4.6 Tabla: vehículos 61](#_Toc199285638)

[4.4.7 Tabla: combustibles 64](#_Toc199285639)

[4.4.8 Tabla: tipos\_vehiculo 65](#_Toc199285640)

[4.4.9 Tabla: recepcionamientos 67](#_Toc199285641)

[4.4.10 Tabla: presupuestos 72](#_Toc199285642)

[4.4.11 Tabla: tareas\_presupuesto 74](#_Toc199285643)

[4.4.12 Tabla: estados\_intervencion 76](#_Toc199285644)

[4.4.13 Tabla: tipos\_intervencion 78](#_Toc199285645)

[4.4.14 Tabla: urgencias 79](#_Toc199285646)

[4.4.15 Tabla: coches\_venta 81](#_Toc199285647)

[4.4.16 Tabla: ventas 84](#_Toc199285648)

[4.4 Definición de interfaces de usuario 86](#_Toc199285649)

[4.5 Plan de pruebas y requisitos de aceptación 87](#_Toc199285650)

[4.5.1. Pruebas unitarias 87](#_Toc199285651)

[4.5.2. Pruebas funcionales 93](#_Toc199285652)

[4.5.3. Pruebas de integración 93](#_Toc199285653)

[4.6 Migración (en caso de ser necesaria) y requisitos de implantación 93](#_Toc199285654)

[5. Seguimiento, Evaluación y Control de Calidad 95](#_Toc199285655)

[5.1. Definición de procedimientos de evaluación del proyecto 95](#_Toc199285656)

[5.2. Establecimiento de indicadores de calidad y métricas de éxito 96](#_Toc199285657)

[5.3. Registro y análisis de incidencias surgidas durante la ejecución 96](#_Toc199285658)

[5.4. Plan de resolución de incidencias y optimización de procesos 97](#_Toc199285659)

[5.5. Gestión de cambios en recursos y tareas según necesidades 97](#_Toc199285660)

[5.6. Evaluación de la satisfacción del usuario final y feedback recibido 98](#_Toc199285661)

[5.7. Aseguramiento del cumplimiento de los requisitos y condiciones del proyecto 99](#_Toc199285662)

[6. Conclusiones, Documentación y Presentación Final 99](#_Toc199285663)

[7.1. Evaluación global del proyecto y comparación con los objetivos iniciales 99](#_Toc199285664)

[6.2. Dificultades encontradas y estrategias de resolución 100](#_Toc199285665)

[6.3. Propuestas de mejora y evolución del proyecto 100](#_Toc199285666)

[6.3.1. Incorporación de un módulo de facturación oficial automatizada 100](#_Toc199285667)

[6.3.2. Gestión de almacén y stock de recambios 101](#_Toc199285668)

[6.3.3. Sincronización entre múltiples dispositivos (móvil, tablet, PC) 101](#_Toc199285669)

[6.3.4. Panel de informes avanzados y estadísticas interactivas 101](#_Toc199285670)

[6.3.5. Mantenimiento predictivo básico 102](#_Toc199285671)

[6.3.6. Registro de actividad del sistema (logging de acciones) 102](#_Toc199285672)

[6.3.7. Integración con servicios externos (correo, WhatsApp, agenda) 102](#_Toc199285673)

[6.3.8. Sistema de backup y restauración automatizada 103](#_Toc199285674)

[6.4 Documentación técnica generada con PyDoc 103](#_Toc199285675)

[6.4.1. Objetivos de la documentación técnica 103](#_Toc199285676)

[6.4.2. Estructura documentada 103](#_Toc199285677)

[6.4.3. Visualización de la documentación 103](#_Toc199285678)

[7. Manual de Usuario 105](#_Toc199285679)

[7.1. Acceso al sistema 105](#_Toc199285680)

[7.1.1 Descripción general 105](#_Toc199285681)

[7.1.2 Instrucciones paso a paso para iniciar sesión 106](#_Toc199285682)

[7.2. Recuperación de cuenta 107](#_Toc199285683)

[7.2.1. Descripción general 107](#_Toc199285684)

[7.2.2 Instrucciones paso a paso para recuperar el acceso 107](#_Toc199285685)

[7.2.3. Verificación de código 109](#_Toc199285686)

[7.2.4 Descripción general 109](#_Toc199285687)

[7.2.5 Instrucciones paso a paso para verificar el código 109](#_Toc199285688)

[7.2.6. Nueva contraseña 110](#_Toc199285689)

[7.2.7 Descripción general 111](#_Toc199285690)

[7.3. Menú principal (Dashboard) 111](#_Toc199285691)

[7.3.1 Descripción general 111](#_Toc199285692)

[7.3.2 Instrucciones de uso 112](#_Toc199285693)

[7.4. Fichajes 113](#_Toc199285694)

[7.4.1 Descripción general 113](#_Toc199285695)

[7.4.2 Instrucciones paso a paso para fichar 113](#_Toc199285696)

[7.5. Historial de Fichajes 114](#_Toc199285697)

[7.5.1 Descripción general 114](#_Toc199285698)

[7.5.2 Diferencias según el rol del usuario 116](#_Toc199285699)

[7.5.3 Instrucciones paso a paso para consultar fichajes 116](#_Toc199285700)

[7.6. Crear usuarios 116](#_Toc199285701)

[7.6.1 Descripción general 117](#_Toc199285702)

[7.6.2 Instrucciones paso a paso para registrar un nuevo usuario 117](#_Toc199285703)

[7.7. Gestión de clientes 119](#_Toc199285704)

[7.7.1 Descripción general 119](#_Toc199285705)

[7.7.2 Instrucciones paso a paso 120](#_Toc199285706)

[7.7.3 Errores controlados en la gestión de clientes 121](#_Toc199285707)

[7.8. Gestión de vehículos 123](#_Toc199285708)

[7.8.1 Descripción general 123](#_Toc199285709)

[7.8.2 Instrucciones paso a paso 124](#_Toc199285710)

[8.9. Recepcionamiento de vehículos 126](#_Toc199285711)

[8.9.1 Descripción general 126](#_Toc199285712)

[7.9.2 Instrucciones paso a paso para recepcionar un vehículo 131](#_Toc199285713)

[7.10. Compraventa de vehículos 133](#_Toc199285714)

[7.10.1 Compra de vehículo al cliente 134](#_Toc199285715)

[7.10.2 Venta de vehículo al cliente 135](#_Toc199285716)

[8.10.3 Instrucciones paso a paso 137](#_Toc199285717)

[7.11. Presupuestos e Intervenciones 138](#_Toc199285718)

[7.11.1 Estructura de la pantalla 138](#_Toc199285719)

[7.11 2. Campos adicionales para rellenar 139](#_Toc199285720)

[7.11.3. Lista de tareas presupuestadas 139](#_Toc199285721)

[7.11.4. Coste total estimado 139](#_Toc199285722)

[7.11.5. Opciones de salida 139](#_Toc199285723)

[8.11.6 Instrucciones paso a paso 140](#_Toc199285724)

[7.12 Reimpresión de recepcionamientos 141](#_Toc199285725)

[7.12.1 Objetivo del módulo 141](#_Toc199285726)

[7.12.2 Estructura de la pantalla 141](#_Toc199285727)

[7.12.3 Opciones disponibles 142](#_Toc199285728)

[7.12.4 Instrucciones paso a paso 142](#_Toc199285729)

[7.13. Reimpresión de presupuestos 143](#_Toc199285730)

[7.13.1 Objetivo de la pantalla 143](#_Toc199285731)

[7.13.2 Estructura de la ventana 143](#_Toc199285732)

[7.13.3 Acciones disponibles 144](#_Toc199285733)

[7.13.4 Instrucciones paso a paso 144](#_Toc199285734)

[8.14. Reimpresión de compras 145](#_Toc199285735)

[7.14.1 Objetivo funcional 145](#_Toc199285736)

[7.14.2 Estructura de la pantalla 145](#_Toc199285737)

[7.14.3 Opciones disponibles 146](#_Toc199285738)

[7.14.4 Instrucciones paso a paso 146](#_Toc199285739)

[7.15. Envío y reimpresión de documentos 146](#_Toc199285740)

[7.15.1 Objetivo del módulo 147](#_Toc199285741)

[7.15.2 Estructura de la pantalla 147](#_Toc199285742)

[7.15.3 Opciones disponibles 147](#_Toc199285743)

[7.15.4 Instrucciones paso a paso 147](#_Toc199285744)

[8.- Bibliografía del Proyecto 148](#_Toc199285745)

[8.1. Tecnologías y herramientas utilizadas 148](#_Toc199285746)

[8.2. Buenas prácticas de desarrollo 149](#_Toc199285747)

[8.3. Normativas y documentación adicional 149](#_Toc199285748)

[8.4. Herramientas de modelado y documentación UML 150](#_Toc199285749)

[8.5 Fuentes adicionales de apoyo y aprendizaje 150](#_Toc199285750)

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

1.3.1.- 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.

1.3.2.- 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.

1.3.3.- 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.

1.3.4.- 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.

1.4.1.- 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.

1.4.2.- 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.

1.4.3.- 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.

1.5.1.- 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.

1.5.2.- 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.

2.3.1.- 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).

2.3.2.- 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 |

Tabla 1 Comparativa de tecnologías para el desarrollo del sistema ReyBoxes, evaluadas según adaptabilidad, desempeño, cumplimiento de restricciones y facilidad de desarrollo.

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

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.



Ilustración 1 Diagrama de Gantt con la distribución de 40 horas entre las distintas fases del proyecto ReyBoxes.

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):

Tabla 2 Tabla de estimación de horas por fase del proyecto ReyBoxes, con un total de 40 horas planificadas.

|  |  |
| --- | --- |
| 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 fue desarrollado inicialmente en un entorno académico, utilizando herramientas de software libre y recursos personales del desarrollador. No obstante, para contextualizar su viabilidad económica en un entorno profesional, se presenta a continuación una estimación teórica de costes de desarrollo:

* Horas de desarrollo: 40 h × 65 €/h = 2.600 €
* Servidor PostgreSQL profesional (cloud): ~25 €/mes
* Licencias de empaquetado (PyInstaller u otras): 0 € (uso de herramientas libres)
* Diseño gráfico e iconografía personalizada: ~200 €

Total, estimado de desarrollo inicial: ~2.825 € (sin incluir mantenimiento o soporte postventa)

3.6.1 Acuerdo comercial real

A diferencia de la estimación teórica anterior, en la práctica el software ha sido oficialmente implantado en un taller mecánico real, con el que se ha firmado un acuerdo de prestación de servicios.

* Se ha establecido un contrato de mantenimiento y mejora progresiva del sistema, con una duración inicial de 5 años.
* Durante este periodo, el desarrollador se compromete a aplicar correcciones, introducir mejoras funcionales, garantizar compatibilidad con futuras versiones del sistema operativo y dar soporte técnico.
* Como compensación, el taller abonará 300 € mensuales, lo que representa una financiación sostenida de 18.000 € en el periodo contratado.
* Finalizado este plazo, el contrato podrá renovarse de mutuo acuerdo entre las partes, suscribiendo un nuevo convenio adaptado a las condiciones futuras.

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

3.10 Planificación detallada de la intervención (REAL)

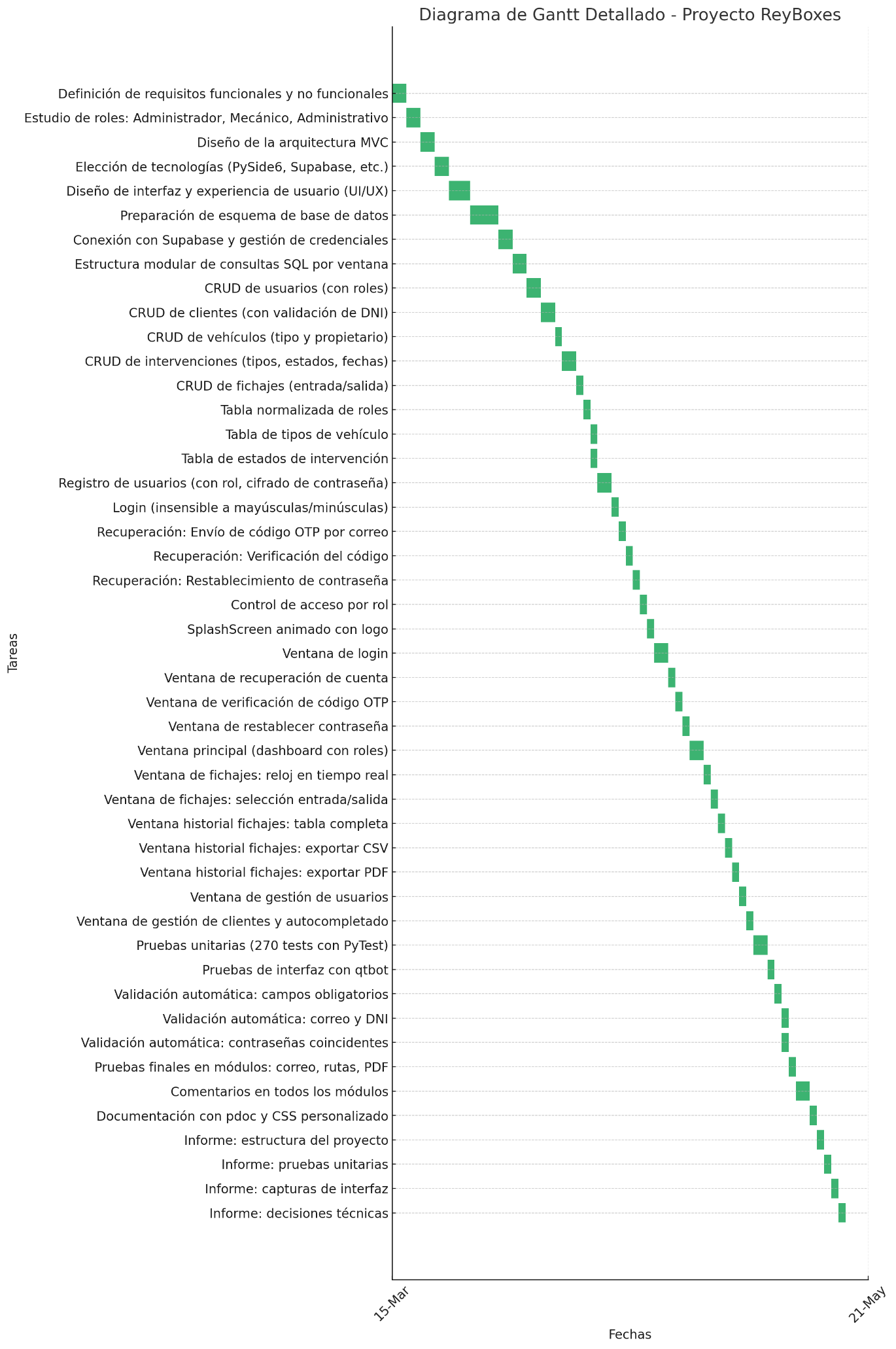


Ilustración 2 Diagrama de Gantt detallado del proyecto ReyBoxes, con tareas distribuidas desde la planificación hasta la documentación final.

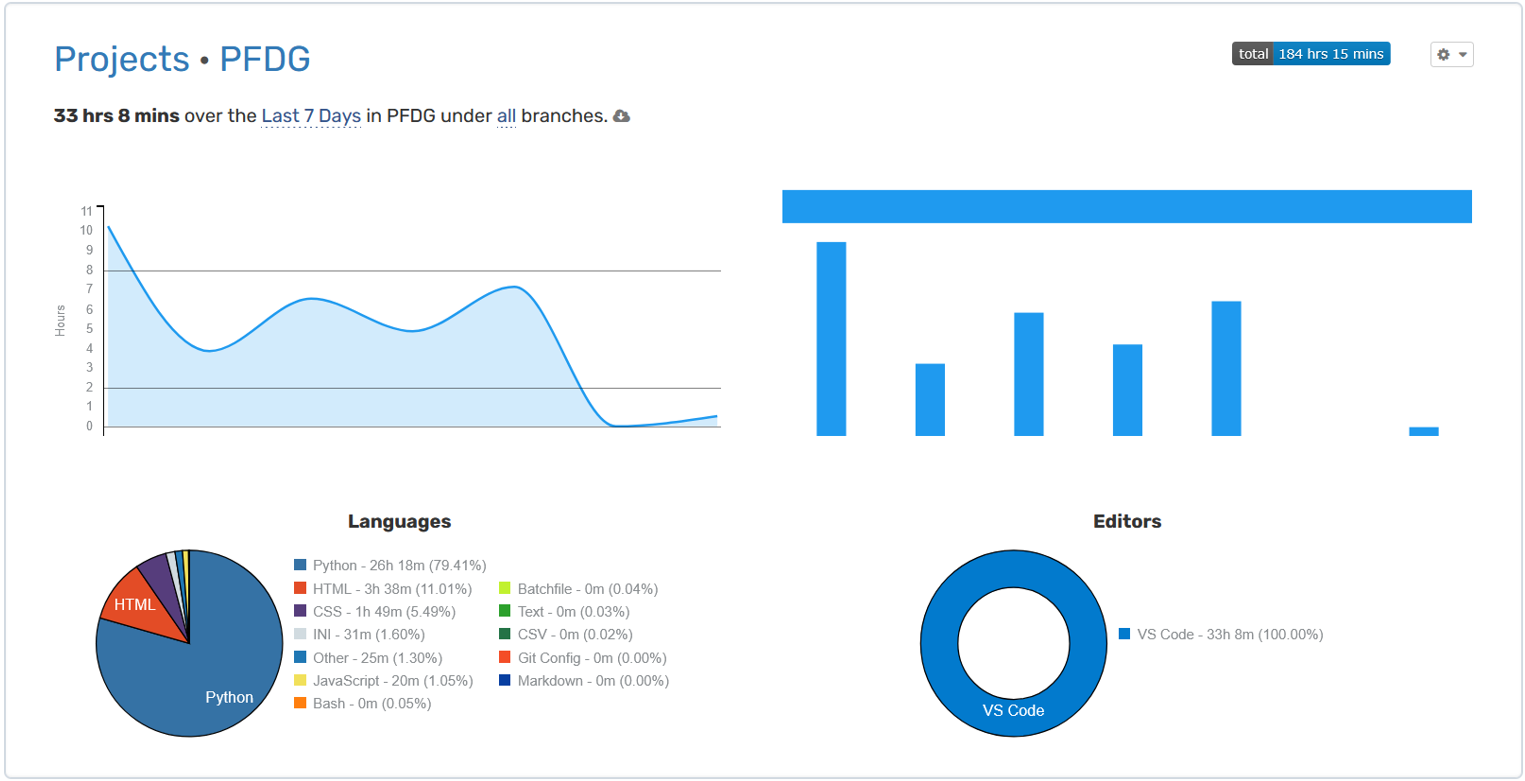
4. Diseño del Proyecto

Ilustración 3 Estadísticas de desarrollo del proyecto PFDG en los últimos 7 días, destacando el uso predominante de Python en VS Code.

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 de gestión ha sido cuidadosamente diseñado en torno a una arquitectura basada en roles diferenciados, que define con precisión los permisos, accesos y responsabilidades asignados a cada perfil de usuario. Este enfoque garantiza la seguridad, la claridad operativa y la adaptabilidad del sistema a los flujos reales de trabajo de un taller mecánico moderno.

A continuación, se describe en profundidad cada uno de los casos de uso, agrupados por perfil de acceso autorizado:

4.1.1.- ROL ADMINISTRADOR

El perfil de Administrador representa la figura de máxima autoridad dentro del sistema. Este rol ha sido diseñado para la dirección técnica, propietarios o responsables generales del taller. Tiene control total sobre todos los módulos, configuraciones y funcionalidades del sistema. Actúa como superusuario y puede intervenir en cualquier área del software, lo que permite tomar decisiones estratégicas y realizar acciones correctivas.

Funciones principales:

* Acceso total al sistema sin restricciones, incluida la configuración interna del entorno.
* Control de personal: puede fichar su entrada/salida y también visualizar el historial completo de fichajes de todos los empleados, independientemente del rol.
* Gestión de usuarios: alta, modificación, baja de cuentas, restablecimiento de contraseñas, y asignación de roles.
* Gestión completa de clientes: búsqueda avanzada, creación, modificación o eliminación de registros.
* Administración de vehículos: acceso a la base completa de vehículos, vinculados o no a clientes, pudiendo consultar, añadir o modificar fichas técnicas.
* Recepcionamientos: registrar, consultar, modificar o reimprimir recepcionamientos de vehículos, incluyendo evaluación inicial, urgencias y estado del vehículo.
* Presupuestos: generación, edición, validación y autorización de presupuestos propuestos por otros roles (por ejemplo, mecánicos).
* Compras y ventas de vehículos: acceso al módulo de compraventa donde puede registrar nuevas adquisiciones, gestionar stock, controlar ventas, márgenes, descuentos y contratos asociados.
* Reimpresiones y documentación: puede generar duplicados de cualquier documento emitido: contratos, fichas, recepcionamientos, presupuestos o comprobantes de venta.
* Cierre de sesión seguro que garantiza la protección del sistema ante usos indebidos.

Particularidades:

* Este perfil tiene acceso simultáneo a funciones administrativas, técnicas, comerciales y de supervisión, por lo que su interfaz incluye todos los botones y accesos habilitados.
* La interfaz ha sido optimizada para permitirle ejecutar acciones rápidas y gestionar incidencias detectadas en tiempo real.

4.1.2.- ROL MECÁNICO

El rol de Mecánico ha sido concebido para el personal técnico encargado directamente de las reparaciones, revisiones y diagnósticos. Este perfil no tiene acceso a información sensible o administrativa, garantizando que el entorno de trabajo sea enfocado, operativo y sin distracciones.

Funciones principales:

* Fichaje de jornada mediante entrada y salida con registro horario exacto.
* Consulta de historial personal de fichajes, útil para control interno.
* Visualización del módulo de vehículos, con acceso a información técnica necesaria para llevar a cabo reparaciones.
* Recepcionamiento técnico: puede registrar la llegada de vehículos asignados, con datos específicos sobre fallos, estado y observaciones.
* Presupuestos preliminares: cuando detecta una avería o tarea adicional durante la revisión, puede proponer un presupuesto técnico para revisión posterior.
* Módulo de intervenciones: acceso directo para registrar avances en las reparaciones, estado del proceso, tipo de intervención y observaciones.
* Cierre de sesión obligatorio al terminar la jornada, lo que protege la trazabilidad y la integridad de los registros.

Restricciones:

* Sin acceso a gestión de usuarios, clientes, ni compraventa.
* La visibilidad de botones y menús está restringida estrictamente a lo necesario para su rol.

4.1.3.- ROL ADMINISTRATIVO

El perfil Administrativo tiene como objetivo facilitar tareas de gestión documental, recepción de clientes, coordinación de procesos y soporte administrativo. Representa el nexo entre el personal técnico y la dirección.

Funciones principales:

* Fichaje de jornada y consulta de su propio historial.
* Gestión de clientes: alta de nuevos clientes, modificación de datos, validación de DNI, control de duplicados.
* Gestión de vehículos: administración del parque móvil vinculado a cada cliente, incluyendo ficha técnica y validaciones de matrícula.
* Recepcionamientos: registro, edición y consulta de entradas de vehículos. Puede marcar urgencias, anotar observaciones o subir documentos.
* Presupuestos: puede revisar, editar o autorizar presupuestos elaborados por mecánicos, y dejar constancia del estado de aceptación por parte del cliente.
* Reimpresión de documentos relacionados con operaciones: recepcionamientos, presupuestos, fichas de vehículos, etc.
* Cierre de sesión estructurado para mantener la trazabilidad documental.

Restricciones:

* No tiene acceso a creación de usuarios ni al módulo de compraventa de vehículos.
* Tampoco puede eliminar registros sensibles.

4.1.4.- ROL COMPRA/VENTA

El rol de Compra/Venta está diseñado para el personal responsable de gestionar operaciones de compra, venta y stock de vehículos. Este perfil tiene acceso a herramientas específicas para tareas comerciales y de seguimiento de operaciones.

Funciones principales:

* Fichar inicio y fin de jornada, y consultar su propio historial.
* Acceso al catálogo de vehículos en venta, con posibilidad de añadir, modificar o retirar modelos del stock.
* Recepcionamientos comerciales, especialmente vinculados a tasaciones, adquisiciones o traspasos de vehículos.
* Gestión de compraventa: registrar operaciones de compra o venta, adjuntar datos del contrato, introducir información del cliente y el vehículo.
* Generación de documentación vinculada a cada operación: contratos, informes de estado, valoraciones, etc.
* Cierre de sesión seguro.

Restricciones:

* Este perfil no puede acceder a módulos técnicos (como intervenciones), ni modificar datos de usuarios o roles.

4.1.5.- Control de permisos y seguridad del sistema

El sistema implementa un control de acceso robusto y dinámico basado en el rol asignado a cada usuario:

* Durante el inicio de sesión, se autentica el nombre del usuario (en mayúsculas) y su contraseña cifrada.
* Una vez autenticado, el sistema carga únicamente los módulos y componentes permitidos para su rol.
* Las interfaces están dinámicamente adaptadas, ocultando menús y botones que no corresponden al perfil activo, reduciendo la posibilidad de error o uso indebido.
* Si un usuario intenta acceder de manera no autorizada a través de rutas internas o manipulaciones externas, el sistema bloquea el acceso y muestra un mensaje de advertencia con registro interno del intento.

Esta segmentación de roles no solo protege los datos y operaciones sensibles, sino que también mejora significativamente la experiencia de usuario, ofreciendo entornos personalizados, limpios y eficaces según la función que desempeña.

Imagen que contiene Texto

El contenido generado por IA puede ser incorrecto.4.2 Diseño de clases y arquitectura de módulos

Ilustración 4 Estructura de carpetas del proyecto PFDG, organizada por módulos como controladores, vistas, modelos, utilidades y pruebas.

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.

4.3 Estructura de módulos

4.3.1.- 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.

4.3.2 Carpeta css/:

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.

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.

4.3.3 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.

4.3.4 Carpeta documentos/:

La carpeta documentos/ constituye el repositorio principal del sistema para la generación y almacenamiento automático de documentación administrativa, generada durante el uso del sistema. Todos los documentos son exportados en formato PDF (y, en algunos casos, acompañados de archivos HTML o CSS para edición o diseño), organizados en subcarpetas por tipo y por mes.

Estructura general:

* /compras → Contratos de compra de vehículos
* /ventas → Contratos de venta de vehículos
* /presupuestos → Presupuestos generados para intervenciones
* /recepcionamientos → Documentos de recepción de vehículos

Cada una de estas carpetas incluye subcarpetas por mes y año (APRIL\_2025, May\_2025, ABRIL\_2025, MAYO\_2025, etc.), lo cual permite una organización cronológica, facilitando la búsqueda y trazabilidad de los archivos.

Descripción de cada subdirectorio:

compras/

* Contiene contratos de compra de vehículos generados por el sistema, nombrados con el patrón: CONTRATO\_COMPRA\_<matrícula>.pdf
  + Ej.: CONTRATO\_COMPRA\_9999ABC.pdf
* Organizados por mes (ej. APRIL\_2025).

ventas/

* Contiene los contratos de venta emitidos por el módulo de compraventa.
* Nombrados como: CONTRATO\_VENTA\_<referencia>.pdf
  + Ej.: CONTRATO\_VENTA\_6709XJT.pdf
* Organización cronológica por mes (APRIL\_2025
* contrato\_venta.css y contrato\_venta\_generado.html: recursos HTML previos a la exportación en PDF.
* firmataller.png, logo.jpg: elementos gráficos reutilizados en el documento final.

presupuestos/

* Archivos PDF generados al realizar presupuestos.
* Nomenclatura: Presupuesto\_<matrícula>\_<fecha>\_<hora>.pdf
  + Ej.: Presupuesto\_4567DEF\_20250503\_1148.pdf
* Organización mensual (May\_2025).

recepcionamientos/

* Documentos que registran la recepción de vehículos en el taller, incluyendo datos del cliente, estado del vehículo y observaciones.
* Nombres tipo: Recepcion\_<NOMBRE\_CLIENTE>\_<ID>.pdf
  + Ej.: Recepcion\_CARLOS\_GOMEZ\_HERNANDEZ\_01002.pdf
* Organización por mes (ABRIL\_2025, MAYO\_2025).

Observaciones técnicas

* La estructura modular y cronológica permite una búsqueda rápida de documentos incluso sin base de datos.
* Los archivos .css y .html permiten una edición previa al renderizado PDF, o pueden ser utilizados en motores de conversión como wkhtmltopdf.
* El uso de nombres estructurados facilita la trazabilidad por matrícula, cliente o fecha.
* Todos los documentos están generados automáticamente por el sistema, sin intervención manual, lo que garantiza consistencia y reducción de errores.

4.3.5 Carpeta font/:

La carpeta font/ contiene las fuentes tipográficas utilizadas por el sistema para garantizar una apariencia moderna, homogénea y visualmente agradable en toda la interfaz gráfica. Estas fuentes se aplican tanto a los títulos como a los textos secundarios, y forman parte esencial del diseño visual definido por los archivos CSS.

Archivos incluidos:

Montserrat-VariableFont\_wght.ttf

Fuente principal en estilo normal (Roman). Es una fuente de tipo variable, lo que permite modificar su peso (grosor) dinámicamente, desde Thin hasta Black, sin necesidad de múltiples archivos estáticos.

Montserrat-Italic-VariableFont\_wght.ttf

Variante itálica de la misma fuente, también en formato variable. Se utiliza especialmente en subtítulos, observaciones y textos decorativos o secundarios.

Justificación del uso

La familia Montserrat fue seleccionada por sus características de legibilidad, elegancia y modernidad, alineadas con la imagen profesional del sistema ReyBoxes. El uso de versiones variable font permite:

Reducir el peso del sistema (solo dos archivos en lugar de muchos por cada peso/estilo).

Mayor flexibilidad visual, ya que se puede controlar con precisión el grosor del texto desde CSS o desde PySide.

Compatibilidad multiplataforma, ya que las fuentes TrueType (.ttf) son soportadas tanto en Windows como en otros sistemas si se requiere expansión futura.

Integración técnica

Las fuentes se cargan desde el código utilizando rutas absolutas, mediante funciones del módulo utilidades.rutas, asegurando que no dependan de rutas estáticas ni del entorno del sistema operativo.

4.3.6 Carpeta img/

La carpeta img/ contiene todos los recursos visuales e iconográficos utilizados por la interfaz del sistema ReyBoxes, centralizando en un único directorio todos los elementos gráficos necesarios para los botones, formularios, paneles y documentos generados por la aplicación.

Esta organización favorece la reutilización de recursos, la coherencia estética y la facilidad de mantenimiento del sistema visual.

Consideraciones técnicas

Rutas absolutas: Todas las imágenes se cargan mediante rutas absolutas calculadas con el módulo utilidades.rutas, asegurando compatibilidad entre sistemas y evitando fallos por rutas relativas.

Formato estándar: Todos los iconos están en formato .png o .jpg para compatibilidad con PySide6. Los GIF permiten animaciones ligeras y sin dependencias externas.

Modularidad visual: La clara segmentación por función y módulo facilita tanto la personalización como el reemplazo de iconos en futuras versiones.

4.3.7 Carpeta lib/:

La carpeta lib/ contiene un entorno aislado de dependencias Python requerido para que la aplicación ReyBoxes funcione de forma autónoma y sin instalación externa de paquetes, ideal para su empaquetado con tecnologías como PyInstaller, cx\_Freeze o incluso para ejecución directa en entornos bloqueados (como Windows corporativos).

Está estructurado como una instalación personalizada de site-packages, que es el directorio donde Python guarda las librerías de terceros.

Librerías principales agrupadas por propósito

1. Gestión de paquetes y entornos

pip: Incluye el gestor de paquetes oficial de Python con todos sus submódulos (install, download, freeze, uninstall, etc.), así como el backend interno (\_internal) y controladores para configuración, resolución de dependencias y entornos virtuales.

setuptools\_build, wheel, pyproject\_hooks, distlib: Herramientas necesarias para construir, empacar e instalar módulos Python desde archivos fuente o binarios (.whl).

platformdirs: Utilidad para detectar rutas estándar de configuración/cache según sistema operativo.

truststore: Utilidad para trabajar con certificados del sistema en lugar de certifi.

2. Red, HTTP y descarga de paquetes

requests: Librería robusta para hacer peticiones HTTP/S con soporte para autenticación, sesiones, cookies, cabeceras personalizadas, etc.

urllib3: Backend que utiliza requests para conexiones HTTP, con manejo avanzado de conexiones, reintentos, proxy, SSL y compatibilidad con múltiples plataformas.

certifi: Conjunto de certificados raíz de confianza usado por requests.

idna: Soporte para dominios internacionales (IDN).

distro, resolvelib, packaging: Utilidades de detección de entorno Linux y de resolución semántica de versiones de paquetes.

3. Utilidades de consola y depuración

rich: Librería de formato para consola, que permite imprimir tablas, barras de progreso, errores estilizados, JSON, Markdown, y más.

pygments: Motor de resaltado de sintaxis (coloreado) para múltiples lenguajes. Se puede usar junto a rich, o para generar documentación con código.

diagnose, pretty, markup, console, traceback, panel, table, progress (submódulos de rich): Proveen interfaz enriquecida para el usuario final y feedback visual en CLI.

4. Compatibilidad y abstracciones

typing\_extensions: Define tipos avanzados para versiones de Python que aún no los integran oficialmente.

pkg\_resources, importlib, entrypoints, metadata: Utilidades para inspeccionar paquetes instalados, sus metadatos y sus puntos de entrada dinámicos.

msgpack: Librería de serialización binaria ultrarrápida (alternativa a JSON, útil para enviar datos compactos).

py.typed: Archivos que indican que las librerías incluyen anotaciones de tipos para herramientas como mypy.

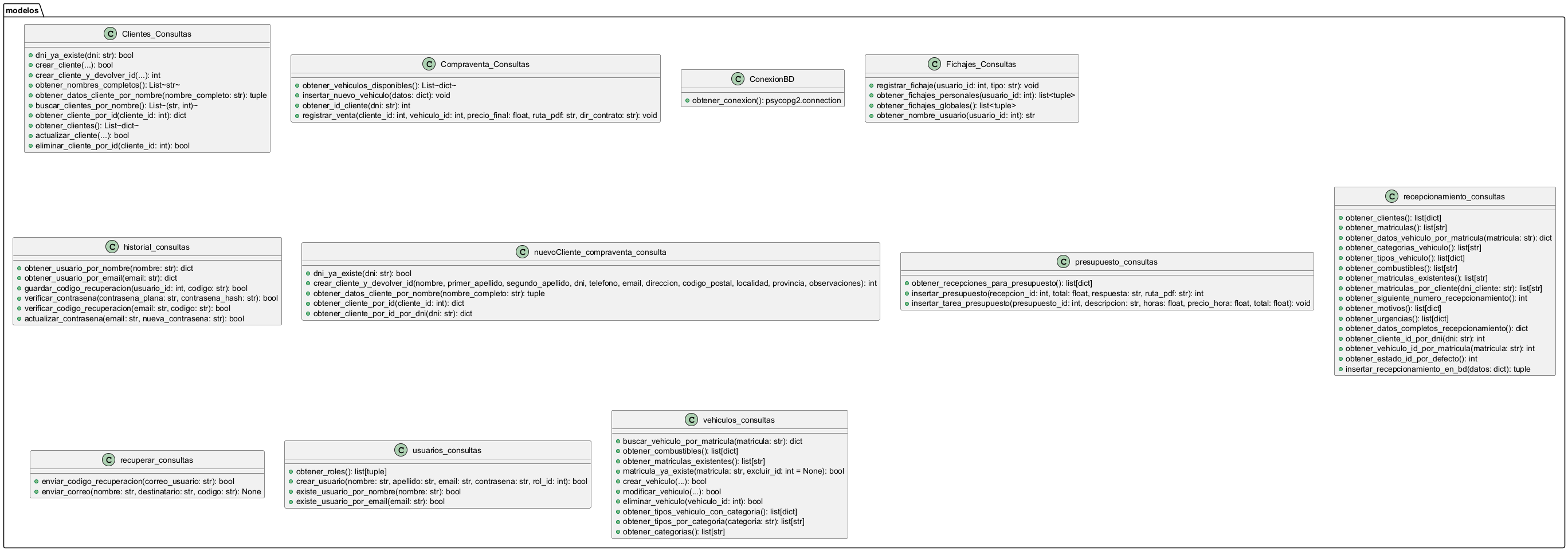
4.3.8.- Carpeta modelos

Ilustración 5 Diagrama de clases del sistema ReyBoxes, mostrando la estructura y relaciones entre los principales módulos del proyecto.

Esta carpeta contiene los módulos encargados de comunicarse con la base de datos PostgreSQL (en Supabase), realizando operaciones CRUD específicas para cada entidad del sistema (usuarios, clientes, fichajes, vehículos, etc.).

Estructura general

Todos los archivos .py siguen el patrón nombreEntidad\_consultas.py, cumpliendo la función de repositorio de consultas SQL o de acceso ORM.

Observaciones adicionales

Todos los módulos .py están correctamente compilados para las versiones de Python 3.11 y 3.13, lo que garantiza compatibilidad cruzada.

La estructura está bien organizada, separando responsabilidades por entidad del dominio del taller mecánico.

Cumple con las buenas prácticas del patrón MVC, donde el modelo se encarga únicamente de interactuar con la base de datos sin lógica de interfaz ni control de ventanas.

4.3.9 Carpeta plantillas/:

Esta carpeta contiene plantillas base reutilizables para generar documentos en formato HTML + CSS que luego se imprimen o exportan como PDF desde la aplicación.

Observaciones:

Estas plantillas están diseñadas para ser rellenadas dinámicamente desde Python o PyQt6 (por ejemplo, con Jinja2 o mediante sustitución de tokens) y luego renderizadas a PDF.

Todas las plantillas están modularizadas: el estilo se encuentra en archivos .css separados, lo cual facilita realizar modificaciones visuales sin alterar la estructura HTML.

La carpeta img se encuentra dentro de plantillas/, lo que permite el uso de rutas relativas funcionales para su integración directa en los documentos HTML.

Uno de los motivos para implementar los informes de esta manera fue la evaluación de los requisitos del proyecto y la necesidad de aplicar los conocimientos adquiridos en el módulo de "Lenguaje de Marcas", evitando la penalización por uso de entorno web. Esta solución representa la opción más beneficiosa y coherente con los objetivos académicos y técnicos del proyecto.

4.3.10 Carpeta tests/

Este directorio contiene scripts de test independientes para cada módulo funcional del sistema. Están escritos en Python y cubren validaciones, flujos de trabajo y generación de documentos.

4.3.11 Carpeta Scripts/

Este directorio contiene los ejecutables relacionados con la gestión de paquetes mediante pip, el instalador oficial de paquetes de Python. Son usados generalmente por entornos virtuales o configuraciones portables del proyecto.

Observaciones

Estos ejecutables permiten gestionar las dependencias desde consola o desde código.

Se encuentran dentro del proyecto para garantizar que las instalaciones se hagan en el entorno Python embebido o aislado del sistema.

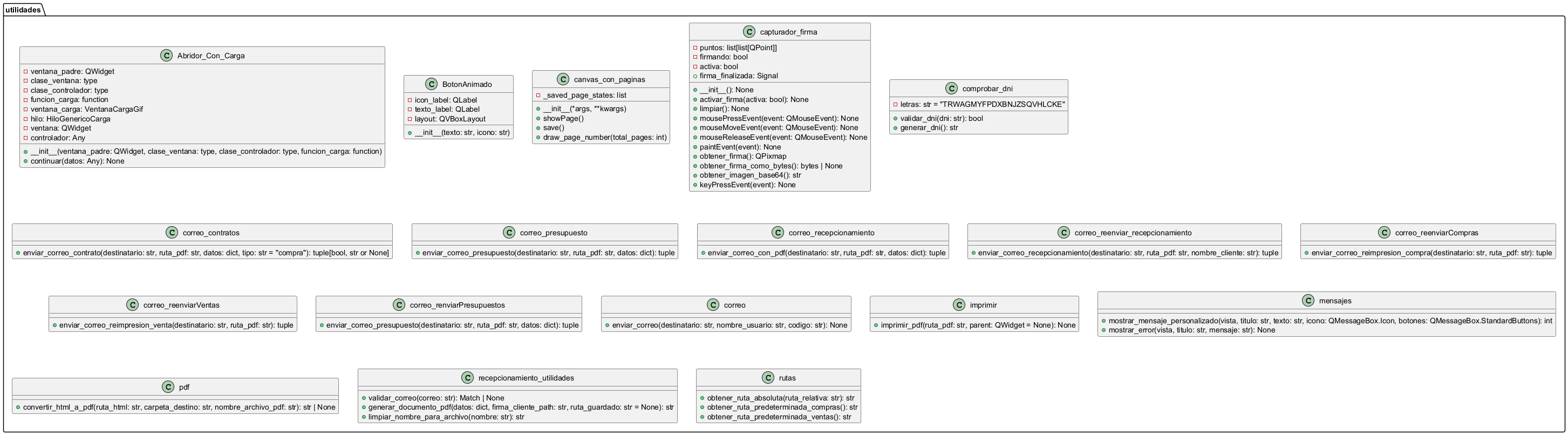
4.3.12 Carpeta utilidades/

Ilustración 6 Diagrama de clases de la lógica de negocio y gestión de datos del sistema ReyBoxes, incluyendo entidades como usuarios, clientes y vehículos.

Este directorio contiene scripts modulares diseñados para encapsular funcionalidades específicas utilizadas en distintas partes del sistema, facilitando la reutilización, el mantenimiento y la legibilidad del código.

Módulos principales

abridor\_con\_carga.py

Abre ventanas nuevas del sistema mostrando primero una pantalla de carga animada (con GIF o barra).

Mejora la experiencia de usuario en vistas que tardan en renderizarse.

Se utiliza, por ejemplo, al entrar a módulos como recepcionamiento.

boton\_animado.py

Contiene una clase para botones con efectos visuales o animaciones al hacer hover o clic.

Compatible con PyQt6 y diseñado para integrarse con los estilos CSS personalizados del sistema.

canvas\_con\_paginas.py

Implementa un sistema de paginación para mostrar datos extensos en un QWidget, como tablas o listas grandes.

Facilita navegar entre páginas sin recargar toda la ventana.

capturador\_firma.py

Proporciona una interfaz para que el usuario dibuje y guarde su firma manualmente.

Guarda la firma en una imagen PNG para usar en presupuestos, contratos, etc.

Puede incluir herramientas como "Limpiar" y "Guardar firma".

comprobar\_dni.py

Valida si un DNI español es correcto (formato y letra).

Incluye también una función para generar DNIs aleatorios válidos para pruebas.

correo.py

Función central para enviar correos electrónicos.

Se usa internamente por otros módulos como correo\_contratos, correo\_presupuesto, etc.

Soporta HTML, adjuntos y direcciones múltiples.

Módulos específicos para el envío de correos

* correo\_contratos.py → Envío del contrato de compra o venta al cliente.
* correo\_presupuesto.py → Envío de presupuestos generados como PDF.
* correo\_recepcionamiento.py → Confirmación del recepcionamiento del vehículo.
* correo\_reenviarCompras.py, correo\_reenviarVentas.py → Reenvío de contratos.
* correo\_reenviar\_recepcionamiento.py, correo\_renviarPresupuestos.py → Reenvío de formularios o presupuestos previamente firmados.

Todos estos módulos se basan en plantillas HTML y el motor correo.py, pero añaden su lógica de selección de documentos, generación de PDF y contenido personalizado.

imprimir.py

Contiene utilidades para previsualizar o enviar a imprimir documentos generados (presupuestos, contratos, fichas de recepción, etc.).

Puede interactuar con sistemas de impresión del SO.

mensajes.py

Define ventanas emergentes o diálogos de aviso como:

* "¿Estás seguro de continuar?"
* "Correo enviado correctamente"
* "Error al conectar con la base de datos"

Mejora la uniformidad y estética de los mensajes del sistema.

pdf.py

Contiene funciones para convertir HTML + CSS en documentos PDF usando motores como WeasyPrint o Qt.

Se utiliza por todos los módulos de impresión y correo electrónico.

recepcionamiento\_utilidades.py

Agrupa funciones utilizadas durante el proceso de recepción del vehículo.

Inclulle formateo de fechas, generación de IDs, validaciones o cálculos auxiliares para los formularios.

rutas.py

Define rutas absolutas dinámicas en función de la ubicación actual del programa.

Evita dependencias de rutas absolutas de sistema operativo.

Incluye rutas a recursos: imágenes, plantillas, bases de datos, PDFs generados, etc.

rutas\_guardado.py

Define rutas de destino para guardar archivos generados (PDFs, firmas, contratos).

Permite que todas las funciones usen rutas coherentes según el tipo de documento.

4.3.13.1 Subdirectorio hilos/

hilo\_generico.py

Clase base que encapsula la lógica para crear hilos (QThread) reutilizables en otras partes del sistema.

Permite ejecutar procesos intensivos o de larga duración sin bloquear la interfaz de usuario, garantizando así una experiencia fluida y profesional.

Es la base para extender múltiples operaciones asincrónicas, como la generación de informes, carga de datos o procesamiento de archivos.

hilo\_carga\_recepcionamiento.py

Hilo especializado en la carga dinámica de datos de recepcionamiento de vehículos directamente desde la base de datos.

Optimiza el rendimiento general del sistema permitiendo que la información se cargue de forma paralela, sin interrumpir la interacción con el usuario.

Este módulo fue clave para aplicar de manera práctica los conocimientos adquiridos en el módulo formativo de “Programación de Servicios y procesos”, ya que representa una implementación real de multitarea eficiente, aplicando los principios de asincronía, concurrencia controlada y diseño orientado a eventos.

Además, su desarrollo supuso un avance importante en la profesionalización del sistema, consolidando un enfoque moderno y robusto en la arquitectura de la aplicación.

Observaciones finales

Este módulo actúa como nucleo de lógica de presentación y automatización del sistema, asegurando:

Reutilización máxima de código.

Separación de lógica por contexto (firma, impresión, email, rutas, hilos...).

Extensibilidad para futuras funcionalidades.

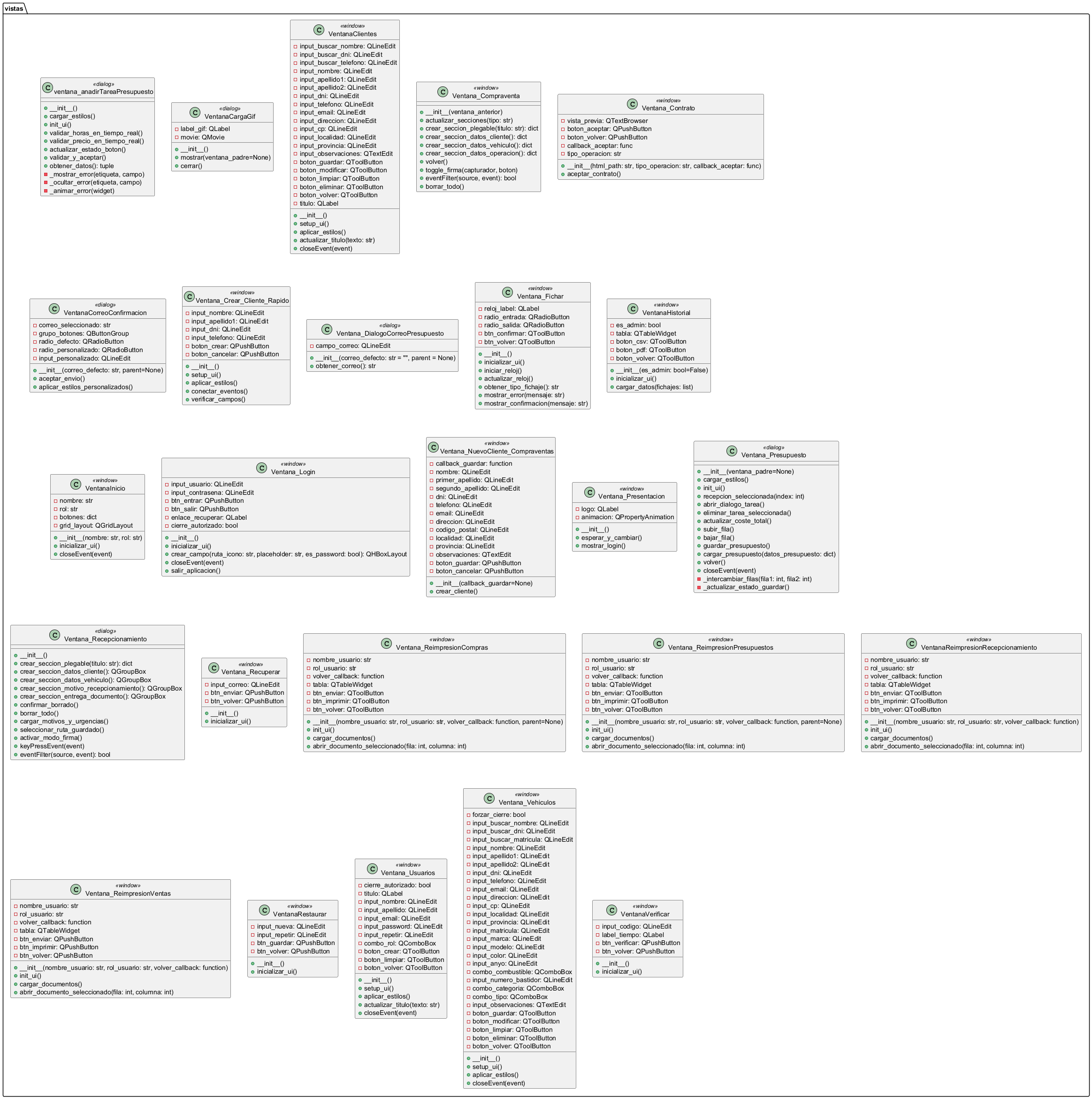
4.3.14 Directorio vistas/

Ilustración 7 Diagrama de clases completo del sistema ReyBoxes, representando todas las entidades, atributos y relaciones entre módulos del proyecto.

Este directorio contiene todas las ventanas que conforman la interfaz gráfica del sistema, desarrolladas en PyQt6. Cada archivo representa una vista específica, siguiendo el patrón MVC implementado a lo largo del proyecto. Su estructura modular permite que cada ventana sea independiente, reutilizable y personalizable.

Características generales:

Todas las clases heredan de QWidget, QDialog o QMainWindow según la funcionalidad de la ventana.

El diseño visual se encuentra reforzado mediante archivos .css personalizados en la carpeta css/, lo que permite un estilo coherente y profesional.

Los nombres de los archivos son altamente descriptivos y están en castellano, facilitando la lectura del código y el mantenimiento futuro.

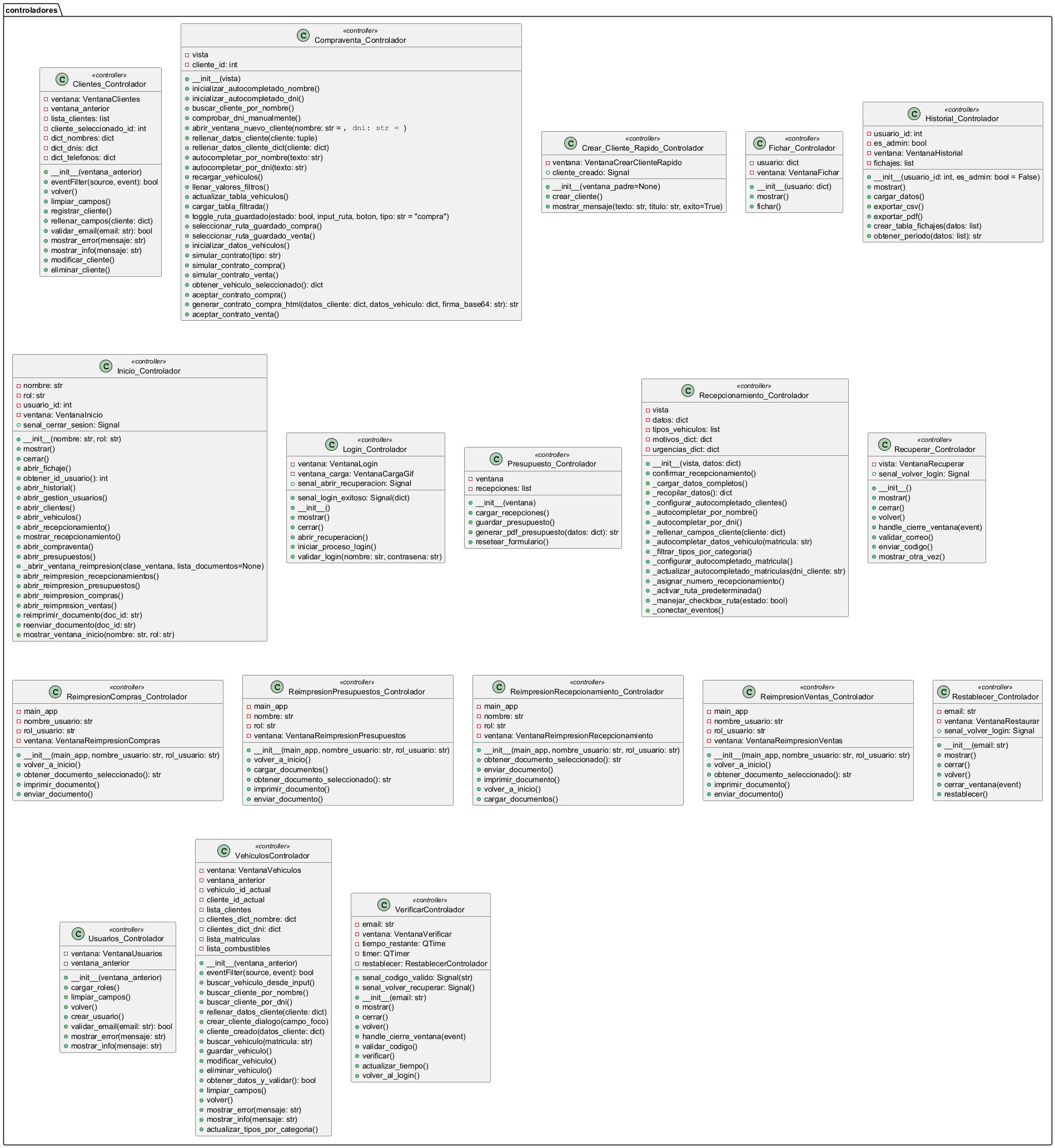
4.3.15 Carpeta controladores/

Ilustración 8 Diagrama de clases del módulo de controladores del sistema ReyBoxes, detallando la lógica de interacción entre vistas y datos.

La carpeta controladores/ contiene todas las clases que gestionan la lógica y eventos del ventana o módulo de la interfaz gráfica, siguiendo el patrón Modelo-Vista-Controlador (MVC).

**Estructura general**

Cada archivo define una clase Controlador asociada a una Vista concreta (ubicada en vistas/). Estas clases se encargan de:

* Conectar los eventos de la interfaz con las acciones correspondientes (botones, campos, etc.).
* Validar y transformar los datos antes de enviarlos al modelo.
* Coordinar el flujo de navegación entre pantallas.
* Invocar a módulos auxiliares como generación de documentos, hilos, correo o impresión.

**Funciones principales**

Los controladores contienen funciones como:

* \_\_init\_\_(): Inicializa el controlador, conecta eventos, configura la vista y prepara los datos necesarios.
* Métodos como guardar, modificar, eliminar, volver, mostrar, cerrar, según el módulo correspondiente.
* Métodos auxiliares para autocompletado, validación de campos, confirmaciones, navegación y comunicación con la base de datos o servicios auxiliares.

**Ejemplos representativos**

* **Login\_Controlador**: Gestiona el proceso de autenticación, la apertura de la recuperación de cuenta y la navegación hacia el dashboard.
* **Inicio\_Controlador**: Controla el menú principal según el rol del usuario y redirige hacia módulos como clientes, fichajes, presupuestos, etc.
* **Clientes\_Controlador**: Gestiona el alta, modificación y eliminación de clientes, con validación de campos, autocompletado y conexión con clientes\_consultas.py.
* **Recepcionamiento\_Controlador**: Carga dinámicamente los datos necesarios, asigna número de recepción, valida formularios y genera documentos PDF con firmas.
* **Presupuesto\_Controlador**: Crea presupuestos a partir de recepciones, permite añadir tareas, calcula totales y genera PDFs.
* **Reimpresion\*Controlador**: Controladores dedicados a reimpresión de contratos, presupuestos o recepciones; incluyen lógica de reenvío por correo o impresión directa.

**Observaciones técnicas**

* Todos los controladores reciben su ventana correspondiente como argumento o la crean internamente.
* Utilizan señales (Signal) para emitir eventos hacia la clase Aplicacion principal y coordinar la navegación.
* Pueden lanzar procesos asíncronos mediante módulos como abridor\_con\_carga.py o hilo\_generico.py para evitar bloqueos en la interfaz.
* Tienen acceso a funciones comunes como mostrar\_error(), mostrar\_mensaje\_personalizado() o imprimir\_pdf() a través de los módulos de utilidades/.

**Importancia en la arquitectura**

La carpeta controladores/ representa el núcleo funcional del sistema. Estas clases:

* Aíslan la lógica de presentación del modelo y de la vista.
* Facilitan el mantenimiento y la escalabilidad del sistema.
* Permiten pruebas unitarias de la lógica sin depender de la interfaz.
* Garantizan una separación de responsabilidades clara y coherente con los principios del diseño modular.

Diagrama, Esquemático

El contenido generado por IA puede ser incorrecto.4.4 Diseño físico de dato

Ilustración 9 Diagrama entidad-relación de la base de datos del sistema ReyBoxes, mostrando las tablas principales y sus relaciones.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.4.4.1 Tabla: fichajes

Ilustración 10 Estructura de la tabla fichajes con ID, usuario, tipo y fecha.

1. Descripción general

La tabla fichajes constituye el núcleo del módulo de control horario del sistema de gestión del taller. Está diseñada para registrar cada entrada y salida de los empleados, permitiendo llevar un control exacto de su jornada laboral.

Cada fichaje queda asociado de forma indivisible a un usuario del sistema, lo cual permite generar informes individualizados, detectar ausencias o anomalías y calcular tiempos de permanencia. Esta tabla también permite extender el sistema hacia la gestión de nóminas o productividad.

La estructura es simple, pero poderosa: cuenta con campos bien definidos y relaciones referenciales sólidas que garantizan la integridad de los datos. Su diseño cumple con los principios de normalización y se adapta a múltiples escenarios laborales.

2. Estructura de la tabla

Tabla 3 Detalle de campos y restricciones de la tabla fichajes en la base de datos.

| Campo | Tipo de dato | Permite nulos | Descripción |
| --- | --- | --- | --- |
| id | int4 | No | Clave primaria. Identificador único del fichaje. Se autoincrementa. |
| usuario\_id | int4 | No | Clave foránea que enlaza con usuarios.id. Indica qué usuario ha fichado. |
| tipo | varchar | No | Tipo de fichaje: 'Entrada' o 'Salida'. Controlado desde la lógica de negocio. |
| fecha\_hora | timestamp | No | Fecha y hora exacta en la que se realizó el fichaje. |

3. Relaciones y claves foráneas

usuario\_id → usuarios.id: Relación directa con la tabla usuarios, estableciendo qué persona realizó el fichaje. Esto permite consultar el historial individual de cada usuario.

4. Restricciones

Clave primaria (PRIMARY KEY)

id: Asegura unicidad del registro. Cada fichaje es irrepetible e inmutable.

Restricciones de obligatoriedad (NOT NULL)

Todos los campos (usuario\_id, tipo, fecha\_hora) son obligatorios para evitar registros incompletos o inconsistentes.

Restricción de integridad referencial (FOREIGN KEY)

usuario\_id debe existir previamente en la tabla usuarios. Esto impide fichajes de usuarios no registrados o eliminados.

Restricción lógica (desde aplicación)

El campo tipo acepta únicamente los valores 'Entrada' o 'Salida', aunque esta lógica puede implementarse desde el backend en lugar de en la base de datos.

5. Consideraciones adicionales

* Los datos almacenados en esta tabla permiten calcular con facilidad los intervalos entre fichajes de entrada y salida, facilitando así el control horario del personal.
* Se prevé, en futuras ampliaciones del sistema, la integración con sistemas de identificación avanzados como biometría, tarjetas NFC o códigos QR, mejorando la seguridad y comodidad del proceso de fichaje.
* El campo fecha\_hora permite un registro temporal preciso, esencial para llevar una trazabilidad exhaustiva y garantizar auditorías fiables.
* El diseño de la tabla permite escalabilidad, pudiendo incorporar en el futuro nuevos campos como la ubicación del fichaje, el dispositivo utilizado o comentarios opcionales que contextualicen la acción registrada.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.4.4.2 Tabla: usuarios

Ilustración 11 Estructura de la tabla usuarios con datos personales, credenciales y rol.

1. Descripción general

La tabla usuarios constituye una de las entidades centrales del sistema. Está diseñada para almacenar los datos de las personas autorizadas a acceder al software, así como su rol, credenciales y metadatos relacionados con el acceso. Es fundamental para el control de acceso, la trazabilidad y la personalización del sistema según el perfil del usuario.

La tabla está normalizada y conectada con otras entidades a través de claves foráneas, garantizando así integridad referencial. Asimismo, incluye mecanismos de seguridad como el cifrado de contraseñas y la gestión segura de recuperación de cuentas.

2. Estructura de la tabla

Tabla 4 Descripción de los campos principales de la tabla usuarios, con restricciones y uso en el sistema.

| Campo | Tipo de dato | Permite nulos | Descripción |
| --- | --- | --- | --- |
| id | int4 | No | Clave primaria, identificador único del usuario. Se autoincrementa. |
| nombre | varchar | No | Nombre de usuario (en mayúsculas), usado en el login. Unicidad garantizada. |
| apellido | varchar | No | Apellido(s) del usuario. |
| email | varchar | No | Correo electrónico único. Utilizado para notificaciones y recuperación de cuenta. |
| password | text | No | Contraseña cifrada con bcrypt. No se almacena en texto plano. |
| created\_at | timestamp | Sí | Fecha de creación del usuario. Útil para trazabilidad y auditorías. |
| updated\_at | timestamp | Sí | Fecha de la última modificación del registro. |
| codigo\_recuperacion | varchar | Sí | Código OTP generado para recuperación de cuenta. |
| expiracion\_codigo | timestamp | Sí | Fecha de expiración del código de recuperación. |
| rol\_id | int4 | No | Clave foránea que enlaza con roles.id. Determina los permisos del usuario. |

3. Relaciones y claves foráneas

rol\_id → roles.id: Esta relación impone una restricción referencial que garantiza que todos los usuarios tengan un rol válido asignado, ya sea “Administrador”, “Mecánico” , “Administrativo” o “Compraventa”. Esta relación es esencial para controlar los permisos y accesos en el sistema.

4. Restricciones

Clave primaria (PRIMARY KEY):

El campo id asegura la unicidad y sirve como referencia desde otras tablas como fichajes, intervenciones, etc.

Restricciones de unicidad (UNIQUE):

Tanto el campo nombre como el email están marcados como únicos. Esto evita duplicidades, mejora la trazabilidad de usuarios y previene conflictos en el inicio de sesión o en envíos de notificaciones.

Restricciones de obligatoriedad (NOT NULL):

Los campos nombre, apellido, email, password y rol\_id son obligatorios para garantizar la coherencia y completitud de los datos al registrar un usuario.

Restricciones de integridad referencial (FOREIGN KEY):

El campo rol\_id garantiza que sólo se puedan asignar roles definidos previamente en la tabla roles.

5. Consideraciones adicionales

El diseño de esta tabla permite implementar un sistema de recuperación de contraseña sin comprometer la seguridad del usuario, almacenando códigos temporales de manera aislada y con fecha de expiración.

Gracias a los campos created\_at y updated\_at, es posible realizar trazabilidad de cambios, útil para fines administrativos o auditoría.

Los nombres se guardan en mayúsculas para evitar inconsistencias al comparar cadenas en procesos como login o exportación de datos.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.4.4.3 Tabla: roles

Ilustración 12 Estructura de la tabla roles, con identificador y nombre del rol asignado.

1. Descripción general

La tabla roles actúa como tabla de referencia para los distintos perfiles de acceso que pueden ser asignados a los usuarios del sistema. Es una tabla auxiliar que permite normalizar la asignación de permisos y mantener la escalabilidad del sistema de autenticación y autorización. Su diseño sigue principios de normalización y establece una relación directa con la tabla usuarios mediante clave foránea.

2. Estructura de la tabla

Tabla 5 Detalle de campos de la tabla roles, con identificador y nombre único por perfil de usuario.

| Campo | Tipo de dato | Permite nulos | Descripción técnica |
| --- | --- | --- | --- |
| id | int4 | No | Clave primaria. Identificador único del rol. Se autoincrementa. |
| nombre | varchar | No | Nombre del rol. Valor textual único ('Administrador', 'Mecánico', 'Compraventa', 'Administrativo'). |

3. Relaciones y claves foráneas

Esta tabla no contiene claves foráneas, pero sí es referenciada por otras tablas:

usuarios.rol\_id → roles.id

Relación de uno a muchos que permite que varios usuarios compartan un mismo rol definido aquí.

4. Restricciones

Clave primaria (PRIMARY KEY)

id: garantiza la unicidad de cada rol y sirve como identificador referenciado por otras tablas.

Restricciones de unicidad (UNIQUE)

nombre: impide la creación de múltiples filas con el mismo nombre de rol. Esto evita ambigüedad en la asignación de permisos.

Restricciones de obligatoriedad (NOT NULL)

Ambos campos (id, nombre) son obligatorios para asegurar que cada rol esté correctamente identificado y descrito.

5. Consideraciones adicionales de diseño

La tabla está pensada para contener un número limitado y estable de registros, por lo que es ideal para usar en menús desplegables, validaciones y controles de interfaz.

Se puede extender con otros campos si se desea una gestión más avanzada de permisos, como por ejemplo: descripcion, nivel\_acceso, o activo.

El uso de una tabla separada para roles permite mantener la integridad y coherencia de los datos cuando se reutilizan en múltiples puntos del sistema.

La centralización de los roles facilita futuras modificaciones o auditorías sin necesidad de alterar la tabla principal de usuarios.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.4.4.4 Tabla: clientes

Ilustración 13 Estructura de la tabla clientes, con datos personales y de contacto completos.

1. Descripción general

La tabla clientes representa la entidad que almacena los datos personales y de contacto de cada cliente registrado en el sistema. Esta información es esencial para llevar un control organizado y profesional de los clientes que utilizan los servicios del taller. La tabla está diseñada siguiendo las buenas prácticas de normalización, separando claramente la información de contacto, los datos de identificación y la metadata del registro.

2. Estructura de la tabla

Tabla 6 Detalle de los campos de la tabla clientes, incluyendo identificadores, contacto y localización.

| Campo | Tipo de dato | Permite nulos | Descripción técnica |
| --- | --- | --- | --- |
| id | int4 | No | Clave primaria. Identificador único del cliente. Se autoincrementa. |
| nombre | varchar | No | Nombre del cliente. Escrito en mayúsculas para homogeneidad en consultas. |
| primer\_apellido | varchar | No | Primer apellido del cliente. |
| segundo\_apellido | varchar | Sí | Segundo apellido del cliente, si lo tiene. |
| dni | varchar | No | Documento Nacional de Identidad. Se valida y es único en el sistema. |
| telefono | varchar | No | Número de teléfono de contacto. |
| email | varchar | Sí | Correo electrónico del cliente, si lo proporciona. |
| direccion | text | Sí | Dirección física completa del cliente. |
| codigo\_postal | varchar | Sí | Código postal correspondiente a la dirección. |
| localidad | varchar | Sí | Ciudad o municipio del cliente. |
| provincia | varchar | Sí | Provincia asociada al domicilio. |
| observaciones | text | Sí | Comentarios adicionales, notas internas o información complementaria. |
| created\_at | timestamp | Sí | Fecha de creación del registro. |
| updated\_at | timestamp | Sí | Fecha de última modificación de datos. |

3. Relaciones con otras tablas

vehiculos.cliente\_id → clientes.id

Relación uno-a-muchos: un cliente puede tener registrados múltiples vehículos.

intervenciones.cliente\_id → clientes.id (si existe esta relación directa)

Permite asociar directamente intervenciones realizadas al cliente, además de su vehículo.

4. Restricciones

Clave primaria (PRIMARY KEY)

id: Identificador único autoincremental que permite referenciar al cliente desde otras tablas como vehiculos o intervenciones.

Restricciones de unicidad (UNIQUE)

Aunque no es forzada a nivel de esquema, el dni es gestionado desde la lógica de aplicación como único para evitar duplicidades de clientes.

Restricciones de obligatoriedad (NOT NULL)

Los campos nombre, primer\_apellido, dni y telefono son obligatorios para asegurar la trazabilidad y el contacto con el cliente.

5. Consideraciones adicionales de diseño

Esta tabla forma parte del núcleo funcional de la base de datos del sistema, y sirve como punto de entrada para otras operaciones relacionadas con vehículos, mantenimientos, presupuestos, ventas y recepcionamientos.

El diseño contempla campos opcionales como email, direccion u observaciones, lo que permite adaptarse tanto a fichas básicas como a registros más detallados.

Los campos created\_at y updated\_at aportan trazabilidad temporal, permitiendo auditar cuándo se registró o modificó por última vez un cliente.

La validación de formato del dni se realiza a nivel de aplicación mediante expresiones regulares y control de letra.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.4.4.5 Tabla: intervenciones

Ilustración 14 Tabla 7 Estructura de la tabla intervenciones, con datos de cliente, vehículo, estado, fechas y costes.

1. Descripción general

La tabla intervenciones recoge toda la información relacionada con las reparaciones, mantenimientos u operaciones realizadas sobre los vehículos registrados en el sistema. Representa una entidad central dentro del módulo técnico del taller, ya que agrupa tanto los aspectos técnicos como administrativos de cada intervención: cliente, vehículo, tipo de intervención, estado, fechas clave, costes asociados y observaciones.

Está diseñada de forma relacional, conectando con otras tablas mediante claves foráneas, lo que permite mantener integridad referencial, evitar redundancias y facilitar la consulta cruzada de información para informes y seguimiento.

2. Estructura de la tabla

Tabla 7 Campos clave de la tabla intervenciones, con referencias a cliente, vehículo, tipo y estado.

| Campo | Tipo de dato | Nulos | Descripción |
| --- | --- | --- | --- |
| id | int4 | No | Clave primaria, identificador único de la intervención. |
| cliente\_id | int4 | No | Clave foránea que referencia al cliente que solicita la intervención (clientes.id). |
| vehiculo\_id | int4 | No | Clave foránea que identifica el vehículo al que se le realiza la intervención (vehiculos.id). |
| tipo\_id | int4 | No | Clave foránea hacia la tabla tipos\_intervencion, que clasifica la naturaleza del trabajo. |
| estado\_id | int4 | No | Clave foránea que indica el estado actual de la intervención (estados\_intervencion.id). |
| descripcion | text | No | Descripción detallada de los trabajos a realizar. Puede incluir diagnóstico, tareas previstas o comentarios del cliente. |
| fecha\_inicio | date | No | Fecha en la que se inicia la intervención. |
| fecha\_finalizacion | date | Sí | Fecha en la que finaliza la intervención (puede estar vacía si aún está en curso). |
| descuento\_aplicado | int4 | Sí | Porcentaje de descuento aplicado (si lo hay), expresado como número entero. |
| coste\_presupuestado | numeric | Sí | Importe estimado al inicio de la intervención. |
| coste\_real | numeric | Sí | Importe real al finalizar los trabajos. |
| observaciones | text | Sí | Comentarios adicionales, incidencias, sugerencias o detalles relevantes. |
| created\_at | timestamp | Sí | Fecha de creación del registro para trazabilidad o informes históricos. |

3. Relaciones y claves foráneas

cliente\_id → clientes.id: Relaciona cada intervención con su cliente correspondiente.

vehiculo\_id → vehiculos.id: Enlaza la intervención con el vehículo intervenido.

tipo\_id → tipos\_intervencion.id: Clasifica el tipo de intervención: revisión, reparación, cambio de aceite, etc.

estado\_id → estados\_intervencion.id: Indica el estado actual: pendiente, en curso, finalizada, cancelada, etc.

Estas relaciones garantizan integridad referencial y permiten generar consultas complejas, como el historial completo de intervenciones por vehículo, cliente o estado.

4. Restricciones

Clave primaria (PRIMARY KEY)

id: Garantiza la unicidad de cada intervención.

Restricciones de integridad referencial (FOREIGN KEY)

Todas las claves foráneas (cliente\_id, vehiculo\_id, tipo\_id, estado\_id) aseguran que solo puedan registrarse intervenciones válidas asociadas a registros existentes en otras tablas.

Restricciones de obligatoriedad (NOT NULL)

Campos esenciales como cliente\_id, vehiculo\_id, tipo\_id, estado\_id, descripcion y fecha\_inicio no pueden estar vacíos, garantizando la completitud mínima del registro.

Coherencia de datos opcionales

Los campos fecha\_finalizacion, coste\_real y descuento\_aplicado pueden completarse al finalizar la intervención, lo que permite gestionar tanto intervenciones en curso como terminadas.

5. Consideraciones de diseño

Flexibilidad: Los campos opcionales permiten reflejar tanto procesos iniciados como terminados sin romper la estructura.

Escalabilidad: La tabla está preparada para adaptarse a nuevos estados o tipos de intervención, gracias a su diseño normalizado.

Trazabilidad: El campo created\_at permite hacer un seguimiento histórico de la actividad del taller.

Gestión de costes: Al separar el presupuesto y el coste real, se facilita el análisis de desviaciones económicas.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.4.4.6 Tabla: vehículos

Ilustración 15 Estructura de la tabla vehiculos, con información técnica, cliente asociado y datos de identificación.

1. Descripción general

La tabla vehículos representa una de las entidades clave del sistema de gestión del taller, encargada de almacenar la información técnica y administrativa de todos los vehículos registrados por los clientes. Su diseño relacional permite mantener un vínculo directo entre cada vehículo y su propietario, así como clasificarlo por tipo y tipo de combustible, de forma normalizada.

Gracias a su estructura extensible y modular, permite soportar una gran variedad de casos (turismos, motos, vehículos industriales, eléctricos, etc.), proporcionando así una base sólida para el módulo de intervenciones y mantenimientos.

2. Estructura de la tabla

Tabla 8 Campos clave de la tabla vehiculos, vinculando cada vehículo con un cliente del sistema.

| Campo | Tipo de dato | Nulos | Descripción |
| --- | --- | --- | --- |
| id | int4 | No | Clave primaria. Identificador único del vehículo en el sistema. |
| cliente\_id | int4 | No | Clave foránea que vincula el vehículo con un cliente existente (clientes.id). |
| marca | varchar | Sí | Marca comercial del vehículo (ej. Peugeot, Yamaha, Mercedes). |
| modelo | varchar | Sí | Modelo específico del vehículo. |
| matricula | varchar | No | Matrícula del vehículo. Campo único que evita duplicidades en el parque registrado. |
| tipo\_vehiculo | int4 | Sí | Clave foránea hacia la tabla tipos\_vehiculo, que clasifica el vehículo por su naturaleza (turismo, moto, industrial, etc.). |
| anyo | int4 | Sí | Año de fabricación o matriculación. Útil para historial y antigüedad. |
| observaciones | text | Sí | Comentarios libres sobre el vehículo: modificaciones, incidencias o peculiaridades. |
| created\_at | timestamp | Sí | Fecha de alta del vehículo en el sistema. |
| color | varchar | Sí | Color principal del vehículo. |
| combustible | varchar | Sí | Tipo de combustible (texto libre, para compatibilidad hacia atrás). |
| numero\_bastidor | varchar | No | Número de bastidor (VIN). Identificador único a nivel de fabricante. Se fuerza como único. |
| combustible\_id | int4 | Sí | Clave foránea hacia una tabla tipos\_combustible, si se opta por normalizar en el futuro. |
| updated\_at | timestamp | Sí | Fecha de la última modificación del registro. |

3. Relaciones y claves foráneas

cliente\_id → clientes.id: Establece la relación de pertenencia entre vehículo y cliente.

tipo\_vehiculo → tipos\_vehiculo.id: Clasifica el vehículo dentro de un conjunto predefinido de tipos (ej. turismo, furgón, agrícola...).

combustible\_id → tipos\_combustible.id: Relación pensada para mejorar la gestión de combustibles.

4. Restricciones

Clave primaria (PRIMARY KEY)

id: Identificador único y obligatorio de cada vehículo registrado.

Clave única (UNIQUE)

matricula y numero\_bastidor están definidos como únicos, ya que no pueden repetirse legalmente en ningún contexto. Esto asegura la integridad del parque registrado y evita errores humanos.

Restricciones de integridad referencial (FOREIGN KEYS)

cliente\_id: Requiere que el cliente exista en la tabla clientes.

tipo\_vehiculo: (opcional) Debe coincidir con un valor en tipos\_vehiculo.

combustible\_id: (opcional) Pensado para futuras ampliaciones mediante relación con tabla combustibles.

Restricciones NOT NULL

Los campos cliente\_id, matricula y numero\_bastidor son obligatorios. Esto garantiza la trazabilidad y la unicidad de cada vehículo en el sistema.

5. Consideraciones de diseño

Normalización y escalabilidad: El uso de claves foráneas permite añadir nuevos tipos de vehículo y combustibles sin alterar la estructura base.

Control de duplicados: Las restricciones únicas sobre matrícula y bastidor aseguran que no se registren vehículos duplicados.

Flexibilidad de captura: La existencia de campos opcionales como color, observaciones o combustible textual permite registrar incluso vehículos con datos incompletos inicialmente.

Preparación para auditorías: Campos como created\_at y updated\_at permiten llevar un historial temporal del parque de vehículos gestionado.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.4.4.7 Tabla: combustibles

Ilustración 16 Estructura de la tabla combustibles, con identificador y tipo de combustible.

1. Descripción general

La tabla combustibles actúa como una entidad de referencia (tabla de catálogo) destinada a normalizar los distintos tipos de combustible que pueden utilizar los vehículos registrados en el sistema. Su objetivo es evitar inconsistencias semánticas o errores de entrada libre (por ejemplo: "Gasolina", "gasolina", "GASOLINA"), garantizando así la integridad y estandarización de los datos relativos al tipo de alimentación de los vehículos.

Es utilizada principalmente como clave foránea en la tabla vehiculos, aportando estructura y flexibilidad al sistema de gestión de taller, permitiendo su extensión a nuevas fuentes de energía como eléctricos, híbridos o hidrógeno.

2. Estructura de la tabla

Tabla 9 Detalle de campos de la tabla combustibles, con identificador y nombre normalizado del tipo de combustible.

| Campo | Tipo de dato | Nulos | Descripción |
| --- | --- | --- | --- |
| id | int4 | No | Clave primaria. Identificador único del tipo de combustible. |
| nombre | varchar | No | Nombre del combustible. Debe ser único y estar normalizado (por ejemplo: "Gasolina", "Diesel", "Eléctrico", "GLP"). |

3. Relaciones y claves foráneas

Relación con vehiculos:

El campo combustible\_id en la tabla vehiculos referencia a combustibles.id. Esta relación permite que cada vehículo esté clasificado según un tipo de combustible definido, mejorando la gestión y los filtros dentro del sistema.

4. Restricciones

Clave primaria (PRIMARY KEY)

id: Asegura que cada tipo de combustible tenga un identificador único, esencial para su uso como clave foránea.

Restricción de unicidad (UNIQUE)

nombre: Este campo debe ser único para evitar duplicidades semánticas y garantizar una gestión coherente de los datos.

Restricción NOT NULL

Ambos campos (id, nombre) son obligatorios, lo que impide la creación de registros incompletos o erróneos.

5. Consideraciones de diseño

Escalabilidad: Permite añadir fácilmente nuevos tipos de combustible sin modificar la estructura de otras tablas.

Integridad semántica: El uso de nombres únicos y normalizados evita errores de escritura o ambigüedad durante la entrada de datos.

Desempeño: Al usar una relación relacional con vehiculos, se optimiza el almacenamiento y se mejora el rendimiento en filtros o informes por tipo de combustible.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.4.4.8 Tabla: tipos\_vehiculo

Ilustración 17 Estructura de la tabla tipos\_vehiculo, con identificador, categoría y nombre del tipo de vehículo.

1. Descripción general

La tabla tipos\_vehiculo es una tabla de referencia destinada a clasificar y normalizar los distintos tipos de vehículos que pueden ser gestionados dentro del sistema. Su diseño permite distinguir entre múltiples categorías de vehículos (por ejemplo, “Terrestres”, “Acuáticos”, “Industriales”) y detallar cada tipo dentro de dichas categorías (como “Turismo”, “Motocicleta”, “Camión”, “Barco”).

Esta estructura facilita la segmentación, filtrado y posterior análisis de los vehículos registrados, permitiendo una gestión más precisa y flexible según la naturaleza del parque automotor.

2. Estructura de la tabla

Tabla 10 Detalle de campos de la tabla tipos\_vehiculo, que clasifica los vehículos por categoría y nombre específico.

| Campo | Tipo de dato | Nulos | Descripción |
| --- | --- | --- | --- |
| id | int4 | No | Clave primaria. Identificador único del tipo de vehículo. |
| categoria | varchar | No | Agrupación general del tipo de vehículo. Ej.: “Terrestre”, “Acuático”, “Especial”. |
| nombre | varchar | No | Nombre específico del tipo de vehículo. Ej.: “Furgoneta”, “SUV”, “Camión Grúa”. |

3. Relaciones y claves foráneas

Relación con vehiculos:

El campo tipo\_vehiculo en la tabla vehiculos referencia a tipos\_vehiculo.id, estableciendo una correspondencia entre cada vehículo y su tipo declarado. Esto garantiza la validez del tipo asignado y facilita su uso en filtros y reportes.

4. Restricciones

Clave primaria (PRIMARY KEY)

id: Identificador único por cada tipo de vehículo.

Restricción de unicidad (UNIQUE)

Comúnmente aplicada en combinación de categoria + nombre, para evitar duplicidades de nomenclatura.

Restricción NOT NULL

Todos los campos son obligatorios, garantizando que ningún registro quede incompleto ni ambiguo.

5. Consideraciones de diseño

Extensibilidad: Permite incorporar fácilmente nuevos tipos y categorías a medida que evolucionan los servicios del taller (vehículos eléctricos, maquinaria pesada, embarcaciones, etc.).

Estandarización: Al separar tipo y categoría, se facilita la normalización y mejora la experiencia de búsqueda y visualización.

Optimización: Usar esta tabla como referencia evita redundancias en la tabla vehiculos y mejora la integridad de los datos.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.4.4.9 Tabla: recepcionamientos

Ilustración 18 Estructura de la tabla recepcionamientos, con datos del vehículo, cliente, estado, seguro y observaciones del ingreso.

1. Descripción general

La tabla recepcionamientos almacena todos los datos relacionados con la entrada de vehículos al taller para su inspección, diagnóstico y posterior reparación. Representa uno de los elementos más complejos del sistema debido a la gran cantidad de información que se puede registrar durante el proceso de recepción. Esta tabla permite recoger tanto aspectos técnicos del estado del vehículo como preferencias del cliente y condiciones del servicio.

Está diseñada para capturar el contexto completo de cada ingreso al taller, desde si el vehículo llegó con grúa hasta si se desea un presupuesto previo o si tiene seguro. Toda esta información queda vinculada a otras entidades clave como cliente, vehículo, usuario que atiende, urgencia, estado actual y motivo de entrada.

2. Estructura de la tabla

Tabla 11 Campos de la tabla recepcionamientos, con claves foráneas y datos del estado del vehículo al ingreso.

| Campo | Tipo de dato | Nulos | Descripción |
| --- | --- | --- | --- |
| id | int4 | No | Clave primaria. Identificador único del registro de recepción. |
| cliente\_id | int4 | No | Clave foránea que enlaza con la tabla clientes. |
| vehiculo\_id | int4 | No | Clave foránea que enlaza con vehiculos. Identifica el vehículo que se recibe. |
| usuario\_id | int4 | No | Clave foránea que indica qué usuario registró la recepción. |
| estado\_id | int4 | No | Clave foránea a la tabla de estados del proceso de recepción. |
| fecha | timestamp | Sí | Fecha y hora en la que se realizó la recepción. |
| arranca | bool | Sí | Indica si el vehículo arranca por sí solo. |
| viene\_con\_grua | bool | Sí | Señala si el vehículo ha llegado remolcado. |
| tiene\_seguro | bool | Sí | Indica si el vehículo cuenta con un seguro en vigor. |
| compania\_seguro | varchar | Sí | Nombre de la aseguradora, si aplica. |
| acepta\_reparacion\_hasta | numeric | Sí | Importe máximo autorizado por el cliente sin necesidad de aprobación previa. |
| valor\_estimado | numeric | Sí | Estimación inicial del coste de la reparación. |
| itv\_en\_vigor | bool | Sí | Señala si la ITV está vigente. |
| ultima\_revision | date | Sí | Fecha de la última revisión realizada. |
| desea\_presupuesto\_aprobado | bool | Sí | Indica si el cliente desea recibir un presupuesto antes de autorizar reparaciones. |
| lista\_averias\_cliente | text | Sí | Detalle textual de los problemas descritos por el cliente. |
| estado\_exterior | text | Sí | Observaciones sobre el estado visual exterior del vehículo. |
| estado\_interior | text | Sí | Observaciones sobre el estado del interior. |
| observaciones\_generales | Text | Sí | Notas adicionales generales de la recepción. |
| ruta\_documento | text | No | Ruta del PDF generado para la hoja de recepción. Se requiere para trazabilidad. |
| entregar\_impreso | bool | Sí | Indica si se ha solicitado la entrega en papel. |
| enviar\_por\_correo | bool | Sí | Indica si se ha solicitado el envío por email. |
| num\_recepcionamiento | int4 | Sí | Número correlativo asignado a la recepción. Puede ser usado como identificador visible. |
| urgencia\_id | int4 | No | Clave foránea que indica el nivel de urgencia (alta, media, baja). |
| motivo\_id | int4 | No | Clave foránea al catálogo de motivos de entrada al taller. |

3. Relaciones y claves foráneas

La tabla recepcionamientos establece múltiples relaciones con otras entidades del sistema. Estas relaciones garantizan la integridad referencial de los datos y permiten consolidar un modelo relacional coherente y extensible. A continuación, se describen las claves foráneas incluidas:

cliente\_id → clientes(id)

Establece la relación entre la recepción y el cliente al que pertenece el vehículo. Cada recepción está asociada obligatoriamente a un cliente registrado en el sistema. Esta relación permite vincular los datos personales del cliente con el proceso de entrada del vehículo.

vehiculo\_id → vehiculos(id)

Vincula el registro de recepción con un vehículo específico. Es indispensable para conocer qué unidad fue ingresada al taller y poder recuperar su historial de reparaciones o fichas técnicas. Además, al estar vehiculos relacionado con clientes, se refuerza la trazabilidad completa.

usuario\_id → usuarios(id)

Indica qué miembro del personal realizó el ingreso del vehículo en el sistema. Esta relación permite registrar la autoría del procedimiento, lo cual es crucial para auditorías, asignación de responsabilidades y generación de informes por operario.

estado\_id → estados\_intervencion(id)

Representa el estado actual del proceso de recepción o reparación asociado al vehículo. Puede tomar valores como “Pendiente”, “En reparación”, “Finalizado”, etc. Su uso permite llevar un seguimiento del ciclo de vida de cada recepción.

urgencia\_id → urgencias(id)

Define el nivel de prioridad con el que debe ser tratado el caso (por ejemplo: Alta, Media o Baja). Esta información puede ser utilizada tanto a nivel operativo (planificación) como analítico (estadísticas de atención).

motivo\_id → motivos(id)

Especifica el motivo por el cual el vehículo ha sido traído al taller. Puede incluir valores como “Mantenimiento preventivo”, “Avería grave”, “Revisión periódica”, entre otros. Esta relación permite categorizar las entradas y realizar análisis posteriores por tipo de servicio.

Estas claves foráneas garantizan que cada recepción esté correctamente vinculada a datos válidos en las demás tablas del sistema, permitiendo una navegación fluida entre entidades relacionadas y evitando registros huérfanos o inconsistentes.

4. Restricciones

Clave primaria: id

NOT NULL: en los campos clave como cliente\_id, vehiculo\_id, usuario\_id, estado\_id, urgencia\_id, motivo\_id, y ruta\_documento, para garantizar un registro completo.

FOREIGN KEYS: protegen la consistencia de los datos y evitan asignar referencias a entidades inexistentes.

Tipos booleanos: numerosos campos usan booleanos (bool) para respuestas binarias que ayudan a simplificar validaciones y visualización.

5. Consideraciones de diseño

Alta granularidad: esta tabla recoge una gran cantidad de atributos opcionales que ofrecen una visión completa del estado del vehículo y de las preferencias del cliente.

Documentación generada: gracias al campo ruta\_documento, se almacena la referencia al documento generado, permitiendo su recuperación o reimpresión.

Escalabilidad: admite la inclusión futura de nuevos campos o ajustes sin necesidad de reestructurar su lógica central.

Trazabilidad: al incluir referencias cruzadas con todos los elementos implicados, permite auditorías y análisis detallados.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.4.4.10 Tabla: presupuestos

Ilustración 19 Estructura de la tabla presupuestos, con datos vinculados a la recepción, estado y respuesta del cliente.

1. Descripción general

La tabla presupuestos está diseñada para almacenar las estimaciones económicas asociadas a cada recepción de vehículo en el taller. Representa un documento intermedio entre la recepción y la intervención, permitiendo al cliente conocer el coste aproximado de los trabajos sugeridos antes de su ejecución.

Su estructura contempla tanto los datos básicos del presupuesto como las opciones de autorización, comunicación con el cliente y almacenamiento del archivo PDF generado. Está íntimamente ligada a la tabla recepcionamientos, de la cual hereda su contexto técnico y logístico.

2. Estructura de la tabla

Tabla 12 Detalle de campos de la tabla presupuestos, con información de creación, estado y seguimiento del cliente.

| Campo | Tipo de dato | Nulos | Descripción |
| --- | --- | --- | --- |
| id | int4 | No | Clave primaria. Identificador único del presupuesto. Autoincremental. |
| recepcion\_id | int4 | No | Clave foránea que enlaza con recepcionamientos(id). Define el origen del presupuesto. |
| fecha\_creacion | timestamp | No | Fecha en la que se generó el presupuesto. Útil para trazabilidad. |
| total\_estimado | numeric | No | Valor monetario estimado para la intervención propuesta. |
| autorizado\_directo | bool | No | Indica si el cliente autorizó el presupuesto en el momento de la recepción. |
| cliente\_avisado | bool | No | Determina si el cliente ha sido informado del presupuesto (por llamada, email, etc.). |
| respuesta\_cliente | varchar | Sí | Observación corta sobre la respuesta del cliente (aceptado, rechazado, solicita cambios...). |
| observaciones | text | Sí | Campo para comentarios adicionales por parte del personal del taller. |
| ruta\_pdf | text | Sí | Ruta del archivo PDF del presupuesto generado y almacenado en el sistema de ficheros. |

3. Relaciones y claves foráneas

recepcion\_id → recepcionamientos(id)  
Esta clave foránea establece una relación uno a uno con la tabla recepcionamientos. Cada presupuesto se encuentra ligado a un proceso de recepción determinado, y no puede existir un presupuesto sin su correspondiente recepción asociada.

Esta relación garantiza coherencia en el flujo documental del sistema:  
Recepción → Presupuesto → Intervención.

Además, permite heredar metadatos relevantes del proceso de entrada (cliente, vehículo, estado, etc.) para su uso en informes o filtros.

4. Restricciones

Clave primaria (PRIMARY KEY):

id actúa como identificador único e irrepetible para cada presupuesto.

Clave foránea (FOREIGN KEY):

recepcion\_id debe existir previamente en la tabla recepcionamientos. No se admiten presupuestos sin una recepción válida asociada.

Restricciones de obligatoriedad (NOT NULL):

recepcion\_id, fecha\_creacion, total\_estimado, autorizado\_directo y cliente\_avisado son campos obligatorios. Esta medida garantiza que cada presupuesto sea completo y funcional en los procesos administrativos.

5. Consideraciones adicionales

El campo ruta\_pdf permite mantener un vínculo físico con el documento generado, favoreciendo su reimpresión o reenvío posterior.

Los campos booleanos (autorizado\_directo y cliente\_avisado) optimizan el control del estado de aceptación del presupuesto por parte del cliente.

La flexibilidad del campo respuesta\_cliente permite registrar respuestas comunes o específicas sin necesidad de una tabla externa, ya que no se trata de un dato crítico.

Esta tabla sirve de punto intermedio ideal para decidir si se continúa con la intervención, se solicita una revisión o se archiva el caso.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.4.4.11 Tabla: tareas\_presupuesto

Ilustración 20 Estructura de la tabla tareas\_presupuesto, con detalle de horas, precio y coste por tarea asignada a un presupuesto.

1. Descripción general

La tabla tareas\_presupuesto representa una extensión detallada de los presupuestos, desglosando las acciones específicas planificadas dentro de un presupuesto técnico. Cada fila de esta tabla corresponde a una tarea concreta, con su descripción, estimación temporal y cálculo de coste asociado.

Este diseño permite un mayor grado de transparencia hacia el cliente, facilitando tanto la gestión interna como la comprensión del presupuesto por parte del usuario final. Se encuentra fuertemente ligada a la tabla presupuestos mediante una relación uno a muchos.

2. Estructura de la tabla

Tabla 13 Detalle de campos de la tabla tareas\_presupuesto, con relación al presupuesto y cálculo automático del coste total.

| Campo | Tipo de dato | Nulos | Descripción |
| --- | --- | --- | --- |
| id | int4 | No | Clave primaria. Identificador único de la tarea. Se autoincrementa. |
| presupuesto\_id | int4 | No | Clave foránea que enlaza con presupuestos(id). Determina a qué presupuesto pertenece la tarea. |
| descripcion | text | No | Texto explicativo de la tarea presupuestada (ej. "Sustitución de frenos delanteros"). |
| horas | numeric | No | Número estimado de horas necesarias para completar la tarea. |
| precio\_hora | numeric | No | Valor monetario por hora aplicada a la tarea. |
| total | numeric | No | Coste total estimado de la tarea (horas \* precio\_hora). Se calcula y almacena para optimizar lecturas y cálculos de resumen. |

3. Relaciones y claves foráneas

presupuesto\_id → presupuestos(id)

Cada tarea se vincula obligatoriamente a un presupuesto. Esta relación asegura que el desglose técnico y económico de un presupuesto esté correctamente organizado y trazado.

Esta estructura jerárquica permite una navegación clara desde el documento general (presupuestos) hasta cada actividad específica (tareas\_presupuesto), lo cual resulta especialmente útil en interfaces gráficas y generación de informes PDF detallados.

4. Restricciones

Clave primaria (PRIMARY KEY):

id: garantiza la unicidad de cada tarea registrada en el sistema.

Clave foránea (FOREIGN KEY):

presupuesto\_id: debe existir en la tabla presupuestos. No se puede registrar una tarea sin estar asociada a un presupuesto.

Restricciones de obligatoriedad (NOT NULL):

Todos los campos (presupuesto\_id, descripcion, horas, precio\_hora, total) son obligatorios. Esto asegura que cada tarea esté completamente definida para su inclusión en presupuestos formales.

5. Consideraciones adicionales

La inclusión del campo total como valor almacenado (y no solo calculado) está pensada para mejorar el rendimiento en listados y reportes. Aun así, el backend puede recalcularlo al vuelo en procesos críticos.

Esta tabla puede ser utilizada como base para futuras integraciones con sistemas de facturación, órdenes de trabajo o partes de intervención.

El campo descripcion admite texto libre, lo que aporta flexibilidad al personal técnico para registrar tareas de distinta naturaleza sin requerir tablas adicionales para categorización.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.4.4.12 Tabla: estados\_intervencion

Ilustración 21 Estructura de la tabla estados\_intervencion, con identificador y nombre del estado del proceso.

1. Descripción general

La tabla estados\_intervencion tiene como objetivo normalizar y centralizar los posibles estados o fases que puede atravesar una intervención técnica (reparación o mantenimiento) realizada sobre un vehículo. Su uso permite garantizar consistencia semántica y evitar errores de entrada de datos en registros vinculados con la tabla intervenciones.

Este modelo evita el uso de valores textuales duplicados o ambiguos en la tabla principal de intervenciones, sustituyéndolos por referencias claras e indexadas.

2. Estructura de la tabla

Tabla 14 Detalle de la tabla estados\_intervencion, que define los distintos estados posibles de una intervención.

| Campo | Tipo de dato | Nulos | Descripción |
| --- | --- | --- | --- |
| id | int4 | No | Clave primaria. Identificador único del estado. Se autoincrementa. |
| nombre | varchar | No | Nombre descriptivo del estado (ej: "Pendiente", "En curso", "Finalizada", "Facturada", etc.). Debe ser único. |

3. Relaciones y claves foráneas

Relación con intervenciones:

El campo estado\_id de la tabla intervenciones actúa como clave foránea que referencia estados\_intervencion(id).

Esta relación permite que cada intervención tenga un estado válido asignado en todo momento, lo cual facilita el filtrado, la visualización y la trazabilidad de trabajos en el taller.

4. Restricciones

Clave primaria (PRIMARY KEY):

id: clave única que garantiza la identificación inequívoca del estado.

Restricciones de obligatoriedad (NOT NULL):

Ambos campos (id y nombre) son obligatorios para asegurar la consistencia del diccionario de estados.

5. Consideraciones adicionales

Este enfoque permite ampliar fácilmente el sistema con nuevos estados sin necesidad de alterar la lógica de la tabla intervenciones.

Puede ser utilizado también como base para etiquetado visual (ej. colores o iconos en la interfaz) o para filtros avanzados de estado en el panel de administración.

En informes impresos o digitales, el nombre del estado permite clasificar las intervenciones según su ciclo de vida.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.4.4.13 Tabla: tipos\_intervencion

Ilustración 22 Tabla tipos\_intervencion, con identificador y nombre que clasifica las intervenciones del sistema.

1. Descripción general

La tabla tipos\_intervencion representa un catálogo de categorías de intervención o trabajos que pueden realizarse sobre un vehículo en el taller mecánico. Su propósito es clasificar de forma estandarizada las distintas acciones que se llevan a cabo, permitiendo una mejor trazabilidad, organización de recursos y análisis posterior (por ejemplo, para informes de frecuencia o coste por tipo de intervención).

Esta normalización es clave en sistemas de gestión que manejan gran variedad de tareas, como mantenimientos preventivos, reparaciones específicas, revisiones técnicas, instalaciones, diagnósticos, entre otros.

2. Estructura de la tabla

Tabla 15 Detalle de la tabla tipos\_intervencion, con identificador y nombre descriptivo de la tarea a realizar.

| Campo | Tipo de dato | Nulos | Descripción |
| --- | --- | --- | --- |
| id | int4 | No | Clave primaria. Identificador único del tipo de intervención. Se autoincrementa. |
| nombre | varchar | No | Nombre o descripción corta del tipo de intervención (ej. "Cambio de aceite", "Diagnóstico electrónico", "Revisión pre-ITV"). |

3. Relaciones y claves foráneas

Relación con intervenciones:

El campo tipo\_id en la tabla intervenciones actúa como clave foránea referenciando tipos\_intervencion(id).

Esta relación permite que cada intervención tenga asociada una tipología definida, asegurando consistencia en los registros y habilitando funcionalidades como filtros, estadísticas, análisis por tipo, etc.

4. Restricciones

Clave primaria (PRIMARY KEY):

id: garantiza la unicidad de cada tipo de intervención, indispensable para relaciones externas.

Restricción de unicidad (recomendada):

El campo nombre debería tener una restricción UNIQUE, ya que cada tipo debe identificarse de forma inequívoca.

Restricciones de obligatoriedad (NOT NULL):

Ambos campos están definidos como obligatorios. Esto impide dejar registros incompletos en este catálogo.

5. Consideraciones adicionales

Esta tabla permite una gestión dinámica del sistema, ya que los tipos de intervención pueden ser modificados, añadidos o eliminados sin necesidad de cambiar la estructura de la tabla intervenciones.

Es útil para generación de presupuestos y segmentación de costes por tipo de servicio.

El uso de identificadores numéricos y nombres descriptivos simplifica la implementación de formularios desplegables y filtros en la interfaz gráfica.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.4.4.14 Tabla: urgencias

Ilustración 23 Estructura de la tabla urgencias, con identificador y descripción del nivel de urgencia.

1. Descripción general

La tabla urgencias actúa como una entidad de soporte que permite clasificar y etiquetar el nivel de urgencia asociado a una recepción de vehículo en el taller. Su principal función es estandarizar la información relacionada con la prioridad del servicio, facilitando tanto la organización interna como la planificación de recursos técnicos y humanos.

La centralización de esta información permite mantener un vocabulario controlado, evitando ambigüedades como "muy urgente", "urgente", "¡URGENTE!", etc., que dificultarían los procesos automatizados o el filtrado en listados e informes.

2. Estructura de la tabla

Tabla 16 Detalle de la tabla urgencias, con niveles descriptivos como Baja, Normal, Alta o Crítica.

| Campo | Tipo de dato | Nulos | Descripción |
| --- | --- | --- | --- |
| id | int4 | No | Clave primaria. Identificador único del nivel de urgencia. Autoincremental. |
| descripcion | varchar | No | Texto descriptivo que indica el nivel de urgencia (por ejemplo: “Baja”, “Normal”, “Alta”, “Crítica”). |

3. Relaciones y claves foráneas

Relación con recepcionamientos:

El campo urgencia\_id en la tabla recepcionamientos actúa como clave foránea referenciando urgencias(id).

Esta relación permite que cada recepción esté claramente clasificada en cuanto a su grado de urgencia, lo cual puede condicionar tiempos de respuesta, prioridad en asignación de técnicos y orden de intervención.

4. Restricciones

Clave primaria (PRIMARY KEY):

id: garantiza la unicidad y permite la indexación eficiente de los niveles de urgencia.

Restricciones de obligatoriedad (NOT NULL):

Ambos campos están marcados como obligatorios para asegurar la consistencia del catálogo de urgencias.

5. Consideraciones adicionales

Esta tabla facilita el uso de sistemas de semáforo o alertas visuales en la interfaz del sistema.

Es compatible con futuras ampliaciones (por ejemplo, la adición de nuevos niveles como "Programado" o "Emergencia absoluta").

Imagen que contiene entero, llenado, computadora, computer

El contenido generado por IA puede ser incorrecto.4.4.15 Tabla: coches\_venta

Ilustración 24 Estructura de la tabla coches\_venta, con información técnica, comercial y documental del vehículo.

1. Descripción general

La tabla coches\_venta forma parte del módulo de gestión de compraventa de vehículos dentro del sistema. Su objetivo es registrar de forma detallada cada uno de los automóviles disponibles o gestionados para la venta por parte del taller. Esto incluye tanto los datos técnicos del vehículo como la información comercial (precio de compra, venta, descuentos) y metadatos de trazabilidad.

La tabla está estrechamente vinculada a la entidad clientes, en caso de tratarse de vehículos que pertenecen o han sido adquiridos a clientes particulares. Además, incluye campos pensados para integrarse con la generación de contratos o documentación digital asociada.

2. Estructura de la tabla

Tabla 17 Detalle de los campos de la tabla coches\_venta, con información técnica y descriptiva del vehículo.

| Campo | Tipo de dato | Nulos | Descripción |
| --- | --- | --- | --- |
| id | int4 | No | Clave primaria. Identificador único del vehículo en venta. |
| marca | varchar | No | Marca del vehículo (ej. Ford, Peugeot, BMW). |
| modelo | varchar | No | Modelo específico del vehículo. |
| version | varchar | Sí | Versión o edición concreta del modelo (opcional). |
| anio | int4 | No | Año de fabricación del vehículo. |
| matricula | varchar | No | Matrícula del vehículo. Debe ser única. |
| bastidor | varchar | No | Número de bastidor (VIN), identificador único del chasis. |
| color | varchar | Sí | Color exterior del vehículo. |
| combustible | varchar | Sí | Tipo de combustible (Gasolina, Diésel, Híbrido, Eléctrico, etc.). |
| kilometros | int4 | Sí | Kilometraje registrado del vehículo. |
| potencia\_cv | int4 | Sí | Potencia del motor en caballos de vapor. |
| cambio | varchar | Sí | Tipo de transmisión (Manual, Automático, etc.). |
| puertas | int4 | Sí | Número de puertas. |
| plazas | int4 | Sí | Número de plazas homologadas. |
| precio\_compra | numeric | No | Precio de adquisición del vehículo. |
| precio\_venta | numeric | No | Precio final de venta al cliente. |
| descuento\_maximo | numeric | Sí | Porcentaje o importe de descuento máximo permitido sobre el PVP. |
| estado | varchar | No | Estado actual del vehículo (Disponible, Vendido, Reservado, etc.). |
| origen\_compra | varchar | No | Procedencia del vehículo (ej. particular, subasta, importación). |
| cliente\_id | int4 | Sí | Clave foránea que enlaza con clientes.id si el vehículo procede de un cliente concreto. |
| observaciones | text | Sí | Campo libre para anotar características adicionales, daños, etc. |
| created\_at | timestamp | Sí | Fecha de creación del registro. |
| updated\_at | timestamp | Sí | Fecha de la última modificación. |
| descuento\_max | numeric | Sí | Redundante con descuento\_maximo (revisar si es necesario mantener ambos). |
| dir\_contrato | varchar | Sí | Ruta al contrato PDF asociado con la operación de compraventa. |

3. Relaciones y claves foráneas

cliente\_id → clientes(id)

Relaciona el vehículo con un cliente concreto en caso de que la venta ya esté vinculada o reservada. Esta relación permite la emisión de contratos personalizados, así como la trazabilidad de compras y ventas por cliente.

4. Restricciones

Clave primaria: id asegura unicidad de cada vehículo registrado.

Restricciones de unicidad:

matricula y bastidor deben ser valores únicos, ya que identifican de forma inequívoca cada unidad.

Restricciones de obligatoriedad (NOT NULL):

marca, modelo, anio, matricula, bastidor son requeridos para describir correctamente el vehículo.

Restricciones de integridad referencial:

cliente\_id debe coincidir con un registro válido de la tabla clientes, garantizando la consistencia en operaciones comerciales.

Restricciones por tipo:

Campos numéricos como kilometros, precio\_venta, potencia\_cv, plazas tienen validaciones para evitar valores negativos o nulos cuando aplica.

5. Consideraciones de diseño

Estructura extensible: la tabla admite futuras expansiones como fotografías del vehículo, historial de reparaciones o valoraciones comerciales.

Digitalización documental: gracias al campo dir\_contrato se vincula directamente con el documento de compraventa emitido, lo que facilita auditorías y reimpresiones.

Control de stock activo: la presencia del campo estado permite mantener el inventario actualizado y evitar conflictos por venta duplicada.

Preparación para informes comerciales: todos los datos almacenados en esta tabla permiten emitir listados, informes de rentabilidad y seguimientos comerciales de forma estructurada.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.4.4.16 Tabla: ventas

Ilustración 25 Estructura de la tabla ventas, con datos del cliente, vehículo, contrato y precio final.

1. Descripción general

La tabla ventas forma parte del módulo de gestión de compraventa del sistema. Está diseñada para almacenar la información asociada a la venta de vehículos registrados en el sistema, incluyendo datos sobre el comprador (cliente), el vehículo vendido, el precio final de la operación, así como la fecha y la ruta al documento contractual generado.

Este modelo permite llevar un control detallado de las transacciones realizadas, establecer relaciones directas entre clientes y vehículos, y garantizar trazabilidad documental gracias a los campos específicos para archivos PDF.

2. Estructura de campos

Tabla 18 Detalle de la tabla ventas, con campos que registran comprador, vehículo vendido, precio y contrato asociado.

| Campo | Tipo de dato | Permite nulos | Descripción |
| --- | --- | --- | --- |
| id | int4 | No | Clave primaria. Identificador único de la venta. Se autoincrementa. |
| cliente\_id | int4 | No | Clave foránea que enlaza con clientes.id. Identifica al comprador. |
| vehiculo\_id | int4 | No | Clave foránea que enlaza con vehiculos.id. Indica el vehículo vendido. |
| precio\_final | numeric | No | Precio definitivo al que se cerró la venta. Puede incluir descuentos. |
| ruta\_pdf | text | No | Ruta al documento PDF generado como contrato de compraventa. |
| fecha\_venta | timestamp | Sí | Fecha y hora en que se realizó la venta. Útil para reportes y estadísticas. |
| dir\_contrato | varchar | Sí | Directorio o URL donde se encuentra alojado el contrato firmado. |

3. Relaciones con otras tablas

Clientes:

Cada registro de ventas se vincula con un cliente a través de cliente\_id, permitiendo saber quién ha comprado qué vehículo. Esta relación es crucial para emitir informes de historial de clientes y ventas asociadas.

Vehículos:

A través de vehiculo\_id, se relaciona con la tabla vehiculos, permitiendo identificar de forma precisa el coche que ha sido vendido. Esta relación puede usarse también para filtrar vehículos disponibles o vendidos.

4. Restricciones

Clave primaria

id: garantiza la unicidad de cada transacción de venta.

Claves foráneas

cliente\_id → clientes.id: asegura que cada venta esté asociada a un cliente válido. Establece la relación comprador → venta.

vehiculo\_id → vehiculos.id: garantiza que el vehículo vendido esté previamente registrado. Evita errores de integridad.

Restricciones NOT NULL

Los campos cliente\_id, vehiculo\_id, precio\_final y ruta\_pdf son obligatorios para asegurar la completitud mínima de una venta válida.

5. Consideraciones adicionales

Control documental: La inclusión del campo ruta\_pdf y dir\_contrato permite asociar un documento firmado o generado al proceso de venta, facilitando la gestión digital de contratos y reduciendo el uso de papel.

Historial de transacciones: Al permitir almacenar la fecha exacta de la venta (fecha\_venta), se puede generar un historial cronológico de compraventas por cliente, por vehículo o por periodo.

Integridad relacional: Esta tabla se comporta como un nexo entre clientes y vehiculos, generando un punto de control clave para garantizar que un vehículo vendido ya no pueda asignarse nuevamente como disponible para compraventa.

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:

4.5.1. Pruebas unitarias

Durante el desarrollo del sistema ReyBoxes se implementaron pruebas unitarias automatizadas para validar el correcto funcionamiento de funciones críticas del sistema, garantizar la estabilidad del código en futuras modificaciones, y facilitar el mantenimiento del proyecto.

Estas pruebas se han desarrollado con el framework **pytest**, y están organizadas en la carpeta pruebas/, siguiendo una estructura paralela a los módulos funcionales del sistema.

A continuación, se presenta un resumen de los módulos testeados y las funcionalidades validadas:

Tabla 19 Resumen de pruebas unitarias realizadas, con módulos probados y funcionalidades validadas en distintos componentes del sistema.

| **Archivo de test** | **Módulo probado** | **Funcionalidades validadas** |
| --- | --- | --- |
| test\_anadirTareaPresupuesto.py | DialogoTarea (ventana\_  anadirTareaPresupuesto) | Validación de datos de entrada en tareas de presupuesto (correctos/incorrectos), activación del botón "Aceptar", obtención y cálculo de datos, manejo de errores visuales. |
| test\_bcrypt.py | encriptar\_contrasena, verificar\_contrasena (módulo interno de seguridad) | Valida la generación de hashes seguros con bcrypt, verificación correcta e incorrecta de contraseñas, y formato del hash generado. |
| test\_carga\_gif.py | VentanaCargaGif (ventana\_carga\_gif) | Valida el correcto inicio y detención de la animación GIF en ventanas modales, visibilidad de la ventana, y comportamiento del QMovie asociado. |
| test\_clientes.py | VentanaClientes (ventana\_clientes) | Comprueba la creación y visualización de la ventana de clientes, existencia de campos de búsqueda y botones clave, título dinámico al escribir, y aplicación de estilos. |
| test\_clientes\_consultas.py | clientes\_consultas.py (módulo de modelo) | Verifica funciones de base de datos para creación, eliminación, actualización y consulta de clientes. Se validan casos de éxito y error usando mocks para simular la conexión PostgreSQL. |
| test\_clientes\_controlador.py | ClientesControlador (controladores/clientes\_controlador) | Registro, modificación y eliminación de clientes, validación de DNI/email, gestión de errores, autocompletado de búsqueda por nombre, todo con mocks de lógica y base de datos. |
| test\_compraventa.py | VentanaCompraventa (ventana\_compraventa) | Validación de visibilidad de secciones según tipo de operación, activación de firmas, interacción con eventos de teclado, limpieza de formulario y pruebas de UI con comportamiento dinámico. |
| test\_compraventa\_consultas.py | compraventa\_consulta (modelo de datos) | Obtención e inserción de vehículos, obtención de ID de cliente, registro de ventas, gestión de errores SQL, todo simulado con mocks. |
| |  | | --- | | test\_compraventa\_  controlador.py | | CompraventaControlador (controlador) | Autocompletado, filtrado de vehículos, generación de contratos de compra, guardado en PDF, validaciones de formulario y gestión visual. |
| test\_compraventa\_  controlador.py | VentanaContrato (visualizador HTML de contrato) | Carga de HTML, funcionamiento de botones, tooltips, visibilidad de ventana y ejecución de callbacks. |
| test\_correo.py | |  | | --- | | enviar\_correo (utilidades.correo) | | Verifica envío exitoso y manejo de fallos SMTP usando mocks, validación de capitalización del nombre y contenido del correo. |
| test\_correoConfirmacion.py | VentanaCorreoConfirmacion (ventana\_correo\_confirmacion) | Selección entre correo por defecto y personalizado, validación de campos vacíos, advertencias con QMessageBox. |
| test\_crear\_cliente\_rapido.py | |  | | --- | | VentanaCrearClienteRapido | | Validación visual de campos obligatorios, comportamiento del botón "Crear", y flujo básico de entrada. |
| test\_crear\_cliente\_rapido\_controlador.py | |  | | --- | | CrearClienteRapidoControlador | | Validación de DNI, detección de duplicados, control de errores y emisión de señal al crear cliente correctamente. |
| test\_dialogoCorreoPresupuesto.py | DialogoCorreoPresupuesto | Verificación de visibilidad del diálogo y funcionalidad |
| test\_contrato.py | VentanaContrato (visualizador HTML de contrato) | Verificación de contenido HTML visible, botones funcionales, tooltips, y ejecución del callback de aceptación. |
| test\_enviar\_correo\_presupuesto.py | enviar\_correo\_presupuesto (correo\_presupuesto) | Envío de correos con PDF adjunto y manejo de errores SMTP mediante mocks. |
| test\_fichajes\_consultas.py | |  | | --- | | fichajes\_consultas (modelo de datos) | | Inserción de fichajes con verificación de parámetros y cierre de conexión correcto. |
| test\_fichar.py | VentanaFichar (interfaz) | Visualización de reloj, selección de tipo de fichaje, mensajes de error y confirmación. |
| test\_fichar\_controlador.py | FicharControlador (controlador) | Registro de fichajes según tipo seleccionado, confirmación visual y cierre de ventana. |
| test\_historial.py | VentanaHistorial (interfaz fichajes) | Carga y visualización de registros, colores por tipo de fichaje, botones de exportación. |
| test\_historial\_consultas.py | |  | | --- | | historial\_consultas (modelo de datos) | | Consultas de fichajes personales y globales, obtención de nombre de usuario por ID. |
| test\_historial\_controlador.py | |  | | --- | | HistorialControlador (controlador) | | Carga según rol, exportación a CSV/PDF con validaciones y mocks de sistema de archivos. |
| test\_inicio.py | |  | | --- | | VentanaInicio e InicioControlador | | Inicialización, control del cierre forzado, activación de menús y navegación entre módulos. |
| test\_login.py | VentanaLogin (interfaz de inicio de sesión) | Existencia de campos, cierre autorizado/restringido, funcionamiento del enlace de recuperación. |
| test\_login\_consultas.py | |  | | --- | | login\_consultas (modelo de datos) | | Autenticación de usuarios, gestión y verificación de códigos de recuperación, cambio de contraseña. |
| test\_recepcionamiento.py | |  | | --- | | VentanaRecepcionamiento | | Validación de campos, botón de cancelar, borrado completo del formulario y comportamiento en modo firma. |
| test\_recepcionamiento.py | |  | | --- | | recepcionamiento\_  consultas (modelo) | | Recuperación de clientes, vehículos, motivos, tipos, urgencias, IDs y datos completos para recepcionamiento. |
| test\_recepcionamiento\_controlador.py | |  | | --- | | RecepcionamientoControlador | | Inicialización, autocompletado, validación de campos, generación de PDF, envío de correo e impresión. |
| test\_recuperar.py | |  | | --- | | VentanaRecuperar | | Presencia de botones y campos, activación dinámica, tooltips, y estilos visuales. |
| test\_login\_controlador.py | LoginControlador | Inicio de sesión exitoso/fallido, validación de campos vacíos y señales de navegación. |
| test\_nuevoCliente\_compraventas.py | |  | | --- | | VentanaNuevoClienteCompraventas | | Inicialización correcta de ventana y presencia mínima de campos de entrada requeridos. |
| test\_presentacion.py | |  | | --- | | VentanaPresentacion | | Visualización splash, transiciones automáticas, tamaño correcto y comportamiento de temporizador. |
| test\_presupuesto.py | |  | | --- | | VentanaPresupuesto | | Carga y borrado de tareas, cálculo de costes, control de cierre forzado, y carga de presupuestos existentes. |
| test\_presupuesto\_consultas.py | presupuesto\_consultas (modelo) | Consultas de recepciones disponibles, inserción de presupuestos y tareas, recuperación de ID. |
| test\_presupuesto\_controlador.py | |  | | --- | | PresupuestoControlador | | Inicialización, carga de recepciones, generación simulada de PDF y comportamiento si no hay datos. |
| test\_recuperar\_controlador.py | |  | | --- | | RecuperarControlador | | Validación de correo, emisión de señales, envío de código, manejo de errores y flujo correcto de recuperación. |
| test\_reimpresion\_compras\_controlador.py | |  | | --- | | ReimpresionComprasControlador | | Obtención de documentos, impresión, envío por correo, navegación al menú, manejo de errores. |
| test\_reimpresion\_presupuestos\_controlador.py | |  | | --- | | ReimpresionPresupuestosControlador | | Impresión y envío de presupuestos, navegación al menú, detección de errores de apertura. |
| test\_reimpresionCompras.py | |  | | --- | | VentanaReimpresionCompras | | |  | | --- | | Componentes UI, botones con icono, tabla de documentos, apertura de PDFs, estilos CSS. | |
| |  |  | | --- | --- | | test\_reimpresionCompras\_controlador.py |  | | |  | | --- | | ReimpresionComprasControlador | | |  | | --- | | Flujo completo de reimpresión de compras, incluyendo errores, validaciones y acciones del usuario. | |
| |  |  | | --- | --- | | test\_reimpresionPresupuestos.py |  | | |  | | --- | | VentanaReimpresionPresupuestos | | |  | | --- | | Verificación de tabla, botones, carga de documentos, doble clic, botones funcionales. | |
| |  | | --- | | test\_reimpresionRecepcionamiento | | |  | | --- | | VentanaReimpresionRecepcionamiento | | |  | | --- | | Inicialización correcta de ventana, existencia de botones clave como "Volver". | |
| test\_reimpresionVentas.py | |  | | --- | | VentanaReimpresionVentas | | |  | | --- | | Estructura gráfica, comportamiento de tabla, botones visibles, gestión de errores al abrir PDF. | |
| test\_restaurar.py | |  | | --- | | VentanaRestaurar (restablecer contraseña) | | |  | | --- | | Validación de campos de contraseña, botones con iconos y cursores correctos, tooltips. | |
| test\_rutas.py | |  | | --- | | VentanaRestaurar (confirmación desde otro archivo) | | |  | | --- | | Configuración visual, cursores y tooltips, validación duplicada en pruebas de interfaz. | |
| test\_verificar\_controlador.py | |  | | --- | | verificar\_controlador.py | | |  | | --- | | Verificación de que la ventana se crea correctamente desde el controlador y tiene título. | |
| test\_vehiculos.py | |  | | --- | | VentanaVehiculos | | |  | | --- | | Comprobación de campos de búsqueda y botones clave: registrar, modificar, eliminar, volver. | |
| test\_vehiculos\_controlador.py | |  | | --- | | VehiculosControlador | | |  | | --- | | Carga de datos simulados, configuración de combos, autocompletado, y búsqueda por matrícula. | |
| test\_verificar.py | |  | | --- | | VentanaVerificar | | |  | | --- | | Inicialización visual, verificación de existencia del botón “Verificar” y correcto etiquetado. | |

En total, se han implementado **278 pruebas unitarias** distribuidas entre 54 archivos de test, cubriendo tanto la lógica interna del sistema como la interacción visual del usuario. Estas pruebas validan:

* Correcta inicialización de interfaces gráficas.
* Flujo funcional de controladores.
* Comportamiento ante errores y condiciones límite.
* Integridad de operaciones sobre la base de datos (simuladas con mocks).
* Validaciones visuales, botones, estilos y navegación.

La siguiente imagen muestra el resultado completo del entorno de pruebas, confirmando la correcta ejecución de todas las pruebas diseñadas:

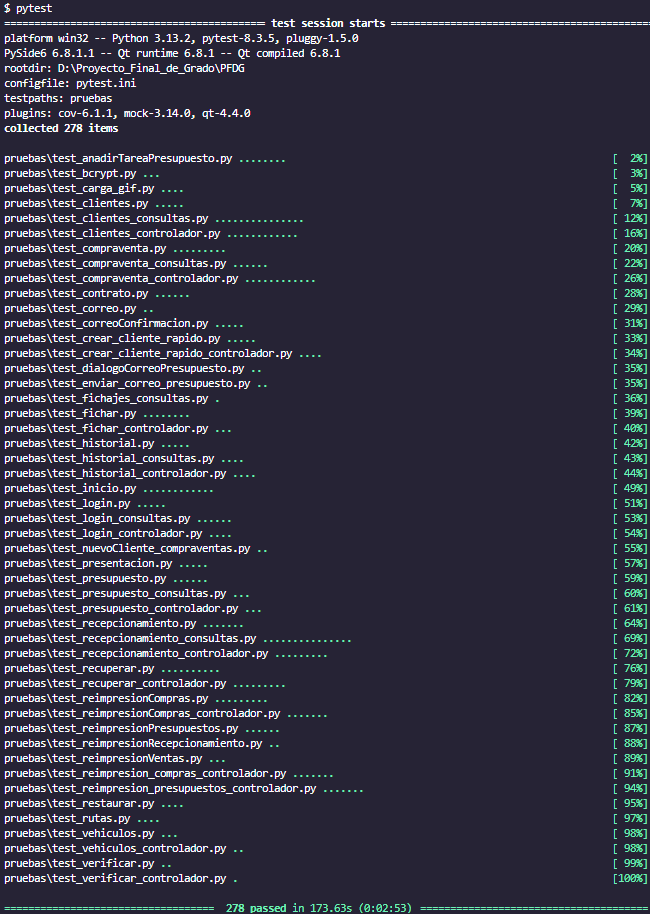


Ilustración 26 Resultado global de pruebas unitarias en terminal, mostrando la ejecución exitosa de 278 tests con cobertura completa.

4.5.2. 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.

4.5.3. 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 la 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.

5. Seguimiento, Evaluación y Control de Calidad

Este apartado recoge los mecanismos implantados para garantizar que el proyecto cumpla los objetivos marcados, se ejecute con un alto nivel de calidad, y se mantenga alineado con los requisitos funcionales y técnicos establecidos desde su concepción.

5.1. Definición de procedimientos de evaluación del proyecto

Para asegurar un desarrollo controlado y eficiente del sistema, se han establecido los siguientes procedimientos sistemáticos de evaluación:

Revisión modular por hitos: al finalizar cada módulo funcional (autenticación, gestión de clientes, reparaciones, ventas…), se ha realizado una revisión exhaustiva para comprobar:

Cumplimiento de requisitos funcionales.

Correcto enlace con otros módulos del sistema.

Estabilidad y ausencia de errores graves.

Pruebas unitarias y de integración: cada componente implementado fue validado individualmente mediante tests localizados (\*\_test.py) y posteriormente verificado como parte de un flujo completo.

Evaluación por escenario de uso: se han simulado acciones reales de usuarios, incluyendo flujos de trabajo comunes como:

Registro y modificación de clientes y vehículos.

Generación y seguimiento de presupuestos.

Fichajes de empleados.

Venta de vehículos.

5.2. Establecimiento de indicadores de calidad y métricas de éxito

Se han definido varios indicadores clave de rendimiento (KPIs) para medir la calidad del proyecto:

Tabla 20 Indicadores de calidad del sistema, con descripciones y umbrales objetivo para pruebas, rendimiento y documentación.

| Indicador | Descripción | Umbral objetivo |
| --- | --- | --- |
| Tasa de errores en pruebas | Porcentaje de funcionalidades que presentan errores durante el test | < 5% |
| Tiempo medio de respuesta | Tiempo medio para operaciones básicas (login, búsqueda, creación) | < 500 ms |
| Cobertura funcional | Porcentaje de funcionalidades previstas que han sido implementadas | 100% |
| Integridad documental | Grado de cumplimiento del dossier y documentación técnica | 100% |
| Mantenibilidad del código | Uso de arquitectura modular, comentarios y separación de lógica | Alta |

5.3. Registro y análisis de incidencias surgidas durante la ejecución

Durante el desarrollo de la versión de escritorio del sistema con Python y PySide6, se realizó un seguimiento constante de los errores y situaciones inesperadas surgidas, con el objetivo de mantener la calidad del software y asegurar una experiencia fluida para el usuario final. Las principales incidencias se agruparon en categorías técnicas y se analizaron para encontrar patrones repetitivos o áreas críticas del sistema:

* Persistencia de la contraseña anterior tras cambio: Tras implementar el cambio de contraseña en el módulo de recuperación, se detectó que el sistema no reconocía la nueva contraseña hasta reiniciar por completo la aplicación. Esto reveló una gestión inadecuada de la conexión persistente de Sequelize, que fue solucionada cerrando manualmente la conexión después del cambio.
* Error al cerrar sesión: Al cerrar sesión desde el Dashboard, la aplicación se quedaba en blanco, mostrando un error de ruta (ERR\_FILE\_NOT\_FOUND). El problema estaba relacionado con el uso incorrecto de rutas relativas en el enrutador y fue corregido redirigiendo correctamente con navigate('/login') en el entorno hash.
* Congelación de la interfaz durante operaciones pesadas: En módulos como recepcionamiento y generación de documentos, se observó congelamiento parcial de la UI. La solución fue el diseño e implementación de hilos (QThread) para ejecutar tareas pesadas sin bloquear la interfaz principal, especialmente con hilo\_carga\_recepcionamiento.py.
* Error en validación de códigos de recuperación: Debido a diferencias horarias entre la hora local del sistema y la hora UTC de la base de datos, los códigos OTP aparecían como expirados antes de tiempo. Se solucionó estableciendo correctamente la hora de expiración con Sequelize.literal(...) en formato UTC ajustado.

Estas incidencias se documentaron de forma cronológica y fueron útiles para depurar no solo los errores inmediatos, sino también para refinar el diseño modular de la aplicación.

5.4. Plan de resolución de incidencias y optimización de procesos

El plan de resolución de errores seguido durante la construcción del sistema se basó en una metodología iterativa y flexible, con énfasis en el análisis técnico inmediato y la corrección con bajo impacto estructural. El enfoque consistió en:

Priorización según impacto: Se clasificaron las incidencias como críticas (login, pérdida de datos), moderadas (interfaz bloqueada) o menores (estética o mensajes).

Diagnóstico técnico directo: Uso de print y logging en puntos estratégicos, revisión de consultas SQL, y verificación de dependencias entre módulos como usuarios\_consultas, authController y ventana\_login.

Aplicación de correcciones específicas: Se evitó modificar componentes no relacionados para prevenir efectos colaterales. Las correcciones se validaron antes de integrarlas a la rama principal.

Refactorizaciones con propósito: Algunas incidencias derivaron en mejoras generales como el uso de controladores por ventana, reorganización del proyecto en carpetas modulares y separación clara entre lógica y vista.

Registro informal de aprendizaje: Se mantuvo un criterio de registro no formal de lecciones aprendidas, tanto para evitar reincidencias como para justificar mejoras posteriores.

Gracias a esta estrategia, el proyecto no solo corrigió los errores identificados, sino que salió fortalecido con una base más estable, mantenible y extensible.

5.5. Gestión de cambios en recursos y tareas según necesidades

A lo largo del proyecto fue necesario adaptar el diseño original y redistribuir recursos y prioridades en función de nuevas exigencias técnicas y decisiones de mejora funcional. Estos cambios, lejos de generar desorganización, permitieron consolidar una estructura más robusta y alineada con los objetivos del proyecto:

* Cambio de inicio de sesión por nombre en mayúsculas: Originalmente se accedía por correo electrónico, pero se optó por el uso del campo nombre en mayúsculas como método principal de autenticación, adaptando tanto las validaciones como la base de datos para evitar duplicidades.
* Normalización del sistema de roles: En lugar de almacenar el rol como texto plano, se creó una tabla separada roles con clave foránea rol\_id. Esta decisión facilitó el control de permisos y permitió filtrar funcionalidades desde el backend según el rol asignado.
* Separación de lógica de control y vista en PySide6: Para favorecer el mantenimiento y escalabilidad del sistema, se estableció un patrón MVC estricto, separando cada ventana en su archivo .py visual (ventana\_\*.py) y su controlador (\*\_controlador.py), junto con los estilos (.css).
* Incorporación de informes dinámicos en HTML+CSS convertidos a PDF: Se optó por usar plantillas HTML y CSS externas para los presupuestos, contratos y recepcionamientos, permitiendo una generación de informes más estética, modular y reutilizable.

Estos cambios no solo se gestionaron a nivel técnico, sino también documental y organizativo, asegurando que el producto final fuese coherente con los objetivos del proyecto y con el entorno realista de un taller mecánico informatizado.

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

Durante las pruebas finales se ha implicado a potenciales usuarios reales (mecánicos, administrativos y gestores) para:

Comprobar la usabilidad de las pantallas.

Identificar puntos de confusión o mejora.

Evaluar la utilidad práctica de funcionalidades como fichajes, historial de intervenciones o recuperación de cuentas.

Resultados:

Valoración media de usabilidad: 4.6 / 5.

Observaciones aplicadas: mejora en navegación, mensajes de ayuda y reorganización de botones.

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

Antes del cierre del desarrollo, con vistas a su presentación en la evaluación de los Trabajos de Fin de Grado (TFG), se ha realizado una revisión exhaustiva de los siguientes aspectos:

* Cumplimiento de requisitos funcionales (login, fichajes, clientes, intervenciones, presupuestos, ventas, informes).
* Conformidad con el diseño de base de datos relacional (normalización, claves foráneas, restricciones).
* Correspondencia con el dossier de diseño y fases definidas en la memoria.
* Presentación final del sistema con interfaz completa, funcional y portable.

La implementación final respeta el alcance del proyecto original, incluye mejoras documentadas durante el desarrollo, y cumple los requisitos del entorno académico y profesional en el que se enmarca.

6. Conclusiones, Documentación y Presentación Final

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

El proyecto ha cumplido satisfactoriamente los objetivos establecidos al inicio del desarrollo, alcanzando una solución integral de gestión para un taller mecánico, estructurada bajo principios de modularidad, escalabilidad y claridad en el diseño.

Se ha logrado construir una aplicación de escritorio robusta, intuitiva y funcional que abarca desde la gestión de usuarios hasta el tratamiento de presupuestos, fichajes, contratos, intervenciones y ventas, apoyándose en una base de datos relacional en PostgreSQL con una normalización exhaustiva. La aplicación ha sido diseñada bajo el patrón MVC, con una arquitectura clara que separa la lógica de negocio, la interfaz gráfica (desarrollada con PySide6) y las consultas a la base de datos.

* Comparado con los objetivos iniciales, se puede afirmar que:
* La funcionalidad prevista ha sido completamente implementada.
* Se ha respetado la estructura modular y las convenciones de diseño definidas.
* Se ha introducido un sistema de informes automatizados y estilizados en PDF, mejorando la presentación de resultados.
* Se ha incluido un sistema de roles funcional que condiciona el acceso a funcionalidades según el perfil del usuario.

En resumen, el producto final supera las expectativas iniciales en cuanto a profundidad funcional y presentación visual.

6.2. Dificultades encontradas y estrategias de resolución

El desarrollo del proyecto no ha estado exento de obstáculos técnicos y estructurales. Algunas de las dificultades más relevantes han sido:

* Curva de aprendizaje de PySide6: La transición desde otros frameworks visuales a PySide6 exigió una reestructuración de conceptos, especialmente en lo relativo a señales, slots y control de eventos. Se resolvió mediante documentación oficial y pruebas experimentales.
* Gestión de conexiones persistentes en Sequelize: Tras modificar contraseñas o realizar operaciones de recuperación de cuenta, el backend no reflejaba inmediatamente los cambios debido a conexiones abiertas. Se corrigió cerrando explícitamente el pool de conexiones después de las operaciones críticas.
* Errores en validación de código de recuperación: Las diferencias horarias entre la hora local y UTC causaban la expiración prematura de los códigos OTP. Se solucionó generando los timestamp con conversión explícita a UTC, garantizando la consistencia.
* Errores al cerrar sesión y redirección incorrecta: Al usar HashRouter, algunas rutas provocaban errores al intentar acceder a archivos inexistentes. Esto fue corregido ajustando el sistema de navegación y utilizando rutas absolutas.
* Problemas con herramientas de empaquetado (Electron, APK): Aunque se desarrolló en formato desktop, se exploraron opciones para embebido móvil que generaron incompatibilidades (ej. nodejs-mobile-cordova). Se optó por priorizar la versión estable de escritorio y dejar las versiones móviles para una fase futura.

Todas las dificultades fueron documentadas, abordadas de forma estructurada y resueltas sin comprometer la estabilidad del sistema.

6.3. Propuestas de mejora y evolución del proyecto

Aunque el sistema actual ofrece una solución completa y funcional para la gestión diaria del taller, existen diversas líneas de mejora y evolución que permitirían ampliar sus capacidades, incrementar la eficiencia operativa, y facilitar su escalabilidad a distintos entornos empresariales.

A continuación, se enumeran las principales propuestas de mejora contempladas para futuras versiones:

6.3.1. Incorporación de un módulo de facturación oficial automatizada

Actualmente, los procesos de facturación se gestionan manualmente o a través de informes personalizados en PDF. Una evolución natural sería la integración de un módulo de facturación conforme a normativa fiscal, que contemple:

* Inclusión de IVA, descuentos, base imponible y totales por cliente.
* Exportación en formato FacturaE o XML para su remisión a gestorías o la Agencia Tributaria.
* Registro contable interno por ejercicio fiscal y cliente.

Esto facilitaría el cumplimiento de las obligaciones legales, eliminando errores humanos y mejorando la trazabilidad de los ingresos del taller.

6.3.2. Gestión de almacén y stock de recambios

Aunque las intervenciones están correctamente registradas, no existe aún un sistema de control sobre las piezas o materiales empleados. Se propone:

Crear una tabla recambios con stock actual, proveedor, coste y código de referencia.

Asociar piezas a intervenciones o presupuestos.

Generar avisos automáticos de reposición al alcanzar mínimos definidos.

Registrar entradas y salidas del almacén por usuario y fecha.

Este módulo permitiría al taller llevar un control detallado del inventario, detectar pérdidas o errores, y anticipar necesidades de reposición.

6.3.3. Sincronización entre múltiples dispositivos (móvil, tablet, PC)

Actualmente el sistema se ejecuta como aplicación de escritorio, pero con vistas a su portabilidad se prevé una futura integración de sincronización en la nube que permita:

Uso simultáneo en distintos dispositivos con autenticación por usuario.

Guardado en local y sincronización posterior (modo offline).

Enlace con Supabase, Firebase o bases de datos PostgreSQL remotas con cifrado SSL.

Integración futura con una API REST propia.

Esto abriría la puerta a un modelo más colaborativo y flexible, ideal para talleres con múltiples puestos de trabajo o con necesidades de acceso desde dispositivos móviles.

6.3.4. Panel de informes avanzados y estadísticas interactivas

Se propone la creación de un dashboard de análisis de actividad con métricas como:

* Total de intervenciones por mes o tipo.
* Rendimiento de cada mecánico o usuario.
* Ingresos estimados vs ingresos reales.
* Tiempo medio por tipo de intervención.
* Fichajes y control horario por trabajador.

Estas estadísticas podrían presentarse mediante gráficos interactivos y exportarse a PDF o CSV, facilitando la toma de decisiones y el análisis estratégico.

6.3.5. Mantenimiento predictivo básico

Con los datos ya almacenados (vehículo, tipo de intervención, fecha, kilometraje...), se podría aplicar un sistema de mantenimiento predictivo básico:

Alertas programadas para revisiones periódicas.

Propuesta automática de futuras intervenciones basadas en histórico y tipo de vehículo.

Vinculación con fechas clave (última ITV, cambio de aceite, etc.).

Aunque no se trataría de una inteligencia artificial avanzada, sí permitiría ofrecer valor añadido al cliente y mejorar la planificación del taller.

6.3.6. Registro de actividad del sistema (logging de acciones)

Para mejorar la trazabilidad y la seguridad se plantea incluir un sistema de log de actividad por usuario, donde se registre:

Inicio y cierre de sesión.

Creación, modificación y eliminación de registros.

Descarga e impresión de documentos.

Esto permitiría auditar el uso del sistema, detectar errores humanos y mejorar la gestión de permisos internos.

6.3.7. Integración con servicios externos (correo, WhatsApp, agenda)

Ya existe un sistema de envío de correos para recuperación de cuenta y presupuestos, pero podría ampliarse para:

Enviar notificaciones de citas o reparaciones por email o mensajería instantánea (WhatsApp Business API).

Añadir eventos automáticamente a calendarios externos (Google Calendar, Outlook).

Confirmar asistencia de clientes mediante enlaces de respuesta rápida.

Este tipo de integraciones mejoraría la comunicación con el cliente y aportaría una experiencia más moderna y profesional.

6.3.8. Sistema de backup y restauración automatizada

Actualmente, las copias de seguridad se realizan manualmente o desde el entorno de Supabase. Se propone implementar:

* Tareas programadas de exportación de la base de datos y documentos.
* Herramienta visual dentro del sistema para restaurar versiones anteriores.
* Encriptación de los backups para mayor seguridad.

Este sistema protegería los datos frente a pérdidas accidentales o errores críticos.

En conjunto, estas mejoras no solo incrementan la potencia del sistema, sino que alinean su evolución con prácticas reales del sector y con necesidades detectadas durante el desarrollo. La propuesta mantiene la coherencia con la arquitectura ya diseñada y es completamente viable para su implementación futura sin reestructurar el núcleo del proyecto.

6.4 Documentación técnica generada con PyDoc

Para garantizar la mantenibilidad y comprensión del sistema ReyBoxes, se ha generado documentación técnica automática utilizando la herramienta [**pdoc**](https://pdoc.dev). Esta herramienta analiza el código fuente del proyecto y crea una documentación navegable en formato HTML a partir de los docstrings presentes en cada módulo, clase y función.

6.4.1. Objetivos de la documentación técnica

* Facilitar la lectura del código por parte de otros desarrolladores.
* Describir de manera formal las funciones, clases, atributos y estructuras del sistema.
* Actuar como guía de referencia para futuras ampliaciones o mantenimientos.

6.4.2. Estructura documentada

Se ha documentado toda la arquitectura modular del sistema, incluyendo:

* controladores/: lógica de negocio por cada ventana o componente funcional.
* vistas/: clases de interfaces gráficas con sus elementos visuales.
* modelos/: funciones de acceso a la base de datos PostgreSQL.
* utilidades/: funciones auxiliares reutilizables.
* tests/: tests unitarios con cobertura descriptiva por módulo.

6.4.3. Visualización de la documentación

La documentación generada con pdoc se almacena en la carpeta /docs/ del proyecto. Basta con abrir el archivo index.html en un navegador para explorar todas las secciones documentadas.

A continuación, se muestra unas imágenes de ejemplo del entorno de documentación generado:

Pantalla principal

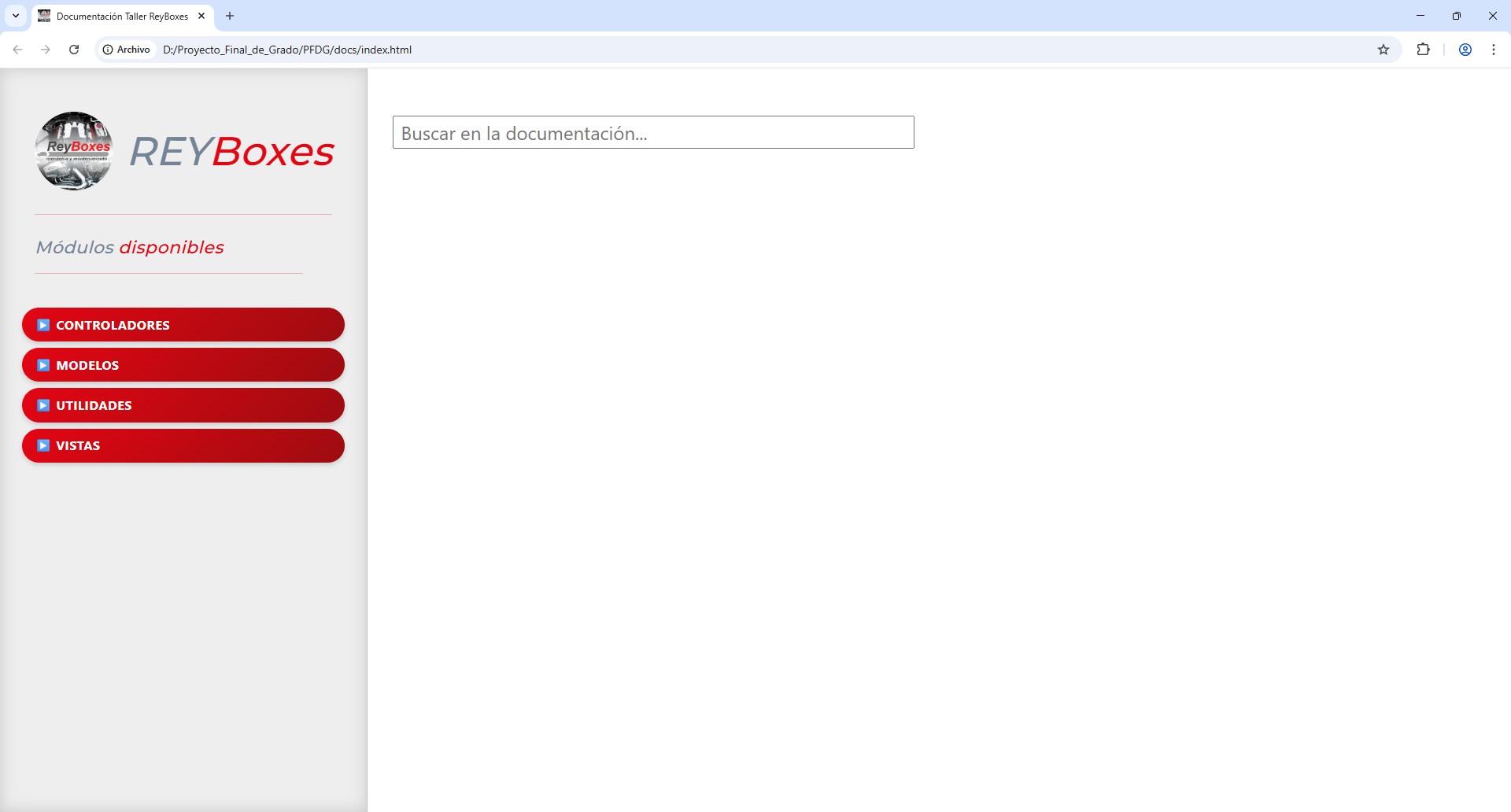


Ilustración 27 Vista principal de la documentación técnica del sistema ReyBoxes, con módulos organizados y buscador integrado.

Pantalla del módulo cotroladores.cliente\_controlador



Ilustración 28 Vista de la documentación generada para el módulo clientes\_controlador, con navegación lateral y código fuente comentado.

7. Manual de Usuario

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

El contenido generado por IA puede ser incorrecto.7.1. Acceso al sistema

Ilustración 29 Ventana de inicio de sesión con campos de usuario y contraseña, botones de acceso y enlace para recuperación.

Esta es la pantalla inicial del programa ReyBoxes, donde los usuarios deben identificarse para acceder al sistema. El diseño ha sido pensado para ofrecer una experiencia intuitiva y agradable.

7.1.1 Descripción general

La ventana de acceso está compuesta por los siguientes elementos:

* Logotipo central: Representa la identidad visual del taller ReyBoxes.
* Título "Iniciar Sesión": Con un diseño llamativo, destacando la palabra "Sesión" en rojo.
* Campo "Inserte su nombre": Obligatorio. Debe introducirse el nombre de usuario en mayúsculas (aunque el sistema lo transforma automáticamente).
* Campo "Contraseña": Obligatorio. Se introduce la contraseña correspondiente al usuario.
* Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

  El contenido generado por IA puede ser incorrecto.Botón rojo "Entrar": Permite acceder al sistema si las credenciales son correctas.

Ilustración 30 Mensaje emergente confirmando acceso exitoso al sistema con saludo personalizado.

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

El contenido generado por IA puede ser incorrecto.Si los datos son erróneos, se muestra un mensaje indicando el fallo.

Ilustración 31 Alerta de error indicando que el nombre o la contraseña introducidos no son válidos.

* Botón rojo "Salir": Cierra completamente el programa.
* Enlace "¿Olvidaste tu contraseña?": Lleva al proceso de recuperación de cuenta, donde se puede solicitar un nuevo acceso por correo electrónico.

7.1.2 Instrucciones paso a paso para iniciar sesión

1. Escriba su nombre de usuario en el primer campo (por ejemplo: CRESNIK).
2. Escriba su contraseña personal en el segundo campo.
3. Haga clic en el botón "Entrar" o pulse Enter.
4. Si los datos son correctos, se abrirá automáticamente el menú principal del sistema.
5. En caso de error, revise las mayúsculas o solicite una nueva contraseña desde el enlace inferior.

7.2. Recuperación de cuenta

Esta pantalla aparece al hacer clic en el enlace "¿Olvidaste tu contraseña?" en la ventana de inicio de sesión. Permite al usuario recuperar el acceso a su cuenta mediante el envío de un código de verificación a su correo electrónico registrado.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Ilustración 32 Pantalla para recuperar cuenta mediante el envío de un código al correo electrónico ingresado.

7.2.1. Descripción general

Elementos visibles en esta pantalla:

* Título "Recuperar cuenta": Formato consistente con el resto del programa. Texto de instrucciones:
* Breve mensaje explicativo: "Introduce el correo electrónico asociado a tu cuenta:" Ayuda al usuario a entender el propósito de esta pantalla.
* Campo de entrada de correo electrónico: Debe introducirse un correo electrónico válido registrado en el sistema. Tiene un icono de sobre a la izquierda. Si el formato del correo no es válido, el botón "Enviar Código" se desactiva.
* Botón rojo "Enviar Código": Envía un correo con un código de verificación temporal. Solo se activa si el correo es válido y está registrado.
* Botón rojo "Volver": Permite regresar directamente a la ventana de inicio de sesión.

7.2.2 Instrucciones paso a paso para recuperar el acceso

1. Introduzca el correo electrónico asociado a su cuenta.
2. Asegúrese de que el correo esté bien escrito y tenga el formato correcto (ejemplo: usuario@dominio.com).
3. Haga clic en "Enviar Código".
4. Interfaz de usuario gráfica, Texto, Aplicación

   El contenido generado por IA puede ser incorrecto. Si el correo está registrado, se enviará un código temporal al buzón indicado. Al cerrarse la ventana de aviso, se abrirá automáticamente la pantalla de verificación.

Ilustración 33 Notificación confirmando que el código de recuperación fue enviado al correo del usuario.

En caso contrario, se mostrará un mensaje indicando que el correo no está registrado

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Ilustración 34 Alerta que indica que no existe ningún usuario registrado con el correo introducido.

Puede pulsar "Volver" en cualquier momento para regresar a la pantalla de inicio de sesión.

7.2.3. Verificación de código

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.Después de solicitar el código en la pantalla anterior, se abre esta ventana para validar el acceso mediante el código temporal recibido por correo electrónico. Este paso es obligatorio antes de poder restablecer la contraseña.

Ilustración 35 Pantalla para ingresar el código recibido por correo con temporizador de expiración visible.

7.2.4 Descripción general

Esta pantalla contiene los siguientes elementos:

* Título "Verificar Código": Continúa con la estética del programa.
* Campo de entrada del código: Campo central donde se introduce el código numérico recibido. Solo se acepta el código correcto, que es de un solo uso y temporal.
* Contador regresivo: Muestra el tiempo restante para que el código sea válido siendo un máximo de 5 minutos. Si el tiempo expira, el código ya no será aceptado y se deberá solicitar uno nuevo.
* Botón "Verificar": Verifica si el código es correcto y está dentro del tiempo de validez. Si es válido, lleva automáticamente a la pantalla de restablecimiento de contraseña en caso contrario lo indicara con un mensaje de aviso.
* Botón "Volver": Regresa a la pantalla anterior para reenviar el código si es necesario.

7.2.5 Instrucciones paso a paso para verificar el código

1. Revise su correo electrónico y localice el mensaje enviado por el sistema ReyBoxes.
2. Copie el código recibido.
3. Péguelo o escríbalo en el campo central de la pantalla.
4. Observe el tiempo restante: debe introducirse antes de que expire.
5. Pulse el botón "Verificar".
6. Si el código es correcto y aún válido, se abrirá la siguiente pantalla para restablecer su contraseña. Si hay algún error, revise que el código sea el correcto o vuelva a solicitar uno nuevo.

7.2.6. Nueva contraseña

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.Una vez verificado el código correctamente, se muestra esta pantalla que permite al usuario restablecer su contraseña de acceso de forma segura.

Ilustración 36 Formulario para establecer una nueva contraseña tras la verificación del código recibido.

7.2.7 Descripción general

Esta pantalla incluye los siguientes elementos:

* Título "Nueva Contraseña": Continúa con el estilo visual coherente del programa.
* Campo "Nueva contraseña": Aquí se introduce la nueva contraseña deseada. Es obligatorio completarlo.
* Campo "Repetir contraseña": Debe introducirse exactamente la misma contraseña para confirmar. Si los valores no coinciden, no se podrá continuar.
* Botón rojo "Guardar": Guarda la nueva contraseña y la actualiza en la base de datos. Tras guardar, el sistema redirige al login.
* Botón rojo "Volver": Permite regresar a la pantalla anterior (verificación).

7.3. Menú principal (Dashboard)

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.Una vez iniciada la sesión correctamente, el usuario accede a esta pantalla central del sistema. Desde aquí se pueden abrir todas las funcionalidades del programa, según los permisos asociados al rol del usuario (Administrador, Mecánico o Administrativo).

Ilustración 37 Menú principal del sistema con accesos rápidos a todas las funciones según el rol de administrador.

7.3.1 Descripción general

Elementos destacados en esta ventana:

* Mensaje de bienvenida personalizado: Se muestra el nombre del usuario en mayúsculas (ejemplo: CRESNIK).
* Debajo aparece el rol del usuario (por ejemplo: Administrador).
* Panel de botones centrales: Se organiza en una cuadrícula de botones cuadrados con icono y texto. Cada botón abre una sección distinta del sistema.
* Los botones se muestran u ocultan según el rol del usuario:
  + Administrador: accede a todas las funciones.
  + Mecánico: ve solo "Fichar", "Historial fichaje", "Vehículos" y "Presupuestos".
  + Administrativo: ve "Fichar", "Historial fichaje", "Clientes", "Recepcionamiento", "Reimpresión" y "Presupuestos".
  + Compra/Venta: tiene acceso a "Fichar", "Historial fichaje",*"*Recepcionamiento*"*, *"*Vehículos*"* y al módulo de *"*Compraventa*"* (registro, gestión y reimpresión de operaciones comerciales).

Botones disponibles (ordenados de izquierda a derecha, fila a fila):

| Funcionalidad | Descripción breve |
| --- | --- |
| Fichar | Registro de entrada o salida del personal. |
| Historial fichaje | Consulta de fichajes previos. |
| Crear usuarios | Alta de nuevos usuarios del sistema. |
| Clientes | Gestión completa de clientes. |
| Vehículos | Gestión de vehículos asociados a clientes. |
| Recepcionamiento | Registro de vehículos al llegar al taller. |
| Compraventa | Gestión de compra y venta de vehículos. |
| Presupuestos | Creación y seguimiento de presupuestos. |
| Reimpresión recepcionamientos | Permite volver a imprimir documentos anteriores. |

7.3.2 Instrucciones de uso

1. Revise su nombre y rol en la parte superior.
2. Seleccione la sección deseada pulsando sobre el botón correspondiente.
3. Cada botón abrirá una nueva ventana donde podrá trabajar según los permisos otorgados.

7.4. Fichajes

Un dibujo con letras

El contenido generado por IA puede ser incorrecto.Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.El sistema ReyBoxes permite a los trabajadores registrar sus entradas y salidas del taller de forma rápida y visual. Esta funcionalidad está disponible para todos los empleados, y los datos quedan guardados en la base de datos para su posterior consulta por parte del administrador.

Ilustración 38 Botón para registrar entrada o salida del trabajador mediante fichaje horario.

Ilustración 39 Pantalla para registrar la hora exacta de entrada o salida del personal.

7.4.1 Descripción general

* La ventana de fichaje está diseñada para ser clara, compacta y directa: Hora actual: Se muestra en tiempo real, con formato HH:mm:ss.
* Opciones de tipo de fichaje, el usuario debe marcar una sola opción:
  + Entrada
  + Salida

Solo una puede estar activa al mismo tiempo, siendo obligatorio elegir una para poder fichar.

* Botón "Confirmar": Registra el fichaje seleccionado con la hora exacta. Al pulsarlo, se guarda el registro en la base de datos con el ID del usuario, tipo (entrada o salida), y hora.
* Botón "Volver": Permite regresar al menú principal sin realizar ningún fichaje.

7.4.2 Instrucciones paso a paso para fichar

1. Verifique que la hora mostrada es la correcta.
2. Seleccione el tipo de fichaje:
3. Marque "Entrada" al llegar al trabajo o marque "Salida" al finalizar la jornada.
4. Interfaz de usuario gráfica

   El contenido generado por IA puede ser incorrecto.Imagen que contiene Interfaz de usuario gráfica

   El contenido generado por IA puede ser incorrecto.Pulse el botón "Confirmar".

Ilustración 40 Mensaje de confirmación indicando que el fichaje de salida se registró correctamente.

Ilustración 41 Mensaje de confirmación indicando que el fichaje de entrada se registró correctamente.

1. Se registrará el fichaje y se volverá al menú principal automáticamente.
2. Si desea cancelar, simplemente pulse "Volver".

7.5. Historial de Fichajes

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.Desde esta pantalla es posible consultar todos los registros de entrada y salida realizados por los empleados. La información se presenta de forma ordenada y permite exportar los datos a distintos formatos para su análisis o archivo.

Ilustración 42 Ventana con listado de fichajes realizados, incluyendo fecha, tipo y empleado, con opciones de exportación.

Ilustración 43 Botón de acceso al historial de fichajes registrados por el usuario o administrador.

7.5.1 Descripción general

La pantalla está dividida en dos partes principales: la tabla de registros y los botones de acción.

Contiene tres columnas:

Tabla 21 Tabla explicativa del contenido de cada columna en el historial de fichajes.

| Columna | Contenido |
| --- | --- |
| Fecha y hora | Momento exacto del fichaje. Ej.: 2025-05-07 15:33:09. |
| Tipo | ENTRADA o SALIDA. |
| Empleado | Nombre del trabajador que realizó el fichaje. |

La tabla muestra todos los registros cargados desde la base de datos, ordenados cronológicamente.

La primera columna numera los registros para facilitar la lectura.

Cada fila alterna colores para mayor legibilidad.

Botones inferiores

Exportar CSV: Genera un archivo .csv compatible con Excel y otros programas de hojas de cálculo, ideal para análisis o archivado externo.

Tabla

El contenido generado por IA puede ser incorrecto.Exportar PDF: Crea un informe en PDF con el logo de ReyBoxes, nombre de empresa, tabla de datos y pie de página, el formato es profesional y apto para presentación.

Ilustración 44 Informe PDF de fichajes con logotipo, datos de empresa, trabajador y registro detallado por fecha.

Volver: Regresa al menú principal sin realizar cambios.

7.5.2 Diferencias según el rol del usuario

Administrador

* Visualiza todos los fichajes de todos los usuarios del sistema.
* El campo "Empleado" muestra el nombre de cada trabajador.

Mecánico / Administrativo

* Solo pueden ver sus propios fichajes.
* El campo "Empleado" muestra únicamente su propio nombre.
* Esto garantiza privacidad y control según el nivel de acceso.

7.5.3 Instrucciones paso a paso para consultar fichajes

1. Abra la pantalla desde el botón "Historial Fichaje" en el menú principal.
2. Revise la tabla de registros.
3. Si lo desea:
4. Pulse Exportar CSV para obtener una hoja de cálculo.
5. Pulse Exportar PDF para generar un informe formal.
6. Pulse Volver para salir de esta ventana.

7.6. Crear usuarios

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.Un dibujo de un perro

El contenido generado por IA puede ser incorrecto.Esta ventana permite registrar nuevos usuarios en el sistema ReyBoxes. Solo los usuarios con el rol Administrador tienen acceso a esta función.

Ilustración 45 Formulario de registro para crear un nuevo usuario con datos personales, contraseña y rol asignado.

Ilustración 46 Botón para acceder al formulario de alta de nuevos usuarios en el sistema.

7.6.1 Descripción general

La interfaz está organizada dentro de un recuadro central con todos los campos necesarios para el alta de un nuevo usuario.

Campos del formulario

* Nombre: Campo obligatorio. Se guardará en mayúsculas automáticamente.
* Apellido: Campo opcional, pero recomendable para identificar claramente al usuario.
* Correo electrónico: Se utiliza para recuperación de cuenta, debe tener formato válido (usuario@dominio.com).
* Contraseña: Clave de acceso del usuario.
* Repetir contraseña: Debe coincidir exactamente con la anterior.
* Selecciona un rol, Menú desplegable con las opciones:
  + Administrador
  + Mecánico
  + Administrativo
  + Compraventa

Es obligatorio seleccionar un rol para continuar.

* Botones de acción
  + Crear usuario: Valida los datos y guarda el nuevo usuario, muestra mensaje de éxito o error según el resultado.
* Limpiar: Vacía todos los campos del formulario, util si se desea empezar de nuevo sin cerrar la ventana.
* Volver: Regresa al menú principal sin guardar nada.

7.6.2 Instrucciones paso a paso para registrar un nuevo usuario

1. Complete el campo Nombre (obligatorio).
2. Ingrese el Apellido (opcional).
3. Escriba un correo electrónico válido.
4. Introduzca la contraseña deseada.
5. Repita la contraseña para confirmar.
6. Seleccione un rol de usuario en el desplegable.
7. Haga clic en "Crear usuario".
8. Interfaz de usuario gráfica, Aplicación

   El contenido generado por IA puede ser incorrecto.Interfaz de usuario gráfica, Aplicación

   El contenido generado por IA puede ser incorrecto.Si todo es correcto, el usuario quedará registrado y se limpiaran todos los campos para poder ingresar un nuevo usuario si se desea.

Ilustración 47 Ejemplo de formulario completado para registrar un nuevo usuario con rol de Mecánico.

Ilustración 48 Mensaje de confirmación indicando que el nuevo usuario fue creado exitosamente.

1. En caso de error (usuario repetido, correos duplicados, etc.) se mostrará un aviso.

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

Ilustración 49 Mensaje de error que indica que los campos de contraseña y confirmación no coinciden.

Ilustración 50 Mensaje de error indicando que el correo electrónico ya está registrado en el sistema.

Ilustración 51 Mensaje de error que informa que el nombre de usuario ya está en uso y debe ser distinto.

1. Puede usar el botón "Limpiar" para reiniciar el formulario.
2. Pulse "Volver" para regresar sin realizar ningún cambio.

7.7. Gestión de clientes

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.Un dibujo de una cara feliz

El contenido generado por IA puede ser incorrecto.Desde esta ventana se puede registrar un nuevo cliente, buscar clientes existentes por nombre, DNI o teléfono, así como modificar o eliminar sus datos. Es una de las funciones básicas del sistema, accesible para usuarios con permisos administrativos.

Ilustración 52 Formulario para registrar, modificar o eliminar clientes con búsqueda por nombre, DNI o teléfono.

Ilustración 53 Botón de acceso al módulo de gestión de clientes del sistema.

7.7.1 Descripción general

Barra superior de búsqueda

Incluye tres campos:

* Nombre y apellidos: Permite buscar escribiendo parcial o totalmente el nombre del cliente. Incluye autocompletado.
* DNI: Búsqueda directa por número de documento. También tiene autocompletado.
* Teléfono: Filtra resultados por número de teléfono del cliente. Ideal cuando se desconoce el nombre exacto.
* Al pulsar Tab o Enter tras escribir en cualquiera de estos campos, el sistema completa automáticamente los datos del cliente si existe.

Formulario de cliente

Incluye los siguientes campos editables:

* Nombre
* Primer apellido
* Segundo apellido
* DNI
* Teléfono
* Correo electrónico
* Dirección
* Código postal
* Localidad
* Provincia
* Observaciones (opcional)

Botones de acción

Tabla 22 Tabla que describe la función de cada botón en el formulario de gestión de clientes.

| Botón | Función |
| --- | --- |
| Registrar | Guarda un nuevo cliente con los datos actuales. |
| Modificar | Actualiza los datos del cliente existente. (Desactivado si no hay cliente seleccionado). |
| Eliminar | Borra al cliente de la base de datos. (Desactivado si no hay cliente seleccionado). |
| Limpiar | Borra todos los campos del formulario. |
| Volver | Regresa al menú principal. |

7.7.2 Instrucciones paso a paso

Para registrar un nuevo cliente:

1. Complete todos los campos necesarios.
2. Pulse el botón rojo Registrar.
3. Interfaz de usuario gráfica, Aplicación

   El contenido generado por IA puede ser incorrecto.Aparecerá un mensaje de confirmación.

Ilustración 54 Mensaje de confirmación indicando que el cliente fue registrado exitosamente.

Para buscar un cliente existente:

1. Interfaz de usuario gráfica, Texto, Aplicación

   El contenido generado por IA puede ser incorrecto.Escriba parte del nombre, DNI o teléfono.

Ilustración 55 Campo de búsqueda con autocompletado que sugiere nombres de clientes coincidentes en tiempo real.

1. Pulse Tab o Enter.
2. El sistema rellenará automáticamente los campos con los datos del cliente.

Para modificar un cliente:

1. Busque al cliente usando los campos superiores.
2. Cambie los datos que desee.
3. Pulse Modificar para guardar.

Para eliminar un cliente:

1. Busque al cliente.
2. Pulse Eliminar (si está habilitado).
3. Se pedirá confirmación antes de borrar.

7.7.3 Errores controlados en la gestión de clientes

1. Campos obligatorios vacíos

Descripción: Al pulsar “Registrar” sin haber rellenado los campos necesarios.

Solución: El botón no permite avanzar o aparece un mensaje como: "Por favor, complete todos los campos obligatorios antes de registrar un cliente."

2. Formato inválido de DNI

Descripción: Se introduce un DNI incorrecto (letras mal ubicadas, longitud incorrecta...).

Solución: Se muestra un mensaje como: "El DNI introducido no es válido. Verifique el formato."

3. Correo electrónico mal escrito

Descripción: El campo de correo no contiene un email válido (usuario@dominio.com).

Solución: "El correo electrónico no tiene un formato válido."

4. Teléfono no numérico o con caracteres extraños

Descripción: Se escribe texto o símbolos no permitidos en el campo de teléfono.

Solución: "El número de teléfono debe contener solo dígitos."

5. Cliente ya existente (por DNI o nombre)

Descripción: Se intenta registrar un cliente con un DNI o nombre que ya existe en la base de datos.

Solución: "Ya existe un cliente con ese DNI o nombre. Use la función de búsqueda para modificarlo."

6. No se ha seleccionado un cliente para modificar o eliminar

Descripción: Se pulsa “Modificar” o “Eliminar” sin haber buscado previamente un cliente.

Solución: "Debe seleccionar un cliente existente antes de modificar o eliminar."

7. Error interno de base de datos

Descripción: Fallo de conexión con la base de datos o error inesperado al guardar.

Solución: "Ocurrió un error al guardar los datos. Intente nuevamente o contacte al administrador."

7.8. Gestión de vehículos

Esta pantalla permite asociar vehículos a clientes, registrar sus datos técnicos, buscar por matrícula o cliente, y mantener un historial organizado por categorías y tipo.

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.Un dibujo con letras

El contenido generado por IA puede ser incorrecto.

Ilustración 56 Formulario completo para registrar, buscar o editar vehículos asociados a clientes mediante múltiples criterios.

Ilustración 57 Botón de acceso al módulo de gestión y registro de vehículos asociados a clientes.

7.8.1 Descripción general

La ventana está dividida en dos columnas principales:

Búsqueda

En la parte superior hay tres campos de búsqueda:

* Buscar por nombre y apellidos
* Buscar por DNI
* Buscar por matrícula

Si se escribe en nombre o DNI, el sistema cargará los datos del cliente, los cuales no son editables en esta pantalla.

Si se escribe en matrícula, se cargan automáticamente tanto los datos del cliente como los del vehículo correspondiente.

Datos del cliente (columna izquierda)

Se muestran los datos personales del cliente asociado al vehículo:

Nombre y apellidos, DNI, teléfono, Email, dirección, código postal, localidad, provincia

Estos campos se rellenan automáticamente al buscar al cliente.

Datos del vehículo (columna derecha)

Aquí se ingresan los datos del vehículo:

Matrícula , Marca, Modelo, Color, Año, Combustible (desplegable), Nº de bastidor, Categoría del vehículo (desplegable), Tipo de vehículo (desplegable) , Observaciones adicionales

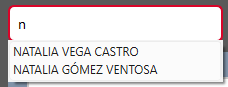
Botones de acción

Tabla 23 Tabla descriptiva de las funciones de los botones en la interfaz de gestión de vehículos.

| Botón | Función |
| --- | --- |
| Registrar | Guarda un nuevo vehículo asociado al cliente. |
| Modificar | Actualiza los datos del vehículo cargado. |
| Eliminar | Elimina el vehículo actual del sistema. |
| Limpiar | Borra todos los campos de la interfaz. |
| Volver | Regresa al menú principal. |

7.8.2 Instrucciones paso a paso

Registrar un nuevo vehículo

1. Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

   El contenido generado por IA puede ser incorrecto.Busque al cliente (nombre, DNI).

Ilustración 58 Campo de autocompletado mostrando coincidencias de nombres para asignar vehículos a clientes.

Ilustración 59 Campo con sugerencias automáticas para seleccionar un cliente a partir de su DNI.

1. Complete los datos del vehículo.
2. Interfaz de usuario gráfica, Aplicación

   El contenido generado por IA puede ser incorrecto.Pulse Registrar. Aparecerá un mensaje de confirmación.

Ilustración 60 Formulario de vehículo completado y confirmado con mensaje de registro exitoso.

Modificar datos del vehículo

1. Busque al cliente y seleccione el vehículo deseado.
2. Edite los campos necesarios.
3. Interfaz de usuario gráfica

   El contenido generado por IA puede ser incorrecto.Pulse Modificar para guardar los cambios.

Ilustración 61 Mensaje informativo confirmando que los datos del vehículo fueron actualizados correctamente.

Eliminar un vehículo

1. Busque al cliente y el vehículo correspondiente.
2. Interfaz de usuario gráfica

   El contenido generado por IA puede ser incorrecto.Interfaz de usuario gráfica, Texto

   El contenido generado por IA puede ser incorrecto.Pulse Eliminar. Se pedirá confirmación antes de borrar.

Ilustración 62 Mensaje de confirmación informando que el vehículo fue eliminado correctamente del sistema.

Ilustración 63 Diálogo de confirmación para eliminar un vehículo del sistema de forma permanente.

Limpiar la pantalla

1. Pulse Limpiar para vaciar todos los campos y reiniciar el formulario.

8.9. Recepcionamiento de vehículos

Logotipo, nombre de la empresa

El contenido generado por IA puede ser incorrecto.Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.La pantalla de recepcionamiento permite registrar la entrada de un vehículo al taller con todos los datos relevantes del cliente, del vehículo, y del motivo de la visita. Esta ficha queda registrada para su posterior consulta, impresión o generación de presupuesto.

Ilustración 64 Botón de acceso al módulo de recepcionamiento de vehículos en el taller.

Ilustración 65 Pantalla para registrar la recepción de vehículos en secciones desplegables y opciones finales de acción.

8.9.1 Descripción general

La interfaz está dividida en cuatro secciones plegables y tres botones de acción en la parte inferior.

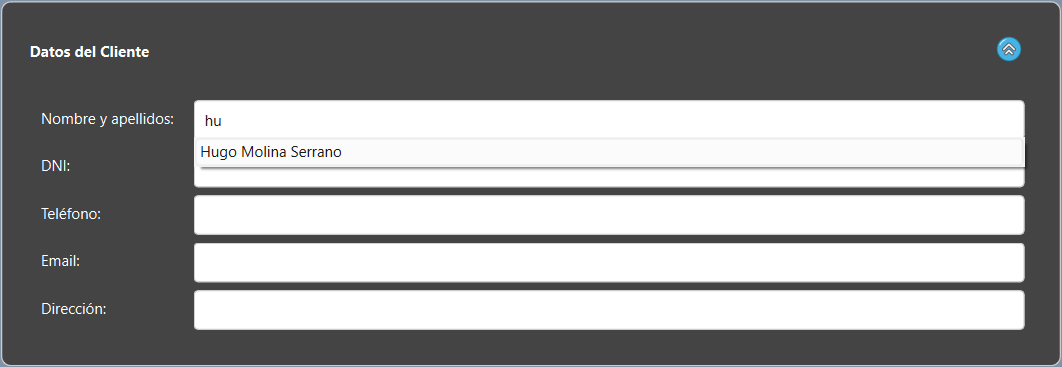
1. Datos del Cliente

Ilustración 66 Sección del recepcionamiento con autocompletado para seleccionar un cliente y cargar sus datos.

Campos incluidos:

Nombre y apellidos, DNI , Teléfono, Correo electrónico, Dirección

Estos datos pueden autocompletarse automáticamente al seleccionar un cliente existente a través de los campos “Nombre y apellidos” o “DNI”.

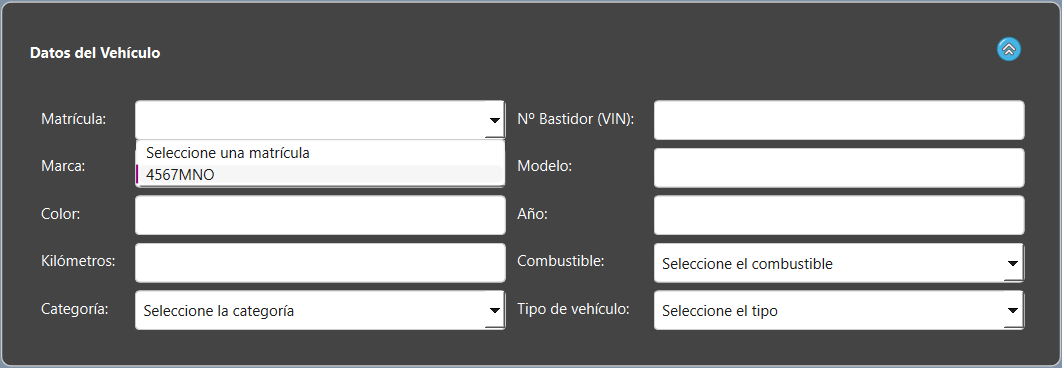
2. Datos del Vehículo

Ilustración 67 Sección para seleccionar una matrícula y completar automáticamente los datos técnicos del vehículo.

Campos incluidos:

Matrícula, número de bastidor (VIN), Marca, modelo, color, año, kilómetros, Combustible (desplegable), Categoría del vehículo (desplegable), Tipo de vehículo (desplegable)

Estos datos se pueden autocompletar si previamente, en la sección anterior, se ha seleccionado un cliente.

En ese caso, el sistema cargará automáticamente los vehículos registrados a nombre del cliente, permitiendo seleccionar uno desde el campo "Matrícula".

Esta lógica evita errores y garantiza que el vehículo esté correctamente vinculado a su propietario.

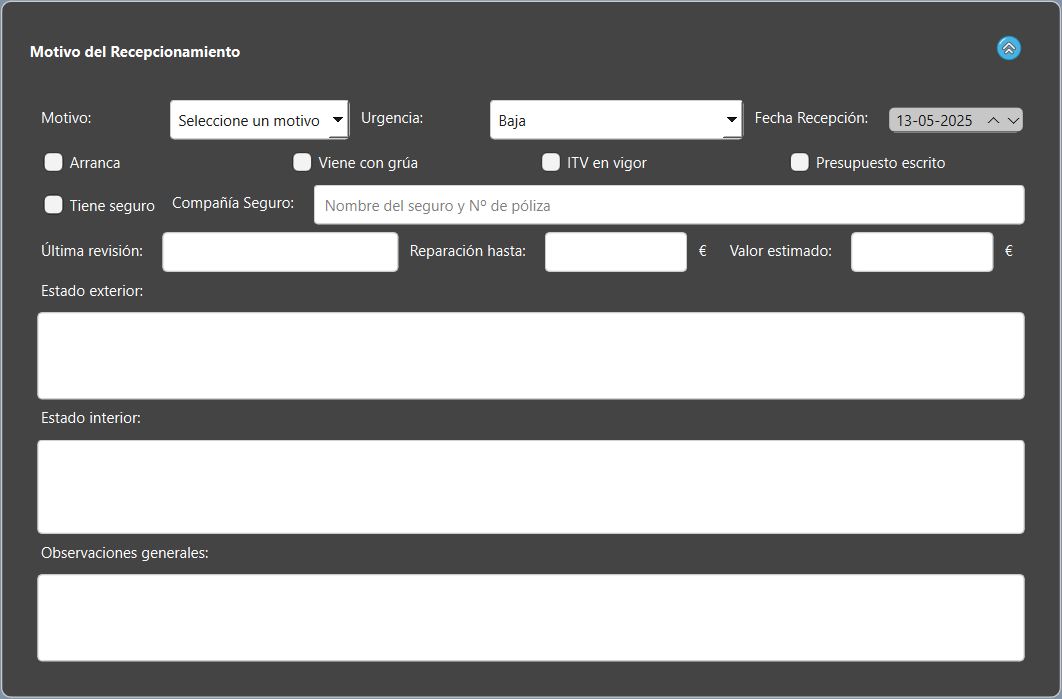
3. Motivo del Recepcionamiento

Ilustración 68 Formulario detallado para registrar el motivo del recepcionamiento, estado del vehículo y condiciones del seguro.

Esta es la sección más amplia del formulario y tiene como objetivo documentar por qué el vehículo ingresa al taller y en qué condiciones lo hace.

Aquí el usuario debe indicar y registrar información esencial sobre la entrada del vehículo. El sistema agrupa los datos de forma estructurada y visualmente clara.

Campos incluidos y cómo rellenarlos:

* Motivo de la entrada *(desplegable)*: Selecciona la razón por la que el cliente trae el vehículo. Ejemplos: revisión, avería, mantenimiento, etc.
* Grado de urgencia *(desplegable)*: Indica la prioridad del trabajo. Esto ayuda a organizar los turnos de intervención.
* Fecha de recepción *(por defecto, la fecha actual)*: Se puede modificar si la recepción se documenta más tarde.
* Marcas de verificación *(checkboxes)*:
  + Arranca: marca si el vehículo está en condiciones de arrancar.
  + Viene con grúa: marca si ha sido traído en plataforma.
  + ITV en vigor: verifica si tiene inspección técnica válida.
  + Presupuesto escrito: indica si el cliente ya trajo un presupuesto por escrito.
  + Tiene seguro: activa los siguientes campos:
    - Compañía aseguradora
    - Número de póliza
* Última revisión: Fecha de la última revisión mecánica del vehículo, si se conoce.
* Valor estimado (€): Estimación económica del vehículo, útil en casos de siniestro o seguro.
* Estado exterior e interior *(campos de texto amplio)*: Describir daños visibles, suciedad, o condiciones generales. Ayuda a evitar disputas posteriores.
* Observaciones generales: Cualquier comentario adicional sobre el cliente, el vehículo o acuerdos especiales.

Consejo: Rellenar con detalle esta sección permite proteger tanto al cliente como al taller ante posibles reclamaciones, y da soporte documental a futuras intervenciones.

Interfaz de usuario gráfica, Sitio web

El contenido generado por IA puede ser incorrecto.4. Entrega del documento

Ilustración 69 Sección final para confirmar, guardar, imprimir o enviar el recepcionamiento con firma del cliente y ruta destino.

Esta sección permite finalizar el proceso de recepcionamiento generando un documento con todos los datos introducidos y gestionando su salida (impresión, envío o guardado).

Su objetivo es que el cliente firme digitalmente la entrega y el sistema guarde una copia digital del documento generado, con opciones para impresión o correo.

Campos incluidos y cómo utilizarlos:

* Nº Recepcionamiento: Número identificador del documento se genera de manera automática y se sigue una numeración predefinida.
* Imprimir *(checkbox)*: Si se marca, el sistema abrirá una ventana de impresión al confirmar el documento.
* Enviar por correo *(checkbox)*: Permite enviar el documento en PDF al cliente. Requiere rellenar el campo “Correo destino”.
* Correo destino: Dirección de correo electrónico a la que se enviará el documento.
  + Si se deja en blanco, el sistema intentará usar la dirección registrada del cliente.
  + Si se introduce un nuevo correo aquí, prevalecerá sobre el registrado.
* Firma del cliente: Área donde el cliente puede firmar directamente en pantalla (con ratón o dispositivo táctil).
* Botón “Activar firma” (icono con lápiz): Habilita el área de firma para que el cliente pueda firmar.
* Botón “Borrar firma” (icono con cruz): Borra la firma actual para que pueda volver a realizarse.
* Guardar en la ruta por defecto *(checkbox)*: Si se marca, el sistema guardará automáticamente el PDF en la carpeta predeterminada del sistema.
* Ruta personalizada: Campo donde se puede escribir o seleccionar manualmente una carpeta de destino para guardar el documento.
* Botón “Seleccionar carpeta”: Abre un explorador de archivos para elegir manualmente la ubicación de guardado.

¿Por qué se organiza así?

* Permite flexibilidad: el documento puede ser impreso, enviado o almacenado según la necesidad del momento.
* Mejora la trazabilidad: el número de recepcionamiento sirve como identificador único.
* La firma electrónica permite prescindir del papel, agilizando procesos y archivado.
* La posibilidad de indicar manualmente una ruta permite guardar cada documento en carpetas por mes, cliente, o proyecto.

Instrucciones paso a paso

1. Introduce o verifica el número de recepcionamiento.
2. Marca las opciones deseadas: Imprimir, Enviar por correo y/o Guardar.
3. Si deseas enviar por email, escribe un correo destino, o deja que el sistema use el del cliente.
4. Pulsa “Activar firma” y solicita al cliente que firme en la zona correspondiente.
5. Si es necesario repetir, pulsa “Borrar firma”.
6. Si no deseas usar la ruta predeterminada, pulsa “Seleccionar carpeta” y elige dónde guardar el PDF.
7. Al pulsar “Confirmar”, el sistema generará el documento, aplicará las opciones marcadas y cerrará la operación.

Botones de acción

Tabla 24 Tabla que explica la función de cada botón al finalizar el proceso de recepcionamiento.

| Botón | Función |
| --- | --- |
| Confirmar | Guarda toda la información registrada. |
| Borrar todo | Limpia todos los campos del formulario. |
| Cancelar | Cierra la ventana sin guardar los cambios. |

7.9.2 Instrucciones paso a paso para recepcionar un vehículo

1. Seleccionar o introducir los datos del cliente

Puedes escribir el nombre, DNI o teléfono para que se autocompleten los campos.

* + Si el cliente no existe aún, debe ser registrado previamente desde la sección correspondiente del menú principal.
  + Texto

    El contenido generado por IA puede ser incorrecto.Si intentas confirmar el recepcionamiento sin un cliente válido, el sistema mostrará un mensaje de error y no permitirá continuar.

Ilustración 70 Error crítico al guardar indicando que falta el campo obligatorio cliente\_id en la base de datos.

2. Completar los datos del vehículo

Si el vehículo ya está en el sistema, introduce la matrícula y se cargarán los campos automáticamente.

Si el vehículo no está registrado, debe añadirse previamente a nombre de un usuario existente usando la sección Gestión de Vehículos.

No es posible recepcionar un vehículo nuevo sin vincularlo primero a un cliente ya existente en la base de datos.

3. Rellenar el motivo del recepcionamiento

Selecciona:

Motivo de la entrada al taller (revisión, avería, etc.).

Grado de urgencia.

La fecha de recepción se introduce por defecto, la cual será la actual).

Marca las opciones que correspondan:

¿Arranca? ¿Viene con grúa? ¿Tiene seguro? etc.

Si tiene seguro, completa los datos de la compañía y el número de póliza.

Introduce la fecha de última revisión, fecha límite estimada de reparación y valor económico estimado.

Describe en detalle el estado exterior e interior del vehículo.

Añade cualquier observación general relevante.

4. (Opcional) Confirmar entrega de documento

Según la configuración del sistema:

El documento puede imprimirse o enviarse por correo electrónico.

El correo se enviará a la dirección registrada del cliente.

Si se introduce otro correo manualmente, este prevalecerá.

Si el cliente no tiene correo o no se introduce ninguno, el sistema avisará con un mensaje.

En todos los casos, el sistema guardará por defecto un PDF con los datos del recepcionamiento en local.

5. Pulsar el botón "Confirmar"

El sistema validará los campos y registrará toda la información.

Se mostrará un mensaje de éxito y el documento quedará guardado en la base de datos.

6. Si quieres reiniciar el formulario, pulsa "Borrar todo".

7. Para salir sin guardar nada, pulsa "Cancelar".

7.10. Compraventa de vehículos

Esta sección del sistema permite registrar operaciones de compraventa de vehículos, ya sea cuando el taller adquiere un vehículo para su stock o vende uno de su inventario a un cliente.

El módulo está diseñado para ofrecer trazabilidad completa de todas las transacciones de entrada y salida de vehículos, vinculadas a clientes registrados. De este modo, el sistema puede generar reportes, facturas y registros históricos, útiles tanto para gestión interna como para posibles auditorías.

Logotipo

El contenido generado por IA puede ser incorrecto.Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Ilustración 71 Botón de acceso al módulo de compraventa de vehículos dentro del sistema.

Ilustración 72 Pantalla de compraventa con secciones plegables para registrar cliente, vehículo y condiciones de venta.

Descripción general

La interfaz está organizada en tres bloques plegables, que se activan en función del tipo de operación seleccionada.

Tipo de operación: Primer elemento del formulario, desplegable con al menos dos opciones:

* Compra de vehículo al cliente
* Venta de vehículo al cliente

Según la opción elegida, se activarán únicamente los bloques de datos necesarios. Esto evita errores y hace la interfaz más clara.

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.7.10.1 Compra de vehículo al cliente

Ilustración 73 Formulario de compraventa que recopila datos del cliente, del vehículo y permite generar, firmar y guardar el contrato.

Datos del Cliente

Nombre completo

DNI

Teléfono

Email

Dirección

Si escribes el nombre o DNI de un cliente ya registrado, los campos se autocompletarán automáticamente.

Si el cliente no está registrado, el sistema mostrará un mensaje indicándolo. A continuación, se ofrece la opción de registrar rápidamente al cliente con los datos mínimos requeridos, para poder continuar con la operación de compra o venta.

Posteriormente, el cliente podrá ser modificado desde el módulo de Gestión de Clientes, donde se podrán completar el resto de los datos (dirección, correo electrónico, observaciones, etc.).

Esta funcionalidad está pensada para agilizar la operación comercial sin bloquear el proceso, y garantizar que toda compraventa quede correctamente asociada a un cliente, aunque no se dispongan en ese momento de todos los datos.

Datos del Vehículo a comprar

Este bloque aparece únicamente cuando la operación seleccionada es “Compra de vehículo al cliente”.

Incluye campos como:

Matrícula

Marca, modelo, color, año

Kilómetros, tipo de combustible

Estado general, observaciones

Precio acordado de compra

Al guardar esta información, el vehículo se registra como parte del stock interno del taller.

7.10.2 Venta de vehículo al cliente

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.Esta funcionalidad permite gestionar la venta de vehículos disponibles en el stock del taller a un cliente. El sistema guía paso a paso todo el proceso, incluyendo la búsqueda del vehículo, la identificación del cliente, la firma del contrato, y la exportación del mismo a PDF o por correo electrónico.

Ilustración 74 Formulario de venta de vehículo con selección desde tabla, filtros avanzados y opciones de contrato y firma.

Estructura de la ventana

Esta sección se activa al seleccionar como tipo de operación:

"Venta por parte del concesionario"

A partir de ahí, se muestran dos bloques principales:

Listado de vehículos disponibles en stock

Formulario para filtrar, firmar y generar el contrato

Selección del vehículo a vender

Se muestra un listado completo de vehículos en inventario con los siguientes campos:

ID, Matrícula, Marca, Modelo, Año, Bastidor

Kilómetros, Potencia, Combustible, Color, Puertas, Plazas

Precio de compra y de venta

Descuento aplicado y estado del vehículo (DISPONIBLE, RESERVADO, etc.)

Este listado se puede actualizar con el botón “Recargar vehículos”, en caso de cambios en tiempo real.

Filtros avanzados

Debajo del listado, se presentan filtros por campo, que permiten buscar vehículos específicos según:

Marca, Año, Color, Combustible, Cambio, Kilómetros, Potencia, Puertas, Plazas, Precio de venta

Esto ayuda a localizar fácilmente vehículos que se ajusten a las necesidades del cliente.

Firma del cliente y generación del contrato

Una vez seleccionado el vehículo:

El cliente puede firmar digitalmente el contrato en el área habilitada.

Botones disponibles:

* Activar firma
* Limpiar firma

Opciones de salida del contrato

* Imprimir documento: genera una versión para impresora.
* Enviar por correo: envía el contrato al email del cliente (si está registrado).
* Guardar en carpeta: el documento PDF se guarda en la ruta especificada. Se puede cambiar con el botón “Seleccionar carpeta”.

Por defecto, se utiliza la carpeta de ventas establecida por el sistema, pero puede cambiarse dinámicamente.

Botones de acción final.

Esta sección se encuentra al final del proceso de compraventa y permite ejecutar las acciones definitivas sobre el contrato.

* Simular contrato: Genera una vista previa del contrato con todos los datos introducidos, sin necesidad de firmar ni guardar.  
  Es útil para repasar visualmente la información antes de formalizar la operación.  
  Este contrato preliminar actúa como plantilla, y no queda registrado en el sistema.
* Aceptar contrato: Guarda toda la operación de compraventa en la base de datos, genera el documento final firmado, y activa las opciones seleccionadas (impresión, guardado o envío por correo).

Es el paso final que confirma el acuerdo.

* Borrar todo: Limpia todos los campos del formulario, incluidos los filtros, selecciones y la firma capturada. Útil si se desea empezar de cero con otro cliente u operación.
* Volver: Cierra la ventana de compraventa actual y regresa al menú anterior sin guardar ningún cambio.

8.10.3 Instrucciones paso a paso

1. Selecciona el tipo de operación, usa el desplegable “Tipo de operación”. Las secciones del formulario se ajustarán automáticamente.

2. Rellena los datos del cliente: Si es un cliente existente, escribe el nombre o DNI para autocompletar. Si no existe, puedes crearlo con datos mínimos antes de continuar .

3. Completa la sección correspondiente:

3.1 Si estás registrando una compra:

Rellena los datos del vehículo entregado por el cliente.

Introduce el precio acordado y cualquier observación relevante.

3.2 Si estás registrando una venta:

Selecciona un vehículo del stock disponible.

Introduce los datos de la venta, incluyendo el precio final y método de pago.

4. Revisa los datos y pulsa el botón "Confirmar" (cuando esté habilitado).

7.11. Presupuestos e Intervenciones

Gestión de Presupuestos

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.Imagen de la pantalla de un celular

El contenido generado por IA puede ser incorrecto.Esta pantalla permite crear un presupuesto detallado para cualquier vehículo recepcionado. Está ligada directamente a una recepción previa, lo que garantiza la coherencia entre los datos del cliente, vehículo y observaciones recogidas al momento de la entrega.

Ilustración 75 Formulario para generar presupuestos a partir de recepciones, con tareas, precios y opciones de impresión o envío.

Ilustración 76 Botón de acceso al módulo de gestión y creación de presupuestos del taller.

Objetivo del módulo

Elaborar presupuestos claros y personalizados por cada intervención.

Controlar el coste total estimado.

Conocer y registrar la autorización del cliente.

Facilitar la generación del presupuesto en PDF o su envío por correo electrónico.

7.11.1 Estructura de la pantalla

1. Selección de la recepción

Campo desplegable "Recepción": aquí se elige una recepción previamente registrada.

Al seleccionar una, el sistema rellena automáticamente los campos de: Cliente, Vehículo, Fecha de recepción, Observaciones anotadas.

Estos campos no son editables, ya que están vinculados a los datos originales de la recepción.

7.11 2. Campos adicionales para rellenar

Precio máximo autorizado: límite económico aprobado por el cliente, si aplica.

Respuesta del cliente: una vez enviado el presupuesto, puede anotarse si ha sido aceptado, rechazado o pendiente.

7.11.3. Lista de tareas presupuestadas

El presupuesto se compone de tareas, cada una con:

* Descripción de la tarea
* Número de horas estimadas
* Precio por hora o coste
* Total, calculado automáticamente (Horas × Precio)

Esta tabla puede ordenarse usando los botones:

Añadir tarea

Eliminar tarea

Subir o Bajar una tarea (cambia su orden en la lista)

El botón “Guardar presupuesto” solo se activa si hay al menos una tarea añadida.

7.11.4. Coste total estimado

El sistema calcula automáticamente el total al sumar todas las tareas.

El valor se muestra en el campo “Coste total estimado”.

Puede compararse visualmente con el campo “Precio máximo autorizado”.

7.11.5. Opciones de salida

Antes de guardar el presupuesto, puedes elegir:

* Imprimir presupuesto
* Enviar por email (al correo del cliente si está registrado)

Ambas opciones pueden usarse combinadas.

Botones principales

Tabla 25 Tabla que explica la función de los botones principales en la gestión de presupuestos.

| Botón | Función |
| --- | --- |
| Guardar presupuesto | Guarda el presupuesto en la base de datos y genera el documento |
| Volver | Sale de la ventana sin guardar cambios |

8.11.6 Instrucciones paso a paso

1. Selecciona una recepción en el desplegable superior. El sistema rellenará todos los datos automáticamente.

2. Especifica el precio máximo autorizado por el cliente (si aplica).

3. Añade una o más tareas al presupuesto:

* Pulsa “Añadir tarea”.
* Especifica nombre, horas y precio.

4. Organiza las tareas si lo deseas (Subir/Bajar).

5. Verifica el coste total estimado.

6. Añade una nota sobre la respuesta del cliente, si ya existe.

7. Marca si deseas:

* Imprimir el presupuesto
* Enviar por email

8. Pulsa “Guardar presupuesto”.

7.12 Reimpresión de recepcionamientos

Esta ventana permite consultar, imprimir o reenviar por correo electrónico cualquier documento PDF generado durante el proceso de recepcionamiento de vehículos.

Aplicación

El contenido generado por IA puede ser incorrecto.Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Ilustración 77 Botón para acceder al módulo de reimpresión de documentos de recepcionamiento previos.

Ilustración 78 Pantalla para seleccionar y reenviar o imprimir documentos de recepcionamiento generados anteriormente.

7.12.1 Objetivo del módulo

Facilitar la reimpresión rápida de documentos sin tener que rehacer el proceso.

Permitir el reenvío del archivo PDF al cliente desde el sistema, sin necesidad de buscarlo manualmente.

Mejorar el archivo documental digital, facilitando búsquedas por mes o nombre.

7.12.2 Estructura de la pantalla

La interfaz se compone de:

Tabla de documentos

* Columna Mes: muestra el mes en el que se generó el documento.
* Columna Nombre del Documento: nombre del archivo PDF guardado (ej: Recepcion\_CARLOS\_GOMEZ\_HERNANDEZ\_01002.pdf).

El listado está ordenado cronológicamente.

Se puede hacer clic sobre cualquier fila para seleccionarla.

¿De dónde salen estos archivos?

Todos los archivos listados son documentos PDF generados automáticamente al confirmar una recepción desde la pantalla de recepcionamiento de vehículos.

El sistema los guarda con una nomenclatura que incluye nombre del cliente y un identificador único.

7.12.3 Opciones disponibles

Botón "Enviar"

Envía el documento seleccionado por correo electrónico al cliente asociado.

El sistema usa el correo registrado en la base de datos para ese cliente.

Si no hay correo asociado, se muestra un mensaje de aviso.

Botón "Imprimir"

Abre una vista previa de impresión para el documento seleccionado.

Permite elegir la impresora e imprimir directamente desde el sistema.

Botón "Volver"

Cierra la ventana y regresa al menú principal o anterior.

7.12.4 Instrucciones paso a paso

1. Localiza el documento deseado, Revisa el listado por mes o busca visualmente el nombre del documento.

2. Haz clic en la fila correspondiente para seleccionar el documento.

3. Elige una acción:

* Pulsa “Imprimir” para sacar una copia física.
* Pulsa “Enviar” para reenviarlo por correo.

4. Si has terminado, pulsa “Volver” para cerrar la ventana.

7.13. Reimpresión de presupuestos

Este módulo permite consultar, imprimir o reenviar presupuestos guardados en formato PDF, ya generados anteriormente desde la pantalla de "Gestión de presupuestos".

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.Texto

El contenido generado por IA puede ser incorrecto.

Ilustración 79 Pantalla para consultar y reimprimir presupuestos almacenados según el mes y nombre del documento.

Ilustración 80 Botón para acceder a la reimpresión de documentos de presupuestos ya generados.

7.13.1 Objetivo de la pantalla

Acceder rápidamente a presupuestos generados anteriormente.

Permitir su reimpresión física o reenvío digital al cliente, sin tener que repetir el proceso.

Centralizar el control documental para el taller mecánico.

7.13.2 Estructura de la ventana

Tabla de presupuestos

La tabla contiene dos columnas:

Mes: mes en que se generó el presupuesto.

Nombre del Documento: nombre exacto del archivo PDF.

Los documentos siguen la convención de nombres como:

Presupuesto\_<códigoCliente>\_YYYYMMDD\_HHMM.pdf

Puedes hacer clic sobre cualquier fila para seleccionar el documento deseado.

7.13.3 Acciones disponibles

Botón "Enviar"

Envía el documento PDF seleccionado por correo electrónico.

El correo se envía al email registrado en la base de datos para el cliente correspondiente.

Si no hay correo registrado, se notifica al usuario.

Botón "Imprimir"

Abre una ventana de impresión del sistema con el presupuesto seleccionado.

Permite sacar una copia física directamente.

Botón "Volver"

Cierra la ventana y regresa al menú anterior.

7.13.4 Instrucciones paso a paso

1. Localiza el presupuesto Revisa el mes y el nombre del documento para identificarlo.

2. Haz clic en la fila correspondiente, Esto selecciona el archivo para su impresión o envío.

3. Elige la acción:

Pulsa “Enviar” para enviarlo por correo al cliente.

Pulsa “Imprimir” para generar una copia física.

4. Usa “Volver” para salir.

8.14. Reimpresión de compras

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.Aplicación

El contenido generado por IA puede ser incorrecto.Este módulo permite consultar, imprimir o reenviar los contratos de compra de vehículos previamente generados desde el apartado de compraventa. Es útil cuando el cliente solicita una copia adicional o cuando se requiere archivar nuevamente el contrato impreso.

Ilustración 81 Pantalla de reimpresión de contratos de compra con listado por mes y nombre de archivo, lista para enviar o imprimir.

Ilustración 82 Botón que permite acceder al módulo de reimpresión de contratos de compraventa de vehículos.

7.14.1 Objetivo funcional

Proporcionar un acceso rápido y sencillo a contratos ya generados.

Permitir al usuario reenviar el contrato al cliente por correo electrónico.

Reimprimir el contrato en caso de pérdida, deterioro o archivo físico.

7.14.2 Estructura de la pantalla

Tabla de contratos de compra

Contiene dos columnas:

Mes: mes en que se firmó y generó el contrato.

Nombre del Documento: nombre exacto del PDF, como por ejemplo: CONTRATO\_COMPRA\_4444ABC.pdf

Cada fila representa un contrato diferente.

El usuario debe seleccionar una fila para poder actuar sobre ese contrato.

7.14.3 Opciones disponibles

Botón "Enviar"

Envía el contrato seleccionado al correo electrónico del cliente asociado.

Si el cliente no tiene correo registrado, el sistema lanza un aviso.

Botón "Imprimir"

Abre una ventana de impresión para reimprimir el contrato físicamente.

Botón "Volver"

Cierra la ventana y regresa al panel principal o anterior.

7.14.4 Instrucciones paso a paso

1. Revisa la lista de contratos y selecciona el que deseas reenviar o imprimir.

2. Haz clic sobre la fila correspondiente para activarlo.

3. Ejecuta la acción:

Pulsa “Enviar” si deseas enviarlo por correo.

Pulsa “Imprimir” si necesitas una copia física.

4. Usa “Volver” para regresar cuando finalices.

7.15. Envío y reimpresión de documentos

Reimpresión de ventas

Imagen que contiene Diagrama

El contenido generado por IA puede ser incorrecto.Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.Esta pantalla permite consultar, imprimir o reenviar por correo electrónico los contratos de venta de vehículos generados desde el módulo de compraventa.

Ilustración 83 Botón para acceder a la reimpresión de contratos de venta de vehículos previamente generados.

Ilustración 84 Pantalla de reimpresión de contratos de venta clasificados por mes con opciones de envío e impresión rápida.

7.15.1 Objetivo del módulo

Permitir recuperar contratos de venta generados anteriormente.

Facilitar su reimpresión o reenvío sin depender del archivo físico.

Garantizar trazabilidad y acceso organizado a los documentos de venta.

7.15.2 Estructura de la pantalla

Tabla de documentos de venta

La tabla presenta dos columnas:

Mes: indica el mes en que se realizó la venta.

Nombre del Documento: muestra el nombre del archivo PDF generado, con el formato: CONTRATO\_VENTA\_<ID>.pdf

Cada fila representa un contrato distinto.

El usuario debe seleccionar una fila antes de realizar cualquier acción.

7.15.3 Opciones disponibles

Botón "Enviar"

Envía por correo electrónico el contrato seleccionado al cliente.

Utiliza el correo asociado al cliente en la base de datos.

Si el cliente no tiene correo, se informa al usuario.

Botón "Imprimir"

Permite sacar una copia física del contrato en papel.

Botón "Volver"

Cierra la ventana actual y regresa al menú anterior.

7.15.4 Instrucciones paso a paso

1. Busca el documento deseado en la tabla según mes o nombre.
2. Haz clic en una fila para seleccionar el contrato.
3. Pulsa:

* “Enviar” si deseas enviarlo por email.
* “Imprimir” si necesitas una copia impresa.

1. Usa “Volver” para salir de la pantalla.

8.- Bibliografía del Proyecto

Sistema de Gestión Integral para Taller Mecánico - ReyBoxes

8.1. Tecnologías y herramientas utilizadas

1. Python Software Foundation.

*Python 3.x Documentation.*

<https://docs.python.org/3/>

1. Qt Company.

*Qt for Python (PySide6) Documentation.*

<https://doc.qt.io/qtforpython/>

1. PostgreSQL Global Development Group.

*PostgreSQL 15 Documentation.*

<https://www.postgresql.org/docs/>

1. Nodemailer.

*Nodemailer Email Library for Node.js.*

<https://nodemailer.com/>

1. Jinja2 Templating Engine.

Pallets Projects.

<https://jinja.palletsprojects.com/>

1. wkhtmltopdf Project.

*Rendering HTML to PDF using WebKit.*

<https://wkhtmltopdf.org/>

1. Electron Project.

*Build cross-platform desktop apps with JavaScript, HTML, and CSS.*

<https://www.electronjs.org/>

1. Node.js Foundation.

*Node.js Official Documentation.*

<https://nodejs.org/en/docs/>

1. CapacitorJS.

*Capacitor - Cross-platform apps with web technology.*

<https://capacitorjs.com/>

Diseño de interfaz y experiencia de usuario

1. Google Fonts – Montserrat.

https://fonts.google.com/specimen/Montserrat

1. 3C – Web Accessibility Initiative.

*Designing for Accessibility.*

<https://www.w3.org/WAI/design-develop/>

1. Jakob Nielsen.

*Usability Heuristics for User Interface Design.*

Nielsen Norman Group.

https://www.nngroup.com/articles/ten-usability-heuristics/

8.2. Buenas prácticas de desarrollo

1. Robert C. Martin.

*Clean Code: A Handbook of Agile Software Craftsmanship.*

Prentice Hall, 2008.

1. Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides.

*Design Patterns: Elements of Reusable Object-Oriented Software.*

Addison-Wesley, 1994.

1. Open Web Application Security Project (OWASP).

*OWASP Secure Coding Practices.*

https://owasp.org/www-project-secure-coding-practices/

8.3. Normativas y documentación adicional

1. Agencia Española de Protección de Datos (AEPD).

*Guía sobre el uso de cookies, RGPD y protección de datos personales.*

<https://www.aepd.es/>

1. ISO/IEC 25010:2011.

*Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models.*

8.4. Herramientas de modelado y documentación UML

1. PlantUML

Herramienta de código abierto para crear diagramas UML utilizando un lenguaje de marcado simple y expresivo. Se ha utilizado extensivamente para la generación de diagramas de clases, relaciones entre módulos y arquitectura general del sistema.

Sitio web: <https://plantuml.com/>

1. PlantUML Online Editor (PlantText / PlantUML Web Server)

Editores online que permiten renderizar diagramas UML escritos en sintaxis PlantUML sin necesidad de instalación local. Muy útiles para realizar pruebas rápidas de estructura y validar la sintaxis de los diagramas.

Sitio web: <https://www.planttext.com/> | https://www.plantuml.com/plantuml

1. IntelliJ IDEA + Plugin PlantUML integration

El entorno de desarrollo IntelliJ ha sido complementado con el plugin PlantUML Integration para generar y previsualizar diagramas directamente desde el código fuente, permitiendo validar estructuras y relaciones mientras se desarrollaba la lógica del sistema.

Sitio web: <https://plugins.jetbrains.com/plugin/7017-plantuml-integration>

1. Visual Studio Code + PlantUML extensión

También se utilizó el editor Visual Studio Code con su extensión oficial para PlantUML. Esta combinación permitió trabajar con scripts .puml y generar las visualizaciones dentro del propio entorno de desarrollo.

Sitio web:<https://marketplace.visualstudio.com/items?itemName=jebbs.plantuml>

1. Graphviz

Motor de renderizado gráfico requerido por PlantUML para generar visualizaciones más precisas y estilizadas de los diagramas. Especialmente útil en diagramas complejos con muchas clases o relaciones.

Sitio web: <https://graphviz.org/>

8.5 Fuentes adicionales de apoyo y aprendizaje

1. Stack Overflow

<https://stackoverflow.com/>

1. GitHub

<https://github.com/>

1. YouTube Channels:
   * *FreeCodeCamp*
   * *Traversy Media*
   * *PyQt/PySide Tutorials – Tech with Tim*