Plan de Proyecto

TT2025

https://github.com/dfleper/

18/12/2025

El Plan de Proyecto de TT2025 define la organización inicial del trabajo: objetivos, fases, tareas, plazos, recursos y metodología prevista para desarrollar la aplicación. Establece la hoja de ruta que guió el arranque del proyecto y su planificación general.

Contenido

Turbo Taller 2025

Nombre del proyecto:

Turbo Taller – Sistema web de gestión de citas para taller de mecánica rápida

Código del proyecto *(APCODE):*

TT2025

Integrantes del equipo:

dfleper (Domingo Fleitas) · GitHub

Fecha y versión del documento:

18/12/2025 – Versión 1.0

1. Resumen ejecutivo

El proyecto **Turbo Taller (TT2025)** consiste en el desarrollo de una aplicación web que permita a los clientes de un taller de mecánica rápida reservar, consultar y gestionar citas de forma online, desde cualquier dispositivo. Al mismo tiempo, el sistema facilitará al personal del taller la planificación de la carga de trabajo, la asignación de recursos y el seguimiento de los servicios realizados.

En la actualidad, muchos talleres pequeños siguen gestionando las citas mediante llamadas telefónicas o agendas en papel, lo que provoca errores de planificación, solapamiento de citas y una mala experiencia para el cliente. **Turbo Taller** pretende digitalizar este proceso, ofreciendo una herramienta sencilla, accesible y centralizada, que mejore la eficiencia operativa del taller y la satisfacción del cliente.

El proyecto se orienta tanto al **público general** que requiere servicios de mantenimiento o reparación rápida de sus vehículos, como al **personal del taller** (recepción, mecánicos, responsable de taller) que necesita organizar su trabajo diario. El resultado esperado es una aplicación web funcional, segura y escalable, lista para desplegarse en un entorno real de producción o en un entorno académico que lo simule.

2. Alcance del proyecto

2.1 Objetivo general

Diseñar y desarrollar una aplicación web que permita **gestionar de forma integral las citas de un taller de mecánica rápida**, ofreciendo a los clientes un sistema de reserva online y al personal del taller una herramienta para la planificación y seguimiento de los servicios.

2.2 Objetivos específicos

a) Permitir que los clientes se registren, inicien sesión y soliciten citas indicando vehículo, servicio y franja horaria deseada.

b) Proporcionar al personal del taller un panel de administración para gestionar citas, clientes, vehículos y estados de los trabajos.

c) Evitar solapamientos de citas mediante un sistema de validación de disponibilidad de horarios y recursos.

d) Generar un historial de servicios por vehículo y por cliente, que permita consultar intervenciones anteriores.

e) Facilitar la comunicación con el cliente mediante confirmaciones y recordatorios (por email u otro canal configurable).

f) Definir una arquitectura técnica clara y una base de datos estructurada que permitan mantener y ampliar el sistema en el futuro.

2.3Límites del proyecto (qué se incluye y qué no)

Se incluye:

- Módulo de registro y autenticación básica de usuarios (clientes y personal del taller).
- Gestión de perfiles de cliente y datos de vehículos.
- Creación, modificación, consulta y cancelación de citas.
- Panel interno para el personal del taller (listado de citas, estados, filtros básicos).
- Registro de servicios realizados asociados a las citas.
- Informes básicos (listado de citas por día, por cliente o por estado).

No se incluye (fuera de alcance en esta fase):

- Pasarela de pago real integrada (tarjeta, PayPal, etc.) – se podrá simular.
- Integraciones complejas con sistemas externos (ERP del taller, contabilidad, etc.).
- App móvil nativa; el alcance se limita a **aplicación web responsive.**

- Módulos avanzados de analítica de negocio o cuadros de mando complejos.

3. Requisitos del proyecto

3.1 Requisitos funcionales

- **RF1.** Alta, baja, modificación y autenticación de clientes.
- **RF2.** Gestión de datos de vehículos asociados a cada cliente (matrícula, marca, modelo, etc.).
- **RF3.** Solicitud de citas por parte del cliente indicando: vehículo, tipo de servicio, fecha y franja horaria deseada.
- **RF4.** Comprobación automática de disponibilidad de cita para evitar solapamientos.
- **RF5.** Panel interno del taller para visualizar el calendario de citas (vista diaria/semana) y filtrar por estado, mecánico o tipo de servicio.
- **RF6.** Cambio de estado de la cita (pendiente, confirmada, en curso, finalizada, cancelada).
- **RF7.** Registro de los servicios realizados sobre el vehículo y de las observaciones del mecánico.
- **RF8.** Consulta del historial de citas y servicios por cliente y por vehículo.
- **RF9.** Envío de confirmación/resumen de cita al cliente (por email o sistema de notificaciones configurable).
- **RF10.** Gestión básica de usuarios internos (creación de cuentas de personal de taller con roles: recepción, mecánico, responsable).

3.2 Requisitos no funcionales

- **RNF1.** La aplicación debe ser accesible desde los principales navegadores web modernos (Chrome, Firefox, Edge).
- **RNF2.** El tiempo de respuesta para las operaciones habituales (consultar citas, crear cita) debe ser menor a 2 segundos en condiciones normales de carga.
- **RNF3.** La interfaz debe ser **responsive**, adaptándose correctamente a dispositivos de escritorio, tablet y móvil.
- **RNF4.** Los datos sensibles (credenciales, sesiones) deben gestionarse de forma segura (hash de contraseñas, sesiones con expiración).
- **RNF5.** El diseño de la interfaz debe ser intuitivo, con navegación clara y mensajes de error comprensibles.
- **RNF6.** El sistema debe permitir la creación de copias de seguridad de la base de datos.

- **RNF7.** La arquitectura debe ser modular, facilitando el mantenimiento y la ampliación futura de funcionalidades.

3.3 Requisitos técnicos

- **RT1.** Backend implementado en un entorno tipo **Java Spring Boot** , con **APIs REST** para las operaciones principales.
- **RT2**. Frontend desarrollado con **HTML5, CSS3 y JavaScript, Thymeleaf**.
- **RT3.** Base de datos **relacional** (MySQL/MariaDB) para gestionar usuarios, vehículos, citas y servicios.
- **RT4.** Uso de sistema de control de versiones (**Git**) con repositorio remoto (GitHub).
- **RT5.** Posibilidad de despliegue en un servidor web o en una plataforma PaaS (Heroku, Railway, Render, etc.), simulando un entorno de producción.
- **RT6.** Estructura del proyecto organizada por capas (presentación, lógica de negocio, acceso a datos) para favorecer la separación de responsabilidades.

4. Estructura de trabajo y cronograma

4.1 Fases y tareas principales

Fase 1 – Análisis y definición (Semana 1)

- **T1.1.** Revisión de la actividad, rúbrica y requisitos académicos.
- **T1.2.** Definición detallada de alcance, objetivos y casos de uso principales.
- **T1.3.** Elaboración de documentos de alcance y plan de proyecto.

Fase 2 – Diseño (Semana 2)

- **T2.1.** Diseño de la arquitectura (capas, módulos principales).
- **T2.2.** Diseño del modelo de datos (tablas, relaciones).
- **T2.3**. Bocetos de interfaz (wireframes de pantallas clave).

Fase 3 – Implementación backend (Semanas 3–4)

- **T3.1.** Configuración del entorno de desarrollo y proyecto.
- **T3.2.** Implementación de entidades y acceso a datos (DAO/Repositories).

- **T3.3.** Implementación de servicios de negocio (gestión de usuarios, vehículos, citas).
- **T3.4.** Implementación de controladores/API REST.

Fase 4 – Implementación frontend (Semanas 4–5)

- **T4.1.** Maquetación de las vistas principales (página de inicio, login, panel de cliente, panel de taller).
- **T4.2.** Integración con el backend mediante llamadas a la API.
- **T4.3.** Validaciones en cliente y mejora de la usabilidad.

Fase 5 – Pruebas e integración (Semana 6)

- **T5.1.** Pruebas unitarias (lógica de negocio).
- **T5.2**. Pruebas de integración (flujo completo cliente → servidor → base de datos).
- **T5.3.** Corrección de errores y refactorización básica.

Fase 6 – Despliegue y documentación final (Semana 7)

- **T6.1.** Preparación del entorno de despliegue (local/servidor).
- **T6.2.** Despliegue de la aplicación y pruebas de aceptación.
- **T6.3.** Documentación final (manual de usuario, guía de despliegue, revisión de entregables).

4.2 Fechas estimadas y responsables

Planificación:

- **Semana 1** (del 21/12/2025 al 27/12/2025): **Fase 1.**
- **Semana 2** (del 28/12/2025 al 03/01/2026): **Fase 2.**
- **Semanas 3–4** (del 04/01/2026 al 10/01/2026): **Fase 3.**
- **Semanas 4–5** (del 11/01/2026 al 17/01/2026): **Fase 4.**
- **Semana 6** (del 18/01/2026 al 24/01/2026): **Fase 5.**
- **Semana 7** (del 25/01/2026 al 31/01/2026): **Fase 6.**

4.3 Herramienta o formato de planificación

Se utilizará una combinación de:

Diagrama de Gantt (LibreOffice, Excel o herramienta online) para visualizar las fases y duración. **Trello, Jira** o herramienta similar para gestionar tareas, checklist y prioridades.

5. Recursos y roles del equipo

5.1 Personal asignado y responsabilidades

dfleper (Domingo Fleitas) · GitHub

- **Rol de Analista:** definición de requisitos, alcance y casos de uso.
- **Rol de Diseñador:** diseño de arquitectura, modelo de datos e interfaces.
- **Rol de Desarrollador Backend y Frontend:** implementación completa del sistema.
- **Rol de Tester:** diseño y ejecución de pruebas.
- **Rol de Administrador de sistemas:** despliegue y configuración de entornos.

5.2 Recursos técnicos y materiales

Ordenador personal con sistema operativo compatible (Linux/Windows).

- Entorno de desarrollo integrado (IDE) para backend (Sprint Developers Tools, VSCode).
- El proyecto se realiza de forma individual, por lo que todas las tareas son responsabilidad de dfleper (Domingo Fleitas) · GitHub
- Navegadores web para pruebas (Chrome, Edge).
- Servidor de base de datos (MySQL/MariaDB).
- Sistema de control de versiones Git con repositorio en GitHub.
- Herramientas de planificación (Trello, Jira o similar).
- Conexión a Internet estable para pruebas, descarga de dependencias y despliegue en la nube (si aplica)

6. Gestión de riesgos

6.1 Identificación de riesgos

R1. Retrasos en la planificación

Descripción: Tareas clave (backend o frontend) requieren más tiempo del previsto.

Probabilidad: Media

Impacto: Alto

R2. Complejidad técnica superior a la esperada

Descripción: Dificultades con el framework elegido, configuración de base de datos integración.

Probabilidad: Media

Impacto: Alto

R3. Errores en la integración entre frontend y backend

Descripción: Problemas con APIs REST, formato de datos o manejo de sesiones.

Probabilidad: Media

Impacto: Medio–Alto

R4. Pérdida de código o datos por falta de copias de seguridad

Descripción: Fallo de equipo, borrado accidental de ficheros o del repositorio.

Probabilidad: Baja–Media

Impacto: Alto

R5. Sobrecarga de trabajo académico

Descripción: Coincidencia con exámenes u otras entregas que limiten el tiempo disponible.

Probabilidad: Alta

Impacto: Medio–Alto

6.2 Plan de contingencia / prevención

Para R1 (Retrasos):

Dividir las tareas en subtareas pequeñas y priorizar las funcionalidades críticas (MVP).

Reservar tiempo adicional en la planificación (margen de seguridad).

Para R2 (Complejidad técnica):

Empezar cuanto antes con una prueba de concepto mínima (prototipo de login y creación de cita).

Documentar problemas y buscar soluciones en la documentación oficial y foros.

Para R3 (Errores de integración):

Definir claramente los contratos de la API (endpoints, formatos JSON) antes de programar.

Probar los endpoints con herramientas como Postman antes de integrar en el frontend.

Para R4 (Pérdida de código/datos):

Realizar commits frecuentes y subir cambios regularmente al repositorio remoto.

Realizar copias de seguridad periódicas de la base de datos del desarrollo.

Para R5 (Sobrecarga académica):

Planificar con antelación teniendo en cuenta las fechas de exámenes y otras entregas.

Priorizar hitos clave de este proyecto para evitar dejar tareas críticas para el último momento.

7. Conclusión

El proyecto **Turbo Taller (TT2025)** aborda una necesidad real de digitalización en los talleres de mecánica rápida, proporcionando una herramienta centralizada para la gestión de citas y servicios.

La planificación presentada define claramente el alcance, los requisitos, la estructura de trabajo, los recursos y los riesgos, lo que permite abordar el desarrollo de forma organizada y profesional.

Desde el punto de vista de viabilidad, el proyecto es alcanzable dentro del marco de la asignatura, siempre que se respeten los plazos establecidos y se prioricen las funcionalidades esenciales del sistema. Además, la arquitectura propuesta y la elección de tecnologías facilitan la futura evolución del sistema, permitiendo añadir módulos como pasarela de pago, informes avanzados o aplicación móvil.

En consecuencia, **Turbo Taller** aporta valor tanto al contexto académico (como proyecto completo de desarrollo web) como al posible entorno real de un taller que desee modernizar su gestión de citas y mejorar la experiencia de sus clientes.