I saw your document

UNIVERSIDAD TECNOLÓGICA DE CHIHUAHUA

Auto-matic

INFORME DE PROYECTO INTEGRADOR

TÉCNICO SUPERIOR UNIVERSITARIO

TECNOLOGÍAS DE LA INFORMACIÓN ÁREA: DESARROLLO DE SOFTWARE MULTIPLATAFORMA

PRESENTA:

Abraham Camacho Rios

Jesús Alejandro Borjon Zapata

Jesús Jordán Ramos López

Chihuahua, Chih., *08 de agosto de 2024*

Resumen

El proyecto aborda el problema de la falta de herramientas accesibles para el monitoreo en tiempo real del estado de los automóviles, lo que lleva a fallos inesperados y costosos. La metodología utilizada incluye la integración de una Raspberry Pi para capturar datos críticos del vehículo, como la temperatura del refrigerante y la presión del aceite, que luego se transmiten y almacenan en Firebase. La información se presenta a los usuarios a través de una aplicación web y móvil, facilitando la toma de decisiones informadas sobre el mantenimiento. Los resultados más importantes muestran que el sistema mejora la seguridad y eficiencia del vehículo, permitiendo a los conductores anticiparse a las necesidades de mantenimiento. Los alcances del proyecto incluyen la implementación exitosa del sistema en diferentes plataformas, mejorando la accesibilidad de los datos críticos para los usuarios. Las limitaciones se centran en la dependencia de una conexión estable a internet para la transmisión de datos y la precisión de los sensores de la Raspberry Pi. Las principales conclusiones destacan la efectividad del sistema para prevenir fallos vehiculares y optimizar los costos de mantenimiento, demostrando la viabilidad de utilizar tecnologías como la Raspberry Pi y Firebase en soluciones de monitoreo automotriz.

ÍNDICE

Índice de tablas

Índice de figuras

Introducción

El Técnico Superior Universitario en Desarrollo de Software Multiplataforma posee las competencias profesionales necesarias para desempeñarse eficazmente en el campo laboral, abarcando los ámbitos local, regional, nacional e internacional. Se caracteriza por la capacidad de desarrollar soluciones tecnológicas para entornos web, empleando fundamentos de programación orientada a objetos, bases de datos y redes de área local. Además, implementa soluciones multiplataforma, en la nube y software embebido en entornos seguros, utilizando la adquisición y administración de datos e ingeniería de software para contribuir a la automatización de procesos en las organizaciones.

El proyecto se centra en la creación de dos aplicaciones, una web y otra móvil, las cuales interactúan con una Raspberry Pi para monitorear y mantener vehículos. Se propone una solución que proporciona información a los usuarios en tiempo real sobre el estado de su vehículo, permitiendo anticiparse a posibles fallos y necesidades de mantenimiento, mejorando la seguridad y eficiencia de utilizar un auto, prolongando su vida útil.

El producto desarrollado utiliza la Raspberry Pi para capturar datos relevantes del automóvil, tales como la temperatura del refrigerante, la presión del aceite, la carga de la batería, la condición del aceite y los kilómetros que faltan para el próximo mantenimiento. Estos datos se transmiten de manera segura y eficiente a una base de datos en la nube, utilizando Firebase, que ofrece una plataforma robusta para la gestión y el análisis de datos.

La aplicación web presenta esta información de manera detallada y comprensible para todos los usuarios, incluyendo indicadores visuales y alertas que facilitan la interpretación de los datos, permitiendo a los propietarios de los vehículos tomar decisiones informadas sobre el mantenimiento

de sus vehículos. La aplicación móvil ofrece una versión simplificada de estos datos, asegurando que la información crítica esté siempre al alcance del usuario, independientemente de su ubicación.

La importancia del producto radica en su capacidad para resolver una problemática actual en el sector automotriz, ya que los propietarios de vehículos a menudo carecen de herramientas accesibles que les permitan monitorear el estado de sus autos en tiempo real. Auto-matic proporciona un medio para estar informados y poder tomar acciones preventivas, lo que no solo mejora la seguridad vial, sino que también optimiza los costos de mantenimiento y reparación al prevenir daños mayores.

Este proyecto refleja la aplicación de los conocimientos y habilidades adquiridos durante el TSU en desarrollo de software multiplataforma. La implementación de la Raspberry Pi y Firebase demuestra la capacidad para integrar diversas tecnologías y crear productos que tienen un impacto en el mundo.

En este documento se definen los siguientes puntos del proyecto:

Capítulo 1: aspectos generales de la empresa: se ofrece una visión general de la empresa Auto-matic, detallando su descripción y la planeación normativa que sigue para asegurar la calidad y eficiencia de sus servicios y productos tecnológicos.

Capítulo 2: definición del proyecto: se aborda el contexto del proyecto, incluyendo antecedentes, análisis de la situación actual y planteamiento del problema. Se define claramente el problema específico a resolver, los objetivos del proyecto, su justificación, el método de trabajo y el cronograma de actividades.

Capítulo 3: desarrollo: se expone el marco teórico que sustenta el proyecto, así como la descripción de las actividades realizadas durante su desarrollo. Se detallan las tareas específicas y los procesos seguidos en cada etapa del proyecto.

Capítulo 4: resultados y análisis: se presentan y analizan los resultados obtenidos, tanto cuantitativos como cualitativos. Además, se incluyen las conclusiones derivadas del proyecto y se ofrecen recomendaciones basadas en los resultados obtenidos, sugiriendo posibles mejoras y futuras líneas de investigación o desarrollo.

Capítulo 1. Aspectos Generales de la Empresa

1.1. Descripción de la Empresa

En Chihuahua, Chihuahua, México, se encuentra Auto-Matic, una empresa enfocada en el desarrollo de aplicaciones web. Auto-Matic ofrece una variedad de servicios tecnológicos que incluyen DevOps, asegurando una infraestructura sólida y continua para las aplicaciones; Back-end, gestionando bases de datos y garantizando la seguridad y escalabilidad para un funcionamiento sin problemas; y Front-end, creando experiencias intuitivas y atractivas para los usuarios, enfocándose en la usabilidad, la accesibilidad y la estética.

A pesar de ser un equipo compacto, los cuatro profesionales altamente cualificados de Auto-Matic desarrollan el producto estrella, Auto-Matic, en el Departamento de Tecnología. Trabajan para superar las expectativas de los clientes y mantenerse a la vanguardia en innovación.

Auto-Matic cree en el mantenimiento preventivo para prolongar la vida útil y optimizar el rendimiento de los vehículos. La aplicación no solo funciona como un dispositivo de mantenimiento, sino que también actúa como un compañero en el viaje de cada conductor, promoviendo viajes seguros y eficientes.

Además, Auto-Matic conecta a los conductores con talleres de confianza, facilitando citas y seguimientos. La dedicación a la calidad y la innovación gana a la empresa un reconocimiento significativo en la comunidad automotriz.

1.2. Planeación normativa

Misión

Crear una aplicación que proporcione a los automovilistas de todo el mundo información en tiempo real sobre el estado de sus vehículos, permitiéndoles tomar decisiones informadas para prolongar la vida útil de sus automóviles.

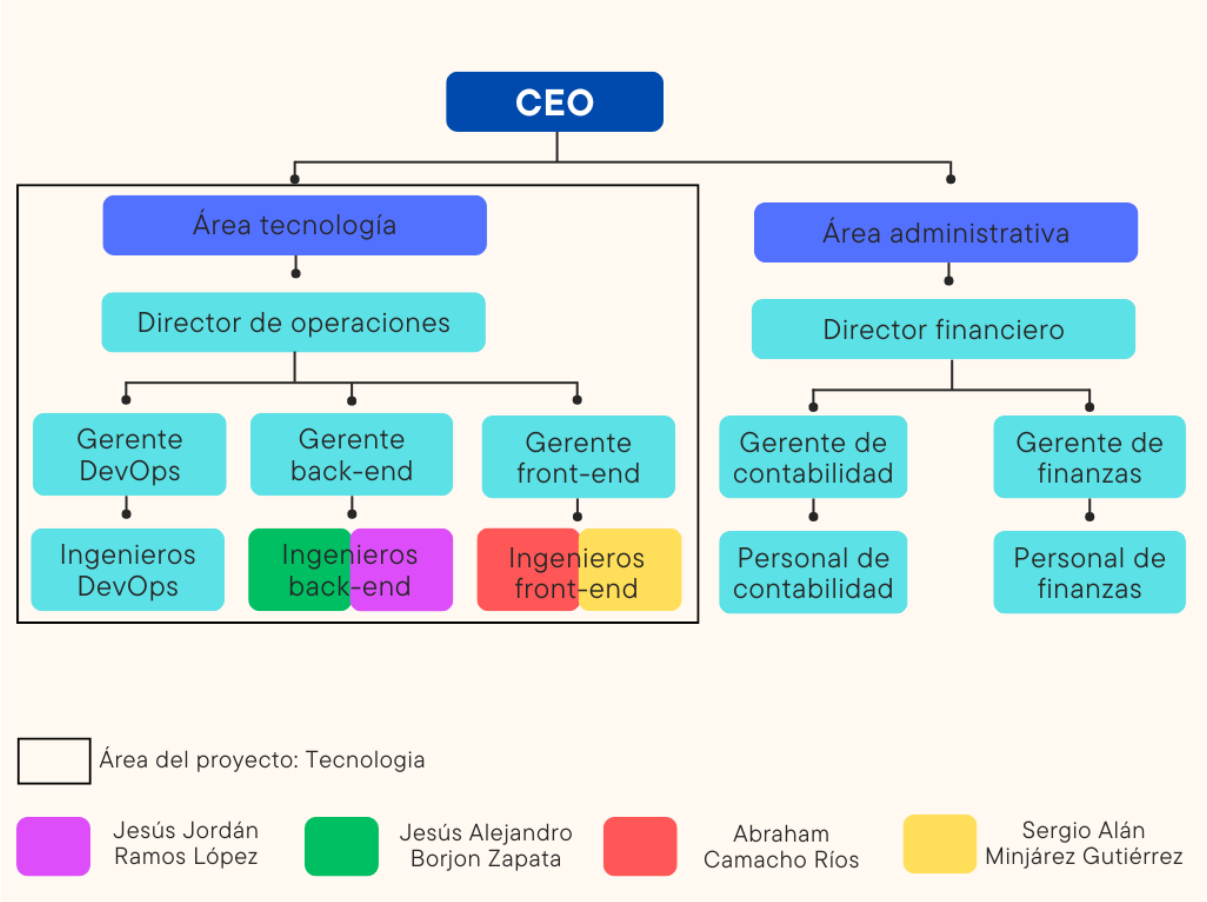
Visión

Ser la primera plataforma en México que integra a mecánicos, refaccionarias y usuarios en un único ecosistema, agilizando todos los procesos relacionados con el mantenimiento y reparación de vehículos.

Organigrama

Figura 1

Organigrama de Auto-matic



Fuente: elaboración propia.

Capítulo 2. Definición del Proyecto

2.1. Antecedentes

En el contexto actual, los propietarios de vehículos enfrentan desafíos significativos relacionados con el mantenimiento y la gestión eficiente de sus automóviles. Tradicionalmente, el monitoreo de las condiciones del vehículo se realiza de manera manual, dependiendo del conocimiento y la experiencia del conductor, así como de visitas periódicas a talleres mecánicos. Este método no solo es ineficiente, sino que también resulta en mantenimiento reactivo en lugar de preventivo, incrementando los riesgos de fallos inesperados y reparaciones costosas.

La falta de sistemas de monitoreo en tiempo real significa que muchos problemas potenciales no se detectan hasta que es demasiado tarde. Por ejemplo, problemas como la baja presión de aceite, la alta temperatura del refrigerante o una batería descargada pasan desapercibidos hasta que causan daños graves al motor. Además, la ausencia de alertas oportunas sobre el mantenimiento regular lleva a la negligencia en el servicio, disminuyendo la vida útil del vehículo y comprometiendo la seguridad del conductor y los pasajeros.

Actualmente, los procesos involucrados en la detección y el manejo de estos problemas dependen en gran medida de revisiones manuales y diagnósticos realizados en talleres mecánicos. Los conductores deben recordar las fechas de servicio, llevar un registro de los kilometrajes y estar atentos a las señales físicas o sonoras de advertencia en el tablero del vehículo. Este enfoque, además de ser propenso a errores humanos, no ofrece una solución integral que permita a los usuarios tener un control total y preciso del estado de su vehículo en todo momento.

2.2. Análisis de la Situación Actual

Lluvia de ideas

Figura 2

Lluvia de ideas a partir de la encuesta realizada por Auto-matic

Lluvia de ideas

Moderador	Jesús Borjon
Participantes	Jesús Ramos Abraham Camacho Sergio Minjárez
Fecha	21 de Junio de 2024

🎯 Resultados de la lluvia de ideas

*La mayoría de las personas son propietarias de un automóvil.
*La mitad de los propietarios de automóviles solo poseen un solo automóvil.
*La mayoría de las personas han realizado el servicio a su automóvil recientemente.
*Todos los encuestados no tienen más que un conocimiento moderado de las condiciones mecánicas de su automóvil.
*Solo 2 de cada 10 personas encuestadas tienen un buen conocimiento sobre el significado de los indicadores en el tablero de su automóvil.
*La mayoría de las personas no conoce los tiempos recomendados para realizar cambio de aceite, filtros de aire o bujías.
*La mayoría de las personas no conoce los procedimientos para revisar el refrigerante del motor o el estado de la batería.
*La simplificación es esencial
*El medir la frecuencia del mantenimiento de los componentes del auto
*El enseñar al usuario como administrar los mantenimientos
*Organizar los tiempos de cambios para el usuario
*Guiar interactiva mente al usuario en la reparación

Fuente: elaboración propia.

A partir de una recolección general de ideas y opiniones de los propietarios de automóviles, se observa cómo se llevan a cabo actualmente las actividades de mantenimiento vehicular. Se realiza una encuesta (anexo 5) para definir la problemática específica.

La lluvia de ideas revela que la mayoría de las personas poseen un automóvil y de estos propietarios, muchos solo tienen uno. Aunque la mayoría realiza el servicio a su automóvil recientemente, todos los encuestados admiten tener un conocimiento limitado sobre las condiciones mecánicas de sus vehículos. Solo dos de cada diez personas conocen bien el significado de los indicadores en el tablero de su automóvil. Además, la mayoría desconoce los tiempos recomendados para el cambio de aceite, filtros de aire o bujías y no sabe cómo revisar el refrigerante del motor o el estado de la batería.

La simplificación del proceso de mantenimiento se identifica como esencial, así como la necesidad de medir la frecuencia del mantenimiento, enseñar al usuario cómo administrarlo, organizar los tiempos de cambio y guiar interactivamente al usuario en las reparaciones.

El área afectada es el mantenimiento del vehículo y el personal involucrado son los propietarios de automóviles. Se identifica un conocimiento limitado de las condiciones mecánicas, ya que todos los encuestados no tienen más que un conocimiento moderado de las condiciones mecánicas de su automóvil. Esto impacta negativamente al aumentar la probabilidad de negligencia en el mantenimiento adecuado, lo que lleva a averías graves. Cuantitativamente, esto resulta en un incremento en los fallos mecánicos y costos de reparación, mientras que cualitativamente se traduce en la pérdida de un medio de transporte adecuado para realizar las actividades diarias.

La falta de conocimiento sobre los indicadores del tablero se evidencia ya que solo dos de cada diez personas encuestadas tienen un buen conocimiento sobre el significado de los indicadores en el tablero de su automóvil. Este desconocimiento lleva a la ignorancia de señales de advertencia importantes que pueden prevenir daños mayores, incrementando así el riesgo de fallos mecánicos y

accidentes. La afectación cualitativa incluye una sensación de inseguridad y falta de control sobre el estado del vehículo.

El desconocimiento de los tiempos de mantenimiento se observa porque la mayoría de las personas no conoce los tiempos recomendados para realizar cambios de aceite, filtros de aire o bujías. Esto impacta en la realización irregular o incorrecta del mantenimiento preventivo, aumentando el desgaste del motor y componentes y generando costos adicionales a largo plazo. La afectación cualitativa es la reducción en la vida útil del vehículo.

Finalmente, el desconocimiento de los procedimientos de revisión se identifica ya que la mayoría de las personas no conoce los procedimientos para revisar el refrigerante del motor o el estado de la batería. Esto lleva a situaciones de emergencia causadas por fallos repentinos, incrementando los costos de reparaciones de emergencia. Cualitativamente, esto genera estrés por los fallos repentinos y disrupción de la rutina diaria.

La falta de conocimiento y habilidades en el mantenimiento preventivo de los vehículos lleva a problemas cuantitativos como el aumento de fallos mecánicos y costos de reparación y a problemas cualitativos como la pérdida de un medio de transporte confiable y el estrés asociado a las emergencias vehiculares.

2.3. Planteamiento del Problema

2.3.1. *Definición del Problema*

La falta de herramientas claras y accesibles para interpretar los datos del vehículo provoca que los usuarios no puedan realizar el mantenimiento adecuado, lo que termina afectando el funcionamiento óptimo de sus automóviles y acarreando mayores gastos en la reparación.

2.3.2. *Objetivos*

Objetivo general

Desarrollar una aplicación web y móvil en Flutter con Firebase para el monitoreo preventivo de vehículos usando Raspberry Pi y OBDII, proporcionando datos cruciales y alertas de servicio en tiempo real. Este proyecto, realizado por Auto-Matic entre mayo y agosto de 2024, busca mejorar la seguridad y optimizar el mantenimiento vehicular.

2.3.3 Justificación

Los vehículos automotores son máquinas complejas, tanto que se necesitan especialidades completas para poder repararlos o montarlos. Las herramientas brindadas al usuario común son tanto básicas como confusas, lo que deja vulnerable al usuario ante la incertidumbre del desconocimiento. Por lo tanto, una forma más clara de transmitir la información es esencial, junto con el acceso constante a esta. Gracias a los conocimientos adquiridos por varios miembros del equipo de trabajo, se pueden proporcionar a los usuarios herramientas para que sean más conscientes de las necesidades de sus vehículos, con una interpretación acertada de sus datos en tiempo real y sugerencias sobre el mantenimiento para que comprendan la magnitud del desgaste de sus vehículos. El dispositivo es factible de vender a los usuarios, así como también enfocado a la venta al por mayor a grandes empresas o pequeños comercios que desean modernizar sus métodos de reparación. Gracias a esto, la aplicación puede llegar a ser trascendente en la manera en cómo los usuarios se relacionan con sus vehículos.

2.4. Método

2.4.1. *Recolección de información.*

Se recaban los datos pertinentes sobre la empresa donde se desarrolla el proyecto. Se verifica la existencia del problema que origina el proyecto mediante una encuesta dirigida a propietarios de

vehículos. A partir del análisis de los resultados de la encuesta, se define, justifica y planifica la ejecución del proyecto, empleando una lluvia de ideas para identificar posibles soluciones. Este proceso asegura una comprensión clara del problema y establece una base sólida para la implementación del proyecto.

2.4.2. *Desarrollo de dispositivo IoT para monitoreo del automóvil.*

Se desarrolla e implementa un dispositivo de IoT para monitorear en tiempo real la información proveniente de los sensores de temperatura, voltaje, kilometraje y revoluciones por minuto de un automóvil (anexo 1). Este sistema utiliza un ordenador de bajo costo, como una Raspberry Pi, que ejecuta la librería Python-OBD para comunicarse con el conector OBDII del vehículo. La recopilación de datos en tiempo real permite una supervisión continua y precisa del estado del automóvil.

2.4.3. *Conexión del dispositivo IoT a la base de datos en la nube (Firebase).*

Se codifica la conexión a la base de datos en la nube Firebase utilizando el lenguaje de programación Python y las funciones proporcionadas por la documentación de Firebase. El objetivo es transmitir los datos obtenidos de los sensores del automóvil para su almacenamiento seguro en la nube. Esto permite que las aplicaciones a desarrollar en los siguientes pasos accedan a los datos de forma remota de manera sencilla.

2.4.4. *Pruebas de funcionamiento del dispositivo IoT.*

Se verifica la transmisión correcta de los datos obtenidos por el dispositivo de IoT hacia Firebase en condiciones de conducción real. Este proceso implica monitorear en tiempo real la integridad y precisión de los datos mientras el vehículo está en movimiento. Se asegura que la conexión entre el dispositivo de IoT y la base de datos en la nube sea estable y confiable.

2.4.5. *Desarrollo de aplicación web con Flutter.*

Se desarrolla una aplicación web utilizando el framework de desarrollo multiplataforma Flutter para presentar de manera sencilla y atractiva los datos provenientes del dispositivo de IoT. Esta aplicación permite a los usuarios visualizar en tiempo real la información recopilada por los sensores del automóvil. Esto requiere el uso de la documentación oficial de Flutter para asegurar un desarrollo ágil y eficiente.

2.4.6. *Desarrollo de aplicación móvil con Flutter.*

Se desarrolla una aplicación móvil utilizando el framework de desarrollo multiplataforma Flutter para presentar de manera sencilla y atractiva los datos provenientes del dispositivo de IoT. Esta aplicación permite a los usuarios visualizar en tiempo real la información recopilada por los sensores del automóvil. Esto requiere el uso de la documentación oficial de Flutter para asegurar un desarrollo ágil y eficiente.

2.4.7. *Pruebas de funcionamiento de la interfaz (móvil/web).*

Se verifica el correcto acomodo de los elementos de la interfaz web y móvil, asegurando que todos los componentes estén alineados de manera coherente y funcional. Se garantiza la ausencia de errores de desbordamiento, revisando que el contenido se adapte adecuadamente a diferentes tamaños de pantalla y resoluciones sin comprometer la usabilidad o la estética. Además, se aprueba la paleta de colores utilizada, asegurando que sea visualmente atractiva y consistente con el diseño general de la aplicación.

2.4.8. *Conexión de la aplicación web/móvil con la base de datos en la nube (Firebase).*

Se desarrolla la conexión a la base de datos en la nube Firebase con el objetivo de obtener datos en tiempo real provenientes del dispositivo de IoT desarrollado previamente. Este proceso

requiere un acceso detallado a la documentación oficial de Flutter y Firebase para utilizar los mecanismos incorporados que facilitan una conexión sencilla y eficiente entre ambos servicios.

2.4.9. *Pruebas de conectividad y despliegue de datos (móvil/web).*

Se verifica la conexión correcta a la base de datos en la nube, asegurando que se establezca y mantenga sin interrupciones. Este proceso incluye la validación de las credenciales de acceso, la integridad de los datos y la eficiencia de las consultas en tiempo real a la base de datos. Además, se garantiza la correcta visualización de los datos en la interfaz web y móvil. Esto implica revisar que los datos obtenidos de la base de datos se muestren de manera precisa y en tiempo real en ambas plataformas.

2.4.10. *Documentación del proyecto.*

Se documentan los datos pertinentes para el proyecto, recolectados de manera oportuna de fuentes apropiadas y confiables. Todas las tecnologías utilizadas durante el desarrollo se exponen de forma clara y concisa, respaldadas por referencias confiables. Se registran detalladamente las actividades realizadas a lo largo del proyecto. El objetivo de esta documentación es comprobar el desarrollo de la solución planteada y evaluar su impacto en el problema abordado.

2.4.11. *Presentación del proyecto.*

Se comparte la solución desarrollada durante el transcurso del proyecto mediante una presentación multimedia dirigida tanto a compañeros como a profesores. Esta presentación incluye una descripción detallada del dispositivo de IoT funcional, la experiencia del desarrollo y los resultados obtenidos. Para una demostración efectiva, se requiere el uso de un vehículo para mostrar el dispositivo en funcionamiento y la plena operatividad de la aplicación web y móvil.

2.5. Cronograma

Tabla 1

Cronograma de actividades del proyecto

Auto-matic en la empresa. mayo - agosto del 2024.														
N°	Actividades		1	2	3	4	5	6	7	8	9	10	11	12	13
1	Recolección de información.	P													
		R													
2	Diseño y desarrollo de dispositivo IoT para monitoreo del automóvil.	P													
		R													
3	Conexión del dispositivo IoT a la base de datos en la nube (Firebase).	P													
		R													
4	Pruebas de funcionamiento del dispositivo IoT.	P													
		R													
5	Diseño y desarrollo de aplicación web con Flutter.	P													
		R													
6	Diseño y desarrollo de aplicación móvil con Flutter.	P													
		R													
7	Pruebas de funcionamiento de la interfaz (móvil/web).	P													
		R													
8	Conexión de la aplicación web/móvil con la base de datos en la nube (Firebase).	P													
		R													
9	Pruebas de conectividad y despliegue de datos (móvil/web).	P													
		R													
10	Documentación del proyecto.	P													
		R													
11	Presentación del proyecto.	P													
		R													

Fuente: elaboración propia.

Capítulo 3. Desarrollo

3.1. Marco Teórico

En el presente marco teórico se busca explicar las tecnologías usadas en el desarrollo de Automatic, como el OBD, que se utiliza para medir y monitorear los sistemas de un vehículo. También se menciona el Raspberry Pi, un microordenador utilizado en aplicaciones de automatización. Además, se describe el uso de Python para trabajar con sensores y sistemas de comunicación. Por último, se incluye la base de datos utilizada, Firebase y el framework Flutter, que se utilizan para el desarrollo de las aplicaciones móvil y web.

3.1.1. *OBD*

Según McCord (2011), los diagnósticos a bordo, u On-Board Diagnostics (OBD), son una serie de mediciones y comprobaciones incorporadas en el motor y los sistemas de un vehículo. Realizar mediciones sobre el estado de los componentes del automóvil permite garantizar que el sistema funcione como está previsto. Si algo no funciona como se espera, es posible enviar una alerta o tomar medidas para recuperar la estabilidad del sistema.

Por otra parte, el OBD se define como una herramienta estándar que permite al conductor determinar las condiciones del motor del vehículo. Para utilizarla, es necesario conectarse al puerto OBD, que generalmente se encuentra debajo del volante. Algunos de los parámetros a los que se puede acceder incluyen el estado del control de emisiones, la velocidad de conducción, el voltaje del sistema y la temperatura del refrigerante del motor (Moniaga et al., 2017).

El funcionamiento del OBD es similar al de una habitación que contiene un termostato. El termostato recibe la temperatura deseada y monitorea continuamente la temperatura del aire. Si esta desciende por debajo de la temperatura deseada, la caldera se enciende; de lo contrario, se apaga.

Como se puede observar, el OBD es un sistema autorregulado. En el caso de la habitación del ejemplo, permite ahorrar costos al mantener la caldera encendida o apagada automáticamente según los parámetros que se le indiquen.

En los vehículos modernos, el OBD se emplea principalmente en el sistema de control de combustible, monitoreando las condiciones en tiempo real del motor y ajustando la alimentación de combustible según las necesidades del vehículo (McCord, 2011).

3.1.2. *Raspberry PI*

De acuerdo con Zhao et al. (2015), Raspberry Pi es una computadora de bajo costo, pequeña y portátil. Es posible conectarla a una variedad de dispositivos, como un monitor de computadora o televisión, teclado, ratón o unidad USB, permitiendo a los usuarios interactuar con el software integrado en el ordenador o desarrollar su propio software mediante el lenguaje Python.

Otra explicación acertada es: la Raspberry Pi (Figura 1) es una computadora de una sola placa, o SBC por sus siglas en inglés, desarrollada por la Fundación Raspberry Pi. Cada modelo incluye un procesador (CPU), un procesador gráfico (GPU), memoria integrada y una entrada de energía de cinco voltios. Todos los modelos también tienen una entrada para conectar una cámara y una serie de pines de entrada y salida (GPIO) para diversos usos (Jolles, 2021).

Por último, Maksimović et al. (2014) explican que la Raspberry Pi es un ordenador del tamaño de una tarjeta de crédito que opera igual que un ordenador estándar. Con un precio asequible de entre veinticinco y treinta y cinco dólares, está orientada a la educación.

3.1.2.1. *Wifi*

Las redes Wi-Fi cambian mucho desde que se crean en los años 80. Wi-Fi Alliance (2024) indica que desde 1999 trabaja para promover Wi-Fi en todo el mundo. Un hito importante es la certificación 802.11 del Instituto de Ingenieros Eléctricos y Electrónicos (IEEE), que asegura que

diferentes proveedores puedan trabajar juntos y da confianza a los usuarios para usar Wi-Fi. Gracias a esto, nuestra forma de trabajar, vivir y divertirnos cambia radicalmente. Impulsa muchas innovaciones en todas las industrias y aplicaciones, desde dispositivos y servicios hasta áreas enteras como el Internet de las cosas (IoT), llegando al Wi-Fi Certificado 7 actual.

Pahlavan y Krishnamurthy (2021) explican cómo Wi-Fi evoluciona y afecta la conectividad inalámbrica en diferentes campos. Además, Rayo y González (2016) detallan cómo gestionar y controlar redes Wi-Fi usando una interfaz en C++, cubriendo temas técnicos como la elección de protocolos, la seguridad y la mejora del rendimiento. Por último, O'Brien et al. (2022) estudian cómo manejar las redes Wi-Fi en universidades para mejorar el aprendizaje y reducir las distracciones, equilibrando la actividad educativa con otras distracciones.

3.1.2.2. *USB (Universal Serial Bus)*

El estándar USB cambia la manera en que se conectan dispositivos. SiliconLabs (2021) narra la historia del USB, desde las primeras versiones hasta las más recientes como USB 3.0, 3.2 y USB-C, además de explorar las futuras tendencias de esta tecnología. Este estándar revoluciona la transferencia de datos y la alimentación de dispositivos, prometiendo velocidades aún más altas y mayor versatilidad en el futuro.

Además, la investigación de Longhua et al. (2008) detalla cómo diseñar sistemas de control embebidos usando Scilab junto con USB, crucial para la comunicación con periféricos y la transferencia de datos.

Finalmente, Hao et al. (2021) analizan las amenazas de seguridad asociadas con dispositivos alimentados por USB, abordando desde ataques de canal lateral hasta vulnerabilidades en

controladores. Proponen medidas para garantizar la confidencialidad e integridad de los datos, contribuyendo al desarrollo de sistemas USB más seguros y robustos.

3.1.2.3. *Python*

Python, como muchos lenguajes, se usa ampliamente en IoT, siendo el principal lenguaje de programación para manejar diferentes componentes. Esto incluye leer, escribir y convertir valores digitales para generar salidas analógicas mediante modulación por ancho de pulso. Además, las capacidades orientadas a objetos de Python y su amplia biblioteca lo hacen ideal para trabajar con sensores, actuadores y para desarrollar sistemas complejos de comunicación y procesamiento de datos en tiempo real (Hillar, 2016).

Asimismo, Python permite usar diversas herramientas web para crear conexiones a Firebase y consumir esta API. (Severance, 2020, p. 151) explica:

> El protocolo de red que hace funcionar la web es en realidad bastante simple y existe un soporte integrado en Python llamado sockets, el cual hace que resulte muy fácil realizar conexiones de red y recuperar datos a través de esas conexiones desde un programa de Python. Un socket es muy parecido a un archivo, a excepción de que proporciona una conexión de doble sentido entre dos programas. Es posible tanto leer como escribir en un mismo socket. Si se escribe algo en un socket, se envía a la aplicación que está al otro lado de este. Si se lee desde el socket, se obtienen los datos que la otra aplicación ha enviado.

3.1.3. *Firebase*

Firebase es una herramienta de gestión de bases de datos que ofrece varias funcionalidades importantes, según menciona Agüero Aguiar (2018). Proporciona servicios esenciales para el desarrollo de aplicaciones modernas: firebase analytics ofrece información detallada sobre el uso de la app, firebase cloud messaging simplifica el envío de mensajes y notificaciones en diversas

plataformas y Firebase Auth facilita la autenticación de usuarios mediante email y contraseña con código en el cliente. Además, Firebase Realtime Database ofrece un backend en tiempo real para sincronizar y almacenar datos en la nube, con soporte multiplataforma. Por otro lado, Firebase Firestore proporciona una base de datos NoSQL flexible que organiza datos en documentos y colecciones, admitiendo una amplia variedad de tipos de datos y estructuras complejas.

Firebase se integra fácilmente con frameworks para el desarrollo de aplicaciones multiplataforma, como se indica en la documentación oficial: "si agregas un complemento de Firebase a tu app de Flutter, el complemento se usará en las versiones para Apple, Android y la Web" (Firebase, 2024, párr. 8). En resumen, Firebase es una herramienta muy versátil que facilita el desarrollo de aplicaciones al reducir la carga de trabajo, permitiendo que los desarrolladores se enfoquen en otros aspectos críticos de la aplicación.

3.1.4. *Flutter*

Flutter es una herramienta de interfaz de usuario y SDK móvil desarrollada por Google en 2018, basada en el lenguaje de programación Dart. Su objetivo es crear aplicaciones nativas de alto rendimiento con una única base de código. Utiliza widgets como el núcleo principal para construir componentes, permitiendo a los desarrolladores personalizarlos ampliamente. El framework incluye una biblioteca de componentes materiales para utilizar componentes predefinidos y ofrece funciones de hot reload para actualizaciones rápidas (Srinivasa Rao et al., 2022).

En cuanto al rendimiento, Wenhao (2018) destaca cómo Flutter puede competir con las aplicaciones nativas en términos de rendimiento de la CPU, con diferencias mínimas entre Flutter en iOS y Android comparado con las aplicaciones nativas desarrolladas en Swift y Kotlin. Además, Flutter requiere menos líneas de código para el desarrollo, lo que mejora la eficiencia. Es importante

mencionar que Flutter utiliza Dart como lenguaje de programación, optimizado para el cliente y compatible con la compilación tanto en código nativo como en JavaScript.

3.1.4.1. *Dart*

En términos generales, Dart es un lenguaje de programación diseñado para ser fácil de usar, escalable y familiar para los programadores. Se basa en clases y está completamente orientado a objetos, con un enfoque particular en el desarrollo de aplicaciones web complejas. Según Hassan (2020), Dart está optimizado para el desarrollo de clientes, lo que lo convierte en una opción ideal para crear rápidamente aplicaciones multiplataforma.

Es notable cómo Dart puede mejorar la experiencia del usuario en aplicaciones en tiempo real, especialmente en proyectos que requieren toma de decisiones basadas en datos (Sri Swathiga et al., 2021), lo cual es una funcionalidad crucial en Auto-matic.

3.2. Descripción de actividades

3.2.1. *Recolección de información.*

Para asegurar una base sólida y bien fundamentada para el proyecto, se lleva a cabo un proceso meticuloso de recolección de información. Este proceso incluye la realización de encuestas y consultas a expertos en el área automotriz, así como la investigación exhaustiva en fuentes confiables en internet.

En primer lugar, se diseña una encuesta (anexo 5) utilizando Google Forms. Esta encuesta está dirigida a propietarios de vehículos y se distribuye a través de redes sociales y entre compañeros universitarios y familiares. El objetivo es recopilar datos sobre las experiencias y necesidades de los usuarios en relación con el mantenimiento y monitoreo de sus vehículos. Las preguntas de la encuesta abarcan desde el conocimiento de los usuarios sobre los sistemas de su automóvil hasta las dificultades que enfrentan para interpretar y actuar sobre los datos de diagnóstico vehicular. Los

resultados obtenidos proporcionan una visión de los problemas comunes y las expectativas de los usuarios, información que es esencial para el desarrollo y la justificación del proyecto.

Además de la encuesta, uno de los miembros del equipo, con experiencia previa en el área automotriz, actúa como una fuente de consulta. Este integrante comparte su conocimiento sobre el funcionamiento básico de los vehículos, los estándares y protocolos utilizados en la industria automotriz y las fallas más comunes que pueden ocurrir, así como los métodos actuales para su análisis y prevención. Su experiencia es particularmente útil para comprender el sistema OBD (On-Board Diagnostics), que es crucial para el proyecto, explicando su funcionamiento y las herramientas necesarias para interactuar con la computadora de los automóviles.

La investigación en internet también juega un papel fundamental en la recolección de información. Se consultan diversas fuentes en línea, incluyendo artículos técnicos, manuales de usuario y documentación oficial sobre OBD y los protocolos de comunicación vehicular. Esta búsqueda permite identificar las herramientas y tecnologías más adecuadas para el proyecto, como las librerías de software necesarias para la comunicación con el OBD y los métodos de integración con bases de datos en la nube.

A través de este proceso de recolección de información, se garantiza una comprensión profunda y detallada de los aspectos técnicos y prácticos necesarios para el desarrollo del Auto-matic.

3.2.2. *Diseño y desarrollo de dispositivo IoT para monitoreo del automóvil.*

Para comenzar con el diseño del dispositivo de IoT se consideran los datos que se desea monitorear; estos incluyen: temperatura del refrigerante del motor, velocidad, revoluciones por minuto, voltaje de la batería y códigos de falla presentes.

Basado en lo anterior, la lista de materiales necesarios es la siguiente:

- Raspberry Pi 4 modelo B 4GB RAM.

- Fuente de alimentación Raspberry Pi de 5 V y 3000 mA.

- Inversor de corriente automotriz de 150 W (12 Vcc a 110 Vca) con plug para encendedor.

- Escáner Automotriz USB Elm327 OBD2.

En el Anexo 6 se ilustran las conexiones entre componentes.

A continuación, se discute el lenguaje a utilizar para la codificación del dispositivo IoT, decidiendo utilizar Python debido al amplio soporte de Raspberry Pi y a la existencia de la librería Python-OBD, que permite una conexión rápida y sencilla a la computadora del vehículo.

Después de finalizar el diseño del dispositivo, se codifica un programa de prueba basándose en la documentación de Python-OBD. El programa permite ingresar comandos de forma manual para obtener datos de la computadora del vehículo.

Al terminar, se realiza el siguiente proceso para comprobar el correcto funcionamiento de los componentes:

- Dentro de la Raspberry Pi se crea el directorio “OBD_reader” y dentro del mismo se crea un entorno virtual de Python.

- El entorno virtual se activa y se instala la librería Python-OBD.

- Los componentes se conectan de acuerdo con el diagrama de componentes.

- El script de prueba se carga e inicia.

- Se insertan los comandos SPEED, RPM, COOLANT_TEMP y GET_DTC de forma manual y se observan los resultados.

La prueba muestra que todos los componentes se encuentran en perfecto estado y brindan las respuestas adecuadas. Además, se descubre que, si se introduce cualquier comando sin estar

conectado a la computadora del vehículo, se recibe un dato ‘NONE’. Lo anterior se registra y se tiene en cuenta para las siguientes etapas del desarrollo.

El desarrollo continuo se realiza mediante la programación del script “realtime_values”, que obtiene los datos requeridos en tiempo real. En este punto se instala la librería Time para permitir realizar pausas en el código cuando se requiere.

El programa hace uso de una clase que contiene los comandos necesarios almacenados como atributos. Por otra parte, contiene las siguientes funciones:

- SetUp: se encarga de regular la llamada a las funciones “ConnectToCar” y “StartReader”. En caso de recibir como respuesta un cero por parte de las funciones mencionadas, vuelve a llamarlas para asegurarse de que siempre se cuenta con conexión a la computadora del vehículo.

- ConnectToCar: crea la conexión hacia la computadora del vehículo y la guarda en una variable para su posterior uso.

- StartReader: inicia el ciclo infinito para obtener los datos solicitados, con un pequeño retraso entre cada solicitud y los imprime en consola.

En el Anexo 7 se adjuntan fotos del dispositivo ensamblado y capturas de pantalla con los datos obtenidos.

3.2.3. *Conexión del dispositivo IoT a la base de datos en la nube (Firebase).*

Para comenzar, se crea la base de datos Firestore y la colección “vehiculeRealtime”, la cual contiene los campos “batteryVoltage”, “failureCodes”, “rpm”, “service”, “speed”, “speedUnit”, “temp” y “tempUnit”, en los cuales se vacían los datos provenientes del dispositivo de IoT. El proceso de creación se describe con mayor detalle en el Anexo 8.

Para realizar la conexión con la base de datos en la nube, se consulta la documentación correspondiente y, de acuerdo con esto, se realizan los siguientes pasos antes de codificar la conexión:

- El paquete `firebase-admin` se instala mediante pip en el entorno virtual de Python, asegurándose de que la versión de Python sea la 3.6 o anterior.

- Para realizar la conexión, son necesarias credenciales de acceso válidas, las cuales se generan desde el portal de Firebase y se obtienen en un archivo JSON.

- Las credenciales se colocan en el directorio "OBD_reader" junto al programa de lectura de datos.

Una vez cumplidos estos pasos, se puede continuar con la codificación de la conexión.

En primer lugar, para realizar la conexión, se importa la librería `firebase_admin`. Después, se agregan dos variables para guardar el nombre del documento al que se cargarán los datos y el nombre de la colección donde se encuentra el documento. Luego, se crea la conexión a Firestore; primero, se inicializa la aplicación Firebase con las credenciales obtenidas previamente y después se crea una variable para almacenar la colección hacia la que se cargarán los datos.

A continuación, se modifica la función "StartReader" para inicializar la variable "dictData" con un diccionario vacío en cada ciclo. Este diccionario se usa para guardar los datos en sus correspondientes campos de acuerdo con el documento de Firebase. Luego, se crea la función "CheckNull", la cual revisa que el dato recibido desde la computadora del vehículo no sea de tipo NONE. Si lo es, el dato se omite para evitar llenar la base de datos con datos inútiles; en caso contrario, se añade a la variable "dictData".

Más adelante, se codifica la función "UpdateData", que actualiza la base de datos solo cuando la variable "dictData" contiene datos, para ahorrar operaciones de escritura.

Con todo esto, el programa está listo para ejecutarse de forma manual. Lo ideal es ejecutarlo de forma automática cada vez que se enciende el dispositivo. Con esto en mente, se continúa con el desarrollo de la siguiente forma:

- Para ejecutar el programa de forma automática, se codifica el script `start_reader.sh`, que tiene las siguientes funciones:

  - Activar el entorno virtual de Python, necesario para contar con las librerías indispensables para el funcionamiento del programa.

  - Iniciar el programa “realtime_values”.

Seguidamente, se crea el archivo `IoTReader.service` en el directorio `/etc/systemd/system/`, el cual se encarga de convertir el script bash en un servicio que se ejecuta cuando se enciende el dispositivo. Dentro del archivo `IoTReader.service`, se indican tres secciones:

- `[Unit]`: esta sección contiene información general del servicio, como su descripción y cuándo debe ejecutarse. En este caso, se indica que el servicio solo debe inicializarse después de que la red esté disponible.

- `[Service]`: indica el comando para iniciar el servicio; en este caso, se inserta la ruta hasta el script `start_reader.sh`.

- `[Install]`: especifica cómo el servicio es instalado e inicializado por el sistema; para esto, se usa la opción por defecto `default.target`.

Por último, se reinicia el controlador de servicios con `systemctl`, se habilita el servicio `IoTReader` y luego se inicia. Una vez hecho todo esto, se comprueba el estado con el comando “sudo systemctl status IoTReader” y, cuando el servicio se encuentra en estado “active”, todo está

listo para realizar las pruebas necesarias. En el anexo 2 se ilustra la interacción entre el dispositivo iot y Firebase.

3.2.4. *Pruebas de funcionamiento del dispositivo IoT*

Durante las pruebas de funcionamiento del dispositivo IoT Auto-matic, se llevan a cabo diversas evaluaciones utilizando vehículos distintos y configuraciones de energía variadas para garantizar su desempeño y estabilidad en condiciones reales de uso.

Primero, se instala el dispositivo en el automóvil del padre de uno de los integrantes. El proceso incluye conectar el Raspberry Pi al sistema eléctrico del vehículo mediante una conexión de 12V con un conversor, con el objetivo de evaluar su autonomía y rendimiento en condiciones reales de conducción. Durante un período determinado, se monitorea el funcionamiento del sistema, recogiendo datos sobre la estabilidad de la conexión, la precisión de las lecturas y la efectividad de las alertas en tiempo real. Se espera verificar la integración y el desempeño del dispositivo con la fuente de energía del vehículo, así como recoger datos para optimizar la configuración y mejorar la experiencia del usuario.

Luego, se realizan pruebas con el vehículo de uno de los integrantes en la clase de la maestra Dynhora Ramírez. En esta fase, se utiliza un hotspot para asegurar una conexión a internet estable y se conecta el Raspberry Pi al inversor de corriente del vehículo a través del conector de cigarrillos, evaluando así su rendimiento energético y funcionalidad. El dispositivo se instala en el vehículo y se configuran tanto el hotspot como el inversor de corriente. Las pruebas se realizan en diferentes condiciones de manejo para evaluar la estabilidad y fiabilidad del sistema. El objetivo es evaluar la viabilidad de utilizar un inversor de corriente como fuente de energía e identificar posibles problemas y áreas de mejora en la conexión y el rendimiento del sistema.

Asimismo, se llevan a cabo pruebas con el profesor Vidal, utilizando un hotspot para asegurar una conexión a internet estable y conectando el Raspberry Pi a un powerbank para evaluar su rendimiento y autonomía con esta fuente de energía portátil. El dispositivo se instala en el vehículo del profesor Vidal y se configuran tanto el hotspot como el powerbank. Las pruebas se realizan durante diferentes períodos de tiempo y en diversas condiciones de manejo. Se espera determinar la efectividad y la autonomía del sistema utilizando un powerbank, recogiendo datos sobre la duración de la batería y la estabilidad del sistema en condiciones reales.

Finalmente, después de realizar pruebas exhaustivas con diferentes configuraciones de energía, se decide utilizar un powerbank para la presentación del prototipo debido a su portabilidad, facilidad de uso y fiabilidad. Se llevan a cabo pruebas finales en el vehículo, utilizando el powerbank para asegurar que el sistema funcione correctamente y sin interrupciones. El objetivo es confirmar que el sistema Auto-matic es estable y funcional utilizando un powerbank, listo para la presentación final del prototipo.

3.2.5. *Diseño y desarrollo de la aplicación web con Flutter.*

Para el diseño y desarrollo de la aplicación web con Flutter, el primer paso consiste en crear el wireframe y elegir la paleta de colores. Se elabora un bosquejo visual de la aplicación y se seleccionan los colores que se utilizan, utilizando herramientas como Canvas para el wireframe (anexo 3) y una paleta de colores adecuada. Después, se realiza un diagrama de caso de uso (anexo 4) para determinar las interacciones que los usuarios tienen a su alcance al utilizar la app.

Una vez definidos estos aspectos, se procede a crear el proyecto de Flutter y el repositorio correspondiente. Esto implica configurar el proyecto con las herramientas de desarrollo necesarias y descargar las extensiones pertinentes. Posteriormente, se sube el proyecto a un repositorio,

generalmente en GitHub, para que todos los miembros del equipo puedan acceder a él y colaborar de manera eficiente.

Con el proyecto en el repositorio, se crean ramas específicas para cada miembro del equipo. Estas ramas permiten a cada integrante trabajar en sus tareas asignadas sin interferir con el trabajo de los demás. Las ramas llevan el nombre de cada miembro para facilitar su identificación y manejo.

El siguiente paso es repartir las interfaces entre los miembros del equipo y comenzar con el desarrollo del frontend. Se planifica detalladamente cómo se ve la aplicación, incluyendo todos los elementos visuales. Cada miembro del equipo recibe una parte equitativa del trabajo con una fecha de finalización establecida. Una vez completada su parte, cada miembro sube su trabajo al repositorio para su revisión.

La revisión del frontend es un proceso crucial donde el trabajo de cada miembro es evaluado por el encargado del equipo. Se realizan las correcciones necesarias basándose en las observaciones y sugerencias del supervisor. Tras estas correcciones, todas las partes del frontend se fusionan en una nueva rama denominada 'dev', consolidando el trabajo visual del proyecto.

Posteriormente, se desarrolla la responsividad de la aplicación. Cada miembro se encarga de adaptar su parte del proyecto para que funcione correctamente en distintos tamaños y tipos de pantalla. Una vez ajustada la responsividad, se realiza una revisión para identificar y corregir cualquier desperfecto.

Las correcciones de responsividad son evaluadas y, una vez resueltas, todo el trabajo se vuelve a juntar en la rama 'dev'. Desde esta rama, se pasa todo el código completamente corregido a la rama principal, 'main', asegurando que el proyecto esté listo para las etapas finales.

Finalmente, se realiza el enrutamiento de las pantallas. Este paso es esencial para conectar correctamente todas las pantallas de la aplicación y evitar errores de navegación. Con todas las pantallas bien enrutadas, se garantiza que el usuario tenga una experiencia fluida y sin problemas al utilizar la aplicación.

A lo largo del desarrollo, se documenta cada paso y se utiliza el repositorio para mantener un control eficiente de las versiones y colaboraciones, asegurando que el proyecto avance de manera ordenada y sistemática.

3.2.6. *Diseño y Desarrollo de la aplicación móvil con Flutter.*

Para el diseño y desarrollo de la aplicación móvil con Flutter, se comienza modificando el enrutamiento de las pantallas para adaptarlas a la experiencia del usuario en dispositivos móviles. Este ajuste asegura que la navegación sea intuitiva y adecuada para la interfaz móvil. A continuación, se realizan cambios específicos para que, desde la aplicación móvil, los usuarios no puedan acceder a la landing page. Esto implica modificar la pestaña de inicio de la aplicación móvil, de manera que el catálogo no sea visible para los usuarios de estos dispositivos.

Además, se implementan restricciones para que los usuarios móviles tampoco puedan acceder a la página de registro. Esta limitación se introduce por razones de seguridad, asegurando que ciertas funciones estén restringidas en la versión móvil. Consecuentemente, la pantalla principal en la aplicación móvil se cambia para que sea el login. Este cambio garantiza que solo los usuarios autenticados puedan acceder a su información, proporcionando una capa adicional de seguridad.

Otro paso crucial en el desarrollo es modificar los permisos en Android para permitir el uso de Internet. Esta modificación es necesaria para que la aplicación móvil pueda aprovechar todas sus funcionalidades. Adicionalmente, se trabaja en la modificación de las pantallas de registro de usuario.

La meta es simplificar estas pantallas, combinándolas en una sola que permita tanto el registro como el login. Esta fusión minimiza el número de pantallas accesibles sin iniciar sesión, optimizando la experiencia del usuario y mejorando la seguridad.

Cada una de estas etapas se lleva a cabo con un enfoque detallado, asegurando que la aplicación móvil no solo sea funcional, sino también segura y fácil de usar. La coordinación entre los miembros del equipo y el uso de herramientas de desarrollo colaborativo como GitHub juegan un papel fundamental en la realización de estas tareas, permitiendo que cada cambio y mejora sea documentado y revisado de manera eficiente. En cada fase del desarrollo, se realizan pruebas para asegurar que todas las modificaciones cumplen con los requisitos establecidos y mejoran la experiencia del usuario en dispositivos móviles.

3.2.7. *Pruebas de funcionamiento de la interfaz (móvil/web).*

Durante el desarrollo de Auto-matic, se llevan a cabo varias pruebas de funcionamiento de la interfaz móvil y web. A lo largo de estas pruebas, se encuentran diversos errores que se solucionan de manera efectiva. Uno de los problemas principales se presenta en la pantalla de inicio (Landing), donde las tarjetas flotantes causan un overflow lateral al superar cierta cantidad de píxeles. Para solucionar esto, se implementa un ajuste que cambia el diseño de Column a Row cuando se supera dicha cantidad de píxeles, asegurando así una visualización correcta en todas las resoluciones.

Otro desafío surge en los formularios de inicio de sesión y registro, los cuales presentan problemas de desbordamiento en pantallas más pequeñas. La solución consiste en utilizar una combinación de widgets Column y Row para ajustar el diseño y evitar el overflow, mejorando así la experiencia del usuario en dispositivos móviles.

Además, se identifica que las tarjetas de información se deforman al comprimirse en pantallas más pequeñas. Para abordar este problema, se reemplazan los widgets de diseño existentes por componentes más adaptables y robustos de Syncfusion, lo que permite mantener la integridad visual de las tarjetas en diferentes tamaños de pantalla.

El proceso de registro también se optimiza mediante la implementación de un timeline que divide el registro en tres pasos claros y manejables. Aunque inicialmente esto causa un overflow, se realiza una reforma en el diseño para solucionar el problema, logrando así una mejora significativa en la experiencia del usuario durante el registro.

Se realizan pruebas exhaustivas de reactividad y funcionamiento de widgets en diversos dispositivos para asegurar que el diseño mínimo funcione correctamente en pantallas de al menos 320 píxeles. Estas pruebas incluyen ajustes precisos para garantizar que todos los elementos sean responsivos y se adapten adecuadamente a diferentes tamaños de pantalla, asegurando una experiencia de usuario coherente y fluida.

En cuanto al perfil del usuario, no se encuentran errores significativos. Las funcionalidades operan como se espera, permitiendo a los usuarios acceder y gestionar su información sin inconvenientes.

En resumen, las pruebas de funcionamiento de la interfaz móvil y web de Auto-matic son esenciales para identificar y solucionar varios errores. Estas mejoras no solo optimizan la visualización y reactividad de la interfaz, sino que también garantizan una experiencia de usuario más intuitiva y satisfactoria.

3.2.8. *Conexión de la aplicación web/móvil con la base de datos en la nube (Firebase).*

Para la configuración del proyecto de Firebase con Flutter, se siguen varios pasos detallados para asegurar una integración correcta y funcional entre la aplicación web/móvil y la base de datos en la nube.

Primero, se crea un nuevo proyecto en Firebase. Para ello, se visita la consola de Firebase y se selecciona la opción "Add project", siguiendo las instrucciones proporcionadas para completar el proceso. Una vez creado el proyecto, se selecciona en la consola de Firebase para proceder con las configuraciones necesarias.

El siguiente paso implica registrar la aplicación con Firebase. En el proyecto de Firebase, se selecciona "Añadir App" y se elige la plataforma correspondiente (iOS, Android o Web). Se siguen las instrucciones para registrar la aplicación, lo que incluye descargar un archivo de configuración específico para cada plataforma: googleService-Info.plist para iOS o google-services.json para Android. Este archivo de configuración se coloca en el directorio apropiado del proyecto Flutter, asegurando que para iOS se coloque en ios/Runner y para Android en android/app.

A continuación, se añaden los paquetes de Firebase necesarios al proyecto Flutter. Para ello, se abre el archivo pubspec.yaml y se agregan las dependencias necesarias, como firebase_core, firebase_auth y cloud_firestore, entre otros paquetes de Firebase según sea necesario. Después de añadir las dependencias, se ejecuta el comando flutter pub get para instalarlas. Luego, se inicializa Firebase en la aplicación Flutter editando el archivo principal (main.dart) para asegurar que Firebase se configure correctamente al iniciar la aplicación.

Una vez configurado Firebase, se procede a añadir Firebase Authentication a la aplicación. Se asegura de haber añadido firebase_auth a las dependencias en pubspec.yaml y se inicializa

Firebase Authentication en la aplicación Flutter de manera similar a como se inicializa Firebase en general. Para manejar el estado de autenticación, se utiliza el stream authStateChanges proporcionado por FirebaseAuth, lo que permite escuchar los cambios en el estado de autenticación y mostrar la pantalla correspondiente (página principal o página de inicio de sesión) según el estado del usuario.

Para habilitar la autenticación con correo y contraseña, se utilizan los métodos createUserWithEmailAndPassword y signInWithEmailAndPassword, que permiten registrar y autenticar usuarios mediante sus correos electrónicos y contraseñas.

Además, se implementa la autenticación con Google. Para ello, se añaden las dependencias necesarias en pubspec.yaml y se configura el proyecto de Firebase para soportar el acceso a la API de Google. Se describe el proceso de autenticación con Google, desde solicitar las credenciales del usuario hasta autenticarlo en la aplicación mediante Firebase.

En cuanto al manejo de la información del usuario, se obtienen los datos de la cuenta de Google del usuario autenticado y se actualiza la información del perfil del usuario utilizando métodos proporcionados por Firebase.

Finalmente, se configura Firestore, la base de datos en tiempo real de Firebase. Se asegura de haber añadido cloud_firestore a las dependencias en pubspec.yaml y se inicializa Firestore en la aplicación Flutter de manera similar a como se inicializa Firebase. Se describe cómo añadir y leer datos en Firestore, explicando los métodos utilizados para interactuar con la base de datos, como set y get en las colecciones o documentos de Firestore.

3.2.9. *Pruebas de conectividad y despliegue de datos (móvil/web).*

Para llevar a cabo las pruebas de conectividad y despliegue de datos tanto en la aplicación móvil como en la web, se inicia con la creación de una cuenta utilizando un correo y una contraseña.

Este método tradicional permite a los usuarios ingresar sus datos de acceso para poder utilizar la aplicación. Posteriormente, se crea una cuenta a través de Google, utilizando el servicio de autenticación de esta plataforma para verificar la información del usuario. Esta integración permite una experiencia más rápida y segura para los usuarios que prefieren utilizar su cuenta de Google.

Una vez creadas las cuentas, se prueba el inicio de sesión con Google, verificando que el sistema pueda autenticar correctamente a los usuarios y permitirles acceder a la aplicación. De igual manera, se realiza la prueba de inicio de sesión con correo y contraseña, asegurando que esta forma tradicional de autenticación funcione sin inconvenientes.

Con los usuarios autenticados, se verifica que los datos personales se visualicen correctamente. Cada usuario debe poder ver su información de manera precisa y ordenada. Además, se comprueba que la lista de vehículos asociada a cada usuario se muestre de forma adecuada, extrayendo toda la información pertinente sin errores. Esto incluye datos estáticos, como el modelo y la marca del vehículo, que deben ser visibles de manera correcta.

El siguiente paso es asegurar que los datos en tiempo real del vehículo, como la velocidad y el estado del motor, se desplieguen correctamente cuando el vehículo está encendido. Este aspecto es crucial, ya que permite a los usuarios monitorear el estado de su vehículo en tiempo real a través de la aplicación.

Cada uno de estos pasos se ejecuta meticulosamente, asegurando que todas las funciones de conectividad y despliegue de datos cumplan con los estándares establecidos y brinden una experiencia de usuario óptima. Las pruebas se realizan tanto en la versión móvil como en la web, garantizando que la funcionalidad sea consistente en ambas plataformas. La colaboración del equipo y la utilización de herramientas de desarrollo colaborativo como GitHub son fundamentales para

documentar y revisar cada fase del proceso, asegurando la calidad y el correcto funcionamiento del sistema.

3.2.10. *Documentación del proyecto.*

Para la documentación del proyecto, se decide designar a uno de los integrantes del equipo como editor principal del documento. Este editor se encarga de coordinar la redacción y organización del contenido, asegurando la claridad y coherencia en cada sección. Las reuniones se realizan durante las clases de la materia Integradora, donde se explica la parte del documento que se está redactando en ese momento.

Cada capítulo del documento se divide en pequeñas secciones para que cada miembro del equipo pueda trabajar en ellas de manera eficiente. Se asignan tiempos de entrega específicos para cada sección, garantizando un flujo continuo de trabajo y permitiendo una revisión regular del contenido. Llegado el tiempo de entrega, el editor principal y el resto del equipo revisan conjuntamente el trabajo realizado por cada miembro. Durante estas revisiones, se evalúa si es necesario realizar ajustes o correcciones.

Una vez que el equipo está de acuerdo con el contenido, el editor principal integra el trabajo de todos en el borrador del documento final. Este borrador se presenta a la profesora para su evaluación. La retroalimentación proporcionada por la profesora se usa para identificar áreas de mejora y realizar las correcciones necesarias. Este proceso de revisión y retroalimentación se repite hasta que la profesora aprueba la sección en cuestión.

Durante las dos semanas de vacaciones, se sigue el mismo proceso, con reuniones que se llevan a cabo los viernes en la biblioteca pública central "Carlos Montemayor". Estas sesiones

adicionales permiten mantener el ritmo de trabajo y asegurar que el proyecto avance según lo planeado.

La documentación abarca todas las actividades y tecnologías utilizadas durante el desarrollo del proyecto. Se registra detalladamente cada paso, desde el diseño y desarrollo de Auto-matic hasta la conexión a la base de datos en la nube. Cada tecnología, como las aplicaciones web y móvil desarrolladas con Flutter y Firebase, se describe de manera clara y precisa, proporcionando un panorama completo del proceso de desarrollo.

3.2.11. *Presentación del proyecto.*

Para la presentación del proyecto, se realizan diversas actividades orientadas a garantizar una exposición clara y efectiva en dos eventos principales: la presentación ejecutiva, programada para el viernes 2 de agosto y la presentación final, fijada para el martes 6 de agosto.

Durante las clases de Expresión oral y escrita, se dedica tiempo a practicar técnicas de oratoria y presentación pública. Estas sesiones incluyen ejercicios para mantener la atención del público, el uso adecuado del lenguaje corporal y la selección de la vestimenta apropiada para un entorno formal. Se elaboran diapositivas con un diseño atractivo y profesional, utilizando gráficos y esquemas que faciliten la comprensión del contenido.

Para practicar las ideas generales de la presentación, se realizan ensayos en la Biblioteca Pública Central "Carlos Montemayor". Estos ensayos permiten afinar el discurso y coordinar la intervención de cada miembro del equipo.

La presentación final se lleva a cabo en la sala de rectores y para ello se coordina con la profesora encargada de los proyectos integradores, Eva Pérez. Ella autoriza realizar la presentación

fuera del edificio de rectoría debido al uso de un vehículo como parte del proyecto. Para esta presentación, se organiza una mesa y una carpa que proporciona sombra y protege al equipo del sol.

Además, se planea llevar una televisión para mostrar las aplicaciones desarrolladas, conectándola a una extensión para garantizar el suministro eléctrico adecuado. Las aplicaciones se presentan en tiempo real, demostrando su funcionalidad y eficiencia.

Para asegurar una presentación fluida y coordinada, se realizan ensayos donde se revisa cada parte del discurso y se ajusta según sea necesario. El editor del documento se encarga de consolidar la información y distribuir las tareas de exposición entre los miembros del equipo, asignando tiempos específicos para cada sección de la presentación.

Capítulo 4. Resultados y análisis

4.1. Resultados

Auto-matic desarrolla una aplicación web y móvil utilizando Flutter y Firebase para el monitoreo preventivo de vehículos. El sistema se conecta a través de un Raspberry Pi al puerto OBD-II del vehículo, permitiendo obtener en tiempo real datos críticos como la temperatura del motor, la velocidad, el voltaje de la batería y las revoluciones por minuto (RPM). Además, la aplicación proporciona alertas de servicio en tiempo real, lo que permite a los usuarios tomar medidas preventivas y mantener sus vehículos en óptimas condiciones.

4.1.1. *Resultados Cuantitativos*

Reducción de visitas al mecánico: se busca reducir la frecuencia con la que los usuarios llevan sus vehículos al mecánico. Se estima que los usuarios de Auto-matic reducen sus visitas al mecánico en un 40%, ya que el sistema proporciona alertas tempranas sobre posibles fallas, permitiendo el mantenimiento preventivo.

Costo Promedio de una Visita al Mecánico: 3,533 pesos mexicanos.

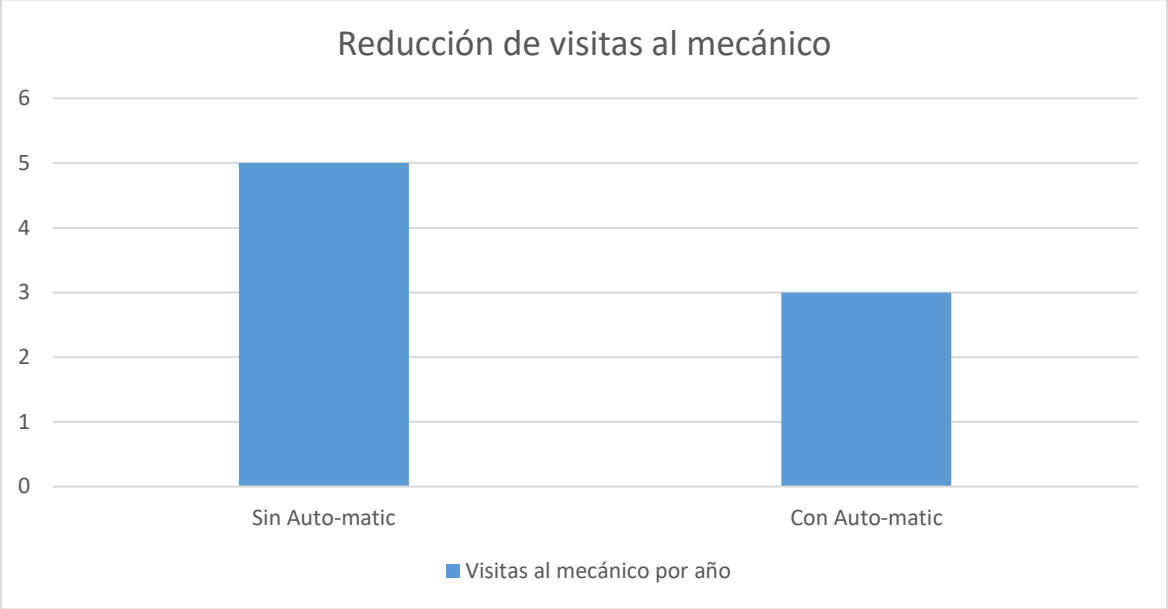
Costo de Auto-matic: 500 pesos mexicanos.

Al reducir el número de visitas al mecánico de cinco a dos veces al año, los usuarios ahorran aproximadamente 10,000 pesos mexicanos anualmente.

Reducción de visitas al mecánico

Figura 3

Comparación del número visitas a un mecánico en un año

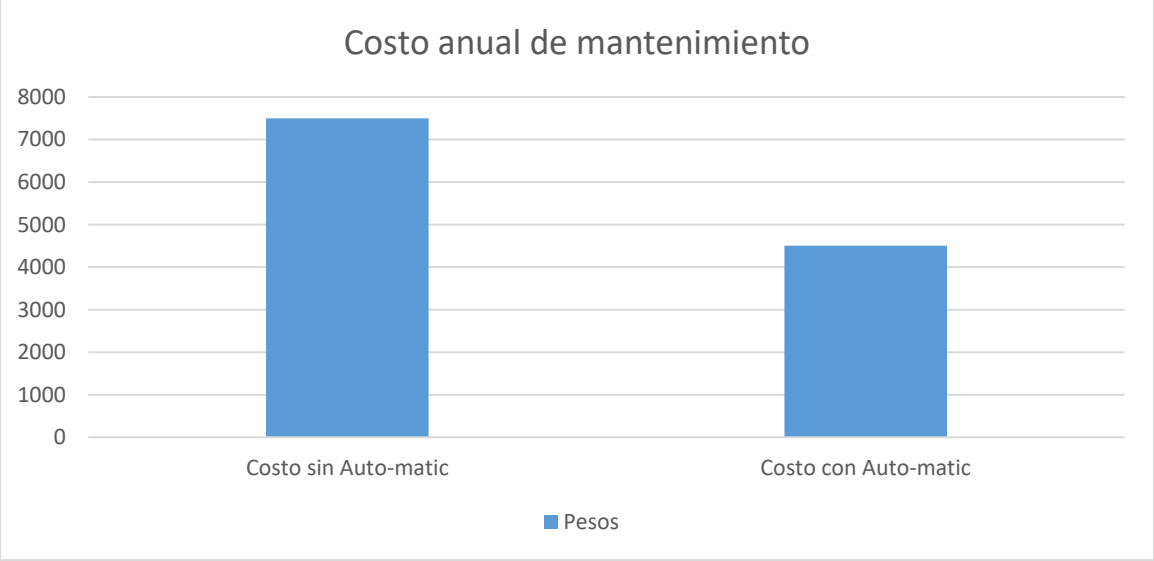


Fuente: elaboración propia.

Costo anual de mantenimiento

Figura 4

Comparación del costo en pesos mexicanos en mantenimiento de un automóvil en un año



Fuente: elaboración propia

Tiempo de respuesta a fallas

Figura 5

Comparación del tiempo que toma solucionar una falla en un automóvil

Fuente: elaboración propia

4.1.2. *Resultados Cualitativos*

Satisfacción del usuario: los usuarios reportan estar muy satisfechos con la funcionalidad y facilidad de uso de la aplicación. La capacidad de recibir alertas en tiempo real y la precisión de los datos monitoreados son los aspectos más valorados.

Percepción de seguridad: los automovilistas sienten que su vehículo es más seguro gracias al monitoreo constante proporcionado por Auto-matic. La tranquilidad de saber que cualquier problema es detectado y reportado de inmediato es un factor clave en esta percepción.

4.2. Conclusiones

El desarrollo del prototipo facilita la provisión de datos cruciales y alertas de servicio en tiempo real, mejorando considerablemente la seguridad y optimizando el mantenimiento vehicular.

Los resultados obtenidos demuestran que el sistema es efectivo para prevenir fallos vehiculares y reducir los costos de mantenimiento, proporcionando una herramienta eficiente tanto para los usuarios finales como para la empresa Auto-matic.

Durante el desarrollo del proyecto, se adquieren habilidades significativas en el desarrollo de aplicaciones móviles y web, así como en la integración de hardware y software. Se trabaja con tecnologías avanzadas como Flutter, Firebase, Raspberry Pi y OBDII, lo cual proporciona una experiencia práctica valiosa y relevante para el futuro profesional. Además, se desarrollan competencias en gestión de proyectos, trabajo en equipo y resolución de problemas, aplicando conocimientos teóricos en un entorno real.

Para la empresa Auto-matic, la implementación de esta solución fortalece la capacidad para ofrecer productos tecnológicos avanzados, traduciéndose en una ventaja competitiva en el mercado.

La experiencia adquirida y la solución implementada destacan la viabilidad de utilizar tecnologías como Flutter, Firebase, Raspberry Pi y OBDII en soluciones de monitoreo automotriz, demostrando su impacto positivo en la seguridad y eficiencia vehicular.

4.3. Recomendaciones

Para garantizar la seguridad y estabilidad del dispositivo en el automóvil, se recomienda conectarlo directamente al sistema eléctrico del vehículo a través de los fusibles existentes. Esta conexión directa asegura una instalación segura y reduce el riesgo de fallas eléctricas. Además, se sugiere reducir el tamaño del dispositivo a una sola caja que contenga todas las piezas individuales. Esta consolidación facilita tanto la instalación como el mantenimiento del dispositivo, haciéndolo más compacto y manejable para los usuarios.

Otra recomendación importante es diseñar el dispositivo para que se alimente directamente desde el conector OBD-II de los escáneres del vehículo. Esta modificación elimina la necesidad de una fuente de alimentación adicional, simplificando la configuración y reduciendo el número de componentes necesarios.

Asimismo, se considera beneficioso añadir una pestaña de historial en la interfaz de la aplicación. Esta pestaña mostraría todos los códigos de error registrados en cualquier momento, permitiendo a los usuarios revisar y gestionar estos códigos fácilmente. Con esta funcionalidad, los usuarios pueden tener un seguimiento detallado de los problemas detectados en su vehículo, facilitando el mantenimiento y las reparaciones.

Se propone también implementar una función que verifique el número de registro del vehículo contra una base de datos de vehículos robados. Esta verificación adicional incrementa la seguridad al asegurar que el vehículo no tenga reportes de robo, ofreciendo a los usuarios una capa extra de tranquilidad.

Para mejorar la experiencia del usuario, se recomienda agregar notificaciones a la aplicación. Estas notificaciones pueden alertar a los usuarios sobre eventos importantes, como la aparición de nuevos códigos de error, recordatorios de mantenimiento, o alertas de seguridad. Las notificaciones garantizan que los usuarios estén siempre informados sobre el estado de su vehículo y cualquier acción que necesiten tomar.

Referencias

Agüero Aguiar, E. (2018). *Firebase en el desarrollo de aplicaciones móviles.* [Tesina de licenciatura, Universidad Politécnica de Sinaloa, Programa Académico de Ingeniería en Informática]. http://repositorio.upsin.edu.mx/formatos/TesinaEdgarUlisesAgueroAguiar6855.pdf

Firebase. (10 de junio de 2024). *Agrega Firebase a tu app de Flutter*. https://firebase.google.com/docs/flutter/setup?hl=es-419&platform=ios

Hao L., Riccardo S., Federico T., Riccardo B. y Mauro C. (2021). USB powered devices: A survey of side-channel threats and countermeasures. *High-Confidence Computing, 1* (1). https://doi.org/10.1016/j.hcc.2021.100007

Hassan, A. (2020). Java and Dart programming languages: Conceptual comparison. *Indonesian Journal of Electrical Engineering and Computer Science, 17* (2), 845-849. http://doi.org/10.11591/ijeecs.v17.i2.pp845-849

Hillar, G. (2016). *Internet of Things with Python.* Packt Publishing. https://books.google.es/books?hl=es&lr=&id=I9FtDQAAQBAJ&oi=fnd&pg=PP1&dq=Internet+of+Things+with+Python&ots=YFO7f588HI&sig=4KbzI429beI61DYlWyX8-25oBjY#v=onepage&q=Internet%20of%20Things%20with%20Python&f=false https://repsitory.udistrital.edu.co/bitstream/handle/11349/2559/GonzalezCarlosEnrique2016.pdf?sequence=1&isAllowed=y

Jolles, J. (2021). Broad-scale applications of the Raspberry Pi: A review and guide for biologists. *Methods in Ecology and Evolution, 12* (9), 1562–1579. https://doi.org/10.1111/2041-210X.13652

Longhua, M., Feng, X. y Zhe, P. (2008). Integrated Design and Implementation of Embedded Control Systems with Scilab. *Sensors, 8,* 5501-5515. https://doi.org/10.3390/s8095501

Maksimović, M., Vujović, V., Davidović, N., Milošević, V., y Perišić, B. (2-5 de junio 2014). *Raspberry Pi as Internet of things hardware: performances and constraints.* 1st International Conference on Electrical, Electronic and Computing Engineering. ETRAN, Vrnjačka Banja, Serbia. https://www.researchgate.net/profile/Vladimir-Vujovic/publication/280344140_ELI16_Maksimovic_Vujovic_Davidovic_Milosevic_Perisic/links/55b3368608ae9289a08594aa/ELI16-Maksimovic-Vujovic-Davidovic-Milosevic-Perisic.pdf

McCord, K. (2011). *Automotive diagnostic systems: Understanding OBD-I OBD-II*. CarTech. https://books.google.es/books?hl=es&lr=&id=kyEtsrPk9ZQC&oi=fnd&pg=PP2&dq=what+is+obd2&ots=O6pGMdTomB&sig=lgW3zghqwVGy8WGO2n2RwfFHOZU#v=onepage&q&f=false

Moniaga, J., Manalu, S., Hadipurnawan, D. y Sahidi, F. (28-30 de noviembre 2017). *Diagnostics vehicle's condition using OBD-II and Raspberry Pi technology: Study literature*. 2nd International Conference on Computing and Applied Informatics. Universitas Sumatera Utara, Medan, Indonesia. https://doi.org/10.1088/1742-6596/978/1/012011

O'Brien, O., Sumich, A., Kanjo, E. y Kuss, D. (2022). WiFi at University: A better balance between education activity and distraction activity needed. *Computers and Education Open, 3.* https://doi.org/10.1016/j.caeo.2021.100071

Pahlavan, K. y Krishnamurthy, P. (2021). Evolution and Impact of Wi-Fi Technology and Applications: A Historical Perspective. *International Journal of Wireless Information Networks, 28,* 3–19. https://doi.org/10.1007/s10776-020-00501-8

Rayo, O. y González, C. (2016). *Diseño e Implementación de una Interfaz Sobre C++ Para Control Inalámbrico de la Iluminación de un Edificio a través de Internet 2 Utilizando un Sistema Embebido*. [Tesis, Universidad Distrital Francisco José de Caldas]. https://repository.udistrital.edu.co/bitstream/handle/11349/2559/GonzalezCarlosEnrique2016.pdf?sequence=1&isAllowed=y

Severance, C. (2020). *Python para todos: Explorando la información con Python 3*. https://www.cartagena99.com/recursos/alumnos/temarios/211001163348-pythonlearn.pdf

SiliconLabs. (2021). *The Past, Present and Future of USB.* https://www.silabs.com/documents/public/white-papers/The-Past-Present-and-Future-of-USB.pdf

Sri Swathiga, U., Vinodhini, P. y Sasikala, V. (2021). An interpretation of dart programming language. *Dogo Rangsang Research Journal, 11* (10). https://www.researchgate.net/publication/358661479_AN_INTERPRETATION_OF_DART_PROGRAMMING_LANGUAGE

Srinivasa Rao, P., Pavan, B., Srivastava, A., Venkata Amani, K. y Sharma, A. (2022). Distinction of mobile frameworks: flutter vs native apps. *International Research Journal of Modernization in Engineering Technology and Science, 4* (6). https://www.irjmets.com/uploadedfiles/paper/issue_6_june_2022/26317/final/fin_irjmets1655531771.pdf

Wenhao, W. (2018). *React Native vs Flutter, cross-platform mobile application frameworks*. [Tesis, Metropolia University of Applied Sciences]. https://www.theseus.fi/bitstream/handle/10024/146232/thesis.pdf

Wi-Fi Alliance. (15 de febrero de 2024). *Wi-Fi Alliance® celebrates 25 years of Wi-Fi® innovation and impact*, Wi-Fi Alliance. https://www.wi-fi.org/news-events/newsroom/wi-fi-alliance-celebrates-25-years-of-wi-fi-innovation-and-impact

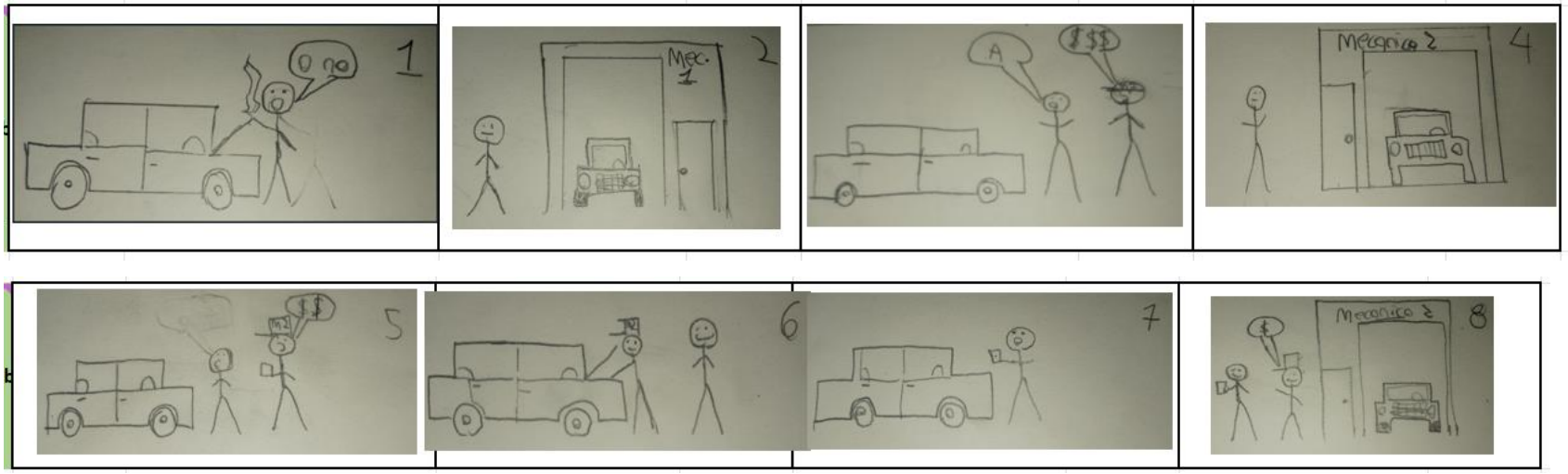
Zhao, C., Jegatheesan, J., y Loon, S. (2015). Exploring IOT application using Raspberry Pi. *International Journal of Computer Networks and Applications, 2* (1), 27-29. https://ijcna.org/Manuscripts/Volume-2/Issue-1/Vol-2-issue-1-M-04.pdf

Anexos

Anexo 1. Storyboard

Proceso para iniciar el Storyboard			
A) Producto	**¿Cuál es el producto o servicio que se venderá?**	**¿Para qué se utilizará?**	
	Un sistema de monitoreo para automóviles, compuesto por una aplicación web y una aplicación móvil. Utilizando un módulo ESP32 para recopilar y transmitir datos relevantes del automóvil a una base de datos	Monitorear en tiempo real diversos parámetros del automóvil, como la temperatura del refrigerante, la presión del aceite, la carga de la batería, la condición del aceite y el tiempo restante hasta el siguiente mantenimiento regular.	
B) Necesidades	**¿Cuál es el público objetivo?**	**¿Cuál es su nivel de habilidad?**	**¿Tienen conocimiento previo?**
	Propietarios de automóviles que buscan una solución para el monitoreo y mantenimiento preventivo de sus vehículos.	Los propietarios de automóviles que utilizarán este sistema no necesitan tener habilidades técnicas avanzadas en mecánica o en tecnología, ya que la aplicación está diseñada para ser intuitiva.	La aplicación está diseñada para proporcionar toda la información de manera clara y comprensible, por lo que no es necesario que los usuarios tengan un conocimiento técnico profundo
C) Materiales	**¿Tiene contenido disponible en este momento para empezar o lo tiene que recolectar?**	**Del que tiene... ¿Es todo importante o hay información secundaria que podría dejar por fuera?**	
	ESP32 Sensores y Periféricos Placa de Prototipado Cables y Jumpers Fuente de Alimentación Computadora y Cable USB Acceso a Internet Generador de imagenes de inteligencia artificial https://create.microsoft.com/es-es/features/ai-image-generator	Dependera del usuario, pero en general todo es util, ya con el producto final, el uso de la breadbord seria minimo o nulo, asi que puede que se elimine en el futuro	
D) Objetivos	**¿Cuál es el objetivo de aprendizaje?**	**¿Tienes objetivos secundarios?**	
	Desarrollar una aplicación web y móvil que permita a los usuarios monitorear en tiempo real el estado de sus vehículos, utilizando datos recopilados por un ESP32 y almacenados en Firebase. La solución proporcionará alertas y gráficas comprensibles para facilitar la toma de decisiones informadas sobre el mantenimiento preventivo de los automóviles.	Notificaciones en Tiempo Real Historial y Análisis de Datos Recordatorios de Mantenimiento Localización del Vehículo Análisis Predictivo	

E) Roles	Escritor	Dibujante	Diseñador
	Es el contructor de ideas, que no solo dibuja letras, es aquella persona que se conecta con todo lo que existe, lo siente, le da forma; para luego plasmarlo con la ayuda de las letras.	Es aquella persona que dibuja lo que no puede expresar con palabras: "tele transporta su imaginación y pensamientos" a un lienzo que puede ser de papel u otro material para transmitir una idea o sentimiento surgido de la subjetividad del escritor.	Es la persona que es el responsable del desarrollo del formato y diseño de las entregas del producto. son los responsables de dejar en claro al equipo los problemas o necesidades de los clientes, los cuales culminan con la producción o construcción de la solución.
	Sergio Alan Minjárez Gutiérrez	Abraham Camacho Ríos	Jesús Alejandro Borjon Zapata



TÍTULO

Auto-matizate

1. Información del guion			
Secuencia	1	Escena	1
Lugar	Mecanico 1	Página	1

2. Cuadro de dibujo				
3. Descripción	A un conductor se le descompone su auto	Tiene que llevarlo con el mecánico por las fallas.	Al hablar con el mecánico, este se sorprende por el precio.	El conductor acude a otro mecánico.
4. Narración	No todos somos unos expertos automotrices, tu auto puede estar a punto de fallar y tu no tienes manera de saberlo	Este decide llevarlo a un mecánico para poder saber qué sucede.	El conductor se ve sorprendido al saber el costo de la reparación de su carro.	Al ver el costo tan elevado, el conductor decide acudir a otro mecánico.
5. Observaciones	El conductor se muestra especialmente frustrado por las fallas de su auto.	El conductor decidió ir al primero que encontró.	El conductor se ve extrañado.	El conductor se ve más temeroso.

1. Información del guion			
Secuencia	2	Escena	2
Lugar	Mecanico 2	Página	2

2. Cuadro de dibujo				
3. Descripción	El conductor decide perguntarle al segundo mecánico como puede evitar que esta situacion se repita	El mecánico le recomienda prevenir fallos graves utilizando Auto-matic.	El conductor recibe una notificación.	El mecánico hará la reparación por menor precio.
4. Narración	Los buenos mecánicos saben como ayudar a sus clientes	Al ver las ventajas que Auto-matic ofrece, el conductor decide instalarlo junto a las reparaciones.	Mientras estaba en su día a día, el conductor recibió una notificación del estado del auto para ir a un chequeo.	Al haber podido acudir a tiempo, la falla fue leve y el costo mucho menor.
5. Observaciones	El conductor se nota frustrado por el poco control que tuvo sobre la situación de su auto.	El mecánico trabaja rápidamente frente al conductor.	La notificación puede ayudar a comprender qué tipo de desperfecto hay en el auto.	El mecánico cobrará mucho menos porque el carro no ha sufrido daños graves.

Instrucciones de uso de la herramienta de inteligencia artificial de Microsoft, Designer:

1- Plantear bien la situación

2- Definir todo lo que se desea ver en la imagen

3- Escribir las instrucciones en inglés de preferencia para evitar errores

Rectificación: en caso de que la imagen no sea lo esperado puedes usar estos pasos para obtener lo que buscas

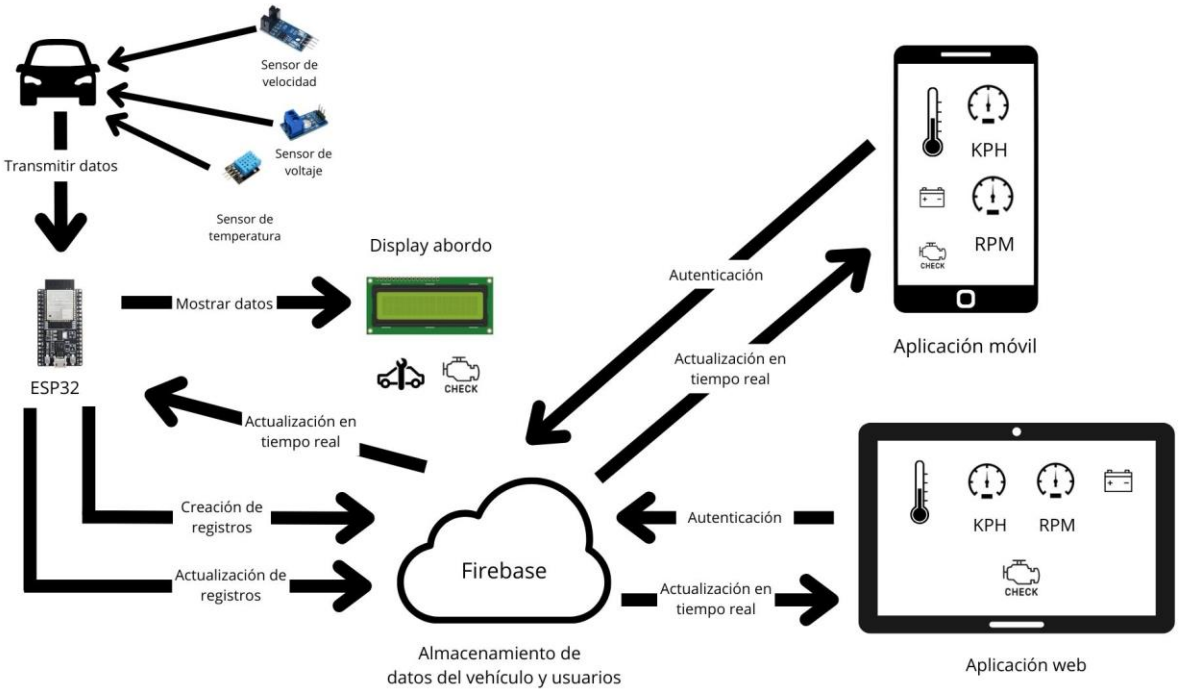
1- Cambiar palabras por sinónimos

2- Replantear la situación o el contexto de la imagen

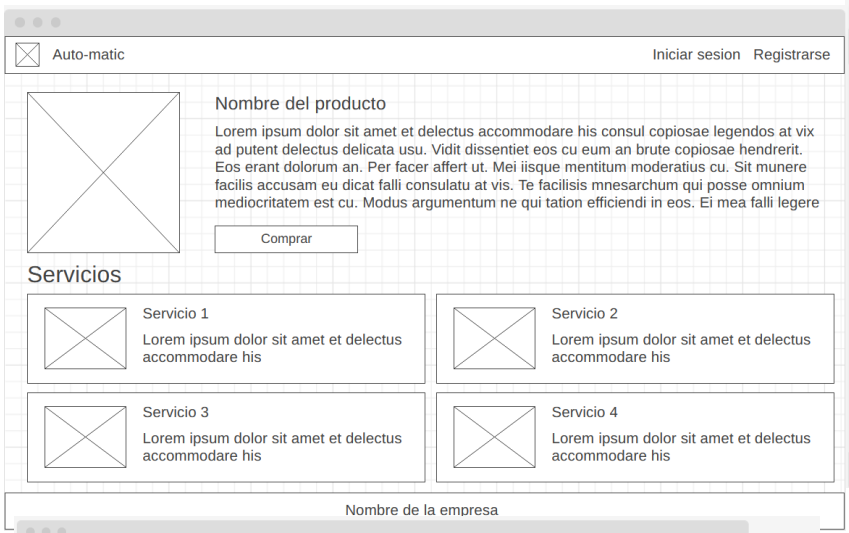
3- Volver a generarla, pues en ocasiones la IA cambia las imágenes levemente

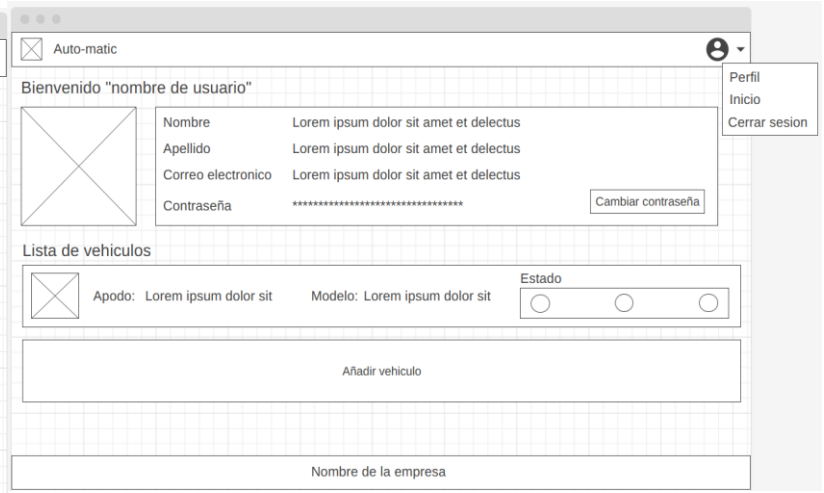
4- Pasarle una imagen y pedirle que realice cambios en esta

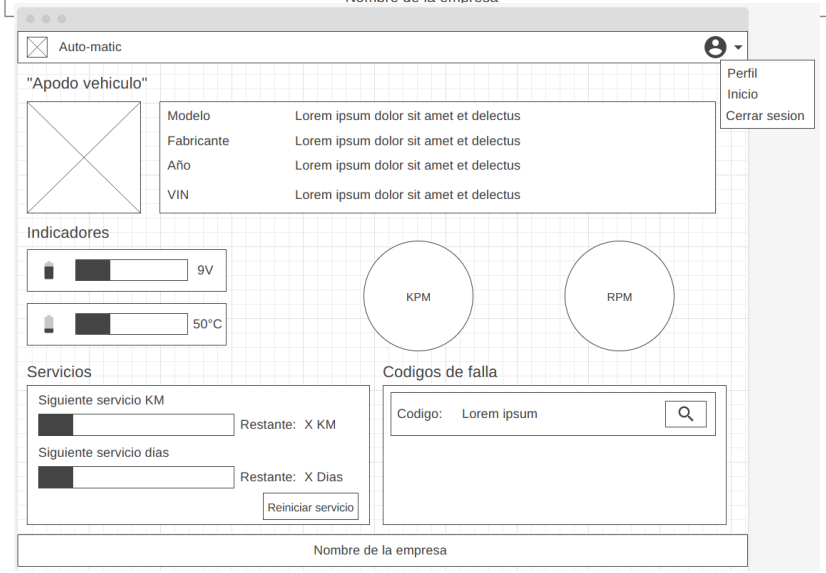
Anexo 2. Diagrama de funcionamiento

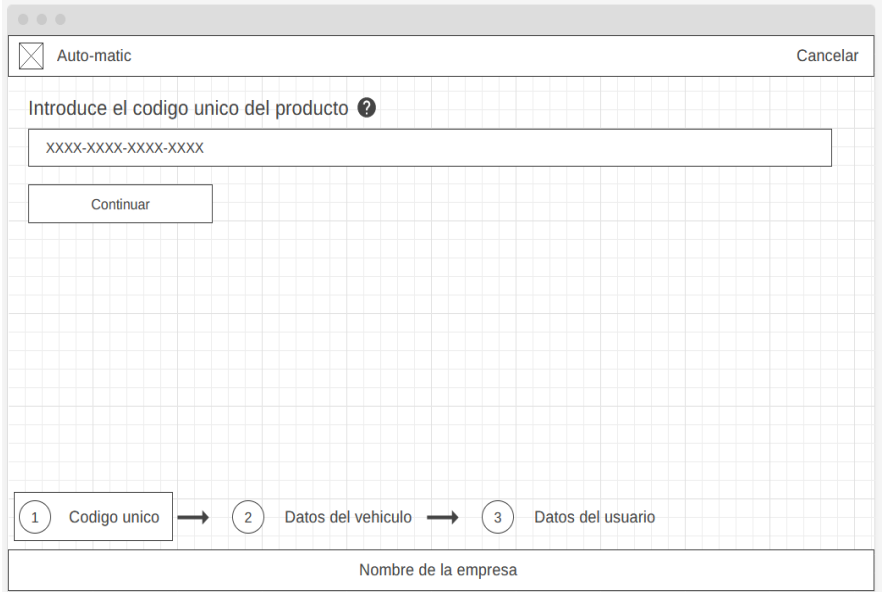


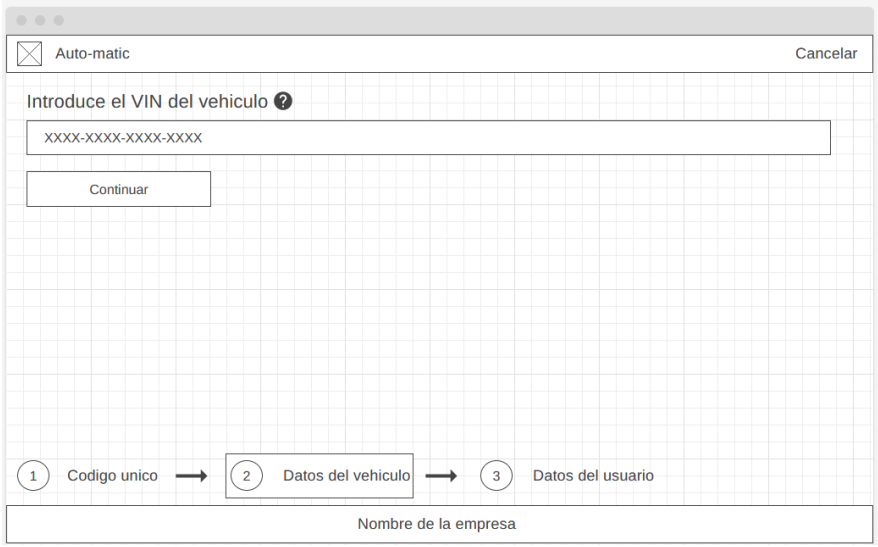
Anexo 3. Wireframe

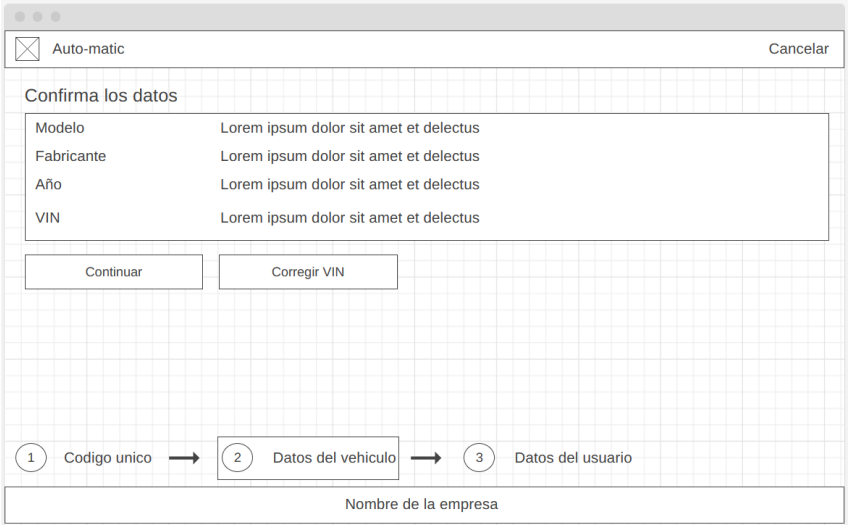


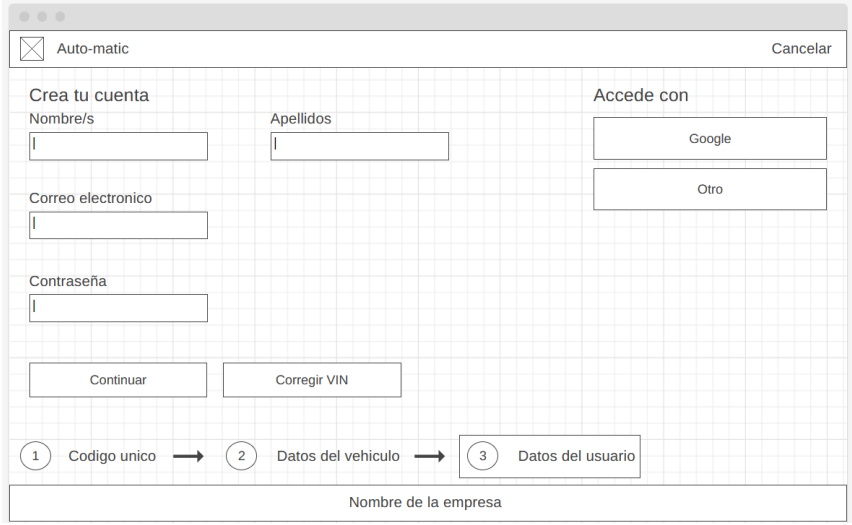




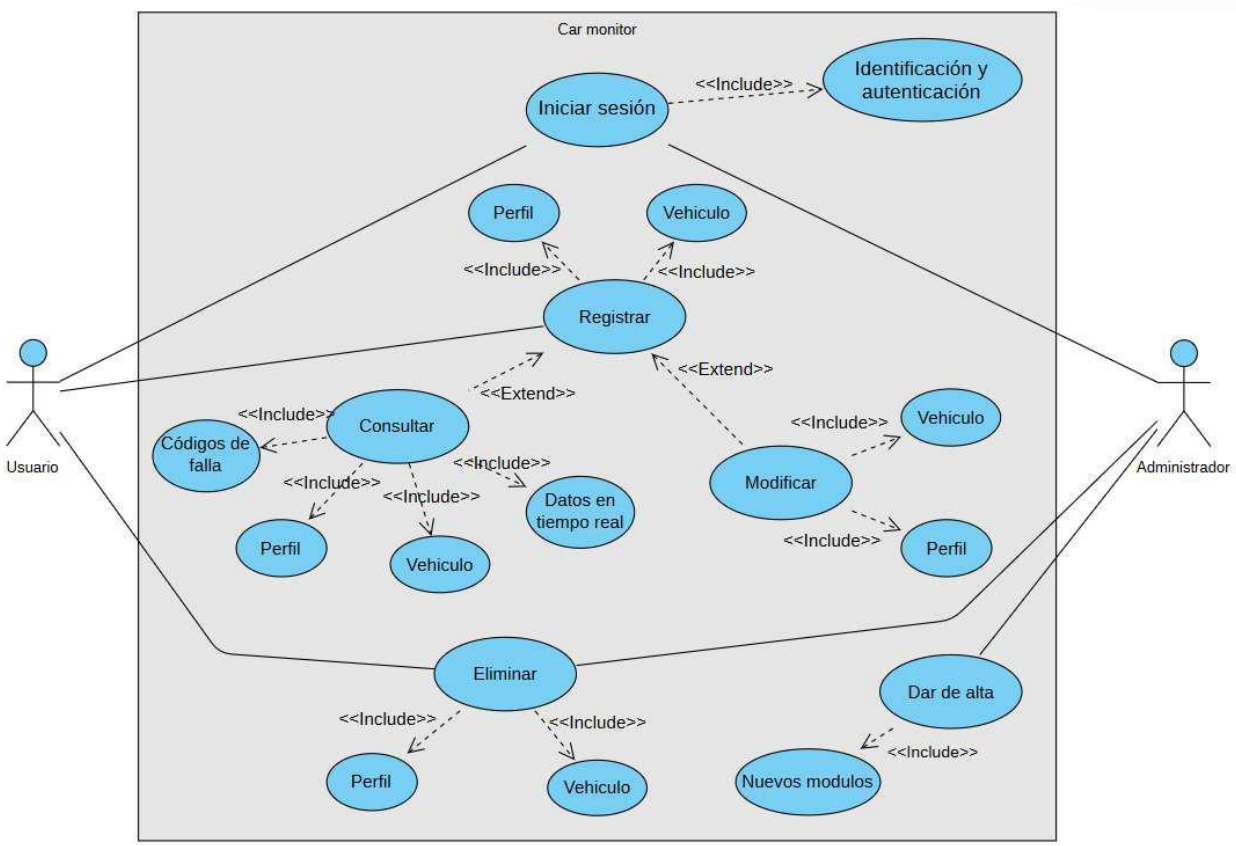



Auto-matic
Cancelar
Introduce el VIN del vehiculo
XXXX-XXXX-XXXX-XXXX
Continuar
1 Codigo unico
2 Datos del vehiculo
3 Datos del usuario
Nombre de la empresa

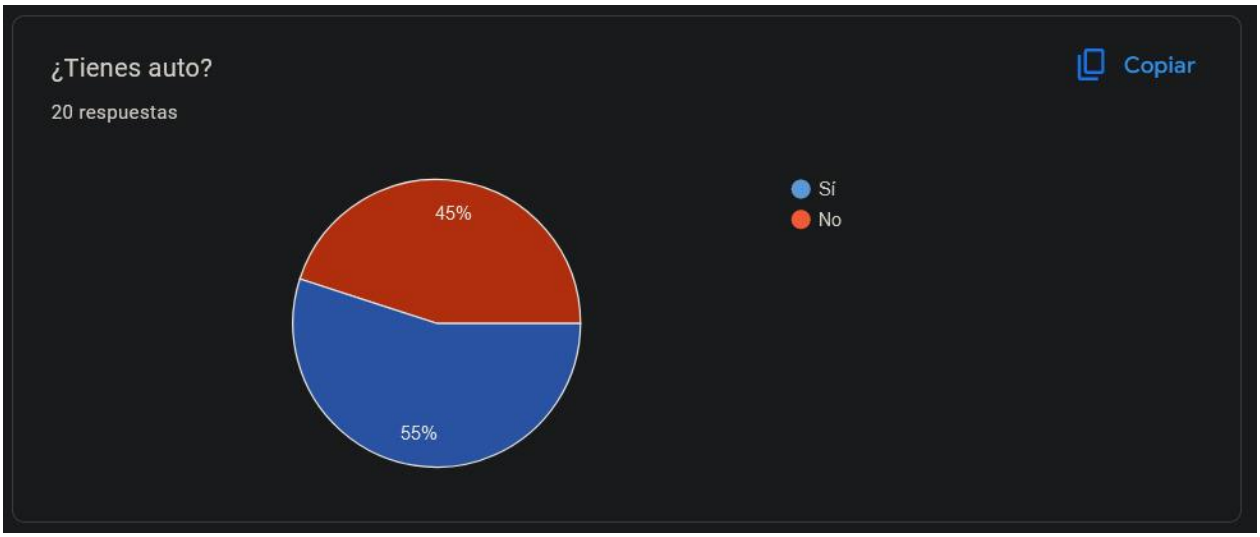

Auto-matic
Cancelar
Confirma los datos
Modelo Lorem ipsum dolor sit amet et delectus
Fabricante Lorem ipsum dolor sit amet et delectus
Año Lorem ipsum dolor sit amet et delectus
VIN Lorem ipsum dolor sit amet et delectus
Continuar
Corregir VIN
1 Codigo unico
2 Datos del vehiculo
3 Datos del usuario
Nombre de la empresa

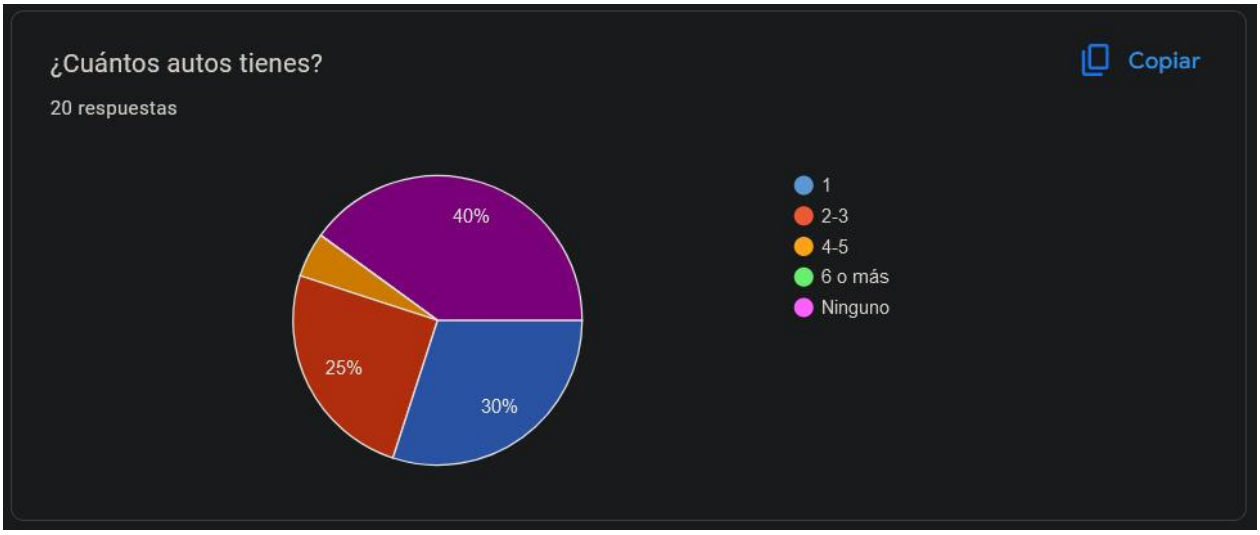

Auto-matic
Cancelar
Crea tu cuenta
Nombre/s
Apellidos
Correo electronico
Contraseña
Accede con
Google
Otro
Continuar
Corregir VIN
1 Codigo unico
2 Datos del vehiculo
3 Datos del usuario
Nombre de la empresa

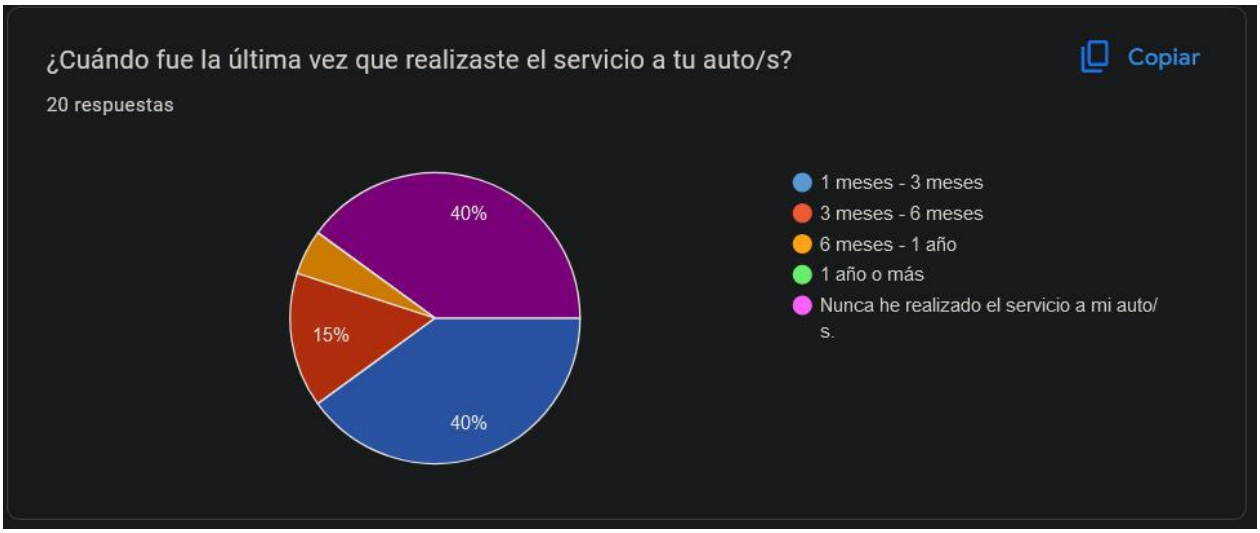
Anexo 4. Diagrama de caso de uso

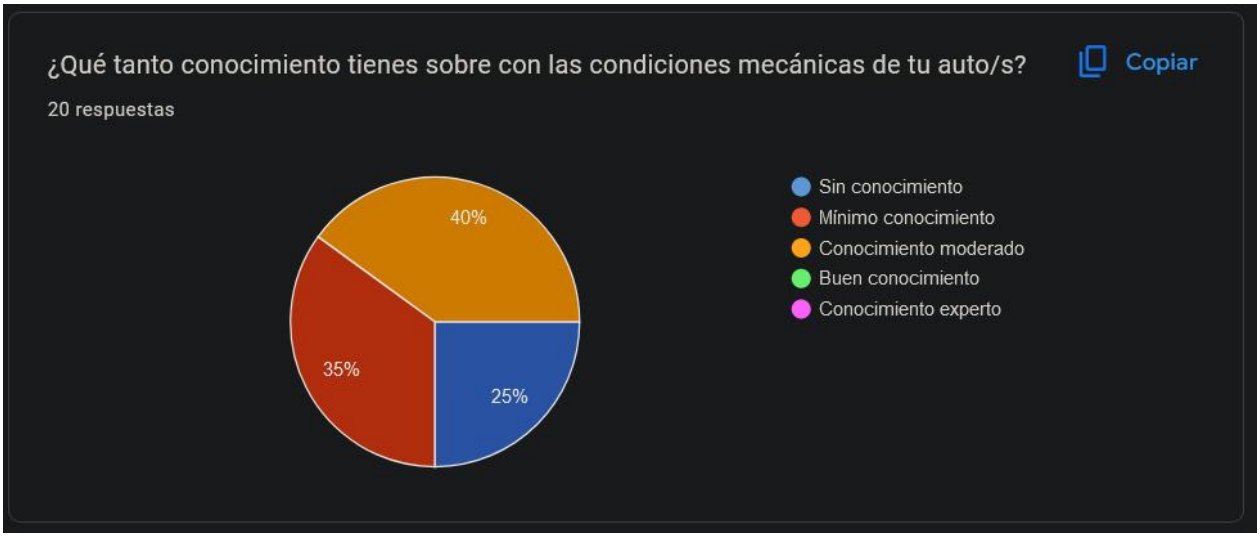


Anexo 5. Encuesta





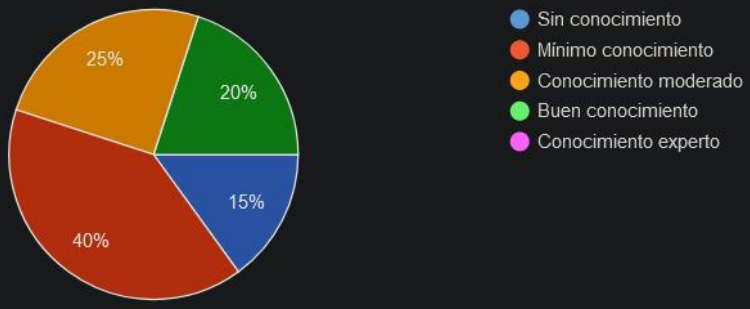

¿Cuándo fue la última vez que realizaste el servicio a tu auto/s?
Copiar
20 respuestas
40%
15%
40%
1 meses - 3 meses
3 meses - 6 meses
6 meses - 1 año
1 año o más
Nunca he realizado el servicio a mi auto/s.


¿Qué tanto conocimiento tienes sobre con las condiciones mecánicas de tu auto/s?
Copiar
20 respuestas
40%
35%
25%
Sin conocimiento
Mínimo conocimiento
Conocimiento moderado
Buen conocimiento
Conocimiento experto

¿Cuál es tu nivel de conocimiento en cuanto a los indicadores en el tablero de tu automóvil?

Copiar

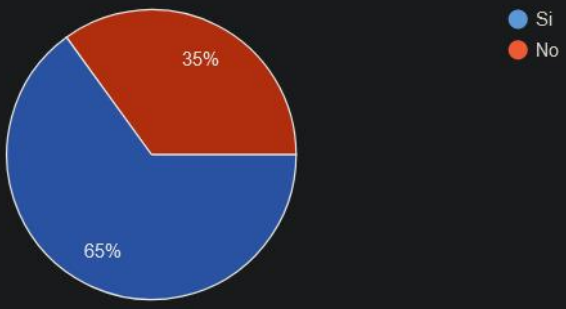
20 respuestas



¿Sabes cuál es la frecuencia con la que debe de realizarse los cambios de aceite en tu automóvil?

Copiar

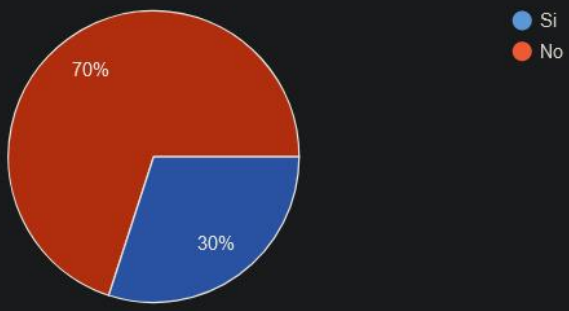
20 respuestas



¿Sabes cuál es la frecuencia con la que debe de remplazarse el filtro de aire en tu automóvil?

Copiar

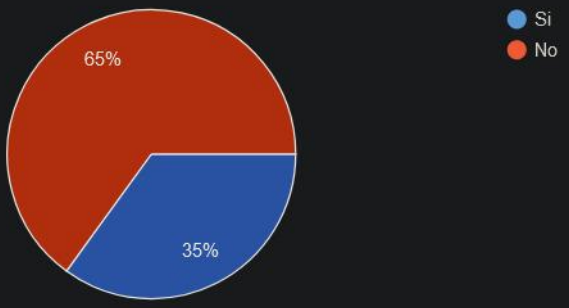
20 respuestas

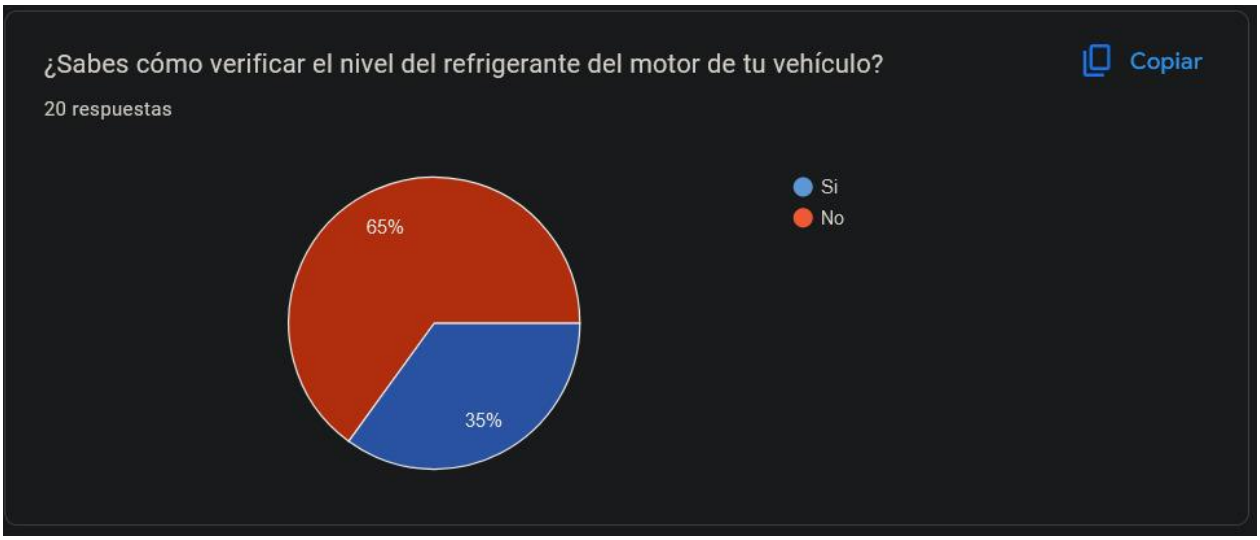


¿Sabes cuál es la frecuencia con la que debe de remplazarse las bujías en tu vehículo?

Copiar

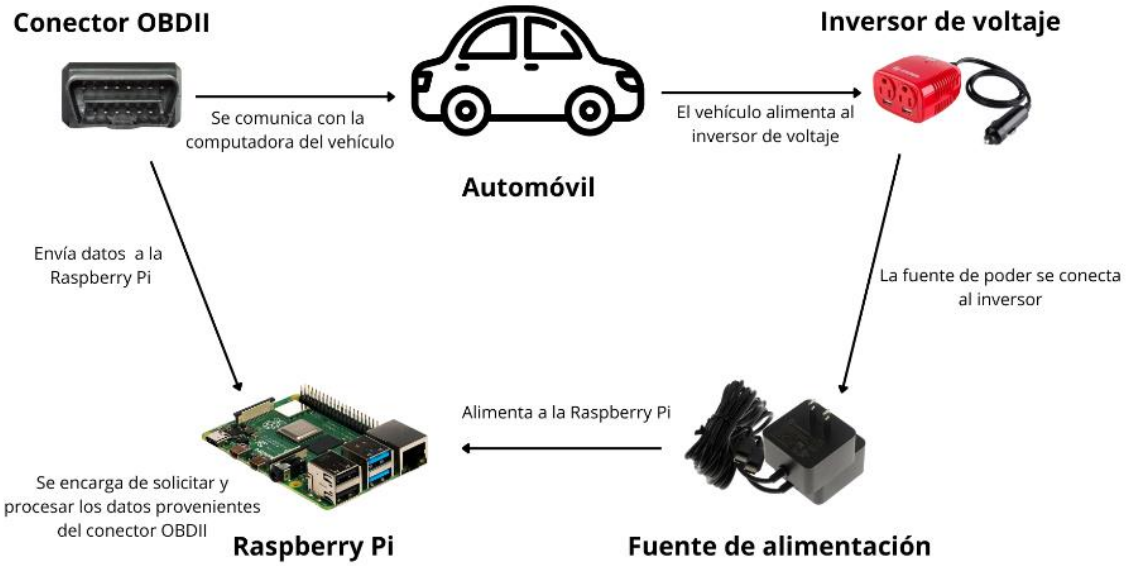
20 respuestas



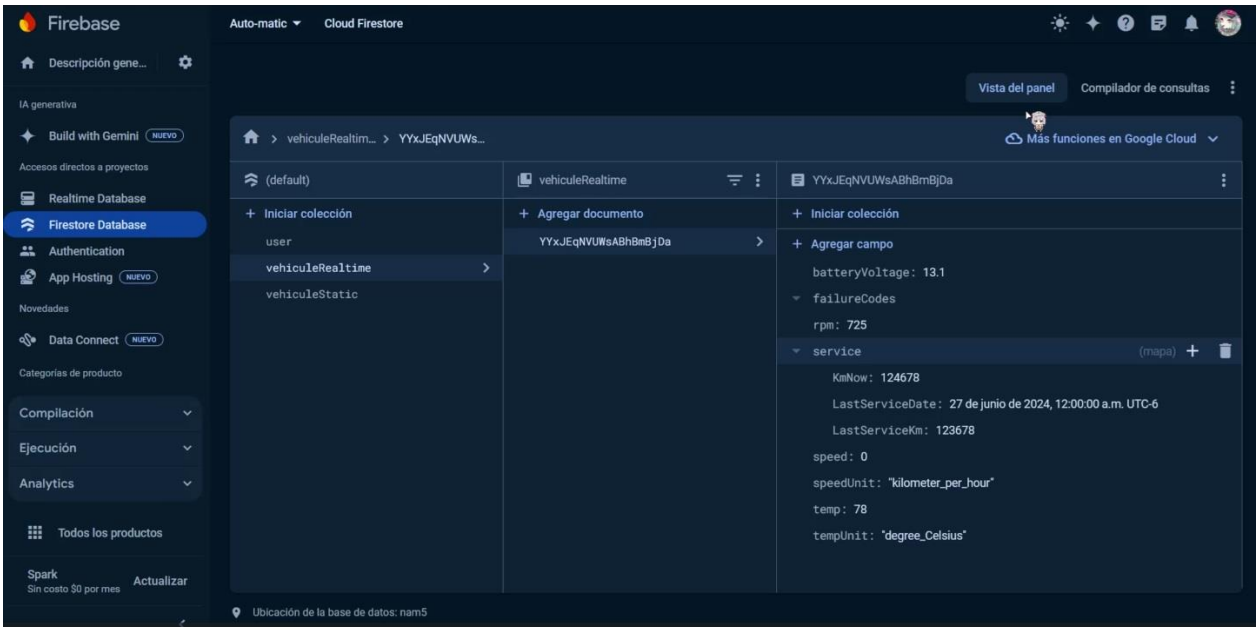

¿Sabes cómo verificar el nivel del refrigerante del motor de tu vehículo?
Copiar
20 respuestas
Si
No
65%
35%

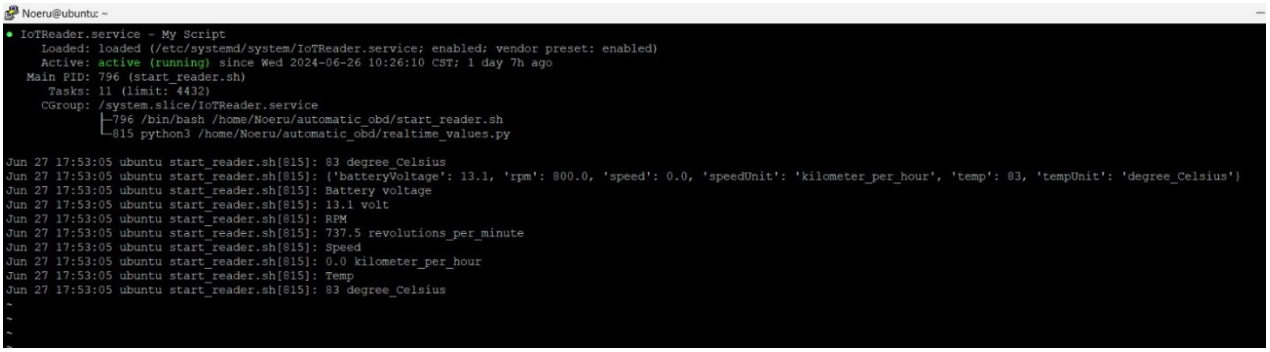

¿Conoces el procedimiento para verificar el estado de la batería de tu coche?
Copiar
20 respuestas
Si
No
90%
10%

Anexo 6. Conexiones entre componentes

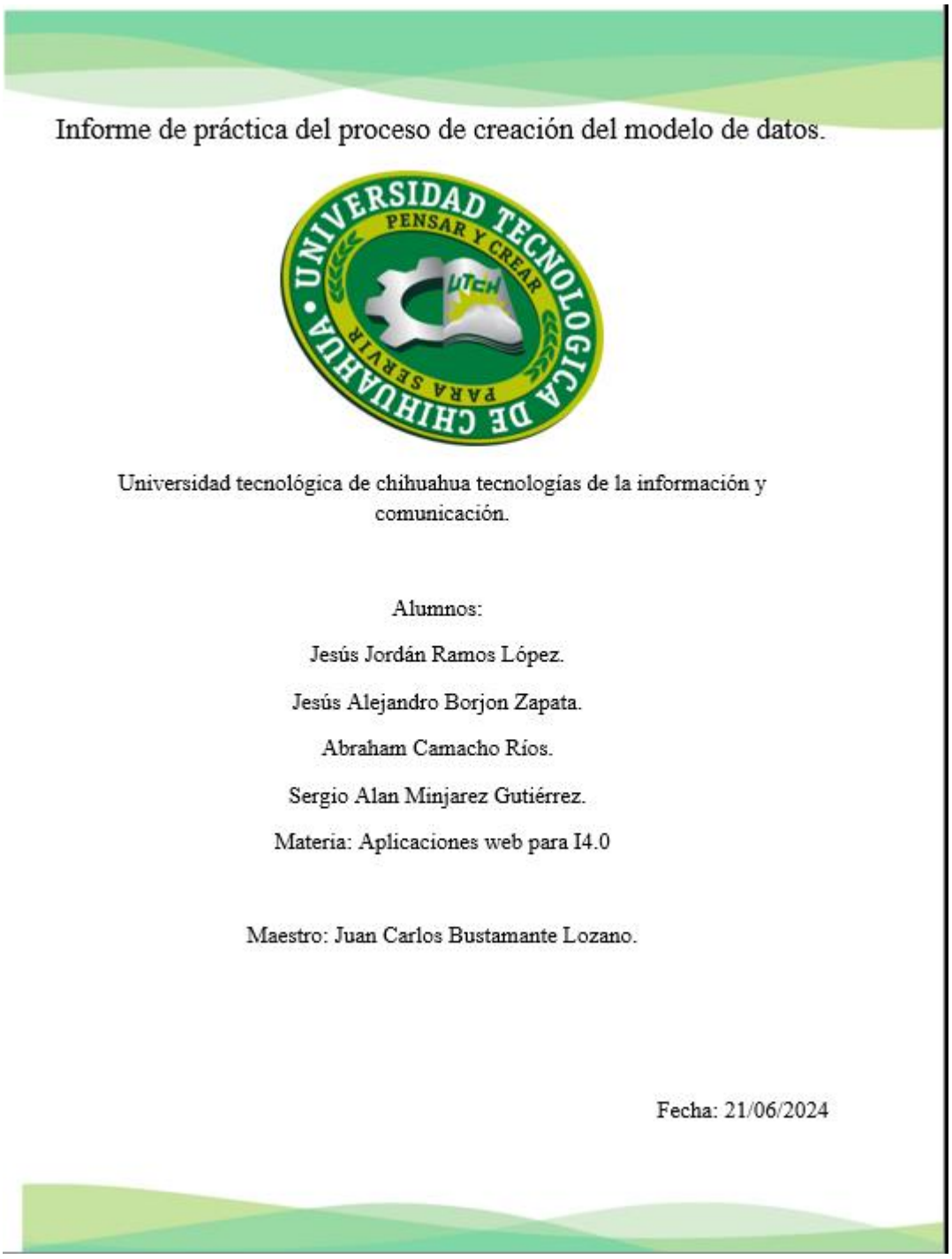


Anexo 7. Dispositivo ensamblado y datos obtenidos


Firebase
Auto-matic
Cloud Firestore
Descripción gene...
IA generativa
Build with Gemini
NUEVO
Accesos directos a proyectos
Realtime Database
Firestore Database
Authentication
App Hosting
NUEVO
Novedades
Data Connect
NUEVO
Categorías de producto
Compilación
Ejecución
Analytics
Todos los productos
Spark
Sin costo $0 por mes
Actualizar
Vista del panel
Compilador de consultas
vehiculeRealtim...
YYxJEqNVUWs...
Más funciones en Google Cloud
(default)
vehiculeRealtime
YYxJEqNVUWsABhBmBjDa
Iniciar colección
user
vehiculeRealtime
vehiculeStatic
Agregar documento
YYxJEqNVUWsABhBmBjDa
Iniciar colección
Agregar campo
batteryVoltage: 13.1
failureCodes
rpm: 725
service
(mapa)
KmNow: 124678
LastServiceDate: 27 de junio de 2024, 12:00:00 a.m. UTC-6
LastServiceKm: 123678
speed: 0
speedUnit: "kilometer_per_hour"
temp: 78
tempUnit: "degree_Celsius"
Ubicación de la base de datos: nam5


Noeru@ubuntu: ~
● IoTReader.service - My Script
Loaded: loaded (/etc/systemd/system/IoTReader.service; enabled; vendor preset: enabled)
Active: active (running) since Wed 2024-06-26 10:26:10 CST; 1 day 7h ago
Main PID: 796 (start_reader.sh)
Tasks: 11 (limit: 4432)
CGroup: /system.slice/IoTReader.service
├─796 /bin/bash /home/Noeru/automatic_obd/start_reader.sh
└─815 python3 /home/Noeru/automatic_obd/realtime_values.py
Jun 27 17:53:05 ubuntu start_reader.sh[815]: 83 degree_Celsius
Jun 27 17:53:05 ubuntu start_reader.sh[815]: {'batteryVoltage': 13.1, 'rpm': 800.0, 'speed': 0.0, 'speedUnit': 'kilometer_per_hour', 'temp': 83, 'tempUnit': 'degree_Celsius'}
Jun 27 17:53:05 ubuntu start_reader.sh[815]: Battery voltage
Jun 27 17:53:05 ubuntu start_reader.sh[815]: 13.1 volt
Jun 27 17:53:05 ubuntu start_reader.sh[815]: RPM
Jun 27 17:53:05 ubuntu start_reader.sh[815]: 737.5 revolutions_per_minute
Jun 27 17:53:05 ubuntu start_reader.sh[815]: Speed
Jun 27 17:53:05 ubuntu start_reader.sh[815]: 0.0 kilometer_per_hour
Jun 27 17:53:05 ubuntu start_reader.sh[815]: Temp
Jun 27 17:53:05 ubuntu start_reader.sh[815]: 83 degree_Celsius

Anexo 8. Dispositivo ensamblado y datos obtenidos



Informe de práctica del proceso de creación del modelo de datos.

Universidad tecnológica de chihuahua tecnologías de la información y comunicación.

Alumnos:

Jesús Jordán Ramos López.

Jesús Alejandro Borjon Zapata.

Abraham Camacho Ríos.

Sergio Alan Minjarez Gutiérrez.

Materia: Aplicaciones web para I4.0

Maestro: Juan Carlos Bustamante Lozano.

Fecha: 21/06/2024

Proceso de creación del modelo de datos.

Colecciones.

Las colecciones usadas en nuestro proyecto integrador son las siguientes:

- VehiculeStatic

Contiene los datos que nunca varían referentes al vehículo.

Campos

- VIN (String): Vehicle Identification Number (VIN), es una serie de dígitos que identifican a cada automóvil, permite diferenciarlos entre sí, pero más importante aún, permitirá obtener los datos que se mencionarán a continuación.
- Model (String): Contiene el modelo del auto registrado por el usuario. Hicimos uso de este campo por ser información básica del automóvil que permite diferenciarlo de otros que pueda tener el usuario.
- Make (String): Este campo almacena el fabricante del auto registrado por el usuario. Hacemos uso de este campo por la misma razón que el anterior campo.
- Year (String): Almacena el año de fabricación del auto registrado por el usuario. Al igual que los dos anteriores, se trata de información básica que permite diferenciar entre vehículos.
- Realtime (Reference): Almacena la referencia a los datos en tiempo real del automóvil. Usamos este campo para tener un acceso fácil a los datos mediante un mecanismo ya incluido en Firebase.

- Alias (String): Contiene un alias con el que el usuario puede identificar fácilmente su vehículo.
- userUID (String): Gurda la ID única del usuario proporcionada por Firebase para relacionar el automóvil con su dueño.

- VehiculeRealtime

Contiene los datos provenientes de los sensores del automóvil.

Campos

- BateryVoltage (Number): Almacena la información del sensor de voltaje del vehículo.
- Temp (Number): Almacena la información del sensor de temperatura del vehículo.
- KPM (Number): Almacena la velocidad actual del vehículo.
- RPM (Number): Almacena las revoluciones por minuto actuales del vehículo.
- FailureCodes (String List): Almacena una lista de códigos de fallas provenientes de las computadora del vehículo.
- LastServiceDate: Almacena la fecha de realización del último servicio.

Proceso de creación de la base de datos.

1- Acceder al Firebase console.

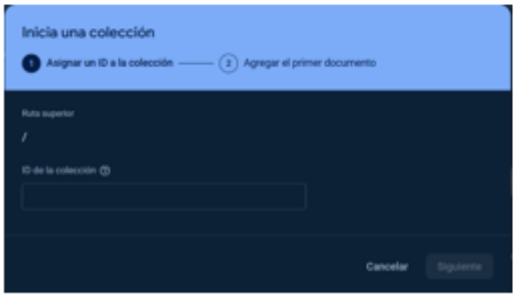
2- Dar clic en compilación > Firestore Database

3- Dar clic en crear base de datos.

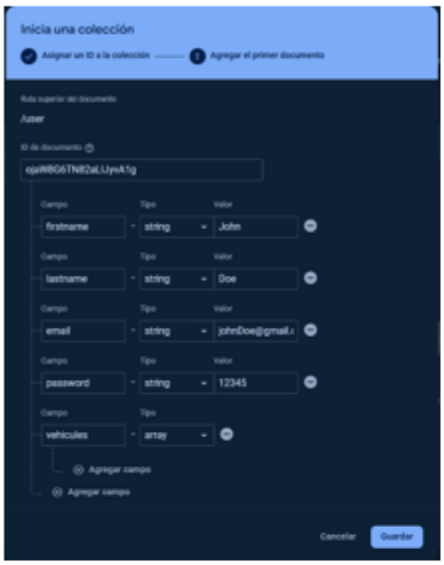
4- Seguimos los pasos que se nos presentan seleccionando la region a usar y las reglas de seguridad.

5- Una vez creada, aparecerá esta pantalla, hacemos clic en iniciar colección.

6- Para comenzar asignamos un nombre a la colección.



7- Después, agregamos los campos para el primer documento de la colección, seleccionando la opción de ID automático.



8- Repetimos el paso 6 y 7 para el resto de las colecciones para terminar con una pantalla como la siguiente.

Glosario

Actuadores: parte de un dispositivo o máquina que le ayuda a realizar movimientos físicos convirtiendo energía, a menudo eléctrica, neumática o hidráulica, en fuerza mecánica.

Aplicaciones multiplataforma: aplicaciones caracterizadas por ser desarrolladas bajo un único lenguaje de programación que les permite ejecutarse en cualquier dispositivo sin importar el sistema operativo.

Application Programming Interface (API): pieza de código que permite a diferentes aplicaciones comunicarse entre sí y compartir información y funcionalidades. Una API es un intermediario entre dos sistemas, que permite que una aplicación se comunique con otra y pida datos o acciones específicas.

Ataques de canal lateral: tipo de ataque cibernético en el que un atacante obtiene información confidencial de un sistema mediante el análisis de características indirectas, en lugar de explotar directamente las vulnerabilidades del sistema en sí.

Autenticación: proceso para confirmar que solo las personas, servicios y aplicaciones adecuados con los permisos correctos pueden acceder a recursos de un sistema informático.

Backend: parte lógica de un sitio, este se encarga de la lógica de negocio, de recibir y devolver datos procesados a las apps y sitios web, de forma que facilite la navegación y se garantice el funcionamiento y la seguridad de diferentes funciones.

Base de datos NoSQL: enfoque para el diseño de bases de datos que permite el almacenamiento y la consulta de datos fuera de las estructuras tradicionales que se encuentran en las bases de datos relacionales.

Bus: sistema digital que transfiere datos entre los componentes de una computadora. Está formado por cables o pistas en un circuito impreso, dispositivos como resistores y condensadores, además de circuitos integrados.

Computadora de una sola placa (SBC): computadora completa en una sola tarjeta o placa de circuito impreso. El diseño se centra en un solo microprocesador y todas las demás características de tamaño reducido.

Controladores: es un programa informático que permite al sistema operativo interactuar con un periférico, haciendo una abstracción del hardware y proporcionando una interfaz para utilizar el dispositivo.

Firebase: plataforma para el desarrollo de aplicaciones web y aplicaciones móviles en la nube que usa un conjunto de herramientas para la creación y sincronización de proyectos que serán dotados de alta calidad.

Flutter: kit de desarrollo de software para interfaces de usuario de código abierto creado por Google. Se usa para desarrollar aplicaciones multiplataforma desde una sola base de código.

Framework: conjunto de reglas y convenciones que se usan para desarrollar software de manera más eficiente y rápida. Estos marcos de trabajo se emplean para ahorrar tiempo y esfuerzo en el desarrollo de aplicaciones.

Modulación por ancho de pulso: técnica en la que se modifica el ciclo de trabajo de una señal periódica, ya sea para transmitir información a través de un canal de comunicaciones o para controlar la cantidad de energía que se envía a una carga.

On-Board Diagnostics (OBD): sistema de diagnóstico a bordo en vehículos que aporta un monitoreo y control completo del motor y otros dispositivos del vehículo. Proporciona el capacidad de autodiagnóstico e información para los técnicos de reparación.

Periféricos: herramientas de hardware que sirven como interface entre el usuario y la computadora. Tienen la finalidad básica de satisfacer algún requerimiento, ya sea introducir, obtener o almacenar información.

Pines de Entrada/Salida de Propósito Genera (GPIO): pin genérico en un chip, cuyo comportamiento se puede controlar (programar) por el usuario en tiempo de ejecución.

Python: lenguaje de alto nivel de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código. Se trata de un lenguaje de programación multiparadigma

Base de datos: recopilación de datos sistemática y almacenada electrónicamente.

Raspberry Pi: serie de ordenadores monoplaca u ordenadores de placa simple de bajo costo desarrollado en el Reino Unido por la Raspberry Pi Foundation

Sensores: todo aquello que tiene una propiedad sensible a una magnitud del medio y al variar esta magnitud también varía con cierta intensidad la propiedad, es decir, manifiesta la presencia de dicha magnitud y también su medida.

Señal analógica: señal generada por algún tipo de fenómeno electromagnético; que es representable por una función matemática en forma de onda.

Señal digital: es aquella señal que sólo puede proporcionar dos estados lógicos (ALTO y BAJO), o 0 y 1 desde el punto de vista digital.

Sistemas embebidos: sistema de computación basado en un microprocesador o un microcontrolador diseñado para realizar una o algunas pocas funciones dedicadas, frecuentemente en un sistema de computación en tiempo real.

Sockets: Los sockets son canales de comunicación que permiten que procesos no relacionados intercambien datos localmente y entre redes. Un único socket es un punto final de un canal de comunicación bidireccional.

Unidad central de procesamiento (CPU): componente de hardware presente en ordenadores encargado de interpretar las instrucciones de un programa informático mediante la realización de las operaciones básicas aritméticas, lógicas y externas.

Unidad de procesamiento grafico (GPU): circuito electrónico diseñado para acelerar el procesamiento por computadora de gráficos e imágenes en una variedad de dispositivos, para aligerar la carga de trabajo del procesador/CPU.

Universal Serial Bus (USB): bus de comunicaciones que sigue un estándar que define los cables, conectores y protocolos usados en un bus para conectar, comunicar y proveer de alimentación eléctrica entre dispositivos.

Vulnerabilidad: cualquier fallo o error en el software o en el hardware que hace posible a un atacante o hacker comprometer la integridad y confidencialidad de los datos que procesa un sistema.

Wireless Fidelity (Wifi): tecnología de redes inalámbricas que permite a los dispositivos electrónicos conectarse entre sí de manera fluida a una red mediante frecuencias de radio.