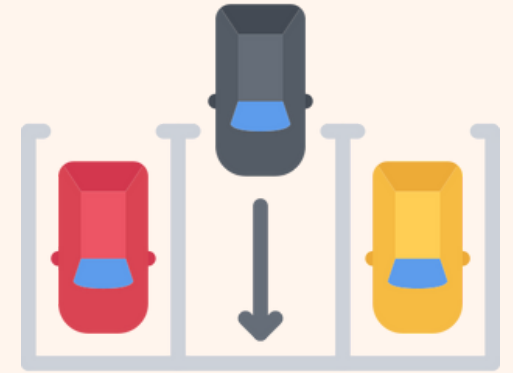


SISTEMA DE PARQUEO



Entorno tecnológico: BD (SQL Server), Sistema de pagos

Servicios disponibles

- ✅ Disponibilidad actual: 25 cupos libres
- 🔒 Seguridad: Cámaras 24/7 + vigilancia en portería
- 📄 Historial: Puedes ver tus visitas anteriores
- 📅 Reservas: Aparta tu cupo desde la app/web
- 🎫 Descuentos activos: 10% para mensualidad, 5% cliente frecuente

PROCESAMIENTO DE LA INFORMACIÓN

1. PROBLEMA ELEGIDO:
Sistema de parqueo
2. ENTRADAS: Identificación,
placa del vehículo
3. PROCESOS: Hora de entrada,
hora de salida, tarifa, aplicar
descuentos, actualización en
tiempo real de cupos
4. SALIDAS: Tiempo de
permanencia, total a pagar,
comprobante de pago, historial
de uso, alertas de seguridad



METODOLOGÍAS DE DESARROLLO

1. Metodología
seleccionada:
SCRUM
2. JUSTIFICACIÓN:
Organiza bien el
trabajo en fases y
asegura
entregables
funcionales desde
el inicio.



DISEÑO

Bienvenido, Cliente #123



Placa: ABC123
Carro

Tu parqueo actual



Hora de entrada:
05/09/2025 - 14:30



Hora estimada de salida:
05/09/2025 - 18:00



Tarifa por hora:
\$4.000 COP



Total acumulado hasta ahora:
\$12.000 COP

PAGAR

- **ENCABEZADO**
- **INFORMACIÓN DEL VEHÍCULO**
- **DATOS DEL PARQUEO ACTUAL**
- **BOTÓN PRINCIPAL DE ACCIÓN**

PRUEBAS

Lenguaje elegido:
C# (.NET con SQL Server)
React Native

Caso 1

- Dado que el cliente entra a las 14:30 y sale a las 18:00.
- Entonces debe mostrar \$16.000 COP (4 horas × \$4.000).

Caso 2

- Dado que hay 100 cupos totales y 75 ocupados.
- Cuando un cliente entra.
- Entonces la disponibilidad debe mostrar 24 cupos libres.

Caso 3

- Dado que el cliente es "frecuente".
- Cuando se calcule el total de \$20.000.
- Entonces el sistema debe aplicar 5% y mostrar \$19.000.

INTEGRACIÓN Y DESPLIEGE

¿Como se verificarían los cambios?

Pruebas unitarias: validar funciones (ej: cálculo de tarifa por hora).

Pruebas de integración: verificar que backend + base de datos interactúen bien (ej: registro en la tabla Historial).

Pruebas funcionales: un usuario real usa la app y se valida que el flujo completo funcione.


Regresión: comprobar que cambios nuevos no dañen lo que ya funcionaba.

¿Como se actualizaría la app?

Desarrollo local
Control de versiones (Git)
Deploy en el servidor

COMUNICACIÓN

 Miguel (Backend)

- Hoy: Implementé cálculo de tarifas.
- Mañana: Conectar descuentos al pago.
- Obstáculo: Endpoint de disponibilidad falla.
-  Laura (Frontend)
- Hoy: Terminé vista "Mi parqueo actual".
- Mañana: Integrar API de reservas.
- Obstáculo: Falta endpoint de reservas.

 Andrés (QA)

- Hoy: Probé entrada/salida.
- Mañana: Casos de prueba de descuentos.
- Obstáculo: Faltan datos de cliente frecuente.



Estudiantes

Miguel Casseres

23158125

Samuel Mejia

23158330

Jose Escobar