Planteamiento del problema

En la actualidad, muchos lavaderos de autos siguen utilizando métodos manuales o poco eficientes para gestionar sus reservas, como agendas físicas, llamadas telefónicas o aplicaciones de mensajería sin control centralizado. Esto genera problemas de organización, sobrecarga de trabajo al personal, errores en los turnos y pérdida de clientes por falta de disponibilidad o tiempos de espera prolongados.

Además, los usuarios no cuentan con una forma rápida y confiable de saber cuándo pueden llevar su vehículo, ni pueden gestionar sus citas de forma autónoma. Esta falta de automatización y accesibilidad afecta tanto la experiencia del cliente como la productividad del negocio.

Por lo tanto, surge la necesidad de desarrollar una aplicación web/móvil que permita a los usuarios reservar turnos de lavado de forma sencilla, consultar disponibilidad en tiempo real y recibir notificaciones sobre su cita. Al mismo tiempo, el sistema debe permitir al lavadero administrar su agenda, controlar la cantidad de vehículos por horario, y obtener reportes sobre la actividad del negocio.

Objetivo General

Desarrollar una aplicación web/móvil para la gestión de reservas en un lavadero de autos, que permita a los clientes agendar turnos de manera rápida y autónoma, y al negocio administrar eficientemente su agenda y operaciones diarias.

Objetivos Específicos

1. Implementar un sistema de reservas en línea con disponibilidad en tiempo real.
2. Permitir a los usuarios crear, modificar o cancelar sus turnos desde la aplicación.
3. Generar notificaciones automáticas para recordar al cliente su cita.
4. Facilitar al personal del lavadero la visualización y gestión de las reservas diarias.
5. Registrar el historial de servicios prestados por vehículo o cliente.
6. Generar reportes sobre la cantidad de reservas, horarios de mayor demanda y servicios realizados.

✅ Requerimientos Funcionales

Estos definen **lo que el sistema debe hacer**:

1. El sistema debe permitir a los clientes registrarse e iniciar sesión.
2. El sistema debe permitir a los clientes reservar un turno para el lavado de su vehículo.
3. El sistema debe mostrar la disponibilidad de horarios en tiempo real.
4. El sistema debe permitir modificar o cancelar una reserva existente.
5. El sistema debe enviar notificaciones (correo o push) de confirmación y recordatorio de la cita.
6. El sistema debe permitir al administrador ver todas las reservas agendadas por día.
7. El sistema debe registrar el historial de reservas de cada cliente.
8. El sistema debe permitir al administrador configurar los horarios disponibles y la capacidad por franja.
9. El sistema debe generar reportes de uso del servicio (por fecha, cliente o tipo de lavado).
10. El sistema debe permitir registrar el tipo de servicio (lavado exterior, interior, completo, etc.) en cada reserva.

🚦 Requerimientos No Funcionales

Estos definen **cómo debe comportarse el sistema**:

1. El sistema debe estar disponible las 24 horas del día, los 7 días de la semana.
2. El tiempo de respuesta del sistema no debe exceder los 2 segundos por operación.
3. La aplicación debe ser accesible desde dispositivos móviles y navegadores web modernos.
4. La interfaz debe ser intuitiva y fácil de usar para personas sin conocimientos técnicos.
5. Los datos de los usuarios deben almacenarse de forma segura y cumplir con políticas de privacidad.
6. El sistema debe realizar copias de seguridad automáticas diariamente.

7. El sistema debe soportar al menos 100 usuarios concurrentes sin pérdida de rendimiento.

8. El sistema debe ser escalable para añadir nuevas funcionalidades en el futuro.

🧱 Estructura Técnica del Sistema

El sistema de reservas para el lavadero ha sido desarrollado utilizando una arquitectura basada en tres capas principales: **Frontend**, **Backend** y **Base de Datos**, con un enfoque modular y escalable. A continuación, se describen las tecnologías utilizadas en cada componente:

🔹 **Frontend (Cliente)**

- **Tecnología:** Ionic con Angular
- **Descripción:** El frontend es una aplicación híbrida desarrollada con el framework Ionic y Angular, lo cual permite una experiencia de usuario fluida tanto en dispositivos móviles como en navegadores web.
- **Funcionalidades:**
    - Registro e inicio de sesión de usuarios
    - Visualización de turnos disponibles
    - Reserva, modificación y cancelación de turnos
    - Notificaciones visuales y confirmaciones

🔹 **Backend (Servidor)**

- **Tecnología:** Java con Spring Boot
- **Descripción:** El backend gestiona toda la lógica de negocio y las operaciones con la base de datos. Está construido con Spring Boot, lo que permite un desarrollo rápido y estructurado mediante servicios REST.
- **Funcionalidades:**
    - Gestión de usuarios y autenticación

- Control de disponibilidad de turnos
- Registro y seguimiento de reservas
- Generación de reportes y estadísticas
- Envío de notificaciones automáticas (si aplica)

🔹 **Base de Datos**

- **Motor:** MySQL
- **Descripción:** Se utiliza MySQL como sistema de gestión de bases de datos relacional, donde se almacenan las entidades principales como usuarios, reservas, horarios, tipos de servicios, etc.
- **Esquema general:**
  - Tabla `usuarios`
  - Tabla `reservas`
  - Tabla `servicios`
  - Tabla `horarios`

🔹 **Entorno de Desarrollo**

- **Tipo:** Desarrollo local
- **Herramientas utilizadas:**
  - IDE: IntelliJ IDEA / Visual Studio Code
  - Gestor de dependencias: Maven o Gradle (para Spring Boot)
  - Servidor local: Spring Boot embebido
  - Cliente: Ionic CLI para pruebas en navegador o emuladores
  - Base de datos: MySQL Server local (ej. XAMPP o instalación manual)

🏗️ Arquitectura del Proyecto

El sistema ha sido diseñado bajo una **arquitectura de microservicios**, lo que permite una alta escalabilidad, mantenibilidad y despliegue independiente de los distintos módulos funcionales de la aplicación.

🔸 Descripción General

La arquitectura se basa en la separación de responsabilidades mediante microservicios que se comunican entre sí a través de APIs REST. Cada microservicio encapsula una funcionalidad específica del sistema, facilitando su evolución independiente, pruebas aisladas y despliegue autónomo.

🔹 Componentes Principales

1. **Frontend (Cliente)**
    - Framework: **Ionic + Angular**
    - Funciona como cliente de los servicios REST del backend.
    - Puede ejecutarse en dispositivos móviles o navegadores.
    - Se comunica con los microservicios a través de HTTP.
2. **Gateway API**
    - Servicio intermediario que dirige las solicitudes del cliente hacia el microservicio correspondiente.
    - Responsable del enrutamiento, autenticación básica y control de acceso.
3. **Microservicio de Autenticación**
    - Maneja el registro, inicio de sesión y validación de tokens JWT.
    - Gestiona credenciales de usuarios y roles.
4. **Microservicio de Reservas**
    - Administra la creación, modificación y cancelación de turnos.
    - Controla disponibilidad de horarios y registra historial de reservas.

5. **Microservicio de Gestión de Servicios**
    - Maneja la definición y configuración de los distintos tipos de lavado disponibles.
    - Permite al administrador gestionar precios, tiempos y descripciones.
6. **Base de Datos (MySQL)**
    - Cada microservicio puede tener su propia base de datos (enfoque distribuido), o bien acceder a un esquema compartido (para proyectos más simples).
    - Motor utilizado: **MySQL**, alojado en entorno local.

🔹 **Comunicación entre Componentes**

- El cliente (Ionic) realiza llamadas HTTP al **API Gateway**.
- El **API Gateway** enruta las solicitudes al microservicio correspondiente.
- Los microservicios exponen APIs REST y pueden comunicarse entre ellos (si es necesario) mediante REST o mensajería asincrónica (en futuros entornos escalables).

🔹 **Ventajas de esta arquitectura**

- Despliegue independiente de cada módulo.
- Mayor facilidad para mantener y escalar el sistema.
- Mejor separación de responsabilidades y control de versiones.
- Posibilidad de migrar partes del sistema a otros lenguajes o tecnologías en el futuro.

📖 Manual de Usuario

Sistema de Reservas para Lavadero de Autos

👤 1. Registro de Usuario

Para comenzar a utilizar la aplicación:

1. Abre la aplicación móvil o web.
2. Presiona el botón **"Registrarse"**.
3. Completa el formulario con tu nombre, correo electrónico, número de teléfono y una contraseña segura.
4. Presiona **"Crear cuenta"**.
5. Recibirás un mensaje de confirmación.

✅ **Nota:** Si ya tienes una cuenta, selecciona **"Iniciar sesión"** directamente.

🔐 2. Iniciar Sesión

1. Ingresa tu correo electrónico y contraseña.
2. Presiona el botón **"Iniciar sesión"**.
3. Si los datos son correctos, accederás a tu panel de usuario.

❗ ¿Olvidaste tu contraseña? Presiona **“¿Olvidaste tu contraseña?”** para recuperarla.

📅 3. Reservar un Turno de Lavado

1. Desde el menú principal, selecciona **"Reservar turno"**.
2. Elige el **tipo de lavado** (exterior, interior, completo, etc.).
3. Selecciona el **día y horario** disponible.
4. Confirma los datos de tu reserva.
5. Presiona **"Confirmar reserva"**.

✅ Recibirás un mensaje de confirmación por correo o dentro de la app.

📝 4. Ver y Gestionar Mis Reservas

1. Ve a la sección **"Mis reservas"**.
2. Allí podrás ver el listado de turnos próximos y anteriores.
3. Para **modificar o cancelar una reserva**, selecciona la cita y elige la opción correspondiente.

❗Solo se puede cancelar hasta 2 horas antes del turno.

🔔 5. Notificaciones

- El sistema te enviará **recordatorios automáticos** antes de tu cita.
- También recibirás alertas en caso de cambios por parte del lavadero.

⚙️ 6. Perfil de Usuario

1. Accede a tu perfil desde el menú.
2. Puedes editar tu información personal y cambiar tu contraseña.