Planteamiento Casos Capstone

Carlos Herrera

Marcelo Escobar

Contenido

[Caso Taller Mecánico 3](#_Toc207141806)

[Requerimientos Funcionales 4](#_Toc207141807)

[Requerimientos No Funcionales 4](#_Toc207141808)

[Borrador de Cronograma 5](#_Toc207141809)

[Caso Supermercado 5](#_Toc207141810)

[Requerimientos Funcionales: 5](#_Toc207141811)

[Requerimientos No Funcionales: 6](#_Toc207141812)

[Borrador Cronograma 6](#_Toc207141813)

# Caso Taller Mecánico

En la actualidad, muchos talleres mecánicos presentan una falta de automatización en sus procesos, lo que afecta directamente la fidelidad, ingresos y satisfacción de los clientes. Es común que se compartan trabajos mediante canales informales como WhatsApp o Instagram, sin documentación adecuada, y que no se entreguen facturas o boletas en condiciones. Además, la ausencia de registros históricos representa una gran desventaja al momento de realizar seguimientos y controlar la calidad de los servicios.

Para dar solución a esta problemática, proponemos una **plataforma digital** que permita a los talleres:

* Registrar clientes y asociar sus vehículos.
* Mantener un historial de visitas y mantenciones realizadas.
* Programar futuras visitas de acuerdo con kilometraje o periodos de tiempo definidos.
* Gestionar repuestos y servicios para próximas atenciones.

Desde la perspectiva del cliente, la plataforma le permitirá:

* Subir bitácoras desde su dispositivo móvil, incluyendo notas, fotografías y videos.
* Agendar visitas con antecedentes del problema detectado.
* Confirmar de manera oficial las citas únicamente a través de la aplicación, aunque también puedan ser solicitadas por teléfono.

Con esta solución, se busca **optimizar la gestión interna del taller**, ofrecer **mejor trazabilidad de los servicios** y **elevar la experiencia del cliente** al recibir un servicio más ordenado, confiable y transparente.

## Requerimientos Funcionales

1. **Registro de clientes y vehículos**: El sistema debe permitir ingresar clientes y asociar sus vehículos.
2. **Historial de visitas**: El sistema debe guardar las mantenciones y reparaciones realizadas a cada vehículo.
3. **Agenda de visitas**: El sistema debe permitir agendar y confirmar horas de atención.
4. **Programación automática**: El sistema debe sugerir próximas visitas según kilometraje o tiempo.
5. **Bitácoras del cliente**: El cliente debe poder subir notas, fotos o videos desde su dispositivo móvil.
6. **Bitácoras del taller**: El especialista debe poder registrar diagnósticos y trabajos realizados.
7. **Confirmación de visitas**: Las visitas solo deben confirmarse a través de la aplicación.
8. **Notificaciones**: El sistema debe enviar recordatorios y confirmaciones de visitas.
9. **Gestión de repuestos y servicios**: El taller debe poder registrar repuestos y trabajos realizados.
10. **Generación de documentos**: El sistema debe generar comprobantes o reportes en PDF para el cliente.

## Requerimientos No Funcionales

1. **Seguridad**: La información de clientes y vehículos debe almacenarse de forma segura.
2. **Disponibilidad**: El sistema debe estar disponible al menos el 95% del tiempo.
3. **Rendimiento**: El sistema debe responder en un tiempo razonable (menos de 3 segundos en operaciones básicas).
4. **Usabilidad**: La plataforma debe ser fácil de usar tanto en computadores como en dispositivos móviles.
5. **Compatibilidad**: El sistema debe funcionar en los navegadores más usados (Chrome, Edge, Firefox).
6. **Mantenibilidad**: El sistema debe permitir actualizaciones sin afectar el servicio.

## Borrador de Cronograma

1. **Análisis:** Levantamiento de requerimientos y definición de alcance
2. **Diseño:** Diseño de la base de datos, pantallas principales y flujos
3. **Desarrollo Inicial:** Implementación de registro de clientes, vehículos y agenda
4. **Desarrollo avanzado:** Módulo de historial, bitácoras con fotos/ videos y notificaciones
5. **Pruebas:** Pruebas internas y corrección de errores
6. **Implementación:** Presentación de prototipo funcional
7. **Cierre:** Documentación final y entrega de proyecto

# Caso Supermercado

Hoy en día existen las aplicaciones para comprar mercadería y que sea llevada a tu hogar, pero el método preferido por la mayoría de las personas es ir al supermercado o centro comercial y comprar sus productos físicamente. Durante esta actividad suele perderse la información precisa de qué es lo que se compra: qué productos, cuántos productos, y cuánto es lo que se pagará. De hecho, es usual ver personas con sus calculadoras en el teléfono, e incluso con lápiz y papel llevando un conteo de su compra. La solución a esta problemática sería una aplicación para el teléfono que sea capaz de identificar productos a partir del nombre o código de barra. Los productos serán añadidos a un carrito de compras para llevar completo control de lo que se lleva. La aplicación podrá diferenciar: productos, supermercado/tienda, ofertas activas.

## Requerimientos Funcionales:

1. Identificar productos mediante el ingreso manual del nombre o el escaneo del código de barras, y obtener información como precio o descripción.
2. Los usuarios deben poder agregar, editar o eliminar productos del carrito para mantener un control de su compra
3. Calcular automáticamente el total a pagar basado en los productos agregados y otros modificadores
4. Seleccionar o detectar el supermercado/tienda actual para ajustar precios y disponibilidad de productos
5. Identificar y aplicar ofertas o descuentos vigentes en los productos agregados

## Requerimientos No Funcionales:

1. Interfaz intuitiva minimizando pasos para escanear o agregar productos.
2. Debe funcionar en diversos dispositivos, y las distintas cámaras para escaneo.
3. Los datos de las compras y preferencias del usuario deben almacenarse de forma segura

## Borrador Cronograma

1. Refinamiento de Requerimientos,
2. Viabilidad y Investigación de Mercado,
3. Selección de Tecnologías y Herramientas,
4. Base de Datos y Recopilación de Datos,
5. Diseño de la Interfaz de Usuario,
6. Desarrollo del Backend,
7. Desarrollo del Frontend,
8. Integración y Pruebas Iniciales,
9. Despliegue