Documentación TFG

Motivación:

Nuestra principal motivación es la falta de aplicaciones para talleres mecánicos en el mercado, lo que dificulta la gestión de los vehículos al momento de llevarlos a reparar. Con nuestra app, modernizamos el sector, dejando atrás el modelo tradicional y ofreciendo una solución más eficiente y actual. Además, brindamos mayor comodidad a los usuarios.

En los talleres mecánicos tradicionales, la gestión de clientes, vehículos, reparaciones, citas y facturación se realiza de forma manual o con herramientas poco eficientes, lo que genera una gran carga de trabajo. Además, los clientes a menudo deben desplazarse al taller para gestionar sus citas o consultar el estado de sus reparaciones, lo que no es cómodo ni práctico.

Diseño Mockup (WIP):

Link al Mockup

Objetivos propuestos (generales, específicos):

Objetivo General:

Desarrollar una aplicación móvil para la gestión de citas y reparaciones en un taller mecánico, permitiendo a los clientes programar servicios y comunicarse con los mecánicos de manera eficiente.

Objetivos Específicos:

1. **Implementar un sistema de gestión de citas** que permita a los clientes reservar horarios disponibles en un calendario, evitando conflictos.

2. **Automatizar la creación de reparaciones** para los mecánicos.

3. **Incorporar un sistema de chat** entre cliente y mecánico, disponible solo durante la reparación, para facilitar la comunicación.

4. **Proporcionar un historial de reparaciones y facturación** accesible para los clientes, con detalles sobre trabajos realizados y costos.

5. **Desarrollar una interfaz intuitiva y accesible**, asegurando una experiencia de usuario clara tanto para clientes como para mecánicos.

6. **Permitir la visualización de facturas y la simulación de pagos**, (sin integrar pasarelas de pago reales)

7. **Ofrecer una gestión de vehículos para los clientes**, permitiendo registrar y visualizar los automóviles que llevarán al taller.

Metodología utilizada:

Para el desarrollo de esta aplicación, utilizaremos la metodología Scrum. Un marco ágil que permite una gestión eficiente del proyecto. Dividiremos el trabajo en sprints y realizaremos reuniones de seguimiento para evaluar el progreso.

Planificación y Diseño:

Se definirán los requisitos principales y se diseñarán los MockUps en Figma para visualizar la interfaz de usuario.

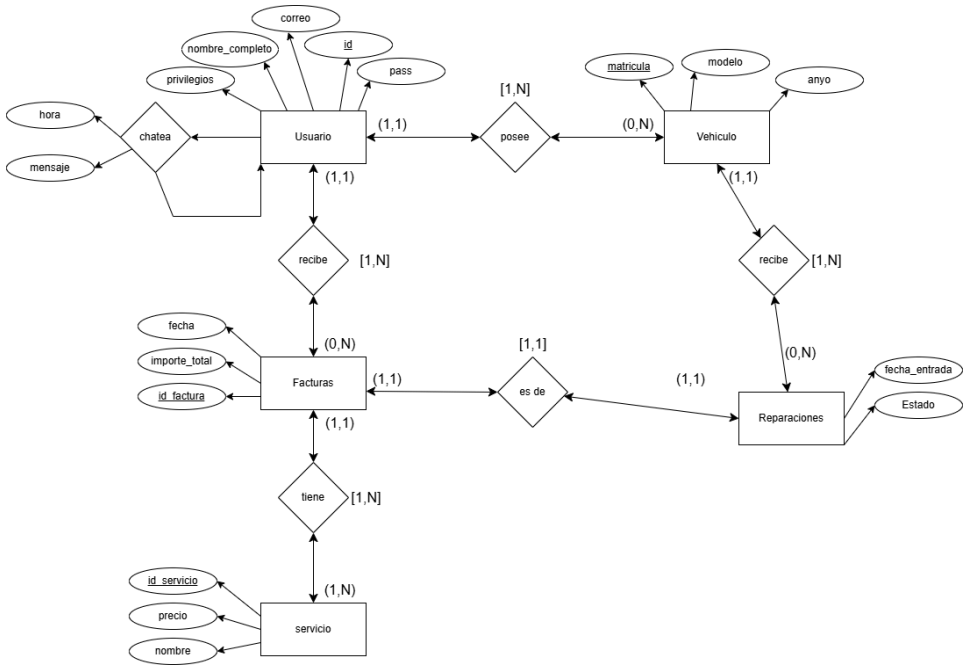
Tecnologías y herramientas utilizadas en el proyecto :

Para el desarrollo de la aplicación del taller mecánico, utilizaremos las siguientes tecnologías y herramientas:

Figma: Para diseñar los MockUps y la interfaz de usuario.
Android Studio: Entorno de desarrollo para la aplicación móvil.
Draw.io: Para la creación del diagrama de casos de uso.
Kotlin y Java: Lenguajes de programación para la lógica de la aplicación.
XML: Para la estructura y diseño de la interfaz gráfica.
Spring Boot: Framework para el desarrollo del backend y la gestión de WebSockets.
Hibernate: Para la gestión de la base de datos en el backend.
MySQL: Sistema de gestión de bases de datos relacional para almacenar los mensajes, usuarios, y otros datos de la aplicación.
Retrofit: Para la comunicación entre la aplicación móvil y el backend mediante API REST.
WebSockets: Para la implementación del chat en tiempo real entre clientes y mecánicos.

Análisis:

Entidad-Relación(WIP):



Requisitos del sistema(WIP)

Requisitos Funcionales

1. La app debe diferenciar entre los roles de cliente y mecánico, asignando permisos según cada perfil.

2. Registro e inicio de sesión de clientes con correo electrónico y contraseña.

3. Registro e inicio de sesión de mecánicos con acceso a funciones administrativas.

4. Registro de vehículos por parte de los clientes, incluyendo marca, modelo, matrícula y año.

5. Visualización y administración de los vehículos registrados en la cuenta del usuario.

6. Creación automática de una reparación al agendar una cita.

7. Consulta del historial de reparaciones con detalles como fecha, tipo de servicio y costos.

8. Seguimiento del estado de la reparación en curso (en espera, en curso, finalizado).

9. Notificaciones sobre actualizaciones en el estado de la reparación.

10. Solicitud de citas en línea seleccionando fecha, hora y tipo de servicio.

11. Visualización de un calendario con fechas y horarios disponibles.

12. Restricción para evitar que dos clientes reserven la misma cita.

13. Generación automática de facturas electrónicas tras completar una reparación.

14. Consulta de facturas anteriores desde la aplicación.

15. Simulación de pago sin integración de pasarelas reales.

16. Chat en tiempo real con el mecánico disponible solo durante la reparación.

17. Un chat exclusivo por cada reparación en curso.

18. Envío de mensajes y notificaciones entre cliente y taller.

Requisitos No Funcionales

19. Interfaz intuitiva y fácil de navegar para clientes y mecánicos.
20. Diseño responsivo y adaptado para diferentes tamaños de pantalla.
21. Código modular y bien documentado para facilitar futuras mejoras.
22. Soporte para dispositivos Android desde la versión 8.0 en adelante.
23. Optimización de consultas a la base de datos para mejorar la velocidad de respuesta.
24. Respuesta rápida en la carga de datos y navegación fluida.
25. La aplicación debe operar en dispositivos Android a partir de la versión 8.0.

26. Se requiere conexión a Internet para gestionar citas, visualizar reparaciones y usar el chat.

Diagrama de Gantt:

(rodri): Tengo que hablar con Martin para implementar un mejor backlog

Diagrama de Gantt