UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
SEDE VIÑA DEL MAR – JOSÉ MIGUEL CARRERA

SOFTWARE DE MANTENIMIENTO AUTOMÓVIL DESCENTRALIZADO EN BLOCKCHAIN "CARCHAIN"

Trabajo de titulación para optar al título profesional de Ingeniero de Ejecución en Software

Alumno:
Nicolás Matías Carrasco Escobar

Profesor guía:
Pedro Francisco Godoy Barrera

CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☒ Memoria o trabajo de título ☐ Tesis de Postgrado

Título del trabajo: SOFTWARE DE MANTENIMIENTO AUTOMÓVIL DESCENTRALIZADO EN BLOCKCHAIN “CARCHAIN”

Nombre del candidato(a): NICOLÁS MATÍAS CARRASCO ESCOBAR

Carrera / Grado: INGENIERIA DE EJECUCIÓN EN SOFTWARE

Campus: VIÑA DEL MAR **Departamento:** DEPARTAMENTO DE ELECTROTECNIA E INFORMÁTICA

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Pedro Francisco Godoy Barrera, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución.

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL (marcar una opción)

☒ El trabajo **NO contiene** información que amerite confidencialidad y puede ser publicado de inmediato en repositorio con acceso abierto.

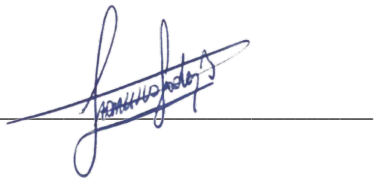
☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (**embargo**) por **(marcar una opción)**:

☐ 6 meses ☐ 12 meses ☐ 2 años ☐ 3 años ☐ 5 años ☐ 10 años

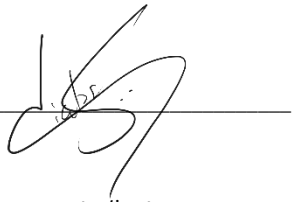
Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis: Pedro Godoy Barrera

Fecha: 29-Oct-2025 **Firma:** 

Estudiante o Candidato(a): Nicolás Carrasco

Fecha: 30-10-2025 **Firma:** 

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.

2024

RESUMEN

KEYWORDS: BITÁCORA, TRAZABILIDAD, BLOCKCHAIN, MANTENIMIENTO VEHICULAR, TRANSPARENCIA.

El presente proyecto tiene como objetivo diseñar un sistema llamado “CarChain”, orientado a talleres de automóviles, plataformas de compra-venta y otros actores relacionados con la industria automotriz que requieran contar una bitácora de actividades realizadas en los vehículos trazable e inmutable. La finalidad de este software es permitir el registro en la nube de las mantenciones y reparaciones realizadas a vehículos motorizados, a modo de una "bitácora" y verificable. De esta manera, se busca garantizar la integridad y transparencia de la información, permitiendo que talleres previamente certificados y autorizados en la herramienta puedan guardar registros de mantenimiento siguiendo una línea exacta que permite ver de una manera clave las mantenciones que se realizaron a un vehículo.

El proyecto abordará aspectos relevantes como los antecedentes de la institución, la situación actual sin el proyecto, los problemas detectados y los objetivos propuestos. Se evaluarán las opciones de solución presentadas, se definirán los requisitos y requerimientos indicados por los usuarios, y se estimarán los costos y tiempos necesarios para llevar a cabo el desarrollo del sistema.

A lo largo del trabajo, se analizará en detalle la solución propuesta, incluyendo diagramas de actividades, usuarios del sistema, requerimientos funcionales y no funcionales, matriz de trazabilidad, modelo conceptual y casos de uso. Además, se presentará el diseño de la arquitectura de software, el diseño de datos con modelos de clases, modelo relacional y diccionario de datos, así como el diseño de componentes y la interfaz propuesta para el sistema.

Finalmente, se expondrán las conclusiones y sugerencias derivadas del proceso de desarrollo, destacando las revisiones y correcciones realizadas en cada etapa para garantizar la entrega de un proyecto satisfactorio que cumpla con los objetivos planteados y las necesidades de los usuarios en el ámbito de la gestión de mantenciones y reparaciones de vehículos motorizados.

INDICE

Tabla de Figuras

Tabla de Ilustraciones

INTRODUCCIÓN

En la actualidad, la industria automotriz enfrenta desafíos significativos en cuanto a la gestión y seguimiento de las mantenciones y reparaciones realizadas a los vehículos motorizados. La falta de un sistema centralizado y confiable para registrar estos datos ha generado problemas como la duplicidad de información, corroborar veracidad de la mantención, la dificultad para acceder a un historial completo de cada vehículo y la falta de transparencia en los procesos.

El presente proyecto tiene como objetivo desarrollar un sistema llamado “CarChain”, orientado a talleres de automóviles, plataformas de compra-venta y otros actores relacionados con la industria automotriz. La finalidad de este software es permitir el registro descentralizado en la nube de las mantenciones y reparaciones realizadas a vehículos motorizados, a modo de una "bitácora" inmutable y verificable. De esta manera, se busca garantizar la integridad y transparencia de la información, permitiendo que talleres previamente certificados y autorizados en la herramienta puedan guardar registros de mantenimiento siguiendo una línea ética y exacta.

Actualmente, la mayoría de los talleres y plataformas de compra-venta de vehículos utilizan métodos manuales o sistemas poco eficientes para registrar las mantenciones y reparaciones. Esto genera problemas como la pérdida de información, errores en los registros y dificultades para acceder a un historial completo y confiable de cada vehículo desde otra localización. Si un cliente se cambia de taller más aún cuando no es de la misma empresa, no existe un historial en linea. Además, la falta de transparencia en estos procesos puede generar desconfianza entre los propietarios de los vehículos y los potenciales compradores.

El sistema “CarChain” busca solucionar estos problemas mediante el uso de tecnologías descentralizadas y seguras, como blockchain, para garantizar la inmutabilidad y trazabilidad de los registros. Además, se implementarán mecanismos de certificación y autorización para los talleres participantes, asegurando que solo aquellos que cumplan con los estándares éticos y de calidad puedan ingresar y actualizar la información en el sistema.

A lo largo de este trabajo, se abordarán los aspectos relevantes del proyecto, incluyendo los antecedentes de la institución, la situación actual sin el proyecto, los problemas detectados y los objetivos propuestos. Se evaluarán las opciones de solución presentadas, se definirán los

requisitos y requerimientos indicados por los usuarios, y se estimarán los costos y tiempos necesarios para llevar a cabo el desarrollo del sistema.

El objetivo final es evaluar las diferentes alternativas del mercado, seleccionar la que tenga mayor puntaje y efectuar todos los procesos de un diseño de software que será de gran utilidad para diferentes empresas del rubro automotriz.

1. CAPÍTULO 1. ASPECTOS RELEVANTES DEL PROYECTO Y SU GESTIÓN

1.1. ASPECTOS RELEVANTES DEL PROYECTO Y SU GESTIÓN.

En este capítulo se explicarán los temas más relevantes en cuanto al proyecto "CarChain" y su gestión. Se describirá la problemática presentada en la industria automotriz en relación al registro de mantenciones y reparaciones de vehículos, el objetivo del sistema propuesto y los resultados esperados de su implementación. Se abordarán los requerimientos de los usuarios, las alternativas de solución evaluadas y los criterios utilizados para seleccionar la mejor opción. Además, se analizarán los riesgos del proyecto y sus costos estimativos.

1.2. Antecedentes de la institución.

La industria automotriz es un sector clave en la economía global, que involucra a diversos actores como fabricantes de vehículos, talleres de reparación, plataformas de compra-venta, entre otros. Sin embargo, actualmente existen desafíos significativos en cuanto a la gestión y seguimiento de las mantenciones y reparaciones realizadas a los vehículos motorizados. La falta de un sistema centralizado y confiable para registrar estos datos ha generado problemas como la duplicidad de información, la dificultad para acceder a un historial completo de cada vehículo y la falta de transparencia en los procesos.

1.3. Situación sin proyecto.

1.3.1. Descripción de la situación sin proyecto.

Actualmente, la mayoría de los talleres y plataformas de compra-venta de vehículos utilizan métodos manuales o sistemas poco eficientes para registrar las mantenciones y reparaciones. Esto genera problemas como la pérdida de información, errores en los registros y dificultades para acceder a un historial completo y confiable de cada vehículo. Además, la falta de transparencia en estos procesos puede generar desconfianza entre los propietarios de los vehículos y los potenciales compradores.

1.3.2. Problemas detectados.

De acuerdo con el análisis realizado a la situación actual de la industria automotriz, se han identificado los siguientes problemas:

1. Falta de una base de datos universal consultable de diferentes talleres, lo que no permite comprobar las últimas mantenciones ni comprobar que reparaciones han sido realizadas por el taller.
2. Duplicidad de información y errores en los registros debido a métodos manuales o sistemas poco eficientes.
3. Dificultad para acceder a un historial completo y verificable de cada vehículo.
4. Falta de transparencia en los procesos, generando desconfianza entre propietarios y compradores.
5. Ausencia de mecanismos de certificación y autorización para los talleres participantes.

1.3.3. Requerimientos del Usuario

Las necesidades manifestadas por los usuarios de la industria automotriz incluyen los siguientes requerimientos:

1. Permitir el registro en la nube de una manera centralizada y seguro de las mantenciones y reparaciones realizadas a los vehículos.
2. Garantizar la inmutabilidad y trazabilidad de los registros mediante tecnologías como blockchain.
3. Implementar mecanismos de certificación y autorización para los talleres participantes.
4. Proporcionar acceso a un historial completo y verificable de cada vehículo.
5. Generar reportes y estadísticas sobre las mantenciones y reparaciones realizadas.
6. Facilitar la integración con plataformas de compraventa de vehículos.
7. Asegurar la privacidad y seguridad de los datos de los usuarios.

Además, los usuarios han manifestado la necesidad del cumplimiento de los siguientes requisitos:

1. El sistema debe ser escalable y capaz de manejar un gran volumen de transacciones.
2. Debe proporcionar una interfaz de usuario intuitiva y de fácil uso.
3. Debe garantizar la disponibilidad y accesibilidad del sistema en todo momento.
4. Debe cumplir con los estándares de seguridad y privacidad de datos.

1.4. Objetivos del proyecto.

El objetivo general del proyecto 'Carchain' es diseñar un sistema que unifique y facilite el acceso a la información de mantenciones y reparaciones de vehículos. Para lograrlo, se empleará tecnología blockchain para garantizar la inmutabilidad y la trazabilidad descentralizada de los registros, generando así una bitácora digital segura que aumenta la transparencia y la confianza en la industria automotriz.

Los objetivos específicos son:

1. Generar reportes y estadísticas sobre las mantenciones y reparaciones realizadas.
2. Facilitar la integración con plataformas de compra-venta de vehículos.
3. Asegurar la privacidad y seguridad de los datos de los usuarios.

1.5. Alternativas globales de solución.

Se presentaron y evaluaron tres alternativas de solución para abordar la problemática detectada en la industria automotriz en cuanto a la gestión de mantenciones y reparaciones. A continuación, se describen las alternativas y se exponen las ventajas, desventajas y costos asociados de cada una.

1.5.1. Alternativa 1: Desarrollo interno del sistema

Ventajas:

a. Control total del desarrollo: La organización tendría completo control sobre todas las etapas del proyecto, permitiendo ajustarse de forma ágil a cambios en los requerimientos o mejoras futuras.
b. Adaptación específica: El sistema se desarrollaría a medida para satisfacer exactamente las necesidades identificadas, lo que asegura un ajuste óptimo a los procesos internos y requisitos de los usuarios.
c. Propiedad del código: El código fuente y todas las funcionalidades serían propiedad total de la empresa, evitando depender de licencias de terceros o limitaciones de uso.

Desventajas:

a. Costos de desarrollo: Si bien se aprovecharía la infraestructura existente, los costos asociados al equipo de desarrollo (remuneraciones, formación, y posibles contrataciones adicionales) pueden ser elevados. Además, habría que considerar los gastos en herramientas de desarrollo, servidores y otros recursos tecnológicos.
b. Tiempo de implementación: El desarrollo desde cero podría prolongarse, ya que el equipo interno podría no tener experiencia en blockchain o en soluciones de trazabilidad descentralizada, lo que puede implicar una curva de aprendizaje significativa.
c. Recursos limitados: Al utilizar solo recursos internos, otros proyectos o actividades de la organización podrían verse afectados al desviar personal al desarrollo de "CarChain".

1.5.2. Alternativa 2: Contratación de empresa externa.

Esta opción consiste en contratar a una empresa especializada en el desarrollo de software para diseñar e implementar "CarChain". La empresa sería responsable de todo el ciclo de vida del proyecto, desde el análisis de requerimientos hasta la entrega y mantenimiento.

Ventajas:

a. Rapidez de implementación: Una empresa especializada ya cuenta con experiencia en proyectos similares, lo que puede reducir el tiempo de desarrollo.
b. Expertise técnico: Las empresas de software suelen tener equipos multidisciplinarios con conocimientos avanzados en tecnologías de punta como blockchain, seguridad en la nube y sistemas de alta disponibilidad, lo que garantiza un producto robusto.
c. Liberación de recursos internos: Al delegar el desarrollo a un proveedor externo, el equipo interno de la organización puede enfocarse en otras prioridades estratégicas sin ver comprometido el desarrollo de "CarChain".

Desventajas:

a. Dependencia del proveedor: Existe el riesgo de depender a largo plazo de la empresa externa para cualquier modificación, actualización o soporte técnico, lo que podría aumentar los costos futuros.

b. Costos elevados: La contratación de una empresa de desarrollo especializada puede tener un costo significativamente más alto en comparación con el desarrollo interno, ya que se incluirían gastos como la personalización del sistema, mantenimiento y posibles actualizaciones.
c. Menor flexibilidad: A pesar de que el sistema será desarrollado según los requerimientos iniciales, cualquier cambio posterior en las especificaciones podría generar costos adicionales y posibles retrasos.

1.5.3. Alternativa 3: Adquisición de software existente

En esta alternativa, se evaluaría la adquisición de un software ya desarrollado que cumpla con la mayoría de los requerimientos de "CarChain". Este software sería personalizado para adaptarse a las particularidades del proyecto, y se integraría con los sistemas internos de la organización.

Ventajas:

a. Implementación rápida: Dado que el software ya existe, el tiempo necesario para su personalización e implementación es considerablemente menor que desarrollar desde cero.
b. Menor riesgo técnico: Se adquiere un software probado en el mercado, lo que disminuye el riesgo de errores graves o fallos en su funcionamiento.
c. Soporte garantizado: La mayoría de los proveedores de software ofrecen contratos de soporte y actualizaciones, lo que asegura que el sistema se mantenga actualizado y funcional sin necesidad de invertir en recursos técnicos internos.

Desventajas:

a. Menor flexibilidad: Al ser un software desarrollado para un público general, puede que no cumpla con todos los requerimientos específicos del proyecto, y las modificaciones pueden ser limitadas o costosas.
b. Costos de licencias: El software generalmente se adquiere bajo un esquema de licenciamiento, lo que puede resultar en un costo recurrente en lugar de una inversión única.

c. Dependencia del proveedor: Al igual que en la alternativa anterior, se genera una dependencia del proveedor para cualquier personalización, mantenimiento o actualización del sistema.

1.6. Evaluación de las alternativas.

1.6.1. Criterios de evaluación.

Se establecieron criterios de evaluación ponderados para analizar cada alternativa, considerando factores como usabilidad, mantenimiento, escalabilidad, seguridad, arquitectura, costo y puesta en marcha.

1.6.2. Ponderación de criterios y escala de evaluación.

Para evaluar los criterios anteriores se usarán notas en una escala de 1 a 5 de evaluación, las cuales se reflejan en ~~la siguiente~~ tabla 1-1:

Tabla 1-1. Escala de evaluación

Nota	Significado
1	Muy Malo
2	Malo
3	Medio
4	Bueno
5	Muy Bueno

Fuente: Elaboración propia.

Tabla 1-2. Ponderaciones de Criterios

Criterio	Ponderaciones
Funcionalidad	20%
Usabilidad	10%
Mantención	10%
Escalabilidad	10%
Seguridad de la información	10%
Arquitectura	15%
Costo	20%
Puesta en Marcha	5%
	100%

Fuente: Elaboración propia

1.6.3. Evaluación.

Una vez establecidos los criterios para la evaluación de cada alternativa presentada, se procederá a definir los puntos clave que permitirán tomar una decisión de manera concreta y objetiva. Estos puntos de evaluación serán fundamentales para comparar las diferentes opciones, considerando factores como costo, tiempo de implementación, flexibilidad, escalabilidad y riesgos asociados.

Tabla 1-3. Evaluación Alternativa 1

Criterio	**Descripción**	**Desarrollo interno**
Funcionalidad	El sistema cumple con todas las funcionalidades requeridas según las necesidades del usuario.	5
Usabilidad	El sistema es intuitivo y fácil de usar, sin necesidad de capacitación adicional.	5
Mantención	La mantención es gestionada internamente, permitiendo un control total sobre las actualizaciones y mejoras.	4
Escalabilidad	El sistema permite una fácil escalabilidad para agregar nuevas funcionalidades.	5
Seguridad de la información	Altos estándares de seguridad implementados para proteger los datos del usuario.	5
Arquitectura	Desarrollado en un lenguaje versátil que facilita la implementación en diversas plataformas.	4
Costo	Los costos son controlados internamente, sin gastos adicionales significativos.	4
Puesta en Marcha	Implementación gradual con pruebas constantes para asegurar la funcionalidad completa.	4
	Nota	**38**

Fuente: Elaboración propia

Tabla 1-4. Evaluación Alternativa 2

Criterio	**Descripción**	**Desarrollo Externo**
Funcionalidad	Cumple con la mayoría de las funcionalidades necesarias con algunas excepciones.	4
Usabilidad	Requiere capacitación inicial debido a su interfaz compleja.	3
Mantención	Depende del proveedor externo, con costos adicionales por soporte y actualizaciones.	3
Escalabilidad	Posible escalabilidad, aunque con costos adicionales por cada actualización o mejora.	3
Seguridad de la información	Cumple con los estándares básicos de seguridad, pero podría mejorarse.	4
Arquitectura	Utiliza un lenguaje de programación moderno, pero con limitaciones en algunas plataformas.	3
Costo	El costo inicial es alto y no incluye todas las funcionalidades necesarias, lo que podría aumentar el gasto total.	2
Puesta en Marcha	La implementación es directa, pero requiere un período de ajuste y adaptación.	3
	Nota	**25**

Fuente: Elaboración propia

Tabla 1-5. Evaluación Alternativa 3

Criterio	Descripción	Compra de Software
Funcionalidad	Requiere adaptaciones significativas para cumplir con los requerimientos específicos.	3
Usabilidad	Interfaz no intuitiva, requiere capacitación extensa para las secretarias.	2
Mantención	Dependiente del proveedor externo, con procesos de aprobación lentos para cotizaciones.	2
Escalabilidad	Limitada capacidad de integración sin servicios adicionales costosos.	2
Seguridad de la información	Cumple con los estándares mínimos de seguridad, pero con áreas de mejora necesarias.	4
Arquitectura	Basado en un lenguaje de programación obsoleto que no soporta múltiples plataformas.	2
Costo	Excede el presupuesto inicial y requiere inversiones adicionales para cada área de especialidad.	1
Puesta en Marcha	Las pruebas internas revelan problemas que necesitan resolución antes de la implementación completa.	2
	Nota	**27**

Fuente: Elaboración propia

Finalmente, evaluadas las tres opciones presentadas, se realiza una tabla comparativa con las calificaciones obtenidas.

Opción 1: Desarrollo Interno
Opción 2: Desarrollo Externo
Opción 3: Software Existente

Tabla 1-6. Resumen de calificaciones alternativas

Criterio	Pond	Calificación			Calificación Ponderada		
		1	2	3	1	2	3
Funcionalidad	20%	5	4	3	1.00	0.80	0.60
Usabilidad	10%	5	3	2	0.50	0.30	0.20
Mantención	10%	4	3	2	0.40	0.30	0.20
Escalabilidad	10%	5	3	2	0.50	0.30	0.20
Seguridad de la información	10%	5	4	4	0.50	0.40	0.40
Arquitectura	15%	4	3	2	0.60	0.45	0.30
Costo	20%	4	2	1	0.80	0.40	0.20
Puesta en Marcha	5%	4	3	2	0.20	0.15	0.10
Total	100%				4.50	3.10	2.20

Fuente: Elaboración propia

1.7. Alternativa Seleccionada.

En base al análisis realizado, se seleccionó la Alternativa 1 de desarrollo interno del sistema “CarChain”. Esta opción permite ajustarse de mejor manera a los requerimientos, brinda mayor flexibilidad y control sobre el proceso de desarrollo principalmente a la remuneración del equipo de trabajo.

Algunos de los beneficios esperados de esta alternativa son:

- Mayor satisfacción de los usuarios al tener un sistema a medida de sus necesidades.
- Validación y consistencia de los datos ingresados.
- Acceso en línea a la información para todos los actores involucrados.
- Facilidad de uso al ser una aplicación web sin necesidad de instalación.
- Generación de reportes de gestión en base a los datos almacenados.
- Control sobre usuarios, talleres y transacciones registradas.
- Posibilidad de agregar nuevas funcionalidades de forma escalable.

1.7.1. Descripción.

Basado en el análisis inicial, se desarrollará “CarChain”, un sistema web destinado a integrarse con plataformas existentes en la industria automotriz, como talleres y sitios de compra-venta de vehículos. Este sistema facilitará un registro centralizado y seguro en la nube de todas las reparaciones y mantenimientos vehiculares, actuando como un registro permanente y seguro.

Tras determinar las necesidades que “CarChain” debe abordar, se diseñarán los procesos necesarios que el software administrará.

El sistema, a través de diversos módulos, permitirá el registro de operaciones de mantenimiento y reparación efectuadas por talleres debidamente certificados. Facilitará la administración de estos registros, permitiendo a los usuarios autorizados acceder a los historiales de mantenimiento, registrar nuevas operaciones y generar informes necesarios para mantener la integridad y transparencia de la información.

1.7.2. Beneficios.

El desarrollo del sistema Carchain traerá múltiples beneficios para los diversos actores involucrados en la industria automotriz:

1. Transparencia y confiabilidad: Al utilizar tecnología blockchain, se garantiza la inmutabilidad y trazabilidad de los registros de mantenimiento y reparaciones. Esto aumenta la transparencia y confiabilidad de la información, generando mayor confianza entre los propietarios de vehículos, talleres y potenciales compradores.

2. Acceso a un historial completo: Carchain permitirá acceder a un historial completo y detallado de las mantenciones y reparaciones realizadas a cada vehículo. Esto facilitará la toma de decisiones informadas por parte de los propietarios y compradores, al conocer el estado real del vehículo y su historial de mantenimiento.

3. Eficiencia en la gestión de información: El sistema centralizado en la nube eliminará la duplicidad de información y los errores en los registros. Los talleres podrán ingresar y actualizar la información de manera eficiente, ahorrando tiempo y recursos en la gestión manual de los datos.

4. Mejora en la calidad del servicio: Al contar con un sistema estandarizado y certificado, los talleres participantes deberán cumplir con estándares éticos y de calidad para poder ingresar y actualizar la información. Esto incentivará la mejora continua en la prestación de servicios de mantenimiento y reparación.

5. Valor agregado para plataformas de compra-venta: Las plataformas de compra-venta de vehículos podrán integrar la información de Carchain en sus listados, brindando a los potenciales compradores un historial confiable y verificable del vehículo de interés. Esto aumentará la confianza en las transacciones y facilitará la toma de decisiones de compra.

6. Ahorro de costos: Al contar con un sistema eficiente y confiable, se reducirán los costos asociados a la gestión manual de la información y a la resolución de problemas derivados de registros incompletos o erróneos.

7. Oportunidades de análisis de datos: La centralización de la información en Carchain permitirá realizar análisis de datos y obtener insights valiosos sobre patrones de mantenimiento, fallas comunes y tendencias en la industria automotriz. Esto podrá ser utilizado por talleres, fabricantes y otros actores para mejorar sus servicios y productos.

En resumen, Carchain ofrecerá una solución innovadora y confiable para la gestión de mantenciones y reparaciones de vehículos, generando beneficios para todos los actores involucrados en la industria automotriz. La transparencia, eficiencia y mejora en la calidad del servicio serán los principales pilares de este proyecto.

1.8. Planificación.

1.8.1. Desarrollo del plan de personal.

Se ha planificado la conformación de un equipo de trabajo que operará bajo un marco ágil, específicamente Scrum. Este equipo estará compuesto por los siguientes roles clave, cuyas responsabilidades son fundamentales para el éxito del proyecto "CarChain".

a. Product Owner (PO): Este rol es el principal responsable de maximizar el valor del producto que resulta del trabajo del equipo de desarrollo. El PO es la voz del cliente y de los stakeholders, encargándose de gestionar y priorizar el Product Backlog. Para "CarChain", sus tareas incluirán definir las historias de usuario, detallar los requisitos de las funcionalidades, como el registro inmutable de mantenimientos en la blockchain, y asegurar que el equipo de desarrollo comprenda claramente cada uno de los elementos a implementar. Será el encargado de que la visión del software se traduzca en características concretas y alineadas con las necesidades del mercado automotriz.
b. Scrum Master (SM): Su función principal es asegurar que el equipo Scrum siga los valores y prácticas de la metodología. Actúa como un líder servicial, eliminando los impedimentos que puedan obstaculizar el progreso del equipo y facilitando los eventos de Scrum (Sprint Planning, Daily Scrum, Sprint Review y Sprint Retrospective). Para este proyecto, el SM garantizará que la comunicación sea fluida entre el PO y el equipo de desarrollo, protegerá al equipo de interrupciones externas y fomentará un ambiente de mejora continua para optimizar la productividad y la calidad del software "CarChain".

c. Ingeniero de Software Full-Stack (IS): Será responsable del desarrollo e implementación de la plataforma Carchain, tanto del backend (blockchain, contratos

inteligentes) como del frontend (interfaz de usuario). Este rol requiere un perfil técnico con experiencia en desarrollo web, blockchain (Solidity, frameworks como Truffle o Hardhat), bases de datos y diseño de APIs. La capacidad de trabajar en todo el stack tecnológico permitirá una mayor flexibilidad y rapidez en el desarrollo.

d. Analista Programador Blockchain (AP): Este rol se centrará en el desarrollo y la implementación de los contratos inteligentes y la lógica de negocio relacionada con blockchain. Deberá tener un profundo conocimiento de Solidity y la plataforma Ethereum (u otra plataforma blockchain elegida), así como experiencia en el diseño e implementación de contratos inteligentes seguros y eficientes. Colaborará estrechamente con el Ingeniero de Software para integrar la lógica blockchain con el resto de la plataforma.

e. Analista de Calidad (QA): Garantizará la calidad del software a través de pruebas exhaustivas. Diseñará y ejecutará planes de pruebas, realizará pruebas funcionales, de integración, de seguridad y de rendimiento. Este rol es crucial para asegurar la estabilidad, la seguridad y la usabilidad de la plataforma Carchain. También se encargará de documentar los errores y trabajar con el equipo de desarrollo para su corrección.

Este equipo de trabajo interdisciplinario, con habilidades complementarias, será fundamental para el éxito del proyecto Carchain. Cada miembro del equipo aportará su experiencia y conocimientos específicos para asegurar el desarrollo de una plataforma robusta, segura y funcional que cumpla con los objetivos establecidos y brinde valor a todos los actores involucrados en la industria automotriz.

Tabla 1-7. Planificación de actividades (Carta Gantt)

Nombre de tarea	Duración	Nombres de los Recursos
Sprint 1: Definición de Requerimientos (5 días)		
Toma de requerimientos	1 día	PO, SM, IS, AP, QA
Refinamiento del Backlog del Producto	2 días	PO, SM, IS, AP, QA
Planificación del Sprint 1	2 días	PO, SM, IS, AP, QA
Sprint 2: Desarrollo del MVP - Backend (10 días)		
Desarrollo de contratos inteligentes	6 días	SM, AP, IS
Pruebas unitarias de contratos	2 días	SM, AP, QA
Integración de contratos con backend	2 días	SM, IS, AP
Sprint 3: Desarrollo del MVP - Frontend (10 días)		
Desarrollo de interfaz de usuario principal	7 días	SM, IS
Pruebas de usabilidad	1 día	PO, SM, QA, IS
Integración frontend-backend	2 días	SM, IS, AP
Sprint 4: Pruebas de Integración y Ajustes (7 días)		
Pruebas de integración del sistema	4 días	PO, SM, QA, IS, AP
Corrección de errores	3 días	SM, IS, AP
Sprint 5: Despliegue y Lanzamiento (5 días)		
Configuración de entorno de producción	1 día	SM, IS
Despliegue de la aplicación	1 día	SM, IS
Pruebas de humo en producción	1 día	SM, QA
Lanzamiento oficial	2 días	PO, SM
Sprint 6: Mejoras y Mantenimiento (13 días)		
Monitoreo y mantenimiento	5 días	SM, IS, AP
Soporte a usuarios	3 días	PO, SM, IS
Desarrollo de nuevas funcionalidades / Refinamiento del Backlog	5 días	PO, SM, IS, AP, QA

Fuente: Elaboración propia.

1.8.2. Administración de riesgos

Para una gestión proactiva del proyecto “CarChain”, es fundamental identificar, evaluar y planificar la mitigación de los posibles riesgos que puedan afectar su desarrollo. Este proceso se divide en tres fases: identificación y categorización, evaluación y priorización, y finalmente, el plan de mitigación.

Tabla 1-8. Riesgos del Proyecto

N°	Categoría	Riesgo	Posibilidad	Impacto	Costo de retiro	Puntaje de riesgo	Prioridad
1	Proyecto	Stakeholders no conformes con la arquitectura blockchain	4	7	6	30	Alta
2	Proyecto	Cambios frecuentes en requisitos del proyecto	6	8	5	25	Muy alta
3	Tecnológico	Fallos en la infraestructura blockchain (conectividad, rendimiento)	3	9	7	35	Media
4	Humano	Ausencia de miembros clave del equipo	5	6	6	30	Alta
5	Tecnológico	Problemas de escalabilidad y rendimiento de la plataforma	4	8	6	28	Alta
6	Humano	Retrasos en la revisión y aprobación de entregables	7	5	4	16	Muy alta
7	Tecnológico	Incompatibilidad de la interfaz de usuario en distintos navegadores o dispositivos	5	6	5	30	Alta
8	Tecnológico	Imprecisiones en los análisis de datos y reportes	4	7	6	30	Alta
9	Proyecto	Solución final no cumple con expectativas de usuarios	6	8	7	21	Muy alta

Fuente: Elaboración propia.

Tabla 1-9. Mitigación de los riesgos

N°	Riesgo	Prioridad	Señal de alerta	Responsable	Plan preventivo
6	Retrasos en revisión y aprobación de entregables	Muy alta	Comentarios pendientes de revisión > 3 días	Jefe de Proyecto	Definir calendario de revisiones con fechas límite y notificaciones automáticas de atraso.
9	Solución no cumple con expectativas de usuarios	Muy alta	Feedback negativo en demos >= 2 sprints consecutivos	Jefe de Proyecto	Reuniones de validación al final de cada sprint con prototipos interactivos y aprobación formal.
2	Cambios frecuentes en requisitos del proyecto	Muy alta	Solicitudes de cambio fuera de proceso formal	Jefe de Proyecto	Implementar tablero de gestión de cambios (JIRA/Trello) y flujos de aprobación definidos.
1	Stakeholders no conformes con arquitectura blockchain	Alta	Objeciones sobre diseño en revisiones de arquitectura	Arquitecto Blockchain	Sesiones de alineación técnicas; talleres de validación de arquitectura con todas las partes.
4	Ausencia de miembros clave del equipo	Alta	Ausencias reportadas > 2 días	Jefe de Proyecto	Plan de sucesión y capacitación cruzada documentada; identificar backups para cada rol.
5	Problemas de escalabilidad y rendimiento	Alta	Métricas de latency o CPU > umbral	Arquitecto Blockchain	Pruebas de carga mensuales y mejoras continuas al diseño; escalamiento horizontal previsto.
7	Incompatibilidad de la interfaz de usuario	Alta	Tickets de soporte de UI por dispositivo/versión	Desarrollador Frontend	Pruebas de regresión en navegadores principales y frameworks de diseño responsive.
8	Imprecisiones en análisis de datos y reportes	Alta	Discrepancias entre fuentes > 5%	Analista de Datos	Definir validaciones en pipelines ETL y auditorías periódicas de calidad de datos.
3	Fallos en infraestructura blockchain	Media	Caída de nodos o errores de conexión	Operaciones IT	Monitoreo 24/7 con alertas; mecanismo de fail-over automático y pruebas de stress trimestrales.

Fuente: Elaboración propia

1.8.3. Estimación de costos.

En primer lugar, es importante definir los costos asociados para cada recurso, por lo que se genera la siguiente tabla con referencias de sueldo de profesionales promedio de junio de 2024.

Tabla 1-10. Tabla de costos mensuales y su respectivo porcentaje

Profesional	Sueldo (UF)	Porcentaje del total.
Product Owner (PO)	75	23%
Scrum Master (SM)	55	17%
Ingeniero de Software Full-Stack (IS)	70	22%
Analista Programador Blockchain (AP)	75	24%
Analista de Calidad (QA)	45	14%
Total	**320**	**100%**

Fuente: Elaboración propia.

Tabla 1-11. Costos mensuales reales por profesional en UF

Profesional	Sueldo (UF)	Porcentaje del total.	Costo mensual real UF
Product Owner (PO)	75	23%	90
Scrum Master (SM)	55	17%	66
Ingeniero de Software Full-Stack (IS)	70	22%	84
Analista Programador Blockchain (AP)	75	24%	90
Analista de Calidad (QA)	45	14%	54
Total	**320**	**100%**	**384**

Fuente: Elaboración propia.

Obtenemos un costo mensual real de **384 UF**.

De la misma manera, es necesario obtener el valor real por día trabajado (costo diario) con la siguiente formula:

$$Costo\ diario = \frac{Costo\ mensual\ real}{20}$$

Tabla 1-12. Costos diarios por profesional

Profesional	Costo mensual real	Costo diario
Product Owner (PO)	90	4,5
Scrum Master (SM)	66	3,3
Ingeniero de Software Full-Stack (IS)	84	4,2
Analista Programador Blockchain (AP)	90	4,5
Analista de Calidad (QA)	54	2,7
Total	**384**	**19,2**

Fuente: Elaboración propia.

Continuaremos con los valores de cada profesional por cada día trabajado, para esto seguimos la siguiente formula:

$$Costo\ recurso = Costo\ diario\ \times Dias\ trabajados$$

Tabla 1-13. Costo de días trabajados por cada profesional en UF

Profesional	Costo diario	días trabajados	Costo recurso
Product Owner (PO)	4,5	20	90
Scrum Master (SM)	3,3	50	165
Ingeniero de Software Full-Stack (IS)	4,2	47	197,4
Analista Programador Blockchain (AP)	4,5	34	153
Analista de Calidad (QA)	2,7	18	48,6
Total Costo Proyecto			**654**

Fuente: Elaboración propia.

Como último paso, se calcula el costo y precio del proyecto para el cliente. Este costo se calcula tomando el costo del proyecto más el costo variable.

Definimos que la utilidad es del 20% sobre el costo del proyecto y con un riesgo del 3%, lo cual obtenemos un total de:

$$Utilidad\ del\ 20\% = utildiad\ \times Costo\ Proyecto\ \div 100$$
$$Riesgo\ del\ 3\% = riesgo\ \times Costo\ Proyecto\ \div 100$$

Tabla 1-14. Precio final del proyecto

Ítem	Total (UF)
Costo Proyecto	654
Costo Variable	0
Costo para proyecto Cliente	654

20% de utilidad	130,8
3% de riesgo	19,62
Total (UF)	**804**

Fuente: Elaboración propia.

Se obtiene como precio final del proyecto un total de **804 UF**, monto que representa la suma del costo del proyecto, más la utilidad y el porcentaje destinado a la cobertura de riesgos.

2. CAPÍTULO 2. ASPECTOS RELEVANTES DEL ANÁLISIS

2.1. ASPECTOS RELEVANTES DEL ANÁLISIS.

En este capítulo se analizará la solución elegida, sus diagramas de actividades, diferentes casos de uso y la estructura funcional propuesta.

2.1.1. Descripción de la solución propuesta

Como se indicó en el capítulo anterior, luego de realizar entrevistas con los representantes de talleres automotrices y plataformas de compra-venta, se evaluó la necesidad de automatizar los procesos de registro de mantenciones y reparaciones. Por lo tanto, se propuso una solución que permita gestionar estas actividades de manera descentralizada y segura, utilizando tecnología blockchain. La solución facilitará el registro, control y generación de reportes de todas las operaciones realizadas a los vehículos, garantizando la integridad y transparencia de la información. Dentro de las principales funcionalidades que tiene el sistema se encuentran:

a) Registrar vehículos y sus propietarios.
b) Registrar mantenciones y reparaciones.
c) Generar y actualizar bitácoras de mantenimiento inmutables.
d) Administrar usuarios y permisos del sistema.
e) Generar reportes detallados y estadísticas.
f) Enviar recordatorios y notificaciones automáticas a los usuarios.

2.1.2. Diagrama de actividades

En la próxima figura, se diagrama la llegada de un nuevo vehículo al taller representa una actividad clave para el sistema.

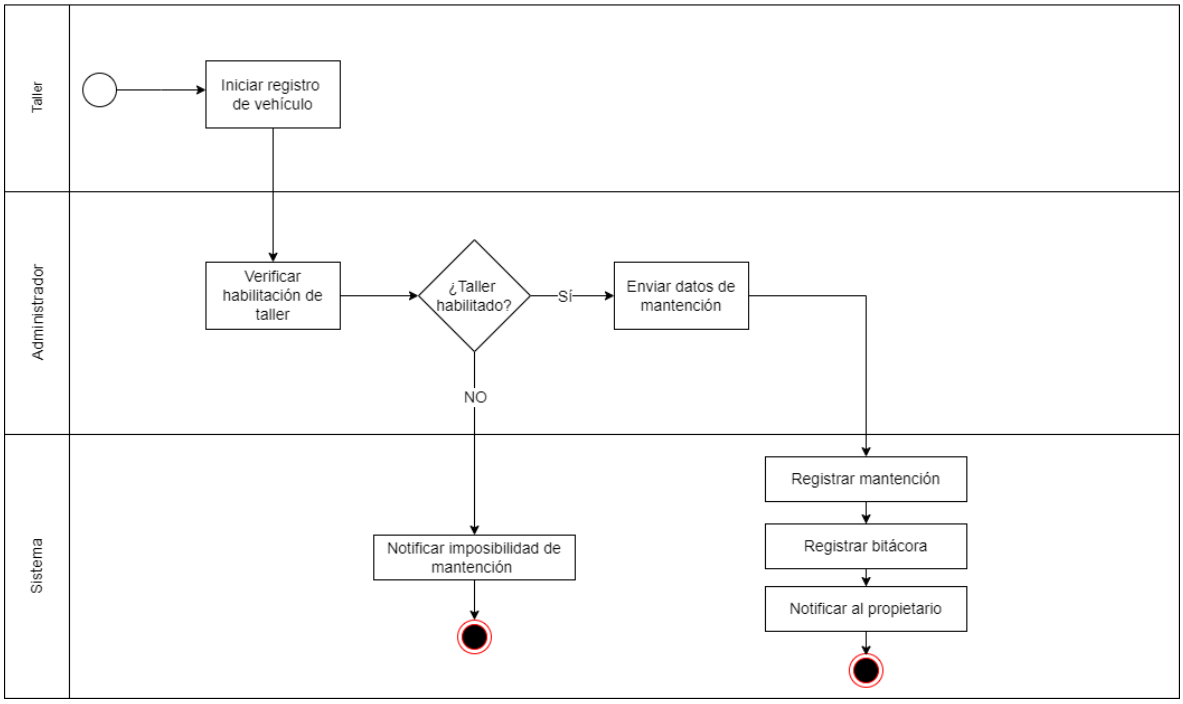


Figura 2.1
Fuente: Elaboración propia

2.1.3. Requerimientos del sistema

Para el correcto desarrollo de la plataforma, se entrevistaron a diferentes locales comerciales, taller de vehículos, páginas de compra venta entre otros para conocer más las necesidades y dolores de cada uno de ellos y crear un software de calidad. En esto surgen los requisitos funcionales y no funcionales.

2.3.1 Requerimientos Funcionales del sistema.

Se describe el comportamiento de cada una de las funcionalidades que el sistema debe cumplir.

ID	Requerimiento Funcional
RF-01	El sistema debe permitir registrar vehículos con datos mínimos: matrícula, marca, modelo, año y datos de contacto del propietario.
RF-02	El sistema debe mostrar la lista de talleres certificados y permitir la evaluación de servicios en una escala de 1 a 5.
RF-03	El sistema debe permitir consultar el historial de mantenimiento de un vehículo por matrícula y rango de fechas.
RF-04	El sistema debe permitir al propietario registrar servicios de mantenimiento realizados fuera de un taller certificado.
RF-05	El sistema debe enviar alertas automáticas de mantenimiento mediante correo electrónico, SMS o notificación push.
RF-06	El sistema debe permitir adjuntar documentos y fotos (formatos JPG, PNG, PDF) asociados a mantenciones y reparaciones.
RF-07	El sistema debe permitir a los talleres certificados registrar mantenciones y reparaciones, indicando tipo de servicio, fecha, mecánico y costo.
RF-08	El sistema debe permitir al administrador generar reportes de mantenciones en formatos PDF y CSV, incluyendo número de servicios, costos y frecuencia.
RF-09	El sistema debe validar la certificación de talleres mediante credenciales (ID, fecha de vigencia).
RF-10	El sistema debe permitir gestionar usuarios y permisos mediante operaciones CRUD (crear, leer, actualizar y eliminar).
RF-11	El sistema debe permitir aprobar o rechazar registros de mantenciones y reparaciones, incluyendo comentarios.
RF-12	El sistema debe enviar notificaciones programadas al actualizar la bitácora de mantenciones.
RF-13	El sistema debe generar una bitácora inmutable de las mantenciones y reparaciones aprobadas utilizando blockchain.
RF-14	El sistema debe permitir exportar el historial de mantenciones en formatos PDF y CSV.
RF-15	El sistema debe permitir a los propietarios evaluar servicios de talleres mediante calificación y comentarios.

2.3.2 Requerimientos No Funcionales del sistema

Corresponde a las restricciones o condiciones que el sistema debe cumplir.

ID	Requerimiento	Métrica de Validación
RNF-01	Inmutabilidad: Los registros aprobados y escritos en la blockchain deben ser inalterables.	Intento fallido de modificar un registro en la blockchain mediante un script de prueba.
RNF-02	Accesibilidad: La aplicación web debe ser accesible desde internet a través de un navegador web estándar.	Conexión exitosa desde una red externa al entorno de desarrollo.
RNF-03	Seguridad: Proteger la información contra accesos no autorizados mediante cifrado, roles y protección contra ataques comunes (OWASP).	Escaneo de vulnerabilidades sin resultados críticos. Pruebas de roles de usuario exitosas.
RNF-04	Escalabilidad: La arquitectura debe soportar un crecimiento del 50% en usuarios sin degradación del rendimiento.	Pruebas de carga simulando +50% de usuarios con un aumento de latencia <15%.
RNF-05	Usabilidad: Un usuario nuevo debe poder completar tareas clave de forma intuitiva y sin capacitación.	Prueba con 5 usuarios: >80% registra un vehículo en <3 min. Puntuación SUS > 70.
RNF-06	Disponibilidad: Disponibilidad del 99.5% (uptime), excluyendo mantenimientos planificados.	Monitoreo con AWS CloudWatch. Inactividad no planificada < 3.6 horas al mes.
RNF-07	Cumplimiento Normativo: Adhesión a la Ley N° 19.628 sobre protección de datos personales de Chile.	Auditoría interna del flujo de datos y políticas de privacidad del sistema.
RNF-08	Recuperabilidad: En caso de desastre, RPO de 24 horas y RTO de 4 horas.	Simulacro de recuperación de BD cumpliendo los tiempos y puntos de restauración definidos.
RNF-09	Rendimiento: Tiempo de carga de páginas < 3s. Confirmación de escrituras en BD < 2s.	Medición con herramientas de navegador y APM bajo carga simulada.
RNF-10	Auditabilidad: Registrar todas las acciones críticas (creación, aprobación, permisos) en un log seguro.	Revisión del log de auditoría para verificar el registro completo de un flujo de prueba.
RNF-11	Integración: Exponer una API RESTful segura y documentada para sistemas externos autorizados.	Un cliente de prueba consume exitosamente un endpoint de la API.
RNF-12	Soporte de Idiomas: Soporte completo y funcional para los idiomas Español (Chile) e Inglés.	Revisión por hablantes nativos y verificación de archivos de recursos de idioma.

RNF-13	Soporte Técnico: Primera respuesta a tickets/correos en menos de 24 horas hábiles.	Medición del tiempo de respuesta para 5 tickets de prueba.
RNF-14	Mantenibilidad: Despliegue de actualizaciones con una interrupción del servicio inferior a 10 minutos.	Cronometrar el tiempo de indisponibilidad durante un despliegue de prueba.
RNF-15	Compatibilidad: Soporte para últimas 2 versiones de Chrome, Firefox, Edge. Diseño responsivo para iOS 15+ y Android 10+.	Batería de pruebas de aceptación manual en las plataformas especificadas.

2.1.4. Estructura funcional del sistema.

El sistema “CarChain” contará con diferentes tipos de permisos, lo cual permitirá una adecuada administración y seguridad dentro de la plataforma. Esto asegurará que cada usuario pueda acceder únicamente a las funcionalidades que le competen, garantizando la integridad y confiabilidad de la información. Los roles y niveles de permiso definidos para el sistema son los siguientes:

1. **Administrador del Sistema**: Tendrá control total sobre la plataforma. Sus responsabilidades y permisos incluyen:
    - **Gestión de usuarios y permisos**: Podrá agregar, modificar y eliminar lógico de usuarios, asignando los roles correspondientes a cada uno.
    - **Aprobación de registros**: Tendrá la facultad de aprobar o rechazar registros de mantenciones y reparaciones ingresados por los talleres certificados.
    - **Visualización de información completa**: Acceso ilimitado a todos los módulos del sistema, incluyendo historiales de vehículos, reportes y estadísticas.
    - **Configuración del sistema**: Podrá realizar ajustes en las configuraciones generales del sistema para garantizar su correcto funcionamiento.
    - **Seguridad y soporte**: Responsable de mantener la seguridad de la plataforma y brindar soporte a los usuarios en caso de inconvenientes.
2. **Taller Certificado**: Representa a los talleres mecánicos autorizados para operar en el sistema. Sus permisos incluyen:

- **Registro de mantenciones y reparaciones**: Podrán ingresar información detallada sobre las mantenciones y reparaciones realizadas a los vehículos de sus clientes.
- **Gestión de citas**: Capacidad para agendar, reagendar y cancelar citas con propietarios de vehículos.
- **Adjuntar documentación**: Podrán adjuntar documentos y fotos relacionados con los servicios prestados.
- **Consulta de historial**: Acceso al historial de mantenciones y reparaciones de los vehículos que han atendido.
- **Recepción de evaluaciones**: Podrán recibir retroalimentación de los propietarios sobre los servicios brindados.

3. **Propietario de Vehículo**: Corresponde a los dueños de los vehículos registrados en el sistema. Sus permisos abarcan:
   - **Registro y actualización de vehículos**: Podrán registrar sus vehículos en la plataforma y actualizar la información cuando sea necesario.
   - **Consulta de historial**: Acceso completo al historial de mantenciones y reparaciones de sus vehículos.
   - **Gestión de citas**: Capacidad para agendar, reagendar y cancelar citas en talleres certificados.
   - **Recepción de notificaciones**: Recibirán alertas y notificaciones sobre mantenciones programadas o pendientes.
   - **Evaluación de servicios**: Podrán evaluar y comentar sobre los servicios recibidos en los talleres.
4. **Plataformas de Compra-Venta de Vehículos**: Entidades que integran "CarChain" para mejorar la transparencia en las transacciones vehiculares. Sus permisos incluyen:
   - **Acceso al historial de vehículos**: Podrán consultar el historial de mantenciones y reparaciones de los vehículos listados en su plataforma.
   - **Visualización limitada**: Acceso restringido únicamente a los vehículos que estén en proceso de compra-venta dentro de su plataforma.
   - **Generación de reportes**: Podrán generar reportes simplificados para presentar a potenciales compradores.

- **Integración con sistemas internos**: Capacidad para integrar la información de “CarChain” con sus propios sistemas, mejorando la confianza en las transacciones.

5. **Usuario Visitante**: Usuarios que, sin estar registrados, desean obtener información general o verificar la autenticidad de la información. Sus permisos son:
   - **Consulta de talleres certificados**: Podrán visualizar la lista de talleres certificados disponibles en el sistema.
   - **Información general**: Acceso a materiales informativos sobre el sistema “CarChain” y sus beneficios.
   - **Verificación básica**: Podrán verificar, de forma limitada, la existencia de registros asociados a un vehículo específico utilizando identificadores públicos como la matrícula.

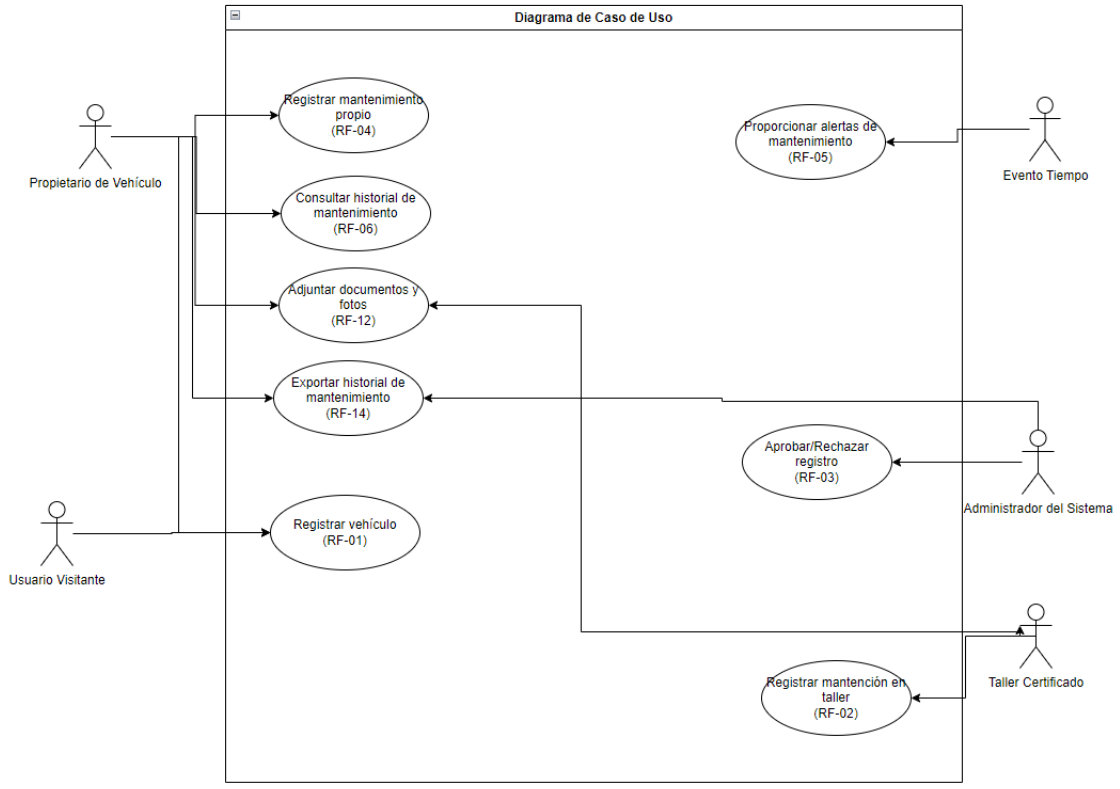


Figura 2.2
Fuente: Elaboración propia

2.2. Matriz de trazabilidad

A continuación, se presenta la matriz de trazabilidad, donde se indica que cada requerimiento es cubierto por los casos de Uso presentados anterior.

Requerimiento Funcional (RF)	CU 01	CU 02	CU 03	CU 04	CU 05	CU 06	CU 07	CU 08	CU 09	CU 10	CU 14	CU 15
RF 01: Registrar vehículo	x											
RF 02: Registrar mantenciones		x										
RF 03: Registrar reparaciones			x									
RF 04: Aprobar registros				x	x							
RF 05: Generar bitácora						x						
RF 06: Consultar historial								x				
RF 07: Integración con plataformas							x					
RF 08: Generar reportes										x		
RF 09: Verificar certificación									x			
RF 10: Gestionar usuarios											x	
RF 14: Exportar historiales												x
RF 15: Evaluar servicios												x

2.3. Modelo Conceptual

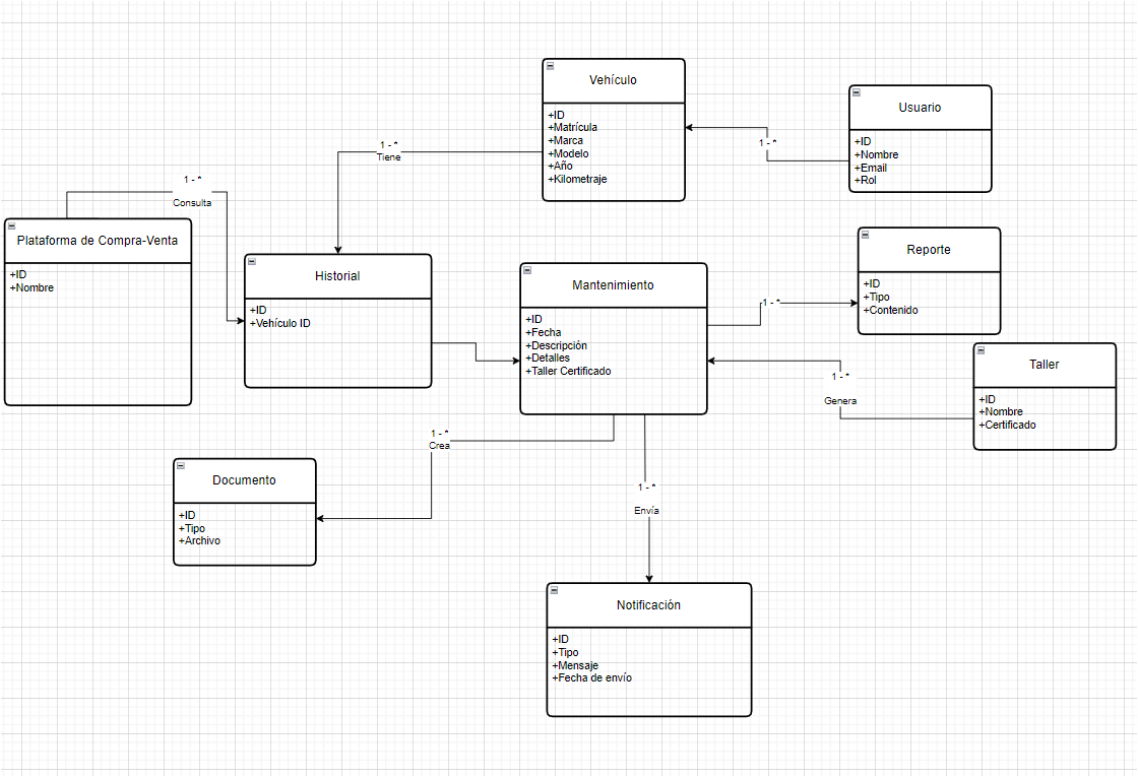


Figura 2.3
Fuente: Elaboración propia

2.4. Modelo de Casos de Uso

A pesar de que los casos de uso pueden ser ilimitados, a continuación, se detallan quince de los más relevantes que el sistema debe implementar para su correcto funcionamiento.

2.3.3 Caso de Uso 01: CU01 - Registrar Vehículo.

Caso de Uso	Registrar Vehículo
Código	CU01
Resumen	El propietario registra un vehículo en el sistema.
Tipo	Evidente
Referencias Cruzadas	Ninguna
Actor	Propietario
Precondiciones	Que el vehículo no esté registrado previamente en el sistema.

Postcondición	El vehículo se registró satisfactoriamente en el sistema.
Flujo Normal	
Usuario	**Sistema**
1.- El propietario ingresa los detalles del vehículo.	2.- El sistema verifica si el vehículo ya está registrado.
	3.- El sistema muestra el formulario de registro vacío.
4.- El propietario completa el formulario con los datos del vehículo y presiona guardar.	5.- El sistema guarda la información en la base de datos.
	6.- El sistema despliega el mensaje "vehículo registrado satisfactoriamente".
Flujo Alternativo	
2.- El sistema verifica si el vehículo ya está registrado	
Usuario	**Sistema**
	2.a.- El sistema busca los datos del vehículo en la base de datos.
	2.b.- El sistema indica que el vehículo ya está registrado.
	2.c.- El sistema muestra un mensaje de error indicando que el vehículo ya existe.

2.3.4 Diagrama de Secuencia: CU01 - Registrar Vehículo.

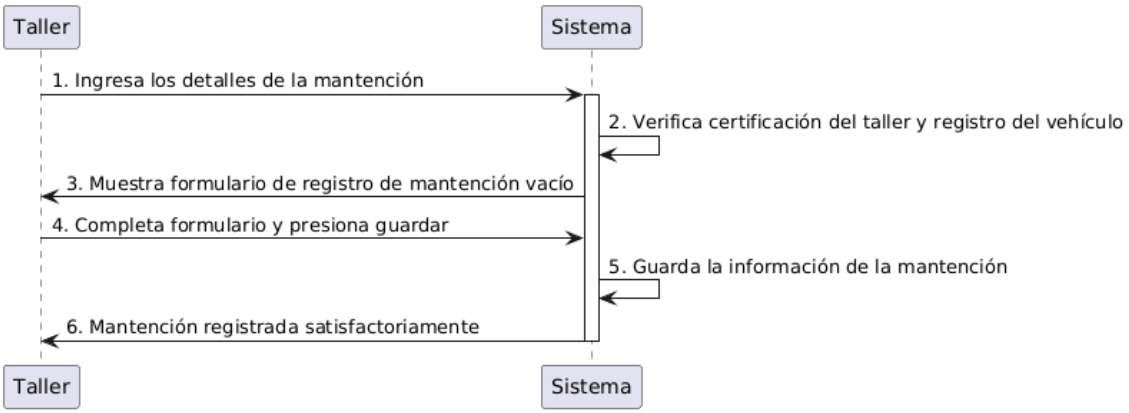


Figura 2.4
Fuente: Elaboración propia

2.3.5 Contrato CU-01 – Registrar Vehículo

Elemento	Detalle
Operación	Registrar un vehículo en el sistema.
Responsabilidades del actor	Proporcionar matrícula, marca, modelo, año y datos de contacto.
Responsabilidades del sistema	Verificar que el vehículo no esté previamente registrado; guardar la información en la base de datos; confirmar registro.
Postcondición	El vehículo quedó registrado satisfactoriamente en el sistema.

2.3.6 Caso de Uso 2: CU02 Registrar Nueva Mantención

Caso de Uso	Registrar Nueva Mantención
Código	CU02
Resumen	El taller certificado registra una nueva mantención para un vehículo en el sistema.
Tipo	Evidente
Referencias Cruzadas	CU11 - Verificar Certificación de Talleres, CU04 - Aprobar / Rechazar Registro
Actor	Taller Certificado
Precondiciones	El taller debe estar certificado y el vehículo debe estar registrado en el sistema.
Postcondición	La mantención se registró satisfactoriamente en el sistema.
Flujo Normal	
Usuario	**Sistema**
1.- El taller ingresa los detalles de la mantención.	2.- El sistema verifica si el taller está certificado y si el vehículo está registrado.
	3.- El sistema muestra el formulario de registro de mantención vacío.
4.- El taller completa el formulario con los datos de la mantención y presiona guardar.	5.- El sistema guardó la información de la intervención, asignándole un estado de 'pendiente de aprobación'.
	6.- El sistema despliega el mensaje “mantención registrada satisfactoriamente”.
Flujo Alternativo	

2.- El sistema verifica si el taller está certificado y si el vehículo está registrado	
Usuario	**Sistema**
	2.a.- El sistema busca los datos del taller y del vehículo en la base de datos.
	2.b.- El sistema indica que el taller no está certificado o el vehículo no está registrado.
	2.c.- El sistema muestra un mensaje de error indicando que la mantención no puede ser registrada.

2.3.7 Diagrama de Secuencia: CU02 Registrar Nueva Mantención.

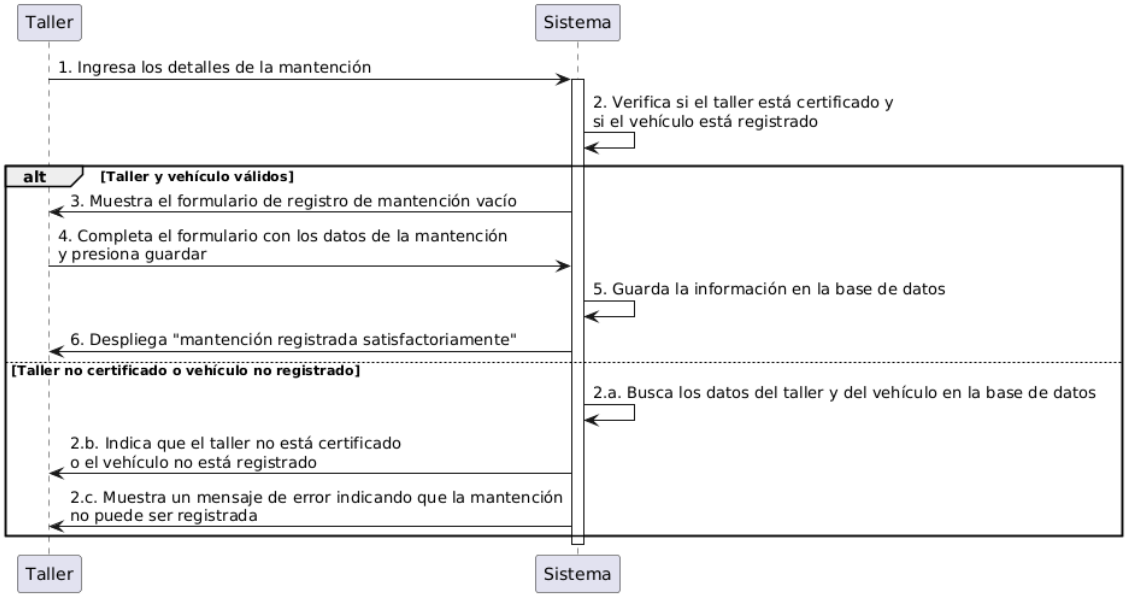


Figura 2.5
Fuente: Elaboración propia

2.3.8 Contrato CU-02 – Registrar Nueva Mantención

Elemento	Detalle
Operación	Registrar una nueva mantención o reparación para un vehículo.
Responsabilidades del actor	Ingresar los detalles de la intervención, como el tipo de servicio, la fecha y los componentes involucrados.
Responsabilidades del sistema	Verificar que el taller esté certificado, validar que el vehículo exista, guardar el nuevo registro de mantención y confirmar la operación.
Postcondición	La nueva mantención quedó registrada satisfactoriamente en el sistema, pendiente de aprobación.

2.3.9 Caso de Uso 3: CU03 Generar Bitácora Inmutable

Caso de Uso	Generar Bitácora Inmutable
Código	CU03
Resumen	El sistema genera y actualiza una bitácora inmutable de mantenimiento para cada vehículo.
Tipo	Automático
Flujo Normal	
Usuario	**Sistema**
1.- El propietario solicita consultar el historial de mantenimiento de su vehículo.	2.- El sistema verifica la identidad del propietario.
	3.- El sistema recupera el historial completo de la bitácora del vehículo.
	4.- El sistema muestra el historial de mantenimiento al propietario.

2.3.10 Diagrama de Secuencia:: CU04 Consultar Historial de Vehículo

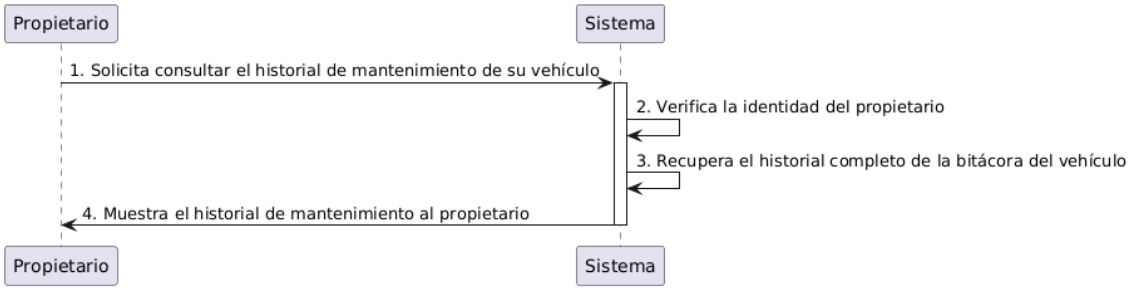


Figura 2.6
Fuente: Elaboración propia

2.3.11 Contrato CU-04 – Consultar Historial de Vehículo

Elemento	Detalle
Operación	Consultar el historial completo de mantenimientos y reparaciones de un vehículo específico.
Responsabilidades del actor	Proporcionar un identificador del vehículo (ej. matrícula) para iniciar la búsqueda.

Responsabilidades del sistema	Verificar la identidad y permisos del solicitante, buscar todos los registros asociados al vehículo en la base de datos y en la blockchain, y presentarlos d forma ordenada.
Postcondición	El historial completo del vehículo fue desplegado al usuario.

2.3.12 Caso de Uso 5: CU05 Integración con Plataformas de Compra-Venta

Caso de Uso	Integración con Plataformas de Compra-Venta
Código	CU05
Resumen	El sistema integra el historial de mantenimiento de vehículos con plataformas de compra-venta.
Tipo	Evidente
Referencias Cruzadas	CU08 – Exportar Historial
Actor	Sistema Externo
Precondiciones	El vehículo debe estar registrado y tener un historial de mantenimiento en el sistema.
Postcondición	La plataforma de compra-venta puede acceder al historial de mantenimiento del vehículo.
Flujo Normal	
Usuario	**Sistema**
	1.- El sistema externo solicita acceso al historial de mantenimiento del vehículo.
	2.- El sistema verifica la autenticidad del sistema externo.
	3.- El sistema recupera el historial de mantenimiento del vehículo.
	4.- El sistema envía el historial de mantenimiento al sistema externo.
	5.- El sistema externo muestra el historial de mantenimiento.
Flujo Alternativo	
2.- El sistema verifica la autenticidad del sistema externo	
Usuario	**Sistema**
	2.a.- El sistema no puede verificar la autenticidad del sistema externo.
	2.b.- El sistema rechaza la solicitud y envía un mensaje de error.

2.3.13 Diagrama de Secuencia:: CU05 Integración con Plataformas de Compra-Venta

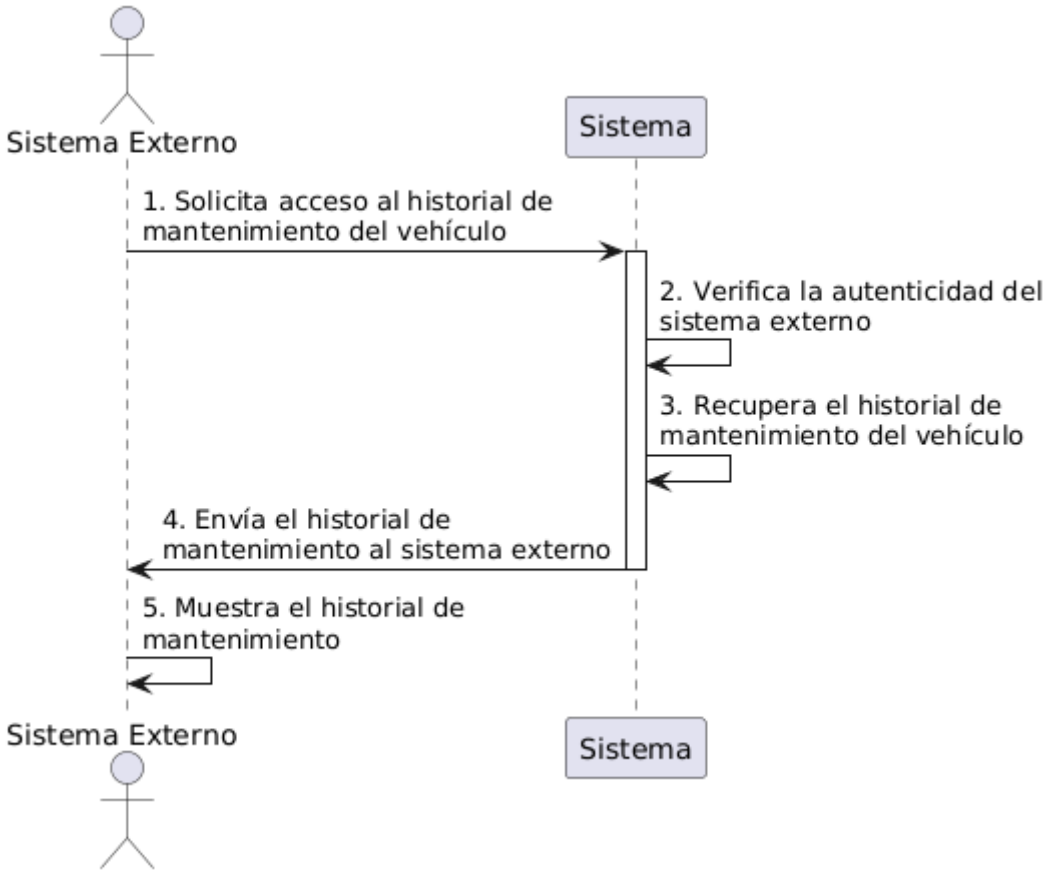


Figura 2.7

Fuente: Elaboración propia

2.3.14 Contrato CU-05 – Integración con Plataformas de Compra-Venta

Elemento	Detalle
Operación	Permitir que un sistema externo (plataforma de compra-venta) consulte el historial de un vehículo.
Responsabilidades del actor	(Sistema Externo) Realizar una solicitud de acceso al historial a través de la API, proveyendo las credenciales de autenticación necesarias.
Responsabilidades del sistema	Autenticar al sistema externo, verificar sus permisos, recuperar el historial del vehículo y enviarlo en el formato acordado.
Postcondición	El historial del vehículo fue entregado de forma segura a la plataforma externa.

2.3.15 Caso de Uso 6: CU06 Verificar Certificación de Talleres

Caso de Uso	Verificar Certificación de Talleres
Código	CU6
Resumen	El sistema verifica si un taller está certificado antes de permitir el registro de datos.
Tipo	Evidente
Referencias Cruzadas	CU02 - Registrar Nueva Mantención, CU03 - Registrar Nueva Reparación
Actor	Sistema
Precondiciones	El taller debe haber solicitado una certificación.
Postcondición	El sistema verifica y confirma si el taller está certificado.
Flujo Normal	
Usuario	**Sistema**
	1.- El sistema recibe una solicitud de verificación de certificación de un taller.
	2.- El sistema consulta la base de datos de talleres certificados.
	3.- El sistema confirma la certificación del taller.
	4.- El sistema permite que el taller registre datos de mantención o reparación.
Flujo Alternativo	
3.- El sistema confirma la certificación del taller	
Usuario	**Sistema**
	3.a.- El sistema no encuentra la certificación del taller en la base de datos.
	3.b.- El sistema rechaza la solicitud y muestra un mensaje de error indicando que el taller no está certificado.

2.3.16 Diagrama de Secuencia: 6: CU06 Verificar Certificación de Talleres

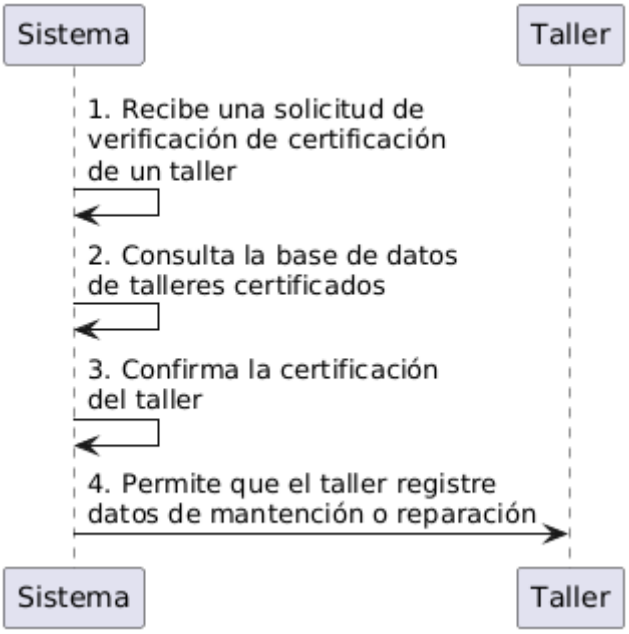


Figura 2.8
Fuente: Elaboración propia

2.3.17 Contrato CU-06 – Verificar Certificación de Talleres

Elemento	Detalle
Operación	Validar si un taller está autorizado para ingresar datos en el sistema.
Responsabilidades del actor	(Sistema) Recibir una solicitud de registro proveniente de un taller.
Responsabilidades del sistema	Consultar la base de datos de talleres para confirmar que las credenciales del taller son válidas y su certificación está vigente.
Postcondición	El estado de certificación del taller fue verificado.

2.3.18 Caso de Uso 07: CU07 Gestionar Usuarios del Sistema

Caso de Uso	Gestionar Usuarios del Sistema
Código	CU7
Resumen	El administrador gestiona usuarios y permisos dentro del sistema.
Tipo	Evidente
Referencias Cruzadas	Ninguna
Actor	Administrador

Precondiciones	El administrador debe estar autenticado en el sistema.
Postcondición	Los usuarios y permisos se actualizaron correctamente en el sistema.
Flujo Normal	
Usuario	**Sistema**
1.- El administrador selecciona la opción de gestión de usuarios.	2.- El sistema muestra la lista de usuarios actuales y sus permisos.
3.- El administrador añade, modifica o elimina usuarios y sus permisos.	4.- El sistema guarda los cambios en la base de datos.
	5.- El sistema despliega un mensaje indicando que la gestión de usuarios se ha completado correctamente.
Flujo Alternativo	
4.- El sistema guarda los cambios en la base de datos	
Usuario	**Sistema**
	4.a.- El sistema encuentra un error al guardar los cambios.
	4.b.- El sistema muestra un mensaje de error indicando que los cambios no pudieron ser guardados.

2.3.19 Diagrama de Secuencia:: CU07 Gestionar Usuarios del Sistema

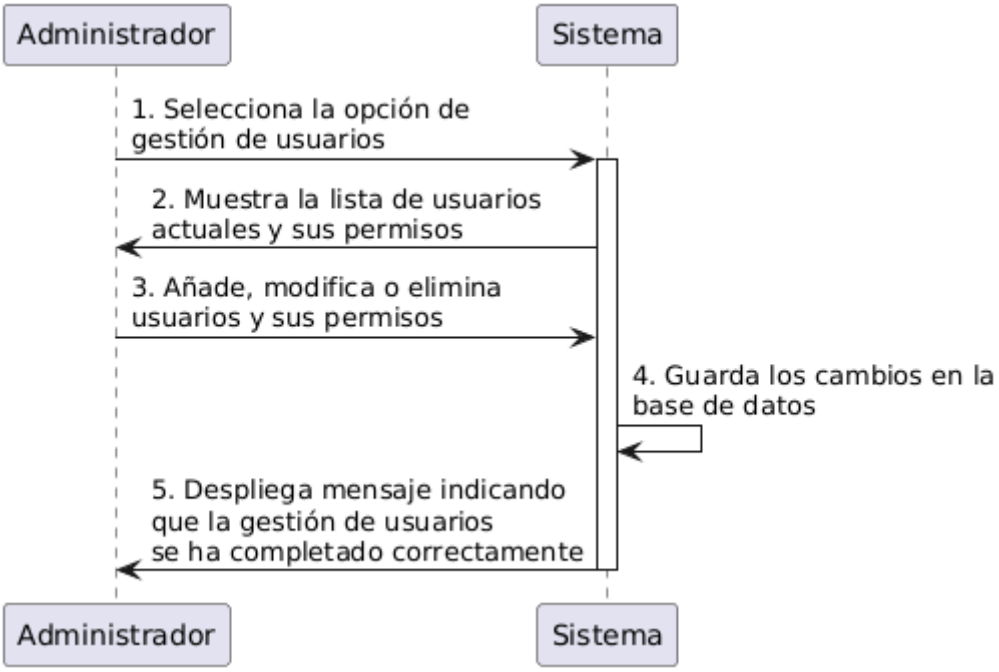


Figura 2.9
Fuente: Elaboración propia

2.3.20 Contrato CU-07: Gestionar Usuarios del Sistema

Elemento	Detalle
Operación	Administrar (crear, leer, actualizar, eliminar) los usuarios y sus respectivos permisos dentro del sistema.
Responsabilidades del actor	(Administrador) Seleccionar la operación de gestión y proporcionar los datos del usuario a modificar.
Responsabilidades del sistema	Aplicar los cambios en la base de datos de usuarios y permisos, y confirmar que la operación se completó exitosamente.
Postcondición	Los datos y permisos del usuario quedaron actualizados en el sistema.

2.3.21 Caso de Uso 8: CU08 Exportar Historial de Mantenimiento

Caso de Uso	Exportar Historial de Mantenimiento
Código	CU08
Resumen	El propietario exporta el historial de mantenimiento de su vehículo en formatos comunes (PDF, CSV).
Tipo	Evidente
Referencias Cruzadas	CU08 - Consultar Historial de Vehículo
Actor	Propietario
Precondiciones	El vehículo debe tener un historial de mantenimiento registrado en el sistema.
Postcondición	El propietario obtuvo un archivo exportado con el historial de mantenimiento.
Flujo Normal	
Usuario	**Sistema**
1.- El propietario selecciona la opción de exportar el historial de mantenimiento.	2.- El sistema muestra las opciones de formato (PDF, CSV).
3.- El propietario selecciona el formato deseado y presiona exportar.	4.- El sistema genera el archivo en el formato seleccionado.
	5.- El sistema proporciona el archivo para descargar al propietario.
Flujo Alternativo	
4.- El sistema genera el archivo en el formato seleccionado	
Usuario	**Sistema**
	4.a.- El sistema encuentra un error al generar el archivo.
	4.b.- El sistema muestra un mensaje de error indicando que el archivo no pudo ser generado.

2.3.22 Diagrama de Secuencia:: CU08 Exportar Historial de Mantenimiento

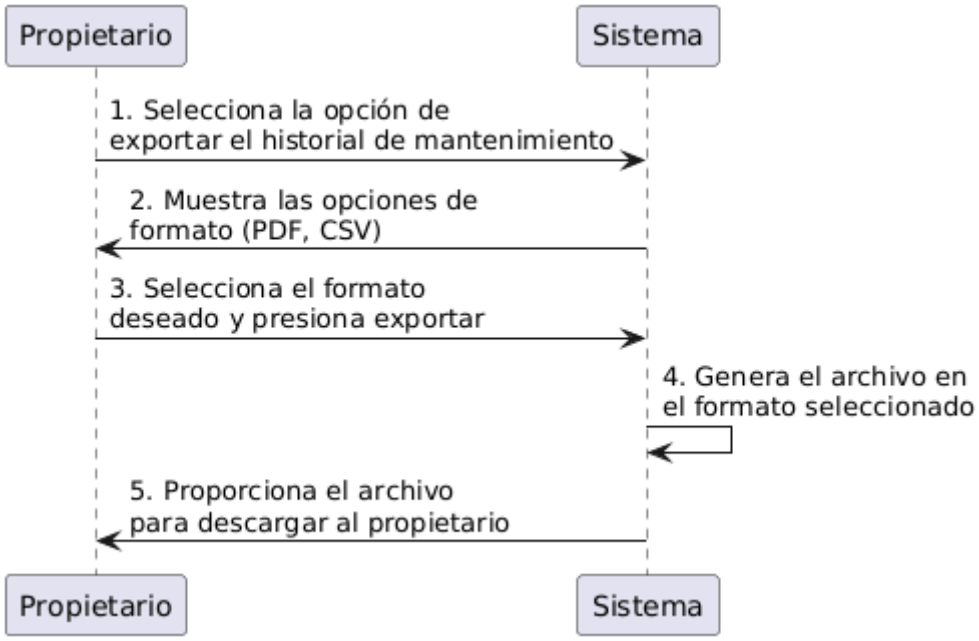


Figura 2.10
Fuente: Elaboración propia

2.3.23 Contrato CU-08: Exportar Historial de Mantenimiento

Elemento	Detalle
Operación	Generar un archivo descargable (PDF, CSV) con el historial de mantenimiento de un vehículo.
Responsabilidades del actor	Seleccionar la opción de exportar y elegir el formato deseado para el archivo.
Responsabilidades del sistema	Recopilar todos los registros del historial, darles formato según la selección del usuario y generar el archivo para su descarga.
Postcondición	El archivo con el historial de mantenimiento fue generado y descargado por el propietario.

2.3.24 Caso de Uso 09: CU09 Cambiar Propietario

Caso de Uso	Cambiar Propietario
Código	CU9
Resumen	El propietario cambia la información de propiedad de un vehículo en el sistema.
Tipo	Evidente
Referencias Cruzadas	-
Actor	Propietario, Taller
Precondiciones	El vehículo debe estar registrado previamente en el sistema
Postcondición	La información del propietario del vehículo se actualizó satisfactoriamente en el sistema.
Flujo Normal	
Usuario	**Sistema**
1.- El propietario, a través del taller, selecciona la opción de cambiar propietario.	
	2.- El sistema solicita la autenticación del propietario actual.
3.- El propietario ingresa sus credenciales de autenticación.	
	4.- El sistema verifica las credenciales y autentica al propietario.
5.- El propietario ingresa los datos del nuevo propietario y confirma la operación.	
	6.- El sistema actualiza la información de propiedad en la base de datos.
	7.- El sistema despliega el mensaje “Propietario actualizado satisfactoriamente”.
Flujo Alternativo	
1.- El cliente (nuevo propietario) accede al sistema e inicia sesión.	
	2.- El sistema verifica las credenciales y autentica al cliente.
3.- El cliente selecciona la opción de cambiar propietario e ingresa los datos del vehículo.	

	4.- El sistema verifica que el vehículo está registrado y solicita la confirmación del cambio.
5.- El cliente confirma la operación.	
	6.- El sistema actualiza la información de propiedad en la base de datos.
	7.- El sistema despliega el mensaje “Propietario actualizado satisfactoriamente”.
Usuario	**Sistema**
3.- El propietario ingresa sus credenciales de autenticación.	
	4.a.- El sistema detecta que las credenciales son incorrectas.
	4.b.- El sistema muestra un mensaje de error indicando que las credenciales son inválidas.
4.c.- El propietario decide reintentar o cancelar la operación.	

2.3.25 Diagrama de Secuencia:: CU009 Cambiar Propietario

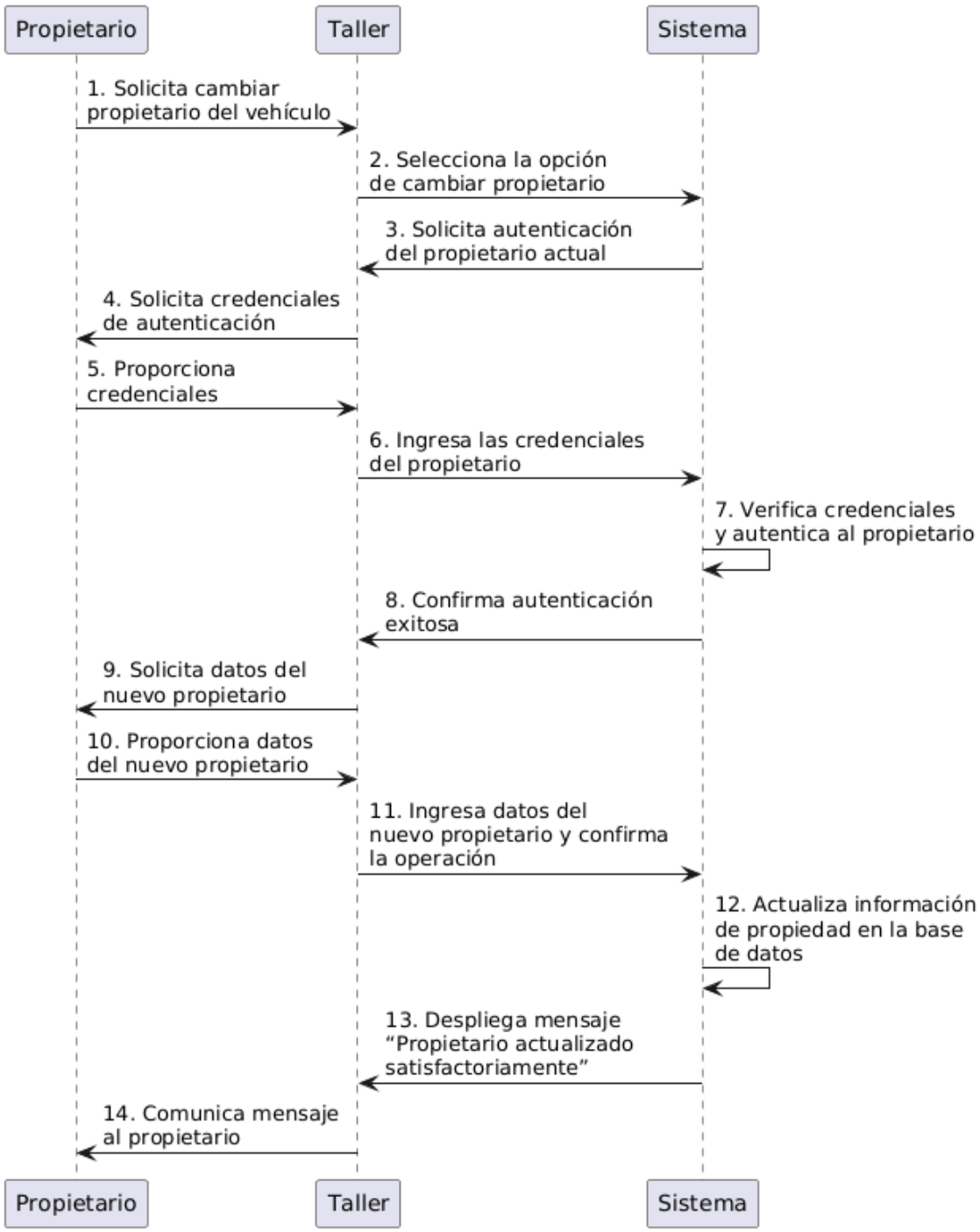


Figura 2.11
Fuente: Elaboración propia

2.3.1 Contrato CU-09: Cambiar Propietario

Elemento	Detalle
Operación	Transferir la propiedad de un vehículo a un nuevo dueño dentro del sistema.
Responsabilidades del actor	(Propietario actual) Autenticarse para confirmar su identidad y proporcionar los datos del nuevo propietario.
Responsabilidades del sistema	Validar las credenciales del propietario actual, actualizar el registro del vehículo con la información del nuevo propietario y notificar a ambas partes.
Postcondición	La propiedad del vehículo y su historial asociado quedaron actualizados con los datos del nuevo dueño.

CAPÍTULO 3: ASPECTOS RELEVANTES DEL DISEÑO PARA LA ALTERNATIVA SELECCIONADA.

3. CAPÍTULO 3. ASPECTOS RELEVANTES DEL DISEÑO

Durante este capítulo se desarrollará los aspectos relevantes para el diseño de la solución que se ha propuesto, esto tomara en cuenta la descripción de la arquitectura del software, diseño de datos, diagramas de secuencias extendidos y de colaboración además de tener los prototipos de interfaz para los usuarios que visualizaran el sistema, esto para lograr tener la idea general del sistema.

3.1. Arquitectura del software.

Se ha optado por una arquitectura híbrida para optimizar el rendimiento y la funcionalidad. La base de datos relacional, PostgreSQL, gestionará datos transaccionales y de uso frecuente que requieren alta velocidad de consulta, como perfiles de usuario, gestión de citas y datos de sesión. Por otro lado, Hyperledger Fabric se utilizará como una capa de 'notaría digital'. Su función es registrar de forma inmutable y auditable las transacciones críticas y ya validadas (mantenimientos y reparaciones finalizadas), garantizando así la integridad y la confianza en el historial del vehículo.

3.1.1. Diagrama de componentes.

En la figura 3-1 se presenta el diagrama de componentes de forma general y a continuación se explicará los diferentes componentes.

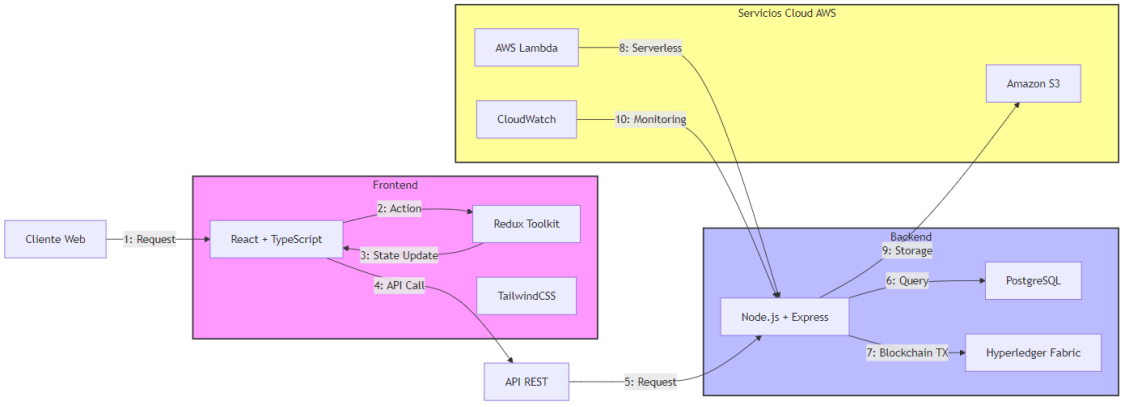


Ilustración 3-1

Figura 3.1. Diagrama de componentes, Arquitectura general del sistema.

Fuente: Elaboración propia.

En la figura 3-1~~antes mostrada~~, se señala de forma general las dependencias que están separadas por capas, las cuales son necesarias para el funcionamiento del sistema y a continuación, se explicara de forma separada cada una de estas:

3.1.2. Capa de presentación Frontend

Interfaces de usuario: Este componente es la parte del sistema con la que los usuarios podrán interactuar de forma directa con el sistema (Solicitar, Actualizar o ingresar).

Para este caso se utilizará:

HTML: Estructura que muestra información y el contenido que se presenta al usuario. Se eligió HTML porque al ser una página web es el lenguaje por defecto en los diferentes navegadores.

• **CSS** (con TailwindCSS): Framework de utilidades CSS para el formato y diseño del HTML. Se eligió TailwindCSS por su enfoque utility-first que permite un desarrollo más rápido y consistente, además de su excelente integración con React.

• **JavaScript** (TypeScript): Se utilizará TypeScript, una extensión tipada de JavaScript, para verificar y obtener una interactividad robusta con el usuario y el sistema. Se eligió TypeScript por su capacidad de detectar errores en tiempo de desarrollo y mejorar la mantenibilidad del código.

• **React**: Se usará para controlar interfaces de usuario complejas y gestionar grandes cantidades de datos. Se eligió React junto con Redux Toolkit para el manejo del estado global, aprovechando su amplio ecosistema de librerías y su extensa documentación y comunidad.

3.1.3. Capa lógica de negocio Backend

Lógica de la aplicación: Este componente se encargará de procesar las solicitudes del usuario, interactuando con la base de datos y determinando la resolución que se le dará al usuario. Se implementa mediante controladores y servicios en Express.js.

Acceso a datos: Este componente se encargará de interactuar directamente con PostgreSQL y Hyperledger Fabric, gestionando las consultas, modificaciones y transacciones en la base de datos relacional y la blockchain respectivamente.

Gestión de sesiones: Este componente manejará la autenticación y autorización de usuarios, manteniendo el estado de las sesiones y la seguridad de las APIs.

Para estos casos se utilizará:

• **Node.js**: Entorno de ejecución para JavaScript del lado del servidor que procesará las solicitudes del cliente y se comunicará con la base de datos. Se eligió Node.js por su naturaleza asíncrona y su excelente rendimiento en operaciones I/O.

• **Express.js**: Framework web para Node.js que facilita la creación de APIs REST y el manejo de rutas. Se eligió Express por su simplicidad, flexibilidad y amplia adopción en la comunidad.

• **AWS Lambda**: Para funciones serverless que requieran procesamiento bajo demanda, reduciendo costos y mejorando la escalabilidad.

• Amazon S3: Para el almacenamiento de archivos y recursos estáticos.

• CloudWatch: Para el monitoreo y logging de la aplicación, permitiendo detectar y resolver problemas rápidamente.

Esta arquitectura moderna permite una mejor escalabilidad, mantenibilidad y rendimiento, aprovechando las ventajas de la computación en la nube y las arquitecturas serverless cuando sea necesario.

3.1.4. Capa de base de datos

Tabla de datos: Este componente se encargará de gestionar el almacenamiento y funcionamiento de los datos.

Para este caso se utilizará:

- PostgreSQL: Base de datos relacional, la cual se utilizará para almacenar los datos de la aplicación. Se eligió PostgreSQL, debido a su amplia documentación y gran variedad de usos.

3.1.5. Sistema operativo y herramientas

A continuación, se señalan el sistema operativo necesario y herramientas que se utilizaron para la creación del sistema.

Sistema operativo: Se utilizará el sistema operativo de Windows, el cual tendrá la versión más utilizada y con mayor soporte por parte de la comunidad y del proveedor, la cual es Windows 10.
Entorno de desarrollo: Se utilizará como IDE Visual Studio Code, el cual cuenta con una gran cantidad de librerías y un fácil uso con las diferentes integraciones con las capas antes mencionadas.
Servicios de la nube: Se utilizará AWS para proporcionar una infraestructura escalable y servicios adicionales de despliegue.

Estas herramientas y lenguajes señalados por capas fueron elegidos en función a su robustez, soporte por parte de la comunidad y lo más importante es que se adecuan a los requisitos del proyecto, además de tener una mayor familiaridad y agilidad por parte del equipo de desarrollo, el cual se encargará de crear el proyecto.

3.2. Diseño de datos.

En esta sección se presentarán los modelos de clases, el modelo relacional y por último el diccionario de datos para cada clase.

3.2.1. Modelo de clases

A continuación, se mostrará el diagrama de clases del sistema.

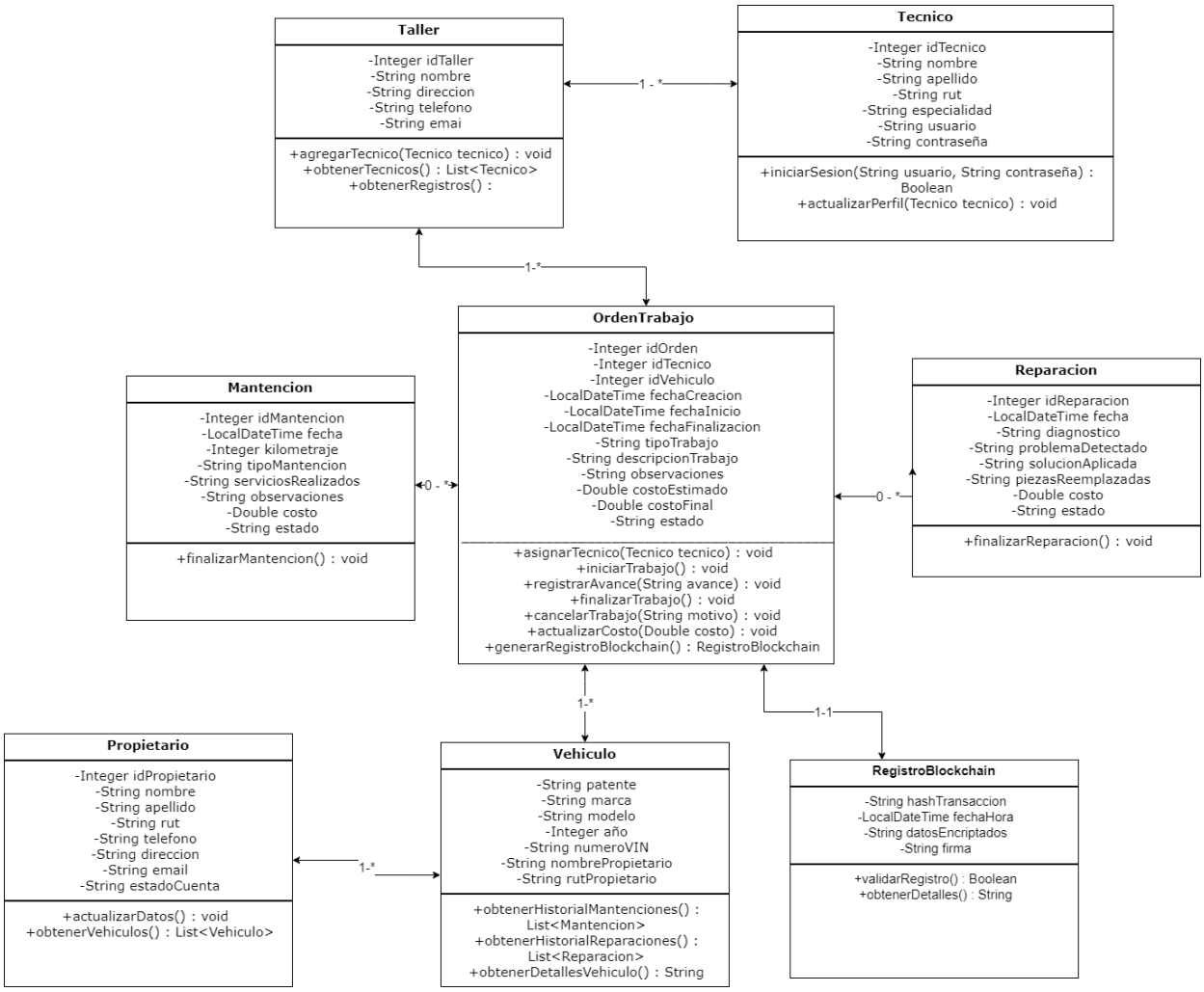


Figura 3.2

Fuente: Elaboración propia.

3.2.2. Modelo relacional

A continuación, se mostrará el modelo relacional para el sistema.

Figura 3.3

Fuente: Elaboración propia.

3.2.3. Diccionario de datos

3.2.3.1. Diccionario de la Tabla: Mantención

Nombre	mantencion			
Descripción	Detalla los servicios de mantención preventiva realizados en una orden de trabajo.			
Clave primaria	id_mantencion			
Clave foránea	id_orden (orden_trabajo)			
Nombre	Tipo	Largo	Nulo	Descripción
id_mantencion	integer		NO	Identificador único para el registro de mantención.
id_orden	integer		NO	Clave foránea que vincula la mantención a una orden de trabajo.
fecha	timestamp		SÍ	Fecha en que se realizó la mantención.
kilometraje	integer		SÍ	Kilometraje del vehículo al momento de la mantención.
tipo_mantencion	varchar	100	SÍ	Tipo de mantención (e.g., 10.000km, 20.000km, Cambio de aceite).
servicios_realizados	text		SÍ	Descripción de los servicios y tareas efectuadas.
observaciones	text		SÍ	Anotaciones adicionales sobre la mantención.
costo	decimal	18,2	SÍ	Costo asociado a esta mantención específica.
estado	varchar	20	SÍ	Estado del servicio de mantención (e.g., Realizado, Pendiente).

3.2.3.2. Diccionario de la Tabla: Orden de Trabajo

Nombre	orden_trabajo			
Descripción	Registra cada solicitud de servicio o trabajo a realizar sobre un vehículo.			
Clave primaria	id_orden			
Clave foránea	id_tecnico (tecnico), vehiculo_patente (vehiculo)			
Nombre	Tipo	Largo	Nulo	Descripción
id_orden	integer		NO	Identificador único para la orden de trabajo.
id_tecnico	integer		NO	Clave foránea que referencia al técnico asignado.
vehiculo_patente	varchar	20	NO	Clave foránea que referencia a la patente del vehículo.
fecha_creacion	timestamp		SÍ	Fecha y hora en que se crea la orden.
fecha_inicio	timestamp		SÍ	Fecha y hora en que comienza el trabajo.

fecha_finalizacion	timestamp		SÍ	Fecha y hora en que finaliza el trabajo.
tipo_trabajo	varchar	50	SÍ	Categoría del trabajo (e.g., Mantención, Reparación).
descripcion_trabajo	text		SÍ	Descripción detallada del problema o solicitud del cliente.
observaciones	text		SÍ	Anotaciones adicionales del técnico o del cliente.
costo_estimado	decimal	18,2	SÍ	Presupuesto inicial del costo del trabajo.
costo_final	decimal	18,2	SÍ	Costo total y final del servicio realizado.
estado	varchar	20	SÍ	Estado actual de la orden (e.g., Creada, En Proceso, Finalizada).

3.2.3.3. Diccionario de la Tabla: Propietario

Nombre	propietario			
Descripción	Guarda la información de los dueños de los vehículos.			
Clave primaria	id_propietario			
Clave foránea	N/A			
Nombre	Tipo	Largo	Nulo	Descripción
id_propietario	integer		NO	Identificador único para cada propietario.
nombre	varchar	100	NO	Nombres del propietario.
apellido	varchar	100	SÍ	Apellidos del propietario.
rut	varchar	20	NO	Rol Único Tributario del propietario.
direccion	varchar	200	SÍ	Dirección de residencia del propietario.
telefono	varchar	20	SÍ	Número de teléfono de contacto del propietario.
email	varchar	100	SÍ	Dirección de correo electrónico del propietario.
estado_cuenta	varchar	20	SÍ	Estado actual de la cuenta del propietario (e.g., Activa, Morosa).

3.2.3.4. Diccionario de la Tabla: Registro Blockchain

Nombre	registro_blockchain
Descripción	Almacena un registro inmutable de las transacciones o eventos importantes de las órdenes de trabajo.
Clave primaria	id_registro

Clave foránea	id_orden (orden_trabajo)			
Nombre	Tipo	Largo	Nulo	Descripción
id_registro	integer		NO	Identificador único del registro en la base de datos.
id_orden	integer		NO	Clave foránea que referencia la orden de trabajo asociada.
hash_transaccion	varchar	255	NO	Hash único que identifica la transacción en la blockchain.
fecha_hora	timestamp		SÍ	Marca de tiempo de cuándo se creó el registro.
datos_encriptados	text		SÍ	Información relevante de la transacción, en formato encriptado.
firma	text		SÍ	Firma digital que asegura la integridad y autoría del registro.

3.2.3.5. Diccionario de la Tabla: Reparación

Nombre	reparacion			
Descripción	Detalla los servicios de reparación correctiva realizados en una orden de trabajo.			
Clave primaria	id_reparacion			
Clave foránea	id_orden (orden_trabajo)			
Nombre	Tipo	Largo	Nulo	Descripción
id_reparacion	integer		NO	Identificador único para el registro de reparación.
id_orden	integer		NO	Clave foránea que vincula la reparación a una orden de trabajo.
fecha	timestamp		SÍ	Fecha en que se realizó la reparación.
diagnostico	text		SÍ	Descripción de la falla o problema diagnosticado por el técnico.
solucion_detectada	text		SÍ	Descripción de la solución propuesta o identificada.
solucion_aplicada	text		SÍ	Detalle de la reparación o solución que se implementó.
piezas_reemplazadas	text		SÍ	Listado de las piezas o repuestos que fueron cambiados.
costo	decimal	18,2	SÍ	Costo asociado a esta reparación específica.
estado	varchar	20	SÍ	Estado del servicio de reparación (e.g., Realizado, Pendiente).

3.2.3.6. Diccionario de la Tabla: Taller

Nombre	taller			
Descripción	Almacena la información de los talleres mecánicos registrados en el sistema.			
Clave primaria	id_taller			
Clave foránea	N/A			
Nombre	Tipo	Largo	Nulo	Descripción
id_taller	integer		NO	Identificador único para cada taller.
nombre	varchar	100	NO	Nombre comercial del taller.
direccion	varchar	200	SÍ	Dirección física del taller.
telefono	varchar	20	SÍ	Número de teléfono de contacto del taller.
email	varchar	100	SÍ	Dirección de correo electrónico del taller.

3.2.3.7. Diccionario de la Tabla: Técnico

Nombre	tecnico			
Descripción	Contiene los datos de los técnicos o mecánicos que trabajan en los talleres.			
Clave primaria	id_tecnico			
Clave foránea	id_taller (taller)			
Nombre	Tipo	Largo	Nulo	Descripción
id_tecnico	integer		NO	Identificador único para cada técnico.
id_taller	integer		NO	Clave foránea que referencia al taller donde trabaja el técnico.
nombre	varchar	100	NO	Nombres del técnico.
apellido	varchar	100	SÍ	Apellidos del técnico.
nif	varchar	20	NO	Número de Identificación Fiscal (o RUT) del técnico.
especialidad	varchar	100	SÍ	Área de especialización del técnico (e.g., motor, frenos).
usuario	varchar	50	NO	Nombre de usuario para acceder al sistema.
contrasena	varchar	255	NO	Contraseña encriptada para el acceso al sistema.

3.2.3.8. Diccionario de la Tabla: Vehículo

Nombre	vehiculo			
Descripción	Almacena los datos específicos de cada vehículo registrado.			
Clave primaria	patente			
Clave foránea	id_propietario (propietario)			
Nombre	Tipo	Largo	Nulo	Descripción
patente	varchar	20	NO	Matrícula o patente única del vehículo.
id_propietario	integer		NO	Clave foránea que referencia al dueño del vehículo.
marca	varchar	100	NO	Marca del vehículo (e.g., Toyota, Ford).
modelo	varchar	100	NO	Modelo del vehículo (e.g., Yaris, F-150).
ano	integer		SÍ	Año de fabricación del vehículo.
tipo	varchar	50	SÍ	Tipo de vehículo (e.g., Sedán, SUV, Camioneta).
nombre_propietario	varchar	200	SÍ	Nombre completo del propietario (puede ser redundante).
rut_propietario	varchar	20	SÍ	RUT del propietario (puede ser redundante).

A continuación, se describe el diccionario de datos, en las cuales se señalará la información de las tablas del sistema.

3.3. Diagrama de secuencia extendido y de colaboración.

3.3.1. Diagrama de secuencia extendido – Iniciar Sesión

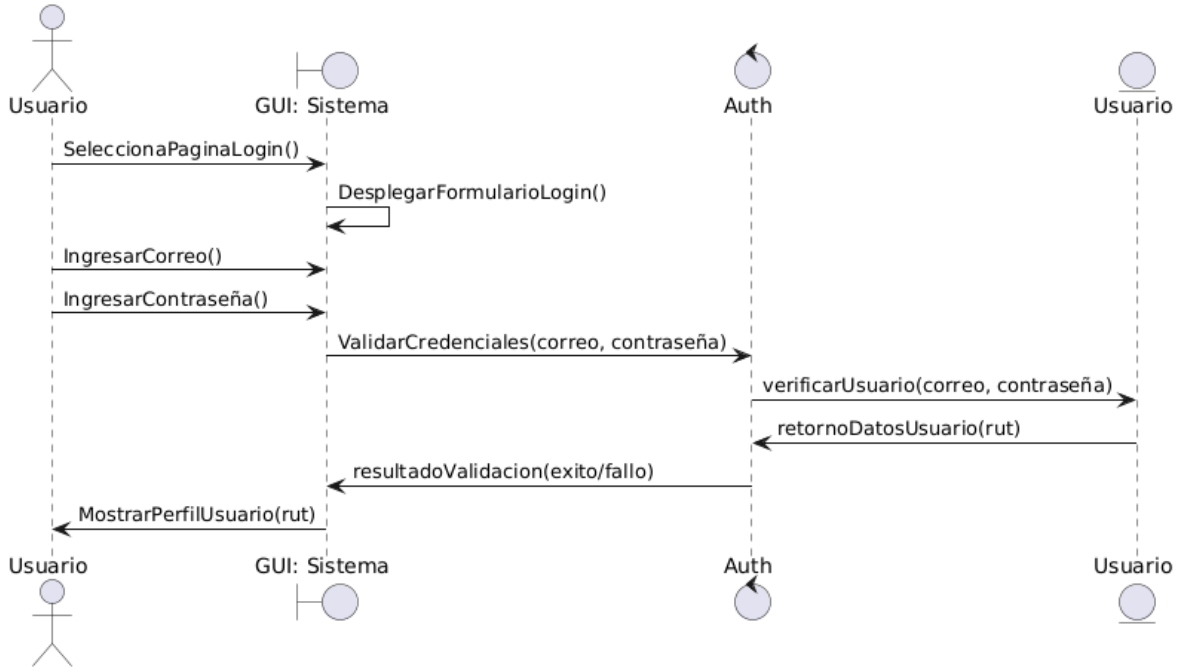


Figura 3.4
Fuente: Elaboración propia.

3.3.2. Diagrama de colaboración– Iniciar Sesión

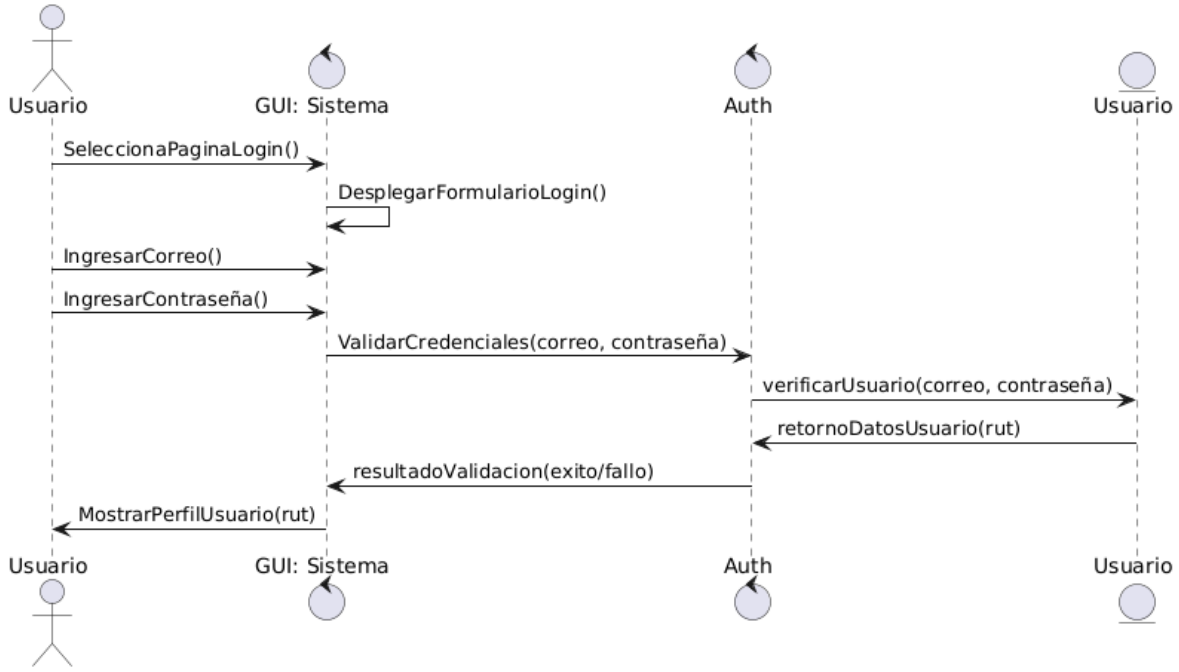


Figura 3.5

Fuente: Elaboración propia.

3.3.3. Diagrama de Secuencia Extendido – Registrar Vehículo

Figura 3.6

Fuente: Elaboración propia.

3.3.4. Diagrama de Colaboración – Registrar Vehículo

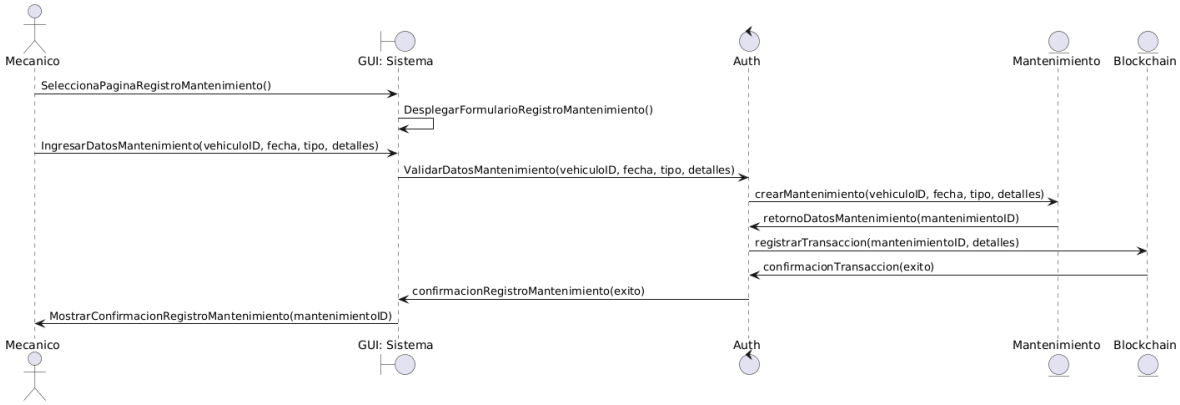


Figura 3.7

Fuente: Elaboración propia.

3.3.5. Diagrama Secuencia Extendido– Registrar Mantenimiento

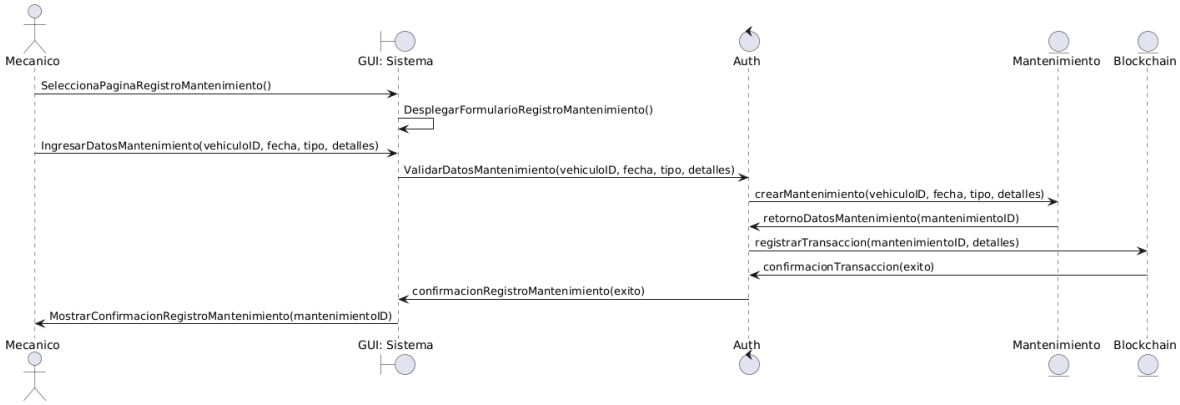


Figura 3.8

Fuente: Elaboración propia.

3.3.6. Diagrama de Colaboración – Registrar Mantenimiento

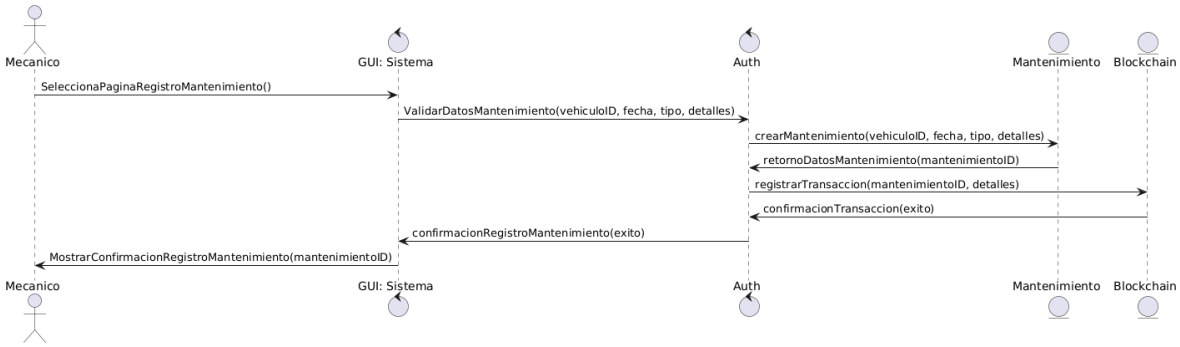


Figura 3.9

Fuente: Elaboración propia.

3.3.7. Diagrama de Secuencia Extendido – Registrar Reparación

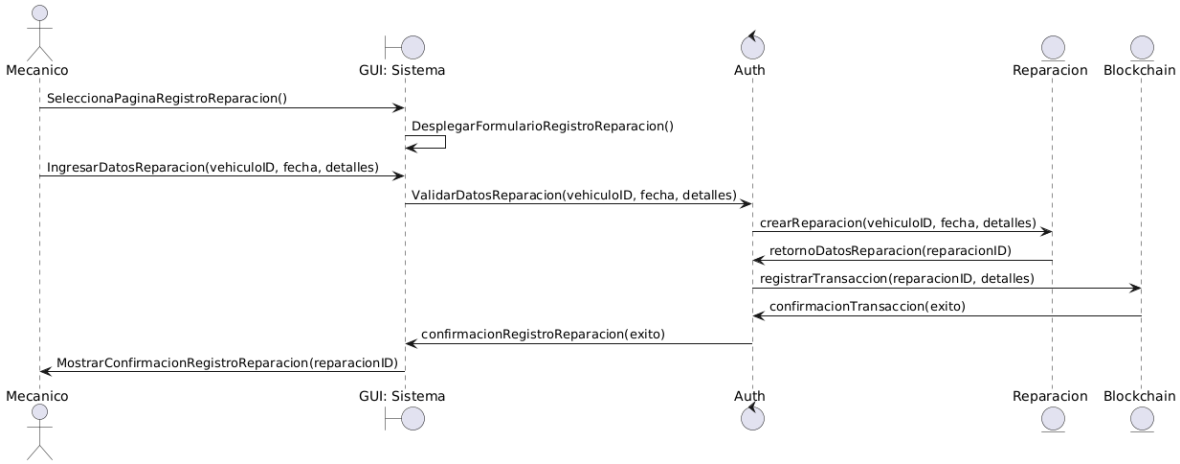


Figura 3.10

Fuente: Elaboración propia.

3.3.8. Diagrama de Colaboración – Registrar Reparación

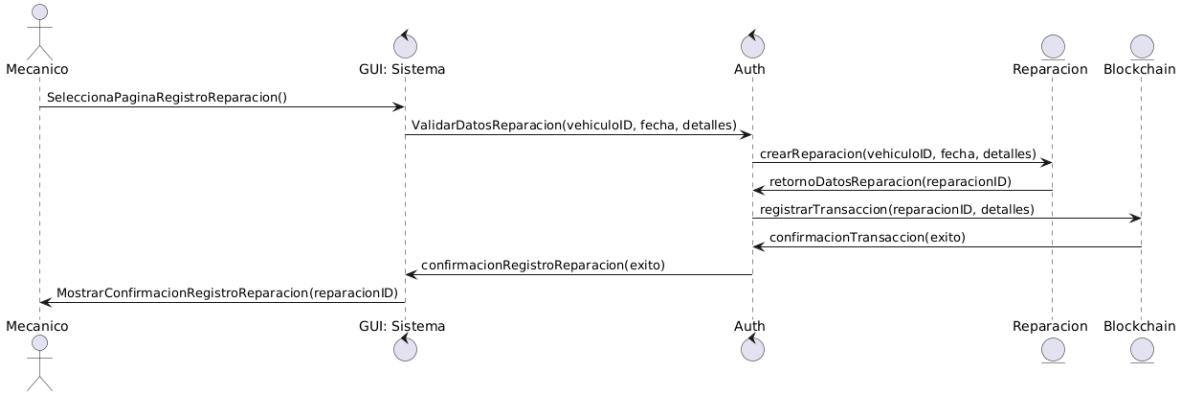


Figura 3.11

Fuente: Elaboración propia.

3.3.9. Diagrama de Secuencia Extendido – Consultar Historial

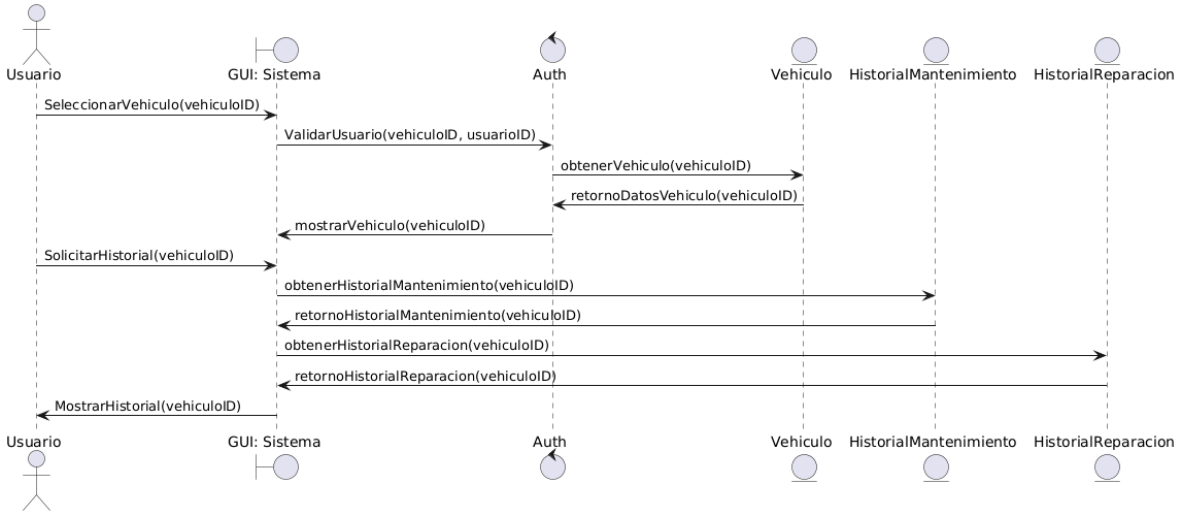


Figura 3.12

Fuente: Elaboración propia.

3.3.10. Diagrama de Colaboración – Consultar Historial

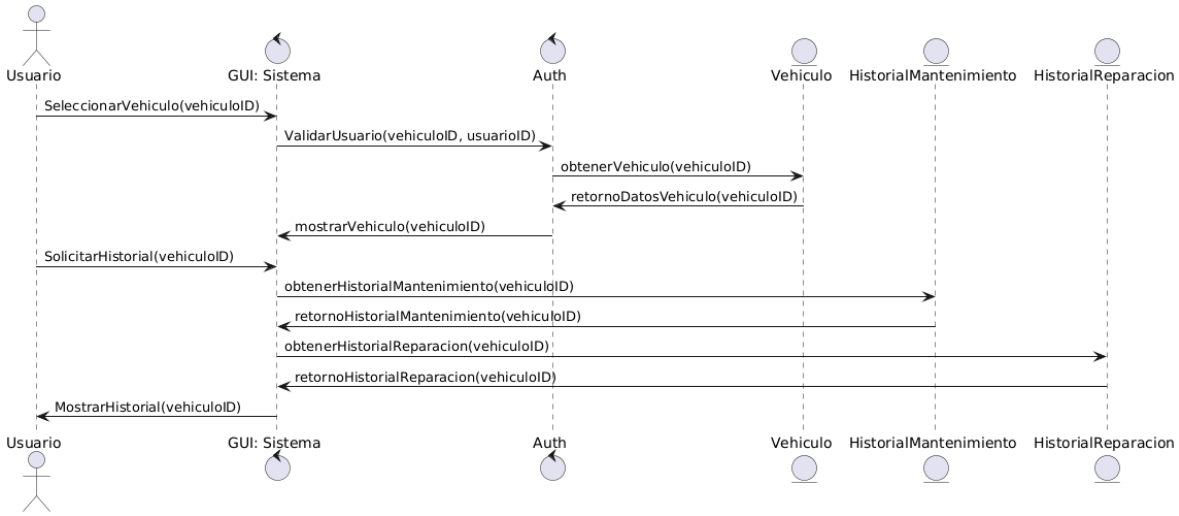


Figura 3.13

Fuente: Elaboración propia.

3.4. Diseño de interfaz.

El diseño de las interfaces de usuario de Carchain se ha desarrollado siguiendo los principios fundamentales de la experiencia de usuario (UX) y las heurísticas de usabilidad establecidas por Nielsen y Schneiderman. Estos lineamientos aseguran una experiencia intuitiva, eficiente y satisfactoria para los usuarios del sistema.

3.4.1. Interfaz – Inicio de Sesión.

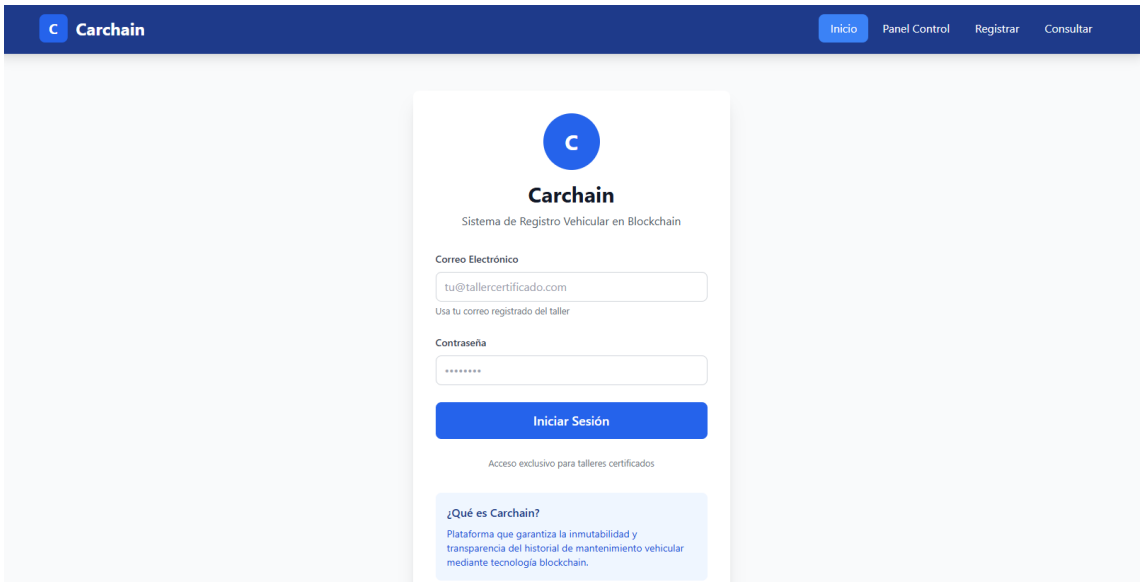


Ilustración 3-2
Fuente: Elaboración Propia

La interfaz de inicio de sesión constituye el punto de entrada al sistema Carchain y ha sido diseñada aplicando los principios de Simplicidad y Claridad Visual. Esta vista permite a los usuarios autenticarse mediante sus credenciales (correo electrónico y contraseña) y acceder al sistema de registro vehicular.

Funcionalidades principales:

- Autenticación segura para talleres certificados
- Información contextual sobre Carchain
- Acceso directo al panel principal tras la validación

El diseño implementa la Visibilidad del Estado del Sistema (N.º 1) mediante mensajes informativos que explican el propósito de la plataforma. Se aplica Diseño Estético y Minimalista (N.º 8) concentrando la atención en los elementos esenciales: campos de autenticación y botón de acceso. La Prevención de Errores (N.º 5) se materializa a través de la validación de formato de email y campos obligatorios claramente marcados. Finalmente, se asegura la Correspondencia entre el Sistema y el Mundo Real (N.º 2) utilizando terminología familiar como "Taller Certificado" y explicaciones comprensibles sobre blockchain.

3.4.2. Interfaz – Panel de Control Principal

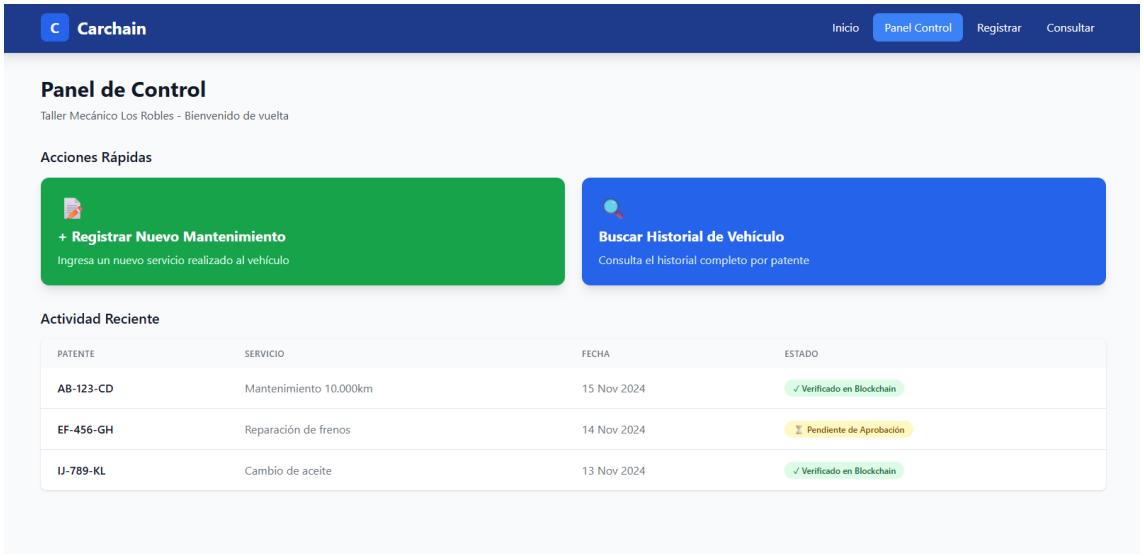


Ilustración 3-3
Fuente: Elaboración Propia

Esta interfaz constituye el centro neurálgico del sistema, diseñada para maximizar la eficiencia operativa del taller mecánico. Presenta un dashboard que permite acceso

inmediato a las funcionalidades más utilizadas y proporciona una visión general del estado de las operaciones.

Funcionalidades principales:

- Acceso rápido a registro de nuevos mantenimientos
- Búsqueda directa de historial vehicular
- Monitoreo de actividad reciente con estados de verificación
- Navegación intuitiva hacia todas las secciones del sistema

La implementación de "Acciones Rápidas" responde directamente a la Flexibilidad y Eficiencia de Uso (N.º 7), permitiendo a usuarios expertos completar tareas frecuentes con mínimos clics. La sección "Actividad Reciente" garantiza Visibilidad del Estado del Sistema (N.º 1) mostrando el progreso de los registros ("Verificado en Blockchain" vs "Pendiente de Aprobación"). El Reconocimiento antes que Recordación (N.º 6) se aplica mediante iconografía intuitiva y etiquetas descriptivas que hacen visible la funcionalidad sin requerir memorización. La Consistencia y Estándares (N.º 4) se mantiene a través de la navegación persistente y el esquema de colores uniforme.

3.4.3. Interfaz – Registro de Mantenimiento Detallado

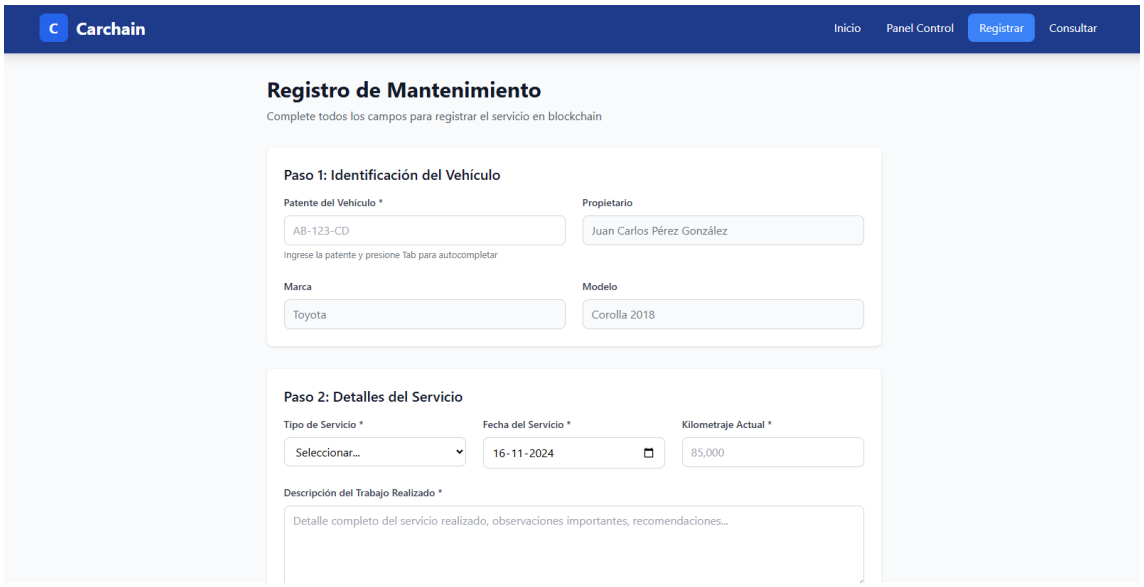


Ilustración 3-4
Fuente: Elaboración Propia

Esta interfaz representa el núcleo funcional del sistema, donde se capturan los datos que posteriormente se registrarán de manera inmutable en blockchain. El diseño de formulario progresivo minimiza la carga cognitiva y guía al usuario a través de un proceso estructurado de tres pasos lógicos.

Funcionalidades principales:

- Identificación automática de vehículos mediante patente
- Registro detallado de servicios con múltiples tipos de datos
- Carga de evidencia fotográfica
- Gestión de piezas reemplazadas con interfaz dinámica
- Opciones de guardado temporal y envío final

La Prevención de Errores (N.º 5) se implementa mediante autocompletado de datos vehiculares, eliminando inconsistencias en la identificación. El diseño de "pasos progresivos" aplica Control y Libertad del Usuario (N.º 3) permitiendo guardar borradores sin perder información. La Correspondencia entre el Sistema y el Mundo Real (N.º 2) se evidencia en la terminología específica del sector automotriz y el flujo de trabajo que replica el proceso natural del mecánico. El Diseño Estético y Minimalista (N.º 8) se logra agrupando información relacionada en secciones claras, evitando la sobrecarga visual que podría generar un formulario extenso presentado como bloque único.

3.4.4. Interfaz – Registro de Mantenimiento Detallado

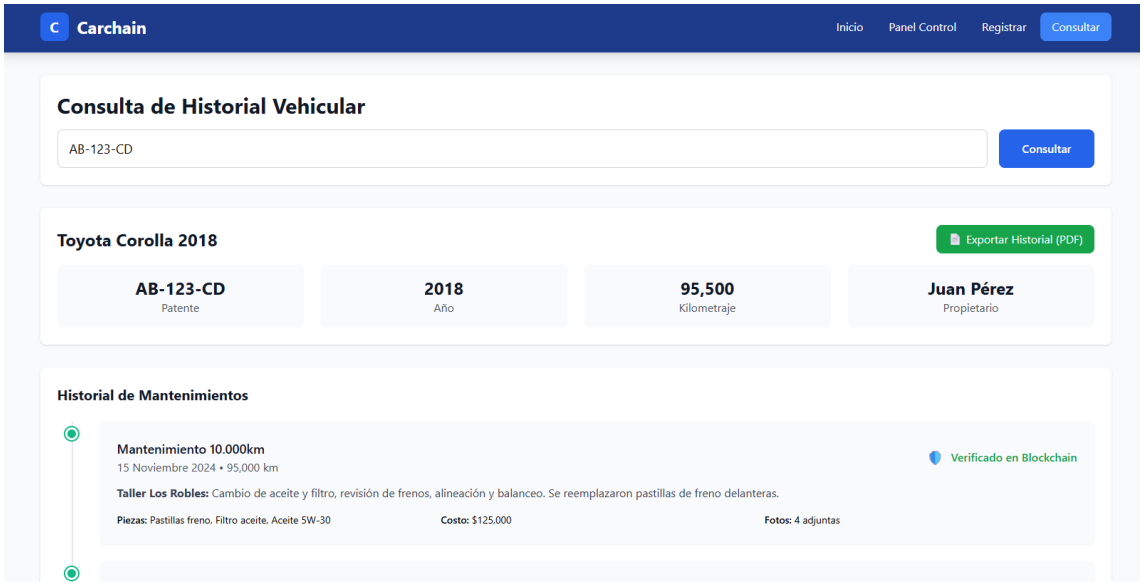


Ilustración 3-5
Fuente: Elaboración Propia

Esta vista materializa la propuesta de valor principal de Carchain: proporcionar transparencia y confianza en el historial vehicular. El diseño de línea de tiempo cronológica transforma datos complejos en una narrativa visual comprensible que comunica efectivamente la "biografía" del vehículo.

Funcionalidades principales:

- Búsqueda de vehículos por patente con resultados inmediatos
- Visualización cronológica del historial de mantenimientos
- Indicadores visuales de verificación blockchain
- Acceso detallado a cada registro con información expandible
- Exportación del historial completo en formato PDF

La Visibilidad del Estado del Sistema (N.º 1) se maximiza a través de los indicadores "Verificado en Blockchain" que comunican constantemente la integridad de los datos. El

formato de línea de tiempo implementa Reconocimiento antes que Recordación (N.º 6), utilizando una metáfora visual universalmente comprensible que no requiere aprendizaje específico. La Flexibilidad y Eficiencia de Uso (N.º 7) se proporciona mediante la función de exportación PDF para usuarios que necesiten documentación física. Finalmente, el Diseño Estético y Minimalista (N.º 8) se aplica ocultando detalles complejos hasta que sean solicitados específicamente, manteniendo la vista general limpia mientras preserva el acceso a información detallada cuando sea necesaria.

CONCLUSIÓN

Durante el desarrollo del capítulo 1 se ha podido comprender el contexto y las necesidades del sistema de registro descentralizado de mantenimiento y reparación automotriz “CarChain”. A través de un análisis exhaustivo de la situación actual sin el proyecto, se han identificado los problemas clave que afectan la gestión y trazabilidad de los registros de mantenimiento y reparaciones en el sector automotriz. Por otra parte, se han establecido los objetivos del proyecto, tanto el objetivo principal como los específicos, orientados a mejorar la eficiencia y la transparencia en la administración de los registros vehiculares mediante la tecnología blockchain. Además, se detallan los requerimientos y requisitos de los usuarios, incluyendo propietarios de vehículos, mecánicos y administradores del sistema, para asegurar una implementación efectiva y práctica del sistema.

Asimismo, se han evaluado diferentes alternativas de solución, considerando las ventajas y desventajas de cada opción en comparación con la creación de un proyecto a medida. A partir de esta evaluación, se ha elegido la aplicación “CarChain” como la mejor solución, destacando por su capacidad de mejorar significativamente la integridad y accesibilidad de los registros de mantenimiento y reparación. Gracias al capítulo 1, se construye una base sólida para los siguientes capítulos del informe, donde se explicarán detalles más técnicos y específicos hacia el desarrollo y su posterior implementación.

Durante el desarrollo del capítulo 2, se ha comenzado definiendo la solución propuesta y su diagrama de actividad. Además, se ha creado un diagrama de casos de uso, un diagrama conceptual y un conjunto de requerimientos tanto funcionales como no funcionales. A partir de esta información, se han desarrollado los casos de uso narrativos para cada uno de los casos de uso, añadiendo los diagramas de secuencias y sus contratos. Estos elementos permiten visualizar y planificar de manera detallada cómo interactuarán los diferentes componentes del sistema y cómo se cumplirán los requisitos del cliente.

Gracias al capítulo 2, se establece que el sistema a desarrollar cumplirá todos los requisitos del cliente y ya se tiene una visión general del sistema. Además, se establecen los datos que deberá cumplir cada perfil y las opciones del sistema, garantizando que todas las funcionalidades necesarias serán implementadas de acuerdo a las especificaciones establecidas. Con esta base, el proyecto “CarChain” está preparado para avanzar hacia las fases de desarrollo e implementación, con la confianza de que

proporcionará una solución eficiente y confiable para la gestión de registros de mantenimiento y reparación automotriz.

BIBLIOGRAFIA

Cook, B. (2018). Blockchain: transforming the seafood supply chain. Link. Accessed 26 Oct 2019.

Lamberti, F., et al. (2018). Blockchains can work for car insurance: using smart contracts and sensors to provide on-demand coverage. IEEE Consumer Electronics Magazine, 7(4), 72-81. Link.

Cebe, M., et al. (2018). Block4forensic: an integrated lightweight blockchain framework for forensics applications of connected vehicles. IEEE Communications Magazine, 56(10), 50-57. Link.

Li, C. (2018). Blockchain Technology for Smart Vehicle: The Mileage and Condition Recording. SpringerLink. Link.

Six Applications for Blockchain in Automotive. (n.d.). AutoFacets Insights. Retrieved from https://www.autofacets.com.

Blockchain in automotive: benefits and use cases. (n.d.). LimeChain. Retrieved from https://limechain.tech.