

UNIVERSIDAD LAICA ELOY ALFARO DE MANABÍ

APLICACIONES PARA EL CLIENTE WEB

Proyecto del Segundo Parcial

Gestión de Parqueadero de la ULEAM

Carrera:

Software

Autor:

Freddy Marcelo Bravo Ponce

Docente:

Patricia Alexandra Quiroz Palma

Manta - Manabí - Ecuador

19/01/2026

Contenido

1. Propósito del sistema web.....	3
2. Requerimientos funcionales.....	4
3. Requerimientos no funcionales.....	5
4. Herramientas y tecnologías de programación (framework, librerías, etc.) y almacenamiento de datos utilizado.....	6
4.1. Framework de Desarrollo: SvelteKit.....	6
4.2. Lenguaje de Programación: JavaScript, HTML y CSS.....	6
4.3. Almacenamiento y Backend: Supabase	6
4.4. Librerías y Recursos Adicionales.....	7
4.5. Despliegue y Hosting: Vercel.....	7
5. Método de publicación y hosting.....	7
5.1 Control de Versiones con GitHub	7
5.2 Hosting y Despliegue en Vercel.....	7
5.3 Resultados de la Publicación.....	8
6. Código fuente completo.....	8
7. Manual de usuario del sistema web.	70
7.1. Acceso al Sistema (Login)	70
8.2. Registro de Nuevos Usuarios	70
8.4. Panel de Administración (Control y Solicituds).....	72
8.5. Panel de Usuario (Alumno / Maestro).....	73
8. Conclusión	75

Este proyecto surge de una necesidad muy común dentro de cualquier institución educativa de gran escala, como lo es la Universidad Laica Eloy Alfaro de Manabí, la gestión eficiente y segura de los espacios de estacionamiento y el control de los vehículos que ingresan al campus.

Tradicionalmente, el ingreso de vehículos a las instalaciones universitarias se ha manejado de forma manual o con registros básicos que no permiten conocer en tiempo real la disponibilidad de espacios o la identidad de quién ocupa cada plaza. Esto genera varios inconvenientes, desde la pérdida de tiempo de los docentes y alumnos buscando un sitio libre, hasta la falta de un historial claro que permita a la seguridad institucional reaccionar ante cualquier eventualidad.

Es por eso que nos hemos enfocado en la gestión del acceso y salida de vehículos, mediante la digitalización de proceso; el sistema no solo se encarga de dejar asar autos, sino que también crea un ambiente sano entre:

1. **El Usuario (Estudiante/Docente):** Quien necesita saber que su vehículo está registrado correctamente y tener un canal directo para actualizar sus datos si cambia de coche, sin tener que llenar formularios físicos interminables.
2. **El Personal de Seguridad (Guardia):** Quien necesita una herramienta visual y rápida. No puede detenerse a escribir en un cuaderno; requiere un mapa que le diga qué plaza está libre y le permita registrar una entrada rápidamente, validando si el vehículo tiene permiso para estar ahí.
3. **El Administrador:** Quien supervisa el inventario total y dar el visto bueno a los cambios de información, asegurando que la base de datos sea verídica y confiable.

Solución Planteada: En lugar de un simple registro de texto, decidimos crear una plataforma web que centraliza la información en la nube. Al implementar este sistema, el parqueadero universitario deja de ser un caos de suposiciones para convertirse en un flujo de datos ordenado. Ahora, cada movimiento queda guardado con fecha y hora, cada plaza tiene un dueño temporal y la comunicación entre el usuario y la administración es inmediata.

1. Propósito del sistema web.

El objetivo principal de este sistema es servir como un puente tecnológico que modernice la forma en que interactuamos con la infraestructura física de la universidad. No se busca reemplazar un cuaderno

de registros por una pantalla, sino crear un flujo de trabajo inteligente que aporte valores fundamentales:

Control: Uno de los propósitos centrales es eliminar la incertidumbre. En un campus tan concurrido como el de la ULEAM, saber exactamente cuántos vehículos hay dentro y dónde están ubicados es vital. El sistema permite que, al instante de que un auto cruza la garita, el mapa se actualice. Esto garantiza que la administración tenga un control total sobre el aforo y la seguridad de las unidades estacionadas, evitando el uso indebido de los espacios.

Agilidad: Para el personal de guardia, el propósito del sistema es facilitar su trabajo bajo presión. En horas pico, el ingreso de vehículos puede ser caótico. La aplicación fue diseñada para que el registro de una entrada o salida sea una tarea de segundos. Al validar automáticamente si una placa está registrada en la base de datos de la universidad, el guardia ya no tiene que dudar o realizar llamadas para verificar permisos; el sistema le da luz verde o roja de forma inmediata.

Transparencia: Para los estudiantes y docentes, el propósito es darles el control sobre su propia información. Antes, si alguien cambiaba de vehículo, el proceso para actualizar sus datos en los registros de la universidad podía ser lento o inexistente. Con esta web, el usuario puede enviar una solicitud de cambio desde su teléfono. Esto crea un historial de transparencia donde el usuario sabe que su petición está siendo revisada y el administrador puede aprobarla con un clic, manteniendo la base de datos siempre depurada y real.

Seguridad: Finalmente, el sistema cumple el propósito de auditoría. Al guardar cada movimiento en una bitácora digital (quién entró, a qué hora y en qué plaza se puso), la universidad cuenta con un respaldo de datos sólido ante cualquier incidente. En resumen, el propósito es transformar el parqueadero de un área de conflicto logístico a un servicio eficiente y ordenado que mejore la experiencia diaria de toda la comunidad universitaria.

2. Requerimientos funcionales

- **Autenticación y Perfiles de Usuario:** El sistema debe permitir el inicio de sesión mediante la cédula de identidad, validando el rol correspondiente (Admin, Guardia, Alumno/Maestro) y otorgando los permisos adecuados según el perfil.

- **Registro de Vehículos:** Los usuarios (estudiantes o docentes) deben ser capaces de registrar la información básica de su vehículo (placa, marca, modelo, color) en la base de datos centralizada.
- **Gestión de Solicitudes de Actualización:** El sistema debe permitir que un usuario envíe peticiones para cambiar los datos de su vehículo, las cuales quedarán en estado "pendiente" hasta que un administrador las revise.
- **Panel Administrativo de Control:** El administrador tiene la potestad de aprobar o rechazar solicitudes de cambio de información y tener acceso a un inventario completo de todos los vehículos registrados.
- **Mapa Interactivo de Estacionamiento:** El guardia debe visualizar un esquema gráfico del parqueadero con 98 plazas numeradas, donde el color cambie automáticamente (rojo/verde) dependiendo de si el espacio está ocupado o libre.
- **Control de Flujo (Ingreso/Salida):** El guardia debe poder seleccionar una plaza vacía, ingresar la placa del vehículo y registrar su entrada. De igual forma, debe validar la placa para autorizar la salida y liberar el espacio en el mapa.
- **Bitácora de Movimientos (Historial):** El sistema debe generar y mostrar una lista en tiempo real de las últimas entradas y salidas registradas, detallando placa, hora, plaza y tipo de movimiento.
- **Disponibilidad y Publicación:** El sistema debe ser accesible desde cualquier dispositivo con conexión a internet (PC, tablet o móvil) al estar publicado en un hosting en la nube (Vercel).

3. Requerimientos no funcionales

- **Persistencia de Datos:** Toda la información de usuarios, vehículos y movimientos debe almacenarse de forma segura y permanente utilizando una base de datos (Supabase).
- **Seguridad y Privacidad:** Las contraseñas de los usuarios deben estar encriptadas y solo los administradores pueden modificar el inventario global de vehículos.
- **Usabilidad y Diseño:** La interfaz debe ser intuitiva y coherente, utilizando una línea gráfica profesional y efectos visuales modernos como para mejorar la experiencia del usuario.

- **Tiempo de Respuesta:** Las actualizaciones en el mapa del parqueadero y las consultas a la base de datos deben procesarse en pocos segundos para no ralentizar el flujo en la garita de seguridad.
- **Escalabilidad:** La arquitectura debe estar preparada para soportar el registro de cientos de usuarios sin degradar el rendimiento del sitio web.

4. Herramientas y tecnologías de programación (framework, librerías, etc.) y almacenamiento de datos utilizado.

Para el desarrollo de este sistema, se seleccionó un conjunto de herramientas que permiten un equilibrio entre rapidez, seguridad y diseño.

4.1. Framework de Desarrollo: SvelteKit

A diferencia de otros frameworks más pesados, Svelte hace que la página sea extremadamente rápida porque "compila" el código antes de que el usuario lo abra. Esto es lo que permite que el mapa del parqueadero cambie de color al instante y que la navegación entre las pantallas de Admin y Guardia sea fluida, casi como si fuera una aplicación instalada en el celular.

4.2. Lenguaje de Programación: JavaScript, HTML y CSS

- **HTML y CSS:** Se utilizaron para construir la estructura y la estética. Se aplicaron técnicas como Flexbox y Grid para que el mapa de 98 plazas se vea ordenado en cualquier pantalla. También se usó efectos de desenfoque, entre otras adiciones.
- **JavaScript:** Es el motor de la lógica. Gracias a JavaScript, el sistema puede "decidir" qué rol tiene la persona que entra y realizar los cálculos de espacios disponibles y ocupados automáticamente además de las validaciones necesarias, brindando seguridad.

4.3. Almacenamiento y Backend: Supabase

Como base de datos, elegimos Supabase. Es una plataforma en la nube que nos permite guardar la información sin necesidad de configurar servidores complejos.

- **Base de Datos Relacional (PostgreSQL):** Aquí guardamos las tablas de Usuarios, Vehículos e Historial. Lo elegimos porque permite relacionar datos; por ejemplo, saber que el vehículo con placa "ABC-123" pertenece exactamente al estudiante "Juan Pérez".
- **Autenticación:** Supabase se encarga de la seguridad. Cuando un usuario se registra o inicia sesión, sus contraseñas viajan encriptadas, garantizando que la información privada de los estudiantes y docentes esté protegida.

4.4. Librerías y Recursos Adicionales

- **FontAwesome:** Para los iconos visuales (el ícono del carro, el candado, el botón de salir) que ayudan a que la interfaz sea más intuitiva sin necesidad de mucho texto.
- **Google Fonts (Poppins):** Utilizamos la fuente *Poppins* por su alta legibilidad y estilo moderno, lo que le da una identidad visual coherente a todo el proyecto.

4.5. Despliegue y Hosting: Vercel

Para que la página pueda abrirse desde cualquier lugar del mundo y no solo en nuestra computadora, utilizamos Vercel. Esta plataforma está integrada con nuestro código y permite que el sistema esté vivo en internet las 24 horas del día con un enlace seguro (HTTPS).

5. Método de publicación y hosting.

5.1 Control de Versiones con GitHub

El primer paso consistió en utilizar GitHub como repositorio central, en donde almacenamos todo el código fuente del proyecto; esto nos permitió llevar un control de cada cambio realizado (Commits) y asegurar que el código estuviera respaldado de forma segura fuera de la computadora local.

5.2 Hosting y Despliegue en Vercel

Para que la aplicación fuera accesible a través de un enlace público, se seleccionó Vercel como plataforma de hosting. Vercel es un servicio de vanguardia optimizado específicamente para frameworks modernos como SvelteKit.

- **Conexión Automática:** Se vinculó el repositorio de GitHub directamente con Vercel. De esta manera, cada vez que hacíamos un "Push" (subida de código), Vercel detectaba el cambio, compilaba el proyecto y lo publicaba automáticamente.
- **Configuración de Servidor:** Dado que no subimos las llaves secretas de la base de datos a GitHub por seguridad, utilizamos el panel de Variables de Entorno en Vercel. Allí configuramos la URL y la llave anónima de Supabase, permitiendo que la web en producción se conectara a la base de datos de forma privada y segura.

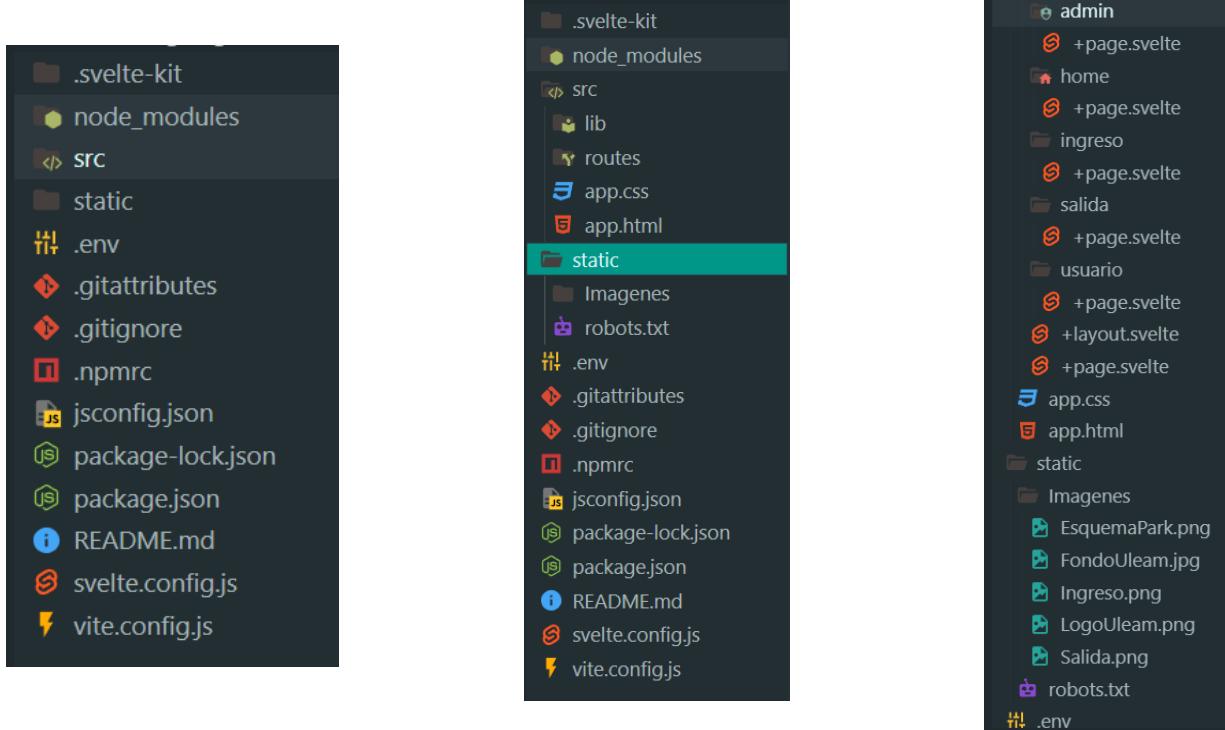
5.3 Resultados de la Publicación

Como resultado de este proceso, el sistema obtuvo las siguientes características:

- **Dominio Propio:** Se generó una URL única y personalizada que puede ser abierta desde cualquier navegador en computadoras o dispositivos móviles.
- **Certificado SSL (HTTPS):** La página cuenta con un candado de seguridad activado, lo que garantiza que los datos de inicio de sesión de los usuarios viajan protegidos.
- **Disponibilidad:** El sistema se encuentra alojado en servidores globales, lo que asegura que esté disponible las 24 horas del día para la comunidad universitaria.

6. Código fuente completo

Esquema General, organización de archivos que se utilizaron:



Los archivos “+page.svelte” son propios de SvelteKit, y se dividen en 3 partes: el scrip, el html y los estilos. Primero comenzaremos con el code del archivo principal, el que dirige el registro e ingreso a la página web:

```
<script>
    import { goto } from '$app/navigation';
    import { supabase } from '$lib/supabaseClient';

    let vistaActual = 'login';
    let verPassword = false;

    // Variables para Login
    let loginCedula = '';
    let loginPassword = '';

    // Variables para Registro
    let regNombres = '';
    let regApellidos = '';
    let regCedula = '';
    let regPassword = '';
    let regCorreo = '';

    // Variables para Olvido Contraseña
    let olvidoCorreo = '';
    let codigoVerificacion = '';
    let codigoGenerado = '';
    let nuevaPassword = '';
    let pasoOlvido = 1;

    function cambiarVista(nuevaVista) {
        loginCedula = ''; loginPassword = '';
        regNombres = ''; regApellidos = ''; regCedula = ''; regPassword = ''; regCorreo = '';
        olvidoCorreo = ''; codigoVerificacion = ''; nuevaPassword = ''; pasoOlvido = 1;
        verPassword = false;
        vistaActual = nuevaVista;
    }

    // --- 1. LÓGICA DE LOGIN ---
    async function manejarLogin() {
        if (!loginCedula || !loginPassword) {
            alert("Por favor complete todos los campos");
            return;
        }

        try {
```

```
// Pedimos correo y rol a la base de datos
const { data: usuarioDB, error: errorDB } = await supabase
    .from('Usuarios')
    .select('correo, rol')
    .eq('cedula', loginCedula.trim())
    .single();

if (errorDB || !usuarioDB) {
    alert("Cédula no encontrada.");
    return;
}

// Revisamos si hay una contraseña recuperada localmente para este correo
const clavesRecuperadas = JSON.parse(localStorage.getItem('clavesRecuperadas') || '{}');
const passwordAUsar = clavesRecuperadas[usuarioDB.correo] || loginPassword;

// Iniciamos sesión en Supabase Auth
const { data, error } = await supabase.auth.signInWithEmailAndPassword({
    email: usuarioDB.correo,
    password: passwordAUsar
});

if (error) {
    alert("Contraseña incorrecta o acceso denegado.");
    return;
}

// Guardamos la sesión
localStorage.setItem('usuarioActual', JSON.stringify(data.user));

const rol = usuarioDB.rol;
if (rol === 'guardia') {
    goto('/home');
} else if (rol === 'admin') {
    goto('/admin');
} else {
    goto('/usuario');
}

} catch (err) {
    console.error(err);
    alert("Error de conexión con el servidor.");
}
}

// --- 2. LÓGICA DE REGISTRO ---
```

```
async function manejarRegistro() {
    if (!regNombres || !regApellidos || !regCedula || !regPassword || !regCorreo) {
        alert("Por favor, complete todos los campos obligatorios.");
        return;
    }

    const correo = regCorreo.toLowerCase().trim();
    const cedula = regCedula.trim();

    if (cedula.length < 10 || !/\d+/.test(cedula)) {
        alert("La cédula debe tener al menos 10 dígitos numéricos.");
        return;
    }

    try {
        const { data: existente } = await supabase
            .from('Usuarios')
            .select('cedula, correo')
            .or(`cedula.eq.${cedula},correo.eq.${correo}`);

        if (existente && existente.length > 0) {
            alert("Error: La cédula o el correo ya están registrados.");
            return;
        }
    } catch (err) { console.error(err); }

    let rolAsignado = '';
    if (correo.endsWith('@live.uleam.edu.ec')) rolAsignado = 'alumno';
    else if (correo.endsWith('@uleam.edu.ec')) rolAsignado = 'maestro';
    else if (correo === '2pq2pugj10@gmail.com') rolAsignado = 'guardia';
    else if (correo === '2pq2pugj9@gmail.com') rolAsignado = 'admin';
    else {
        alert("Error: Usa un correo institucional (@live.uleam.edu.ec o
@uleam.edu.ec)");
        return;
    }

    if (regPassword.length < 6 || !/[A-Z]/.test(regPassword) || !/[0-
9]/.test(regPassword)) {
        alert("La contraseña debe tener mín. 6 caracteres, una mayúscula y un
número.");
        return;
    }

    const { error } = await supabase.auth.signIn({
        email: correo,
        password: regPassword,
```

```

        options: {
          data: { cedula, nombres: regNombres, apellidos: regApellidos, rol:
rolAsignado },
          emailRedirectTo: `${window.location.origin}/home`
        }
      });

      if (error) alert("Error al registrar: " + error.message);
      else {
        alert("¡Revisa tu correo! Se ha enviado un enlace de confirmación.");
        cambiarVista('login');
      }
    }

// --- 3. LÓGICA DE OLVIDO DE CONTRASEÑA ---
async function enviarCodigo() {
  if (!olvidoCorreo) { alert("Ingrese su correo"); return; }

  // Verificar si el correo existe
  const { data, error } = await
supabase.from('Usuarios').select('correo').eq('correo',
olvidoCorreo.trim()).single();
  if (error || !data) { alert("Correo no registrado."); return; }

  codigoGenerado = Math.floor(100000 + Math.random() * 900000).toString();
  alert(`🔑 CÓDIGO DE RECUPERACIÓN: ${codigoGenerado}`);
  pasoOlvido = 2;
}

async function verificarYCambiar() {
  if (codigoVerificacion !== codigoGenerado) { alert("Código incorrecto.");
return; }
  if (nuevaPassword.length < 6) { alert("Mínimo 6 caracteres."); return; }

  // Guardamos la nueva clave localmente para simular el cambio sin afectar Auth
  const clavesRecuperadas = JSON.parse(localStorage.getItem('clavesRecuperadas'))
|| '{}';
  clavesRecuperadas[olvidoCorreo] = nuevaPassword;
  localStorage.setItem('clavesRecuperadas', JSON.stringify(clavesRecuperadas));

  alert("¡Contraseña actualizada localmente! Ya puedes iniciar sesión.");
  cambiarVista('login');
}

</script>

<div class="login-wrapper">
```

```

<div class="top">
    <button on:click={() => cambiarVista('login')}>
        
    </button>
</div>

<div class="fondo"></div>
<div class="base-blanca"></div>

{#if vistaActual === 'login'}
    <div class="card">
        <div class="titulo">Parking ULEAM</div>
        <div class="form-group">
            <label for="cedula">Cédula</label>
            <input id="cedula" type="text" bind:value={loginCedula} placeholder="Ingresa tu cédula">
        </div>
        <div class="form-group">
            <label for="pass">Contraseña</label>
            <div class="input-password-container">
                <input id="pass" type={verPassword ? "text" : "password"} bind:value={loginPassword} placeholder="Contraseña">
                <button type="button" class="toggle-password" on:click={() => verPassword = !verPassword}>
                    {verPassword ? "👁️" : "ܑ"}
                </button>
            </div>
        </div>
        <button class="boton1" on:click={manejarLogin}>Iniciar Sesión</button>
        <div class="links">
            <button class="link-btn" on:click={() => cambiarVista('olvido')}>Olvidé mi contraseña</button>
            <p class="centrado">¿No tienes cuenta? <button class="link-btn" on:click={() => cambiarVista('registro')}>Regístrate</button></p>
        </div>
    </div>
{/if}

{#if vistaActual === 'registro'}
    <div class="card">
        <div class="card-registro">
            <div class="titulo">Registro</div>
            <div class="grid-form">
                <div class="form-group"><label>Nombres</label><input type="text" bind:value={regNombres}></div>
                <div class="form-group"><label>Apellidos</label><input type="text" bind:value={regApellidos}></div>
            </div>
        </div>
    </div>
{/if}

```

```

        <div class="form-group"><label>Cédula</label><input type="text"
bind:value={regCedula}></div>
        <div class="form-group"><label>Correo Institucional</label><input
type="email" bind:value={regCorreo}></div>
        <div class="form-group">
            <label>Contraseña</label>
            <div class="input-password-container">
                <input type={verPassword ? "text" : "password"}>
                bind:value={regPassword} placeholder="Mín. 6 caracteres">
                <button type="button" class="toggle-password" on:click={() =>
verPassword = !verPassword}>
                    {verPassword ? "👁️" : " setHidden()"}>
                </button>
            </div>
        </div>
    </div>
    <button class="boton1" on:click={manejarRegistro}>Enviar Confirmación</button>
    <p class="centrado">¿Ya tienes cuenta? <button class="link-btn" on:click={() => cambiarVista('login')}>Inicia Sesión</button></p>
</div>
</if>

{#if vistaActual === 'olvido'}
    <div class="card">
        <div class="titulo">Recuperar Cuenta</div>
        {#if pasoOlvido === 1}
            <div class="form-group">
                <label>Correo Electrónico</label>
                <input type="email" bind:value={olvidoCorreo} placeholder="Correo
institucional">
            </div>
            <button class="boton1" on:click={enviarCodigo}>Enviar Código</button>
        {:#else}
            <p class="centrado">Código enviado a: <strong>{olvidoCorreo}</strong></p>
            <div class="form-group">
                <label>Código</label>
                <input type="text" bind:value={codigoVerificacion} placeholder="Ingrese
el código">
            </div>
            <div class="form-group">
                <label>Nueva Contraseña</label>
                <input type="password" bind:value={nuevaPassword} placeholder="Nueva
clave">
            </div>
            <button class="boton1" on:click={verificarYCambiar}>Cambiar
Contraseña</button>
        {:#else}
        {:#endif}
    {:#endif}
{:#endif}

```

```
  {/if}
  <button class="link-btn cancelar" on:click={() =>
cambiarVista('login')}>Cancelar</button>
</div>
{/if}
</div>
```

De la mano de este va el archivo css de los estilos globales (app.css):

```
/* 1. VARIABLES Y ESTILOS GLOBALES */
:root {
  --primary: #004AAD;
  --primary-dark: #003580;
  --danger: #dc3545;
  --danger-dark: #b91c1c;
  --success: #28a745;
  --bg: #e8eaf0;
  --card: #ffffff;
  --text: #1f2937;
  --text-muted: #6b7280;
  --border: #d1d5db;
  --shadow: 0 4px 12px rgba(0,0,0,0.08);
  --shadow-hover: 0 6px 20px rgba(0,0,0,0.12);
}

* { box-sizing: border-box; margin: 0; padding: 0; }

body {
  font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, 'Helvetica Neue', Arial, sans-serif;
  background-color: var(--bg);
  color: var(--text);
  line-height: 1.6;
  min-height: 100vh;
  position: relative;
}

body::before {
  font-family: 'Poppins', 'Segoe UI', Arial, sans-serif;
  font-size: 14px;
  line-height: 1.5;
  content: '';
  position: fixed;
  top: 0; left: 0; width: 100%; height: 100%;
```

```
background-size: cover;
background-position: center;
background-attachment: fixed;
background-repeat: no-repeat;
filter: blur(4px);
z-index: -1;
}

/* 2. CONTENEDORES Y LAYOUT */
.container {
  max-width: 1280px;
  margin: 30px auto;
  padding: 30px;
  background: rgba(255, 255, 255, 0.95);
  border-radius: 12px;
  box-shadow: var(--shadow);
  backdrop-filter: blur(5px);
}

header {
  background: linear-gradient(135deg, var(--primary) 0%, var(--primary-dark) 100%);
  color: #fff;
  padding: 18px 24px;
  border-radius: 10px;
  margin-bottom: 20px;
  text-align: center;
  box-shadow: 0 2px 8px rgba(0,74,173,0.2);
}

header h1 { font-size: 1.75rem; font-weight: 600; letter-spacing: 0.5px; }

.main-layout { display: flex; gap: 20px; }
.left-panel { width: 360px; display: flex; flex-direction: column; gap: 16px; }

/* Paneles de Información */
.info-panel, .info-box, #resultadoSalida {
  background: #fff;
  padding: 16px;
  border-radius: 10px;
  box-shadow: var(--shadow);
}

.info-panel > div, #resultadoSalida div {
  display: flex;
  justify-content: space-between;
  align-items: center;
  padding: 10px 0;
```

```
font-size: 0.95rem;
color: var(--text);
border-bottom: 1px solid #f3f4f6;
}

.info-panel > div:last-child, #resultadoSalida div:last-child { border-bottom: none;
}

.info-panel span {
font-weight: 700;
color: var(--primary);
background: #f9fafb;
padding: 5px 10px;
border-radius: 6px;
border: 1px solid var(--border);
}

/* Estilos específicos para caja de instrucciones */
.info-box h3 { color: var(--primary); margin-bottom: 10px; font-size: 1.1rem; }
.info-box p, .info-box li { font-size: 0.9rem; color: var(--text-muted); }
.info-box ol { padding-left: 20px; margin: 10px 0; }

/* Responsive */
@media (max-width: 900px) {
  .main-layout { flex-direction: column; }
  .left-panel { width: 100%; order: 2; }
  .map-container { order: 1; }
}

/* 3. MAPA Y PLAZAS (Estacionamiento) */
.map-container {
  position: relative;
  width: 100%; /* Ocupa todo el ancho disponible */
  border-radius: 12px;
  overflow: hidden;
  background: #1f2937; /* Fondo oscuro por si la imagen tarda en cargar */
  box-shadow: var(--shadow);
  /* El contenedor del mapa se ajusta al tamaño de la imagen */
  display: flex;
  align-items: flex-start;
}

.map-container img {
  width: 100%;
  height: auto;    /* La altura se calcula sola según la proporción de la imagen */
  display: block;  /* Evita un pequeño espacio blanco debajo de la imagen */
  object-fit: contain; /* Asegura que la imagen nunca se recorte */
}
```

```
}

.overlay {
  position: absolute;
  inset: 0;           /* Se pega a Los 4 bordes del contenedor */
  pointer-events: none;
}

.plaza {
  position: absolute;
  display: flex;
  align-items: center;
  justify-content: center;
  font-size: 13px;
  font-weight: 700;
  color: #fff;
  background: linear-gradient(135deg, #10b981 0%, var(--success) 100%);
  border-radius: 4px;
  cursor: pointer;
  pointer-events: auto;
  transition: all 0.2s ease;
  box-shadow: 0 1px 3px rgba(0,0,0,0.2);
  user-select: none;
  border: 1px solid rgba(255,255,255,0.3);
  /* Para evitar que el contenido se extienda más allá del borde */
  box-sizing: border-box;
}

.plaza:hover {
  transform: scale(1.1);
  z-index: 10;
  box-shadow: 0 4px 8px rgba(0,0,0,0.3);
}

.plaza.ocupada {
  background: linear-gradient(135deg, var(--danger) 0%, var(--danger-dark) 100%);
}

/* 4. BOTONES */
/* Botón Base */
button, .btn {
  display: inline-flex;
  align-items: center;
  justify-content: center;
  gap: 8px;
  padding: 12px 16px;
  border-radius: 8px;
}
```

```
border: none;
cursor: pointer;
font-weight: 600;
transition: all 0.2s ease;
font-size: 0.95rem;
background-color: var(--primary);
color: white;
text-decoration: none;
}

button:hover, .btn:hover { transform: translateY(-2px); box-shadow: var(--shadow-hover); }
button:active, .btn:active { transform: translateY(0); }

/* Variantes de Botones */
.btn.ingreso { background: linear-gradient(135deg, var(--primary) 0%, #0056d6 100%); }
.btn.salida, .btn.danger { background: linear-gradient(135deg, var(--danger) 0%, #e74c5c 100%); }

.btn.small {
  padding: 8px 14px;
  font-size: 0.85rem;
  background: #fff;
  color: var(--text);
  border: 1px solid var(--border);
}
.btn.small:hover { background: #f9fafb; border-color: var(--text-muted); }

/* Botones Grandes del Home */
.buttons-row {
  display: flex;
  gap: 16px;
  justify-content: center;
  background: #fff;
  padding: 16px;
  border-radius: 10px;
  box-shadow: var(--shadow);
}

.buttons-row button {
  width: 100%;
  height: 120px; /* Botones cuadrados grandes */
  flex-direction: column;
  font-size: 1.1rem;
}
```

```
.buttons-row img { width: 48px; height: 48px; margin-bottom: 8px; }

.footer-buttons {
  display: flex;
  gap: 10px;
  margin-top: auto;
}
.footer-buttons button { flex: 1; }

/* 5. MODALES Y TABLAS */
.modal {
  position: fixed; inset: 0;
  display: flex; align-items: center; justify-content: center;
  background: rgba(0,0,0,0.6);
  z-index: 1000;
  backdrop-filter: blur(3px);
  animation: fadeIn 0.2s ease-out;
}

@keyframes fadeIn { from { opacity: 0; } to { opacity: 1; } }

.modal-content, .modal-content2 {
  background: #fff;
  padding: 24px;
  border-radius: 12px;
  box-shadow: 0 20px 50px rgba(0,0,0,0.3);
  max-height: 85vh;
  overflow-y: auto;
  position: relative;
  width: 95%;
}
.modal-content { max-width: 400px; }
.modal-content2 { max-width: 800px; } /* Para la tabla ancha */

.close {
  position: absolute; right: 20px; top: 15px;
  font-size: 24px; color: var(--text-muted);
  cursor: pointer;
}
.close:hover { color: var(--danger); }

/* Tabla */
table { width: 100%; border-collapse: collapse; margin-top: 10px; }
th { background: var(--primary); color: #fff; text-align: left; padding: 12px; }
td { padding: 12px; border-bottom: 1px solid #eee; }
tr:hover { background: #f9fafb; }
```

```
/* 6. FORMULARIOS (Login, Registro, Modales) */
form { display: flex; flex-direction: column; gap: 15px; }

label { font-size: 0.9rem; font-weight: 500; margin-bottom: 4px; display: block; }

input {
  width: 100%;
  padding: 10px 12px;
  border: 1px solid var(--border);
  border-radius: 6px;
  font-size: 1rem;
  transition: all 0.2s;
}

input:focus {
  outline: none;
  border-color: var(--primary);
  box-shadow: 0 0 0 3px rgba(0, 74, 173, 0.1);
}

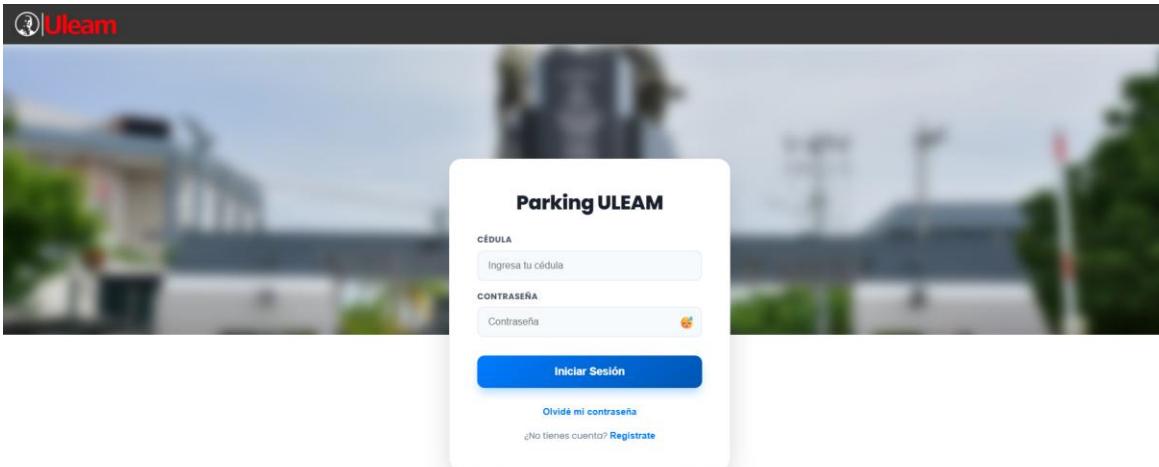
/* Estilos Específicos del Login (Tarjetas) */
.login-wrapper {
  display: flex; justify-content: center; align-items: center;
  min-height: 80vh;
}

.card {
  width: 100%; max-width: 400px;
  background: #fff;
  padding: 30px;
  border-radius: 16px;
  box-shadow: 0 10px 25px rgba(0,0,0,0.1);
}

.titulo-parking {
  font-size: 1.5rem; color: var(--primary);
  text-align: center; margin-bottom: 20px; font-weight: bold;
}

.links { margin-top: 15px; font-size: 0.9rem; text-align: center; }
.link-btn {
  background: none; color: var(--primary);
  padding: 0; text-decoration: underline;
  display: inline; width: auto; font-weight: normal;
  box-shadow: none;
}
.link-btn:hover { background: none; color: var(--primary-dark); transform: none; }
```

Facultad de Ciencias de la Vida y Tecnologías



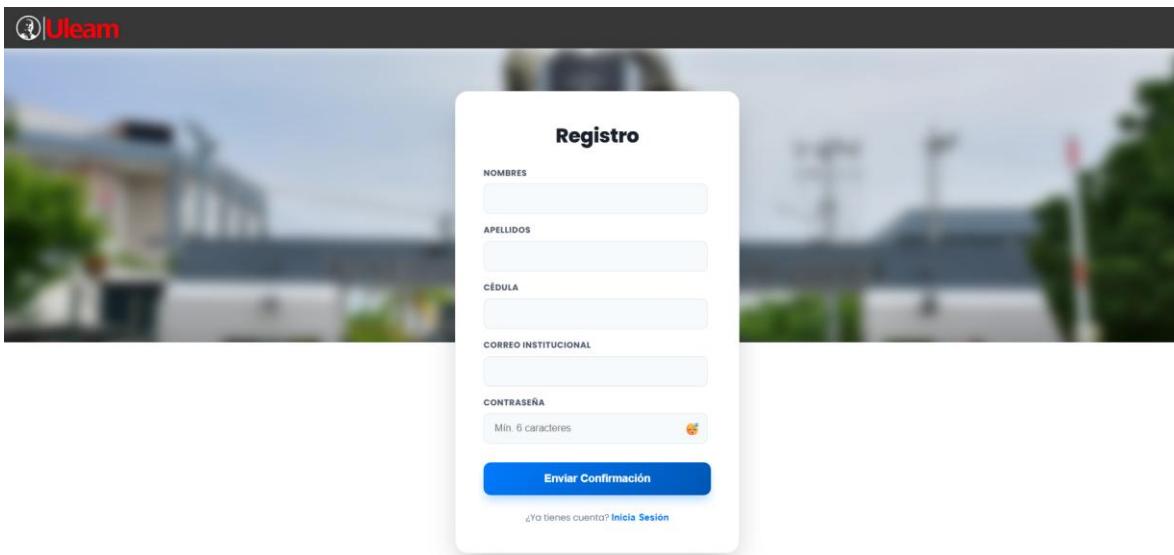
Parking ULEAM

CÉDULA

CONTRASEÑA
 

Iniciar Sesión

[Olvidé mi contraseña](#)
[¿No tienes cuenta? Regístrate](#)



Registro

NOMBRES

APELLIDOS

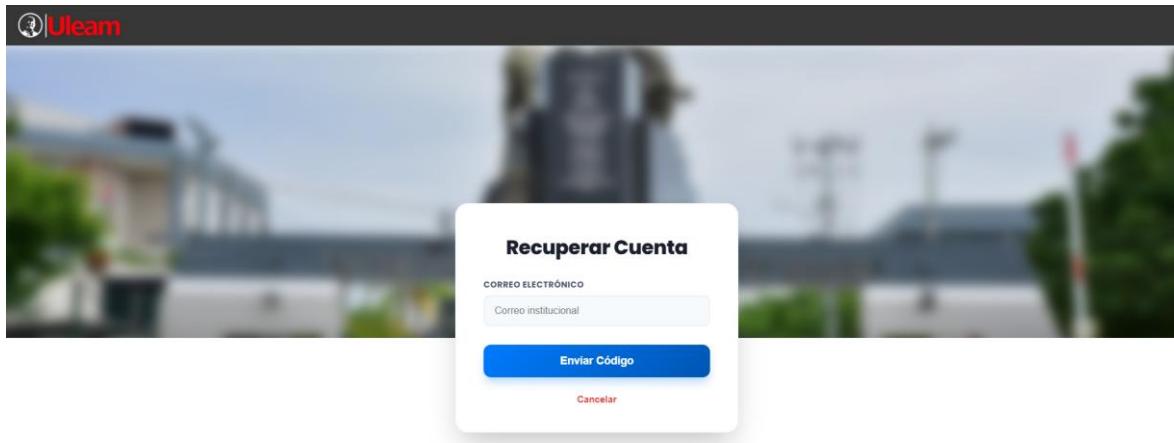
CÉDULA

CORREO INSTITUCIONAL

CONTRASEÑA
Min. 6 caracteres 

Enviar Confirmación

[¿Ya tienes cuenta? Inicia Sesión](#)



Recuperar Cuenta

CORREO ELECTRÓNICO

Enviar Código

[Cancelar](#)

Seguido de esto pasaremos con las interfaces principales del proyecto, las que aparecen luego de iniciar sesión. Dependiendo de que rol tenga asociada la cuenta, se le ingresará a un lado u otro. Comenzaremos por las cuentas con el rol: “alumno” o “maestro”. Si se ingresa con alguno de estos roles se le llevará a la siguiente pestaña:

```

<script>
  import { onMount } from 'svelte';
  import { goto } from '$app/navigation';
  import { supabase } from '$lib/supabaseClient';

  let usuario = null;
  let cargando = true;
  let tieneVehiculo = false; // Estado clave para bloquear/desbloquear paneles
  let vehiculoActual = null;

  // --- DATOS FORMULARIO IZQUIERDO (Registro Nuevo) ---
  let regPlaca = '';
  let regMarca = '';
  let regModelo = '';
  let regColor = '';

  // --- DATOS FORMULARIO DERECHO (Solicitud Cambio) ---
  let solPlaca = '';
  let solMarca = '';
  let solModelo = '';
  let solColor = '';
  let motivoCambio = '';

  onMount(async () => {
    // 1. Verificar sesión
    const sesion = localStorage.getItem('usuarioActual');
    if (!sesion) return goto('/', { replaceState: true });
    usuario = JSON.parse(sesion);

    // 2. Consultar si ya tiene vehículo registrado
    const { data, error } = await supabase
      .from('Vehiculos')
      .select('*')
      .eq('propietario_id', usuario.id)
      .maybeSingle();

    if (data) {
      tieneVehiculo = true;
      vehiculoActual = data;
      regPlaca = data.placa;
    }
  });

```

```

    regMarca = data.marca;
    regModelo = data.modelo;
    regColor = data.color;
}

cargando = false;
});

// --- FUNCIÓN PANEL IZQUIERDO: REGISTRAR VEHÍCULO ---
async function registrarVehiculo() {
    if (!regPlaca || !regMarca || !regModelo || !regColor) {
        return alert("Por favor completa todos los datos del vehículo.");
    }

    const { error } = await supabase.from('Vehiculos').insert([
        propietario_id: usuario.id,
        placa: regPlaca.toUpperCase(),
        marca: regMarca,
        modelo: regModelo,
        color: regColor,
        estado: 'Registrado'
    ]);

    if (error) {
        alert("Error al registrar: " + error.message);
    } else {
        alert("¡Vehículo registrado con éxito!");
        window.location.reload(); // Recargamos para actualizar el estado de los
panelos
    }
}

// --- FUNCIÓN PANEL DERECHO: SOLICITAR CAMBIO ---
async function enviarSolicitud() {
    if (!motivoCambio) return alert("Por favor indica un motivo o qué deseas
cambiar.");

    // Preparamos el JSON con los datos nuevos que el usuario escribió
    const datosNuevos = {
        placa: solPlaca || vehiculoActual.placa,
        marca: solMarca || vehiculoActual.marca,
        modelo: solModelo || vehiculoActual.modelo,
        color: solColor || vehiculoActual.color,
        motivo: motivoCambio
    };

    const { error } = await supabase.from('Solicitudes').insert([

```

```

        usuario_id: usuario.id,
        vehiculo_id: vehiculoActual.id,
        tipo_solicitud: 'actualizacion',
        estado: 'pendiente',
        datos_nuevos: datosNuevos
    ]]);
}

if (error) {
    alert("Error al enviar solicitud: " + error.message);
} else {
    alert("Solicitud enviada al Administrador. Te notificaremos cuando se apruebe.");
    // Limpiar formulario derecho
    solPlaca = ''; solMarca = ''; solModelo = ''; solColor = ''; motivoCambio =
';
}
}

function cerrarSesion() {
    localStorage.removeItem('usuarioActual');
    goto('/', { replaceState: true });
}
</script>

<div class="main-container">
<div class="top">
<div class="logo-area">
    
    <span>Panel de Usuario ({usuario?.user_metadata?.rol || 'Estudiante'})</span>
</div>
<button class="btn-logout" on:click={cerrarSesion}>
    <i class="fas fa-sign-out-alt"></i> Cerrar Sesión
</button>
</div>

<div class="fondo"></div>

<div class="content-wrapper">
{#if cargando}
    <div class="loading">Cargando datos...</div>
{:else}
    <div class="grid-paneles">

        <div class="card panel {tieneVehiculo ? 'bloqueado' : ''}">
            <div class="header-panel">
                <i class="fas fa-car"></i>
                <h3>{tieneVehiculo ? 'Vehículo Registrado' : 'Registrar Vehículo'}</h3>
            </div>
    

```



```
        <input type="text" bind:value={solPlaca}>
placeholder={vehiculoActual.placa}>
    </div>
    <div>
        <label>Nuevo Color</label>
        <input type="text" bind:value={solColor}>
placeholder={vehiculoActual.color}>
    </div>
    </div>

    <div class="fila-doble">
        <div>
            <label>Nueva Marca</label>
            <input type="text" bind:value={solMarca}>
placeholder={vehiculoActual.marca}>
        </div>
        <div>
            <label>Nuevo Modelo</label>
            <input type="text" bind:value={solModelo}>
placeholder={vehiculoActual.modelo}>
        </div>
        </div>
    </div>

    <label>Motivo del cambio (Obligatorio)</label>
    <textarea rows="2" bind:value={motivoCambio} placeholder="Ej: Vendí el
auto anterior..."></textarea>

        <button class="boton1 warning" on:click={enviarSolicitud}>Enviar
Solicitud</button>
        </div>
    </div>
</div>

<style>
    @import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600;700&display=s
wap');

:global(body) { margin: 0; font-family: 'Poppins', sans-serif; background:
#f0f2f5; }

.main-container { min-height: 100vh; display: flex; flex-direction: column; }
```

```
.top {  
background: rgba(45, 45, 45, 0.95);  
backdrop-filter: blur(10px);  
height: 64px;  
display: flex;  
justify-content: space-between;  
align-items: center;  
padding: 0 20px;  
color: white;  
z-index: 100;  
box-shadow: 0 4px 6px rgba(0,0,0,0.1);  
}  
.logo-area { display: flex; align-items: center; gap: 15px; font-weight: 600; }  
.logo-area img { height: 40px; }  
.btn-logout { background: #e74c3c; color: white; border: none; padding: 8px 15px;  
border-radius: 6px; cursor: pointer; font-weight: 600; transition: 0.3s; }  
.btn-logout:hover { background: #c0392b; }  
  
.fondo {  
position: fixed; top: -200px; left: -20px; width: 102%; height: 150%;  
background-image: url("/Imagenes/FondoUleam.jpg");  
background-size: cover; background-position: center;  
filter: blur(8px) brightness(0.6);  
z-index: -1;  
}  
  
.content-wrapper {  
flex: 1;  
display: flex;  
justify-content: center;  
align-items: center;  
padding: 20px;  
}  
  
.grid-paneles {  
display: grid;  
grid-template-columns: 1fr 1fr;  
gap: 100px;  
width: 100%;  
max-width: 900px;  
}  
  
.card {  
background: rgba(255, 255, 255, 0.95);  
border-radius: 20px;  
padding: 30px;
```

```
        box-shadow: 0 15px 35px rgba(0,0,0,0.2);
        transition: transform 0.3s ease, opacity 0.3s ease;
        display: flex;
        flex-direction: column;
    }

.header-panel { display: flex; align-items: center; gap: 10px; margin-bottom: 20px; border-bottom: 2px solid #eee; padding-bottom: 10px; }
.header-panel i { font-size: 1.5rem; color: #007bff; }
.header-panel h3 { margin: 0; color: #333; font-weight: 700; }

.bloqueo input {
    background-color: #e9ecef;
    color: #6c757d;
    border-color: #ced4da;
    cursor: not-allowed;
}

.status-badge {
    margin-top: 20px;
    background: #d4edda; color: #155724;
    padding: 15px; border-radius: 10px;
    text-align: center; font-weight: bold;
    display: flex; align-items: center; justify-content: center; gap: 10px;
}

.desactivado {
    opacity: 0.6;
    pointer-events: none;
    filter: grayscale(0.8);
}

.mensaje-espera {
    height: 100%; display: flex; align-items: center; justify-content: center;
    text-align: center; color: #777; font-style: italic;
}

/* --- FORMULARIOS --- */
.form-body { display: flex; flex-direction: column; flex: 1; }
label { font-size: 0.85rem; color: #555; font-weight: 600; margin-bottom: 5px;
margin-top: 10px; }
input, textarea {
    width: 100%; padding: 12px;
    border: 2px solid #eee; border-radius: 8px;
    background: #f9f9f9; font-family: inherit;
    box-sizing: border-box;
}
```

```

input:focus, textarea:focus { outline: none; border-color: #007bff; background: white; }

.fila-doble { display: grid; grid-template-columns: 1fr 1fr; gap: 15px; }

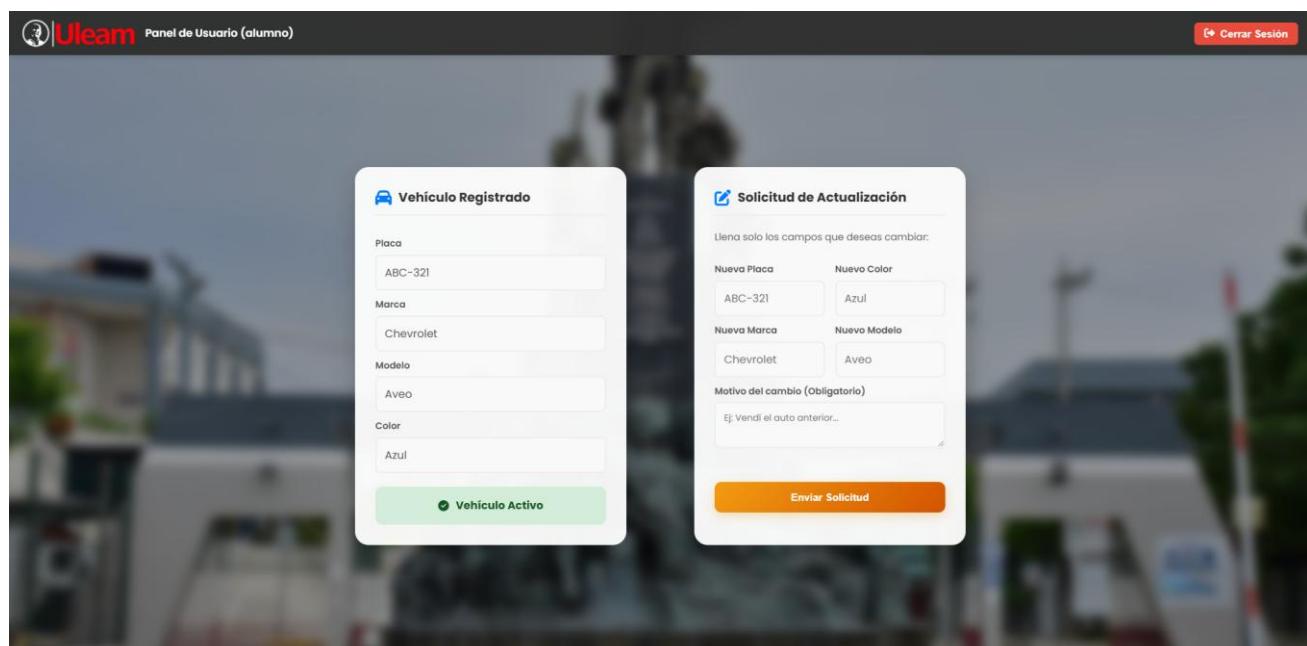
.boton1 {
    width: 100%; padding: 14px;
    background: linear-gradient(135deg, #007bff 0%, #0056b3 100%);
    color: white; border: none; border-radius: 10px;
    margin-top: 50px; font-weight: 700; cursor: pointer;
    box-shadow: 0 5px 15px rgba(0, 123, 255, 0.3);
    transition: 0.2s;
}
.boton1:hover { transform: translateY(-2px); }

.warning {
    background: linear-gradient(135deg, #f39c12 0%, #d35400 100%);
    box-shadow: 0 5px 15px rgba(243, 156, 18, 0.3);
}

.instrucion { font-size: 0.9rem; color: #666; margin-bottom: 15px; }
.loading { text-align: center; font-size: 1.2rem; color: white; font-weight: bold; margin-top: 50px; }

/* Responsive para móviles */
@media (max-width: 768px) {
    .grid-paneles { grid-template-columns: 1fr; }
}
</style>

```



Si se inicia sesión con una cuenta cuyo rol asociado es “guardia” se le llevará a la siguiente pestaña, que sería el home para los guardias que controlarán el ingreso y salida de vehículos:

```
<script>

    import { onMount } from 'svelte';
    import { goto } from '$app/navigation';
    import { supabase } from '$lib/supabaseClient';

    let usuarioGuardia = null;

    onMount(async () => {
        const sesion = localStorage.getItem('usuarioActual');
        if (!sesion) {
            goto('/', { replaceState: true });
        } else {
            usuarioGuardia = JSON.parse(sesion);
            await cargarDatosSupabase();
        }
    });

    // --- 1. ESTADO ---
    let plazasEstacionamiento = new Array(98).fill(false);
    let registrosVehiculos = [];

    $: ocupados = plazasEstacionamiento.filter(Boolean).length;
    $: disponibles = plazasEstacionamiento.length - ocupados;

    let mostrarModalRegistro = false;
    let tablaRegistroHTML = '';

    // Coordenadas de Las filas de plazas
    const filas = [
        { start: 1, count: 26, top: 4.4, marginLeft: 10.4, marginRight: 1.5,
plazaHeight: 13, widthFactor: 0.8, innerOffset: 0, shiftPct: 0 },
        { start: 27, count: 23, top: 34.2, marginLeft: 10.4, marginRight: 11.7,
plazaHeight: 13, widthFactor: 0.8, innerOffset: 0.05, shiftPct: 0 },
        { start: 50, count: 23, top: 54, marginLeft: 10.4, marginRight: 11.7,
plazaHeight: 13, widthFactor: 0.8, innerOffset: 0.05, shiftPct: 0 },
        { start: 73, count: 26, top: 83, marginLeft: 10.4, marginRight: 1.5,
plazaHeight: 13, widthFactor: 0.8, innerOffset: 0.05, shiftPct: 0 }
    ];

    // --- 2. LÓGICA SUPABASE ---

    //Cargar el Mapa (Estado del Parking)
```

```

async function cargarDatosSupabase() {
    try {
        // Buscamos autos que están dentro
        const { data, error } = await supabase
            .from('Vehiculos')
            .select('plaza_actual')
            .eq('en_estacionamiento', true);

        if (error) throw error;

        let nuevoEstado = new Array(98).fill(false);

        if (data) {
            data.forEach(auto => {
                if (auto.plaza_actual && auto.plaza_actual >= 1 &&
auto.plaza_actual <= 98) {
                    nuevoEstado[auto.plaza_actual - 1] = true;
                }
            });
        }
        plazasEstacionamiento = nuevoEstado;
    } catch (error) {
        console.error('Error cargando mapa:', error);
    }
}

//Generar Tabla de Historial
async function generarTablaRegistro() {
    const { data: historial, error } = await supabase
        .from('Historial_Ingresos')
        .select('*')
        .order('created_at', { ascending: false })
        .limit(98);

    if (error) {
        console.error(error);
        return;
    }

    // 2. Construir la tabla HTML
    let tabla = `<table class="registro-table">
        <thead>
            <tr>
                <th>Hora</th>
                <th>Movimiento</th>
                <th>Placa</th>
        </thead>
        <tbody>
            ${data.map(item => `<tr>
                <td>${item.created_at}</td>
                <td>${item.movement}</td>
                <td>${item.plate}</td>
            </tr>`)}
        </tbody>
    </table>`;
}

```

```

        <th>Plaza</th>
    </tr>
</thead>
<tbody>`;

historial.forEach((r) => {
    const fecha = new Date(r.created_at).toLocaleTimeString([], {hour: '2-digit', minute:'2-digit'});
    const color = r.tipo_movimiento === 'entrada' ? '#27ae60' : '#c0392b';
    const movimiento = r.tipo_movimiento.toUpperCase();

    tabla += `<tr>
        <td>${fecha}</td>
        <td style="color:${color}; font-weight:bold;">${movimiento}</td>
        <td>${r.placa}</td>
        <td>${r.plaza} || '-'</td>
    </tr>`;
});

tabla += `</tbody></table>`;
tablaRegistroHTML = tabla;
}

async function mostrarRegistroVehiculos() {
    await generarTablaRegistro(); // Esperar a que traiga los datos
    mostrarModalRegistro = true;
}

function cerrarSesion() {
    localStorage.removeItem('usuarioActual');
    goto('/', { replaceState: true });
}

function irAIngreso() { goto('/ingreso'); }
function irASalida() { goto('/salida'); }
</script>

<div class="top">
    <div class="logo-section">
        
        <span>Parking ULEAM - Guardia</span>
    </div>
    <button class="btn-exit" on:click={cerrarSesion}>
        <i class="fas fa-sign-out-alt"></i> Cerrar Sesión
    </button>
</div>

```

```

<div class="fondo-app"></div>

<main class="main-layout">
    <div class="left-panel">
        <div class="buttons-row">
            <button on:click={irAIngreso} class="btn ingreso">
                
                <span>INGRESO</span>
            </button>
            <button on:click={irASalida} class="btn salida">
                
                <span>SALIDA</span>
            </button>
        </div>

        <div class="info-panel">
            <div>Espacios Disponibles: <span
id="espaciosDisponibles">{disponibles}</span></div>
            <div>Espacios Ocupados: <span
id="espaciosOcupados">{ocupados}</span></div>
        </div>

        <div class="footer-buttons">
            <button on:click={mostrarRegistroVehiculos} class="btn
small">Registro</button>
        </div>
    </div>

    <div class="map-container">
        
        <div id="estacionamiento" class="overlay">
            {#each filas as row}
                {@const count = row.count}
                {@const marginLeft = row.marginLeft || 2.5}
                {@const marginRight = row.marginRight || 2.5}
                {@const spacing = (100 - marginLeft - marginRight) / count}
                {#each Array(count) as _, i}
                    {@const numero = row.start + i}
                    {@const leftPct = marginLeft + i * spacing + spacing *
(row.innerOffset || 0.05) + (row.shiftPct || 0)}
                    <div
                        class="plaza"
                        class:ocupada={plazasEstacionamiento[numero - 1]}
                        style="left: {leftPct}%; top: {row.top}%; width:
{spacing * (row.widthFactor || 0.9)}%; height: {row.plazaHeight || 6}%;"
                    >
                
```

```
                {numero}
            </div>
        {/each}
    {/each}
</div>
</main>

{#if mostrarModalRegistro}
<div class="modal">
    <div class="modal-content">
        <span class="close" on:click={() => mostrarModalRegistro =
false}>&times;</span>
        <h3>Registro de Vehículos</h3>
        <div class="table-wrapper">
            {@html tablaRegistroHTML}
        </div>
    </div>
</div>
{/if}

<style>
    @import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;600;700;800&display=s
wap');
:global(body) {
    margin: 0;
    padding: 0;
    font-family: 'Poppins', sans-serif;
    background-color: #f4f4f4;
    overflow: hidden;
}

.top {
    background: rgba(45, 45, 45, 0.95);
    backdrop-filter: blur(10px);
    height: 64px;
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 0 20px;
    color: white;
    z-index: 100;
    box-shadow: 0 4px 6px rgba(0,0,0,0.1);
}
```

```
.logo-section { display: flex; align-items: center; gap: 15px; font-weight: 600; }

.logo-section img { height: 40px; }

.btnExit { background: #e74c3c; color: white; border: none; padding: 8px 15px; border-radius: 6px; cursor: pointer; font-weight: 600; transition: 0.3s; }
.btnExit:hover { background: #c0392b; }

.fondo-app {
    position: fixed; top: -200px; left: -20px; width: 102%; height: 150%; background-image: url("/Imagenes/FondoUleam.jpg");
    background-size: cover; background-position: center; filter: blur(8px) brightness(0.6); z-index: -1;
}

.main-layout {
    max-width: 1300px;
    margin: 200px auto;
    height: calc(60vh - 60px);
    display: flex;
    align-items: center;
    justify-content: center;
}

.left-panel {
    width: 320px;
    height: 467px;
    background: rgba(255, 255, 255, 0.9);
    padding: 20px;
    border-radius: 15px;
    display: flex;
    flex-direction: column;
    justify-content: space-between;
    box-shadow: 0 10px 30px rgba(0,0,0,0.3);
}

.map-container {
    flex: 1;
    position: relative;
    background: rgba(255, 255, 255, 0.9);
    border-radius: 15px;
    padding: 10px;
    box-shadow: 0 10px 30px rgba(0,0,0,0.3);
    display: flex;
    align-items: center;
    justify-content: center;
}
```

```
#mapImage { max-width: 100%; max-height: 100%; object-fit: contain; }
.overlay { position: absolute; top: 0; left: 0; width: 100%; height: 100%; pointer-events: none; }
.plaza {
    position: absolute;
    border: 2px solid #28a745;
    background: rgba(40, 167, 69, 0.6);
    color: white;
    font-weight: bold;
    font-size: 0.7rem;
    display: flex;
    align-items: center;
    justify-content: center;
}
.plaza.ocupada { border-color: #dc3545; background: rgba(220, 53, 69, 0.8); }

/* BOTONES */
.buttons-row { display: grid; grid-template-columns: 1fr 1fr; gap: 10px; }
.btn {
    border: none; border-radius: 10px; padding: 15px 5px; cursor: pointer;
    display: flex; flex-direction: column; align-items: center; gap: 5px; font-weight: bold;
}
.ingreso { background: rgba(31, 149, 58, 0.8); }
.salida { background: rgba(220, 53, 69, 0.8); }
.btn img { height: 50px; }

.info-panel { background: #ffffff; padding: 15px; border-radius: 10px; font-weight: bold; font-size: 0.9rem; }
.footer-buttons { display: flex; flex-direction: column; gap: 10px; }
.small {
    width: 100%; padding: 14px;
    background: linear-gradient(135deg, #007bff 0%, #0056b3 100%);
    color: white; border: none; border-radius: 10px;
    margin-top: 50px; font-weight: 700; cursor: pointer;
    box-shadow: 0 5px 15px rgba(0, 123, 255, 0.3);
    transition: 0.2s;
}
.small:hover { background: linear-gradient(135deg, #0056b3 0%, #003f7f 100%); }

.modal {
    position: fixed; top: 0; left: 0; width: 100%; height: 100%;
    background: rgba(0,0,0,0.7);
    display: flex; justify-content: center; align-items: center;
    z-index: 2000;
```

```
}

.modal-content {
    background: white;
    padding: 40px;
    border-radius: 20px;
    width: 90%;
    max-width: 900px;
    max-height: 90vh;
    overflow-y: auto;
    box-shadow: 0 20px 50px rgba(0,0,0,0.3);
}

.modal-content h3 {
    margin-top: 0;
    margin-bottom: 25px;
    font-size: 1.8rem;
    color: #1a202c;
    text-align: center;
}

:global(.registro-table) {
    width: 100%;
    border-collapse: collapse;
    margin-top: 10px;
}

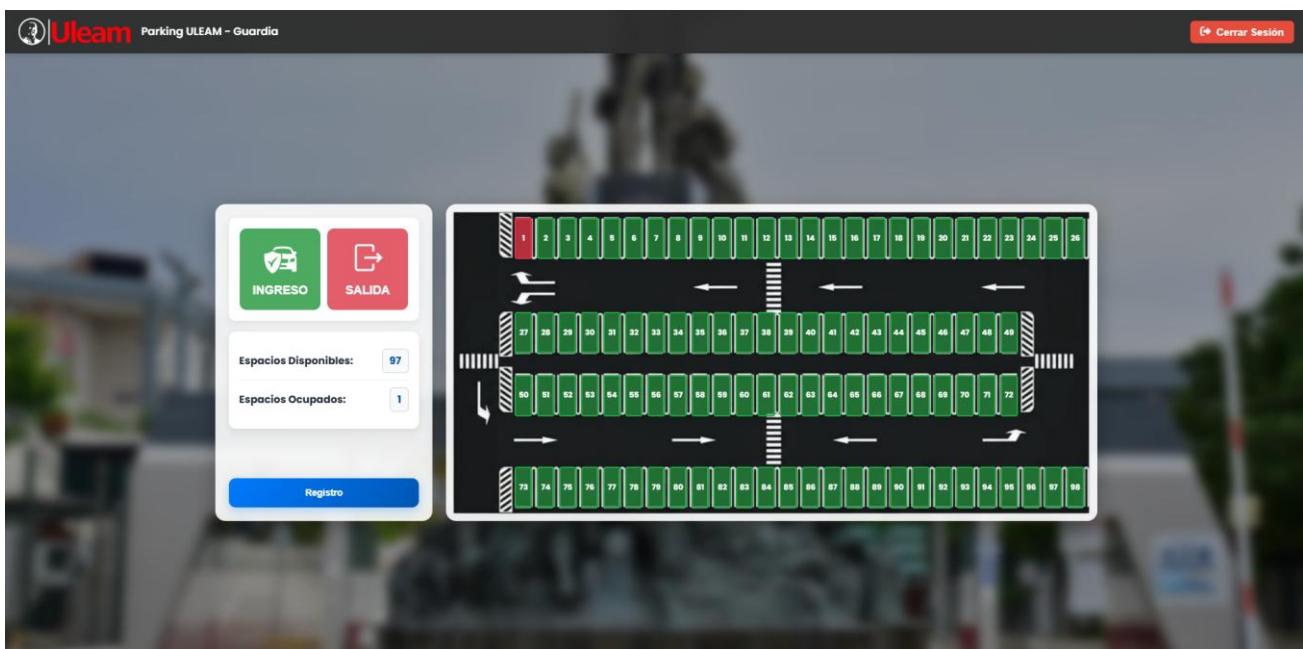
:global(.registro-table th) {
    background: #f1f5f9;
    padding: 16px;
    text-align: left;
    font-size: 1rem;
    color: #334155;
    font-weight: 800;
    border-bottom: 2px solid #e2e8f0;
}

:global(.registro-table td) {
    padding: 16px;
    border-bottom: 1px solid #e2e8f0;
    font-size: 1rem;
    color: #1e293b;
    font-weight: 500;
}

.close {
    float: right;
```

```

        cursor: pointer;
        font-size: 2rem;
        font-weight: bold;
        color: #ef4444;
        line-height: 20px;
    }
    .close:hover { color: #dc2626; }
</style>
```



Dentro de esta pestaña y todas las demás asociadas a este rol, se va a encontrar el botón de “registro”, en donde se desplegará información relacionada a los vehículos estacionados; además de esto hay otros 2 grandes botones, uno para gestionar el ingreso de vehículos y otro para gestionar la salida de ellos. Si damos clic en el apartado de ingreso se nos redireccionará a la siguiente pestaña:

```

<script>
    import { onMount } from 'svelte';
    import { goto } from '$app/navigation';
    import { supabase } from '$lib/supabaseClient';

    onMount(() => {
        const usuario = localStorage.getItem('usuarioActual');
        if (!usuario) {
            goto('/', { replaceState: true });
        }
        cargarDatosSupabase();
    });

    // --- 1. ESTADO GLOBAL ---
    let plazasEstacionamiento = new Array(98).fill(false);
    let registrosCargados = [];

    // Contadores reactivos
    $: ocupados = plazasEstacionamiento.filter(Boolean).length;
    $: disponibles = plazasEstacionamiento.length - ocupados;

    // --- 2. MODALES ---
    let mostrarModalPlaza = false;
    let plazaSeleccionada = null;
    let inputNombre = '';
    let inputPlaca = '';

    let mostrarModalRegistro = false;
    let tablaRegistroHTML = '';

    // Configuración de las plazas
    const filas = [
        { start: 1, count: 26, top: 4.4, marginLeft: 10.4, marginRight: 1.5,
plazaHeight: 13, widthFactor: 0.8, innerOffset: 0, shiftPct: 0 },
        { start: 27, count: 23, top: 34.2, marginLeft: 10.4, marginRight: 11.7,
plazaHeight: 13, widthFactor: 0.8, innerOffset: 0.05, shiftPct: 0 },
        { start: 50, count: 23, top: 54, marginLeft: 10.4, marginRight: 11.7,
plazaHeight: 13, widthFactor: 0.8, innerOffset: 0.05, shiftPct: 0 },
        { start: 73, count: 26, top: 83, marginLeft: 10.4, marginRight: 1.5,
plazaHeight: 13, widthFactor: 0.8, innerOffset: 0.05, shiftPct: 0 }
    ];

    // --- 3. LÓGICA DE DATOS (SUPABASE) ---
    async function cargarDatosSupabase() {
        try {
            const { data, error } = await supabase

```

```
.from('Vehiculos')
.select('*')
.eq('en_estacionamiento', true);

if (error) throw error;

registrosCargados = data;

let nuevoEstadoPlazas = new Array(98).fill(false);
data.forEach(v => {
    if (v.plaza_actual && v.plaza_actual >= 1 && v.plaza_actual <= 98) {
        nuevoEstadoPlazas[v.plaza_actual - 1] = true;
    }
});
plazasEstacionamiento = nuevoEstadoPlazas;

} catch (error) {
    console.error('Error al cargar datos:', error.message);
}
}

function openRegistroPlazaModal(numero) {
    if (plazasEstacionamiento[numero - 1]) {
        alert('Esta plaza ya está ocupada');
        return;
    }
    inputNombre = '';
    inputPlaca = '';
    plazaSeleccionada = numero;
    mostrarModalPlaza = true;
}

function cerrarModalPlaza() { mostrarModalPlaza = false; }

// LÓGICA PRINCIPAL: REGISTRAR EL INGRESO
async function manejarRegistroPlaza(e) {
    e.preventDefault();

    if (!inputPlaca) {
        alert("Por favor ingrese la placa del vehículo.");
        return;
    }

    const placaBuscar = inputPlaca.trim().toUpperCase();

    try {
        // 1. Verificar si el vehículo existe
```

```
const { data: vehiculo, error } = await supabase
    .from('Vehiculos')
    .select('*')
    .eq('placa', placaBuscar)
    .maybeSingle();

if (error) throw error;

if (!vehiculo) {
    alert(`⚠️ ERROR: Vehículo no registrado en el sistema.`);
    return;
}

// 2. Verificar si ya está adentro
if (vehiculo.en_estacionamiento) {
    alert(`⚠️ Este vehículo ya figura dentro del estacionamiento (Plaza ${vehiculo.plaza_actual}).`);
    return;
}

// 3. Registrar Ingreso
const { error: updateError } = await supabase
    .from('Vehiculos')
    .update({
        en_estacionamiento: true,
        plaza_actual: plazaSeleccionada
    })
    .eq('id', vehiculo.id);

if (updateError) throw updateError;

// 4. GUARDAR EN HISTORIAL
await supabase.from('Historial_Ingresos').insert({
    placa: placaBuscar,
    plaza: plazaSeleccionada,
    tipo_movimiento: 'entrada'
});

alert(`✅ Acceso concedido.\nVehículo: ${vehiculo.marca}
${vehiculo.modelo}\nPlaza: ${plazaSeleccionada}`);

cargarDatosSupabase();
cerrarModalPlaza();

} catch (err) {
    alert("Error de conexión: " + err.message);
}
```

```

}

function generarTablaRegistro() {
    let tabla = `<table class="registro-
table"><thead><tr><th>#</th><th>Vehículo</th><th>Placa</th><th>Plaza</th><th>Estado</th></tr></thead><tbody>`;

    registrosCargados.forEach((r, i) => {
        tabla += `<tr>
            <td>${i + 1}</td>
            <td>${r.marca || ''} ${r.modelo || ''}</td>
            <td><strong>${r.placa}</strong></td>
            <td>${r.plaza_actual}</td>
            <td style="color:green; font-weight:bold">En Sitio</td>
        </tr>`;
    });
    tabla += `</tbody></table>`;
    tablaRegistroHTML = tabla;
}

function mostrarRegistroVehiculos() {
    generarTablaRegistro();
    mostrarModalRegistro = true;
}

function cerrarSesion() {
    localStorage.removeItem('usuarioActual');
    goto('/', { replaceState: true });
}

function volverHome() { goto('/home'); }
</script>

<div class="top">
    <div class="logo-section">
        
        <span>Parking ULEAM - Guardia</span>
    </div>
    <button class="btn-exit" on:click={cerrarSesion}>
        <i class="fas fa-sign-out-alt"></i> Cerrar Sesión
    </button>
</div>

<div class="fondo-app"></div>

<main class="main-layout">
    <div class="left-panel">
```

```

<div class="info-box-ingreso">
    <h3>Registrar Vehículo</h3>
    <p>Seleccione una plaza en el mapa</p>
</div>

<div class="info-panel">
    <div>Espacios Disponibles: <span
id="espaciosDisponibles">{disponibles}</span></div>
    <div>Espacios Ocupados: <span
id="espaciosOcupados">{ocupados}</span></div>
</div>

<div class="footer-buttons">
    <button on:click={mostrarRegistroVehiculos} class="btn
small">Registro</button>
    <button on:click={volverHome} class="btn small" style="background:
#e74c3c;">Volver</button>
</div>
</div>

<div class="map-container">
    

    <div id="estacionamiento" class="overlay">
        {@each filas as row}
        {@const count = row.count}
        {@const marginLeft = row.marginLeft || 2.5}
        {@const marginRight = row.marginRight || 2.5}
        {@const plazaHeightPct = row.plazaHeight || 6}
        {@const widthFactor = row.widthFactor || 0.9}
        {@const spacing = (100 - marginLeft - marginRight) / count}

        {@each Array(count) as _, i}
            {@const numero = row.start + i}
            {@const leftPct = marginLeft + i * spacing + spacing *
(row.innerOffset || 0.05) + (row.shiftPct || 0)}
            {@const widthPct = spacing * widthFactor}
            {@const topPct = row.top}

            <div
                class="plaza"
                class:ocupada={plazasEstacionamiento[numero - 1]}
                style="left: {leftPct}%; top: {topPct}%; width: {widthPct}%;
height: {plazaHeightPct}%;">
                data-numero={numero}
                on:click={() => openRegistroPlazaModal(numero)}
            >
        
```

```
                {numero}
            </div>
        {/each}
    {/each}
</div>
</div>
</main>

{#if mostrarModalPlaza}
<div class="modal">
    <div class="modal-content">
        <span class="close" on:click={cerrarModalPlaza}>&times;</span>
        <h3>Registro en Plaza {plazaSeleccionada}</h3>

        <form on:submit|preventDefault={manejarRegistroPlaza} class="form-
registro">
            <div style="margin-bottom: 15px;">
                <label style="display:block; font-weight:600; margin-
bottom:5px;">Placa Vehículo</label>
                <input bind:value={inputPlaca} type="text" required
placeholder="Ingrese Placa" style="width:100%; padding:12px; border:2px solid #eee;
border-radius:8px; font-size: 1.1rem; text-transform: uppercase;" />
            </div>
            <button type="submit" class="btn small" style="margin-top:
10px;">Validar e Ingresar</button>
        </form>
    </div>
</div>
{/if}

{#if mostrarModalRegistro}
<div class="modal">
    <div class="modal-content">
        <span class="close" on:click={() => mostrarModalRegistro =
false}>&times;</span>
        <h3>Vehículos Dentro</h3>
        <div class="table-wrapper">
            {@html tablaRegistroHTML}
        </div>
    </div>
</div>
{/if}

<style>
    @import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;600;700;800&display=s
wap');

```

```
:global(body) {  
    margin: 0; padding: 0;  
    font-family: 'Poppins', sans-serif;  
    background-color: #f4f4f4;  
    overflow: hidden;  
}  
  
.top {  
    background: rgba(45, 45, 45, 0.95);  
    backdrop-filter: blur(10px);  
    height: 64px;  
    display: flex;  
    justify-content: space-between;  
    align-items: center;  
    padding: 0 20px;  
    color: white;  
    z-index: 100;  
    box-shadow: 0 4px 6px rgba(0,0,0,0.1);  
    position: relative;  
}  
.logo-section { display: flex; align-items: center; gap: 15px; font-weight: 600;  
}  
.logo-section img { height: 40px; }  
.btn-exit {  
    background: #e74c3c;  
    color: white;  
    border: none;  
    padding: 8px 15px;  
    border-radius: 6px; cursor:  
    pointer; font-weight: 600;  
    transition: 0.3s;  
}  
.btn-exit:hover { background: #c0392b; }  
  
.fondo-app {  
    position: fixed; top: -200px; left: -20px; width: 102%; height: 150%;  
    background-image: url("/Imagenes/FondoUleam.jpg");  
    background-size: cover; background-position: center;  
    filter: blur(8px) brightness(0.6);  
    z-index: -1;  
}  
  
.main-layout {  
    max-width: 1300px;  
    margin: 200px auto;  
    height: calc(60vh - 60px);
```

```
display: flex;
align-items: center;
justify-content: center;
gap: 20px;
}

.left-panel {
  width: 320px;
  height: 467px;
  background: rgba(255, 255, 255, 0.9);
  padding: 20px;
  border-radius: 15px;
  display: flex;
  flex-direction: column;
  justify-content: space-between;
  box-shadow: 0 10px 30px rgba(0,0,0,0.3);
}

.map-container {
  flex: 1;
  position: relative;
  background: rgba(255, 255, 255, 0.9);
  border-radius: 15px;
  padding: 10px;
  box-shadow: 0 10px 30px rgba(0,0,0,0.3);
  display: flex;
  align-items: center;
  justify-content: center;
  height: 467PX;
}

#mapImage { max-width: 100%; max-height: 100%; object-fit: contain; }
.overlay { position: absolute; top: 0; left: 0; width: 100%; height: 100%; pointer-events: none; }

.plaza {
  position: absolute;
  border: 2px solid #28a745;
  background: rgba(40, 167, 69, 0.6);
  color: white;
  font-weight: bold;
  font-size: 0.7rem;
  display: flex;
  align-items: center;
  justify-content: center;
}

.plaza.ocupada { border-color: #dc3545; background: rgba(220, 53, 69, 0.8); }
```

```
.info-box-ingreso { margin-bottom: 20px; text-align: center; }
.info-box-ingreso h3 { margin: 0; color: #333; font-weight: 800; font-size: 1.5rem; }
.info-box-ingreso p { color: #666; font-size: 0.9rem; margin: 5px 0 0; }

.info-panel { background: #ffffff; padding: 15px; border-radius: 10px; font-weight: bold; font-size: 0.9rem; }

.footer-buttons { display: flex; flex-direction: column; gap: 10px; }

.btn { width: 100%; border: none; border-radius: 10px; padding: 15px 5px; cursor: pointer; font-weight: bold; display: block; text-align: center; }

.small {
    width: 100%; padding: 14px;
    background: linear-gradient(135deg, #007bff 0%, #0056b3 100%);
    color: white;
    box-shadow: 0 5px 15px rgba(0, 123, 255, 0.3);
    transition: 0.2s;
}
.small:hover { background: linear-gradient(135deg, #0056b3 0%, #003f7f 100%) }

/* ESTILOS DEL MODAL IGUALADOS AL HOME */
.modal {
    position: fixed; top: 0; left: 0; width: 100%; height: 100%;
    background: rgba(0,0,0,0.7);
    display: flex; justify-content: center; align-items: center;
    z-index: 2000;
}

.modal-content {
    background: white;
    padding: 40px; /* Igual al Home */
    border-radius: 20px; /* Igual al Home */
    width: 90%; /* Igual al Home */
    max-width: 900px; /* Igual al Home */
    max-height: 90vh; /* Igual al Home */
    overflow-y: auto;
    box-shadow: 0 20px 50px rgba(0,0,0,0.3); /* Igual al Home */
}

.modal-content h3 {
    margin-top: 0;
    margin-bottom: 25px;
    font-size: 1.8rem;
    color: #1a202c;
```

```
        text-align: center;
    }

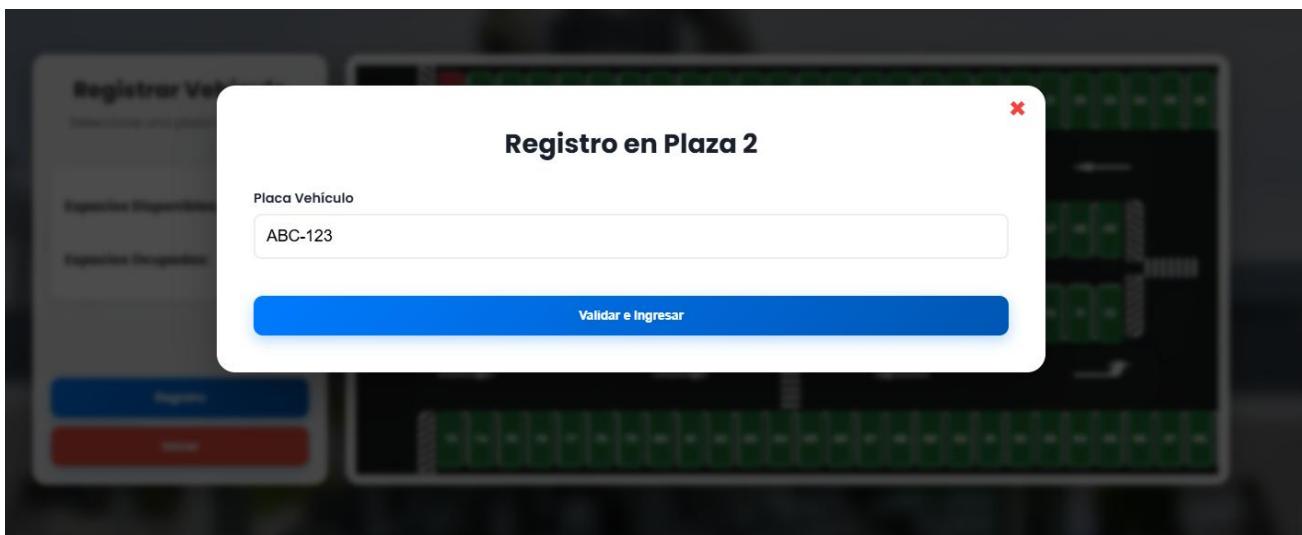
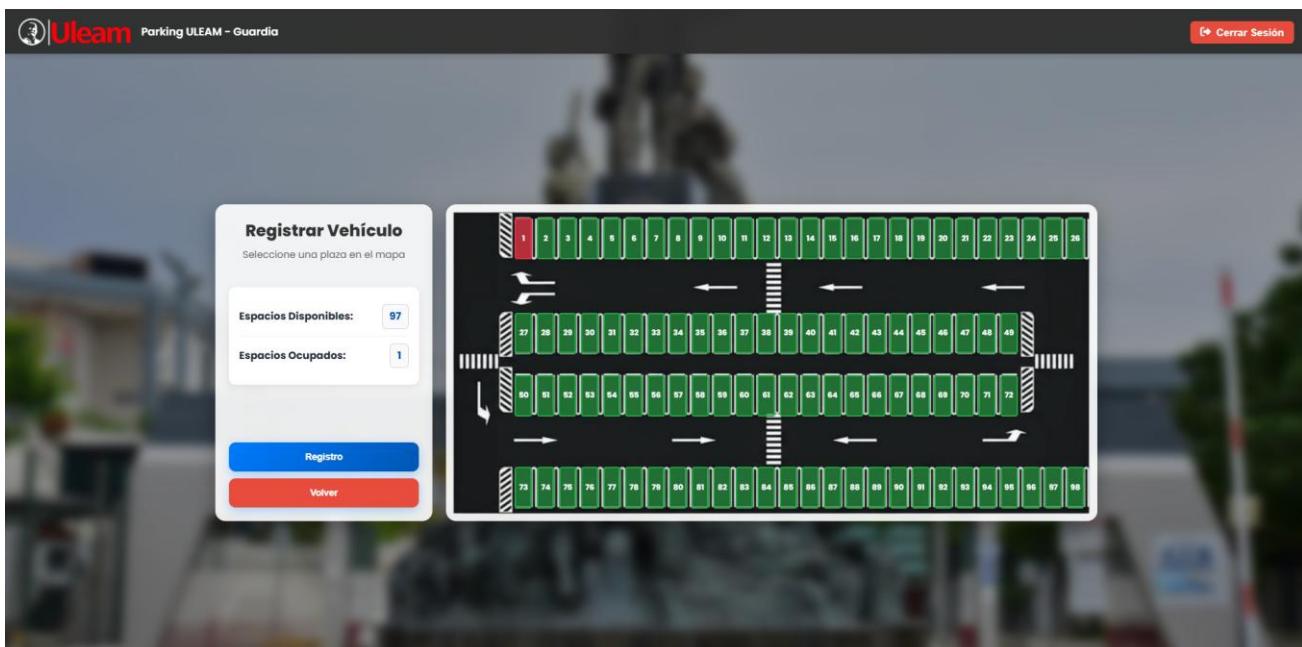
.table-wrapper { overflow-x: auto; }

:global(.registro-table) {
    width: 100%;
    border-collapse: collapse;
    margin-top: 10px;
}

:global(.registro-table th) {
    background: #f1f5f9;
    padding: 16px;
    text-align: left;
    font-size: 1rem;
    color: #334155;
    font-weight: 800;
    border-bottom: 2px solid #e2e8f0;
}

:global(.registro-table td) {
    padding: 16px;
    border-bottom: 1px solid #e2e8f0;
    font-size: 1rem;
    color: #1e293b;
    font-weight: 500;
}

.close {
    float: right;
    cursor: pointer;
    font-size: 2rem;
    font-weight: bold;
    color: #ef4444;
    line-height: 20px;
}
.close:hover { color: #dc2626; }
</style>
```



Para poder liberar una plaza al momento en el que se retira un vehículo, simplemente damos en volver y seleccionamos la opción de “salida”, con esto, se nos llevará al siguiente apartado:

```
<script>
  import { onMount } from 'svelte';
  import { goto } from '$app/navigation';
  import { supabase } from '$lib/supabaseClient';

  onMount(() => {
    const form = document.querySelector('#registro-form');
    const inputPlaca = document.querySelector('#placa');
    const buttonValidar = document.querySelector('#validar');
    const buttonIngresar = document.querySelector('#ingresar');

    buttonValidar.addEventListener('click', () => {
      const placa = inputPlaca.value;
      if (placa) {
        goto('/salida');
      }
    });
  });
</script>
```

```
const usuario = localStorage.getItem('usuarioActual');
if (!usuario) {
    goto('/', { replaceState: true });
}
cargarDatosSupabase();
});

// --- 1. ESTADO GLOBAL ---
let plazasEstacionamiento = new Array(98).fill(false);

// Contadores reactivos
$: ocupados = plazasEstacionamiento.filter(Boolean).length;
$: disponibles = plazasEstacionamiento.length - ocupados;

// --- 2. MODALES ---
let mostrarModalConfirmacion = false;
let mostrarPanelResultado = false;

let plazaSeleccionada = null;
let inputPlacaConfirmacion = '';
let registroSeleccionado = null;
let mostrarModalRegistro = false;
let tablaRegistroHTML = '';
let registrosCargados = [];

// Configuración de las plazas
const filas = [
    { start: 1, count: 26, top: 4.4, marginLeft: 10.4, marginRight: 1.5,
plazaHeight: 13, widthFactor: 0.8, innerOffset: 0, shiftPct: 0 },
    { start: 27, count: 23, top: 34.2, marginLeft: 10.4, marginRight: 11.7,
plazaHeight: 13, widthFactor: 0.8, innerOffset: 0.05, shiftPct: 0 },
    { start: 50, count: 23, top: 54, marginLeft: 10.4, marginRight: 11.7,
plazaHeight: 13, widthFactor: 0.8, innerOffset: 0.05, shiftPct: 0 },
    { start: 73, count: 26, top: 83, marginLeft: 10.4, marginRight: 1.5,
plazaHeight: 13, widthFactor: 0.8, innerOffset: 0.05, shiftPct: 0 }
];

// --- 3. LÓGICA DE DATOS ---
async function cargarDatosSupabase() {
    try {

        const { data, error } = await supabase
            .from('Vehiculos')
            .select('*')
            .eq('en_estacionamiento', true);

        if (error) throw error;

    
```

```
registrosCargados = data;

let nuevoEstadoPlazas = new Array(98).fill(false);
data.forEach(v => {
    if (v.plaza_actual && v.plaza_actual >= 1 && v.plaza_actual <= 98) {
        nuevoEstadoPlazas[v.plaza_actual - 1] = true;
    }
});
plazasEstacionamiento = nuevoEstadoPlazas;

} catch (error) {
    console.error('Error al cargar datos:', error.message);
}
}

// Al hacer clic en una plaza del mapa
function manejarClickPlaza(numero) {
    if (plazasEstacionamiento[numero - 1]) {
        plazaSeleccionada = numero;
        inputPlacaConfirmacion = '';
        mostrarPanelResultado = false;
        mostrarModalConfirmacion = true;
    } else {
        alert('Esta plaza está libre. Solo puede registrar salida de plazas ocupadas.');
    }
}

function cerrarModalConfirmacion() {
    mostrarModalConfirmacion = false;
    plazaSeleccionada = null;
}

// VERIFICACIÓN DE PLACA
async function manejarVerificacionPlaca(e) {
    e.preventDefault();

    const placaIngresada = inputPlacaConfirmacion.trim().toUpperCase();

    if (!placaIngresada) {
        alert('Por favor ingrese la placa del vehículo.');
        return;
    }

    try {
        // Se busca el vehículo en la plaza seleccionada
    }
}
```

```
const { data: vehiculoEnPlaza, error } = await supabase
    .from('Vehiculos')
    .select('*')
    .eq('plaza_actual', plazaSeleccionada)
    .eq('en_estacionamiento', true)
    .maybeSingle();

if (error || !vehiculoEnPlaza) {
    alert("Error: No se encontró registro activo en esta plaza en la
base de datos.");
    return;
}

// Validamos coincidencia
if (vehiculoEnPlaza.placa === placaIngresada) {

    registroSeleccionado = vehiculoEnPlaza;
    mostrarModalConfirmacion = false;
    mostrarPanelResultado = true; n
} else {
    alert(`La placa ingresada (${placaIngresada}) no coincide con el
vehículo en la plaza ${plazaSeleccionada}. (El sistema indica:
${vehiculoEnPlaza.placa})`);
}
} catch (err) {
    alert("Error de conexión: " + err.message);
}
}

// CONFIRMAR SALIDA FINAL
async function confirmarSalida() {
    if (!registroSeleccionado) return;

    try {
        // Liberar el Vehículo (Actualizar BD)
        const { error: updateError } = await supabase
            .from('Vehiculos')
            .update({
                en_estacionamiento: false,
                plaza_actual: null
            })
            .eq('id', registroSeleccionado.id);

        if (updateError) throw updateError;

        // Guardar en HISTORIAL
        await supabase.from('Historial_Ingresos').insert({

```

```
        placa: registroSeleccionado.placa,
        plaza: plazaSeleccionada,
        tipo_movimiento: 'salida'
    });

    alert(`Salida registrada exitosamente.`);

    cargarDatosSupabase(); // Actualizar mapa
    mostrarPanelResultado = false;
    registroSeleccionado = null;

} catch (err) {
    alert('Error al procesar la salida: ' + err.message);
}
}

function generarTablaRegistro() {
    let tabla = `<table class="registro-table"><thead><tr><th>#</th><th>Placa</th><th>Marca/Modelo</th><th>Plaza</th></tr></thead><tbody>`;
    registrosCargados.forEach((r, i) => {
        tabla += `<tr>
            <td>${i + 1}</td>
            <td><strong>${r.placa}</strong></td>
            <td>${r.marca || ''} ${r.modelo || ''}</td>
            <td>${r.plaza_actual}</td>
        </tr>`;
    });
    tabla += `</tbody></table>`;
    tablaRegistroHTML = tabla;
}

function mostrarRegistroVehiculos() {
    generarTablaRegistro();
    mostrarModalRegistro = true;
}

function cerrarSesion() {
    localStorage.removeItem('usuarioActual');
    goto('/', { replaceState: true });
}

function volverHome() { goto('/home'); }

</script>

<div class="top">
    <div class="logo-section">
```

```

<span>Parking ULEAM - Guardia</span>
</div>
<button class="btn-exit" on:click={cerrarSesion}>
    <i class="fas fa-sign-out-alt"></i> Cerrar Sesión
</button>
</div>

<div class="fondo-app"></div>

<main class="main-layout">
    <div class="left-panel">
        <div class="info-box-ingreso">
            <h3>Registrar Salida</h3>
            <p>Seleccione una plaza ocupada (roja) para liberar</p>
        </div>

        <div class="info-panel">
            <div>Especios Disponibles: <span
id="espaciosDisponibles">{disponibles}</span></div>
            <div>Especios Ocupados: <span
id="espaciosOcupados">{ocupados}</span></div>
        </div>

        {#if mostrarPanelResultado && registroSeleccionado}
            <div id="resultadoSalida" class="info-panel resultado-salida">
                <div style="margin-bottom:5px; font-size:1.1em;
color:#d9534f;"><strong>Vehículo a liberar:</strong></div>
                <div>Vehículo: {registroSeleccionado.marca}
{registroSeleccionado.modelo}</div>
                <div>Placa: <strong>{registroSeleccionado.placa}</strong></div>
                <div>Plaza: {registroSeleccionado.plaza_actual}</div>

                <div style="margin-top:15px">
                    <button on:click={confirmarSalida} class="btn small"
style="background: #dc3545; box-shadow: none;">Confirmar Liberación</button>
                </div>
            </div>
        {/if}

        <div class="footer-buttons">
            <button on:click={mostrarRegistroVehiculos} class="btn
small">Registro</button>
            <button on:click={volverHome} class="btn small" style="background:
#e74c3c;">Volver</button>
        </div>
    </div>
</div>
```

```

<div class="map-container">
    
    <div id="estacionamiento" class="overlay">
        {#each filas as row}
            {@const count = row.count}
            {@const marginLeft = row.marginLeft || 2.5}
            {@const marginRight = row.marginRight || 2.5}
            {@const plazaHeightPct = row.plazaHeight || 6}
            {@const widthFactor = row.widthFactor || 0.9}
            {@const spacing = (100 - marginLeft - marginRight) / count}

            {#each Array(count) as _, i}
                {@const numero = row.start + i}
                {@const leftPct = (row.marginLeft || 2.5) + i * spacing +
spacing * (row.innerOffset || 0.05) + (row.shiftPct || 0)}
                {@const widthPct = spacing * widthFactor}
                {@const topPct = row.top}

                <div
                    class="plaza"
                    class:ocupada={plazasEstacionamiento[numero - 1]}
                    style="left: {leftPct}%; top: {topPct}%; width: {widthPct}%;
height: {plazaHeightPct}%;"
                    data-numero={numero}
                    on:click={() => manejarClickPlaza(numero)}
                >
                    {numero}
                </div>
            {/each}
        {/each}
    </div>
</main>

{#if mostrarModalConfirmacion}
    <div id="modalConfirmacion" class="modal">
        <div class="modal-content">
            <span class="close" on:click={cerrarModalConfirmacion}>&times;</span>
            <h3>Confirmar Salida - Plaza {plazaSeleccionada}</h3>

            <form on:submit|preventDefault={manejarVerificacionPlaca} class="form-
registro">
                <p style="margin-bottom: 15px; color: #555;">Ingrese la placa para
verificar que corresponde al vehículo correcto:</p>
                <div style="margin-bottom: 15px;">

```

```
        <label style="display:block; font-weight:600; margin-bottom:5px;">Placa del vehículo</label>
            <input bind:value={inputPlacaConfirmacion} type="text"
placeholder="Ej: ABC-1234" required style="width:100%; padding:8px; border:1px solid #ccc; border-radius:5px; font-size:1.1rem; text-transform:uppercase;" />
        </div>

        <div class="form-actions">
            <button type="button" on:click={cerrarModalConfirmacion}
class="btn-cancel">Cancelar</button>
            <button type="submit" class="btn-confirm">Verificar</button>
        </div>
    </form>
</div>
</div>
{/if}

{#if mostrarModalRegistro}
<div class="modal">
    <div class="modal-content table-content">
        <span class="close" on:click={() => mostrarModalRegistro =
false}>&times;</span>
        <h3>Vehículos en Estacionamiento</h3>
        <div class="table-wrapper">
            {@html tablaRegistroHTML}
        </div>
    </div>
</div>
{/if}

<style>
/* CSS ORIGINAL DE TU CÓDIGO (INTACTO) */
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;600;700;800&display=swap');

:global(body) {
    margin: 0; padding: 0;
    font-family: 'Poppins', sans-serif;
    background-color: #f4f4f4;
    overflow: hidden;
}

.top {
    background: rgba(45, 45, 45, 0.95);
    backdrop-filter: blur(10px);
    height: 64px;
```

```
display: flex;
justify-content: space-between;
align-items: center;
padding: 0 20px;
color: white;
z-index: 100;
box-shadow: 0 4px 6px rgba(0,0,0,0.1);
position: relative;
}
.logo-section { display: flex; align-items: center; gap: 15px; font-weight: 600;
}
.logo-section img { height: 40px; }
.btnExit { background: #e74c3c; color: white; border: none; padding: 8px 15px;
border-radius: 6px; cursor: pointer; font-weight: 600; transition: 0.3s; }
.btnExit:hover { background: #c0392b; }

.fondo-app {
    position: fixed; top: -200px; left: -20px; width: 102%; height: 150%;
    background-image: url("/Imagenes/FondoUleam.jpg");
    background-size: cover; background-position: center;
    filter: blur(8px) brightness(0.6);
    z-index: -1;
}

.main-layout {
    max-width: 1300px;
    margin: 200px auto;
    height: calc(60vh - 60px);
    display: flex;
    align-items: center;
    justify-content: center;
    gap: 20px;
}

.left-panel {
    width: 320px;
    height: 467px;
    background: rgba(255, 255, 255, 0.9);
    padding: 20px;
    border-radius: 15px;
    display: flex;
    flex-direction: column;
    justify-content: space-between;
    box-shadow: 0 10px 30px rgba(0,0,0,0.3);
}

.map-container {
```

```
flex: 1;
position: relative;
background: rgba(255, 255, 255, 0.9);
border-radius: 15px;
padding: 10px;
box-shadow: 0 10px 30px rgba(0,0,0,0.3);
display: flex;
align-items: center;
justify-content: center;
height: 91.5%;
```

```
}
```

```
#mapImage { max-width: 100%; max-height: 100%; object-fit: contain; }
.overlay { position: absolute; top: 0; left: 0; width: 100%; height: 100%; pointer-events: none; }
```

```
.plaza {
  position: absolute;
  border: 2px solid #28a745;
  background: rgba(40, 167, 69, 0.6);
  color: white;
  font-weight: bold;
  font-size: 0.7rem;
  display: flex;
  align-items: center;
  justify-content: center;
}
```

```
.plaza.ocupada { border-color: #dc3545; background: rgba(220, 53, 69, 0.8); }
```

```
.info-box-ingreso { margin-bottom: 20px; text-align: center; }
.info-box-ingreso h3 { margin: 0; color: #333; font-weight: 800; font-size: 1.5rem; }
.info-box-ingreso p { color: #666; font-size: 0.9rem; margin: 5px 0 0; }
```

```
.info-panel { background: #ffffff; padding: 15px; border-radius: 10px; font-weight: bold; font-size: 0.9rem; }
.resultado-salida { border-left: 5px solid #dc3545; background: #fff5f5; margin-bottom: 10px; }
```

```
.footer-buttons { display: flex; flex-direction: column; gap: 10px; }
```

```
.btn { width: 100%; border: none; border-radius: 10px; padding: 15px 5px; cursor: pointer; font-weight: bold; display: block; text-align: center; color: white; transition: 0.2s; }
.small {
  width: 100%; padding: 14px;
  background: linear-gradient(135deg, #007bff 0%, #0056b3 100%);
```

```
color: white;
box-shadow: 0 5px 15px rgba(0, 123, 255, 0.3);
transition: 0.2s;
}
.small:hover { background: linear-gradient(135deg, #0056b3 0%, #003f7f 100%) }

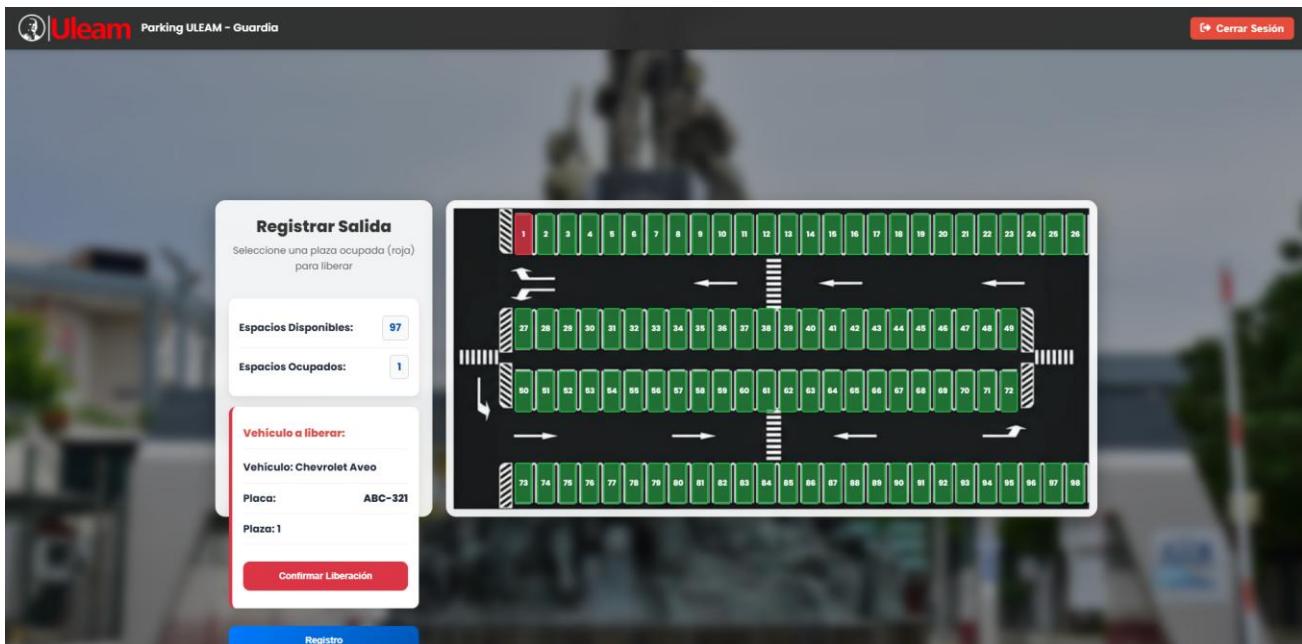
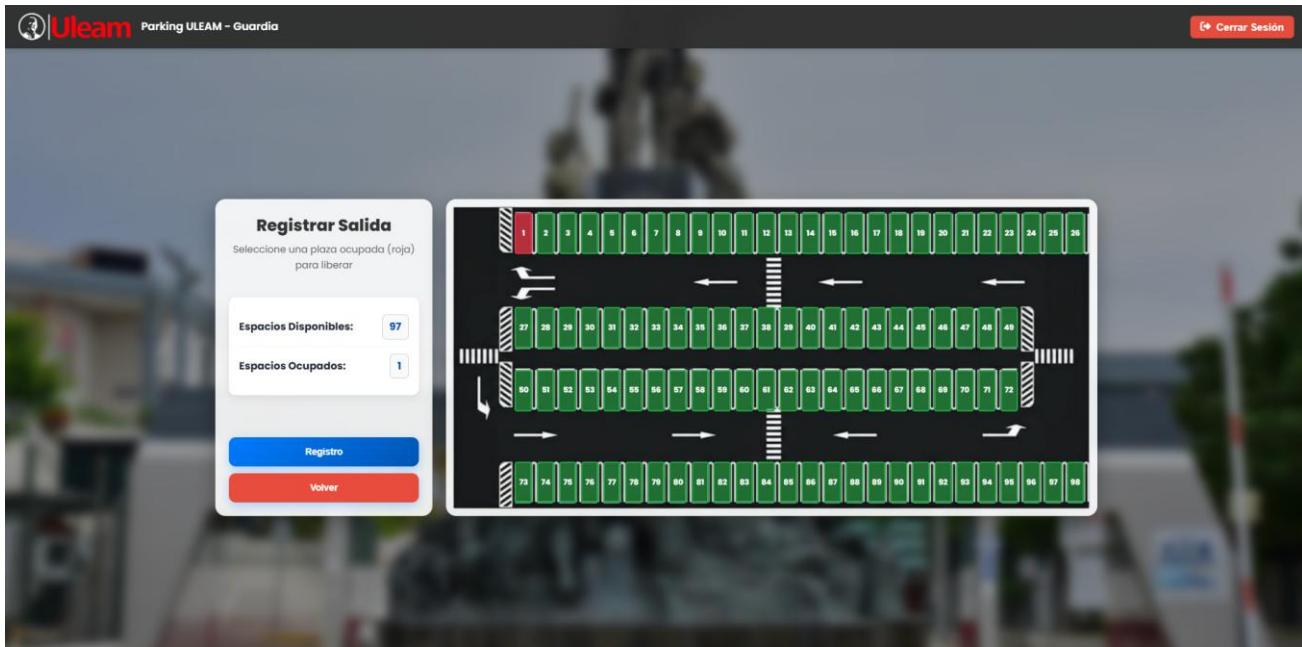
/* MODAL */
.modal { position: fixed; top: 0; left: 0; width: 100%; height: 100%; background: rgba(0,0,0,0.7); display: flex; justify-content: center; align-items: center; z-index: 2000; }
.modal-content { background: white; padding: 30px; border-radius: 15px; width: 90%; max-width: 400px; box-shadow: 0 25px 50px rgba(0,0,0,0.5); }
.table-content { max-width: 900px; max-height: 80vh; overflow-y: auto; } /* Ajuste para tabla grande */

.close { float: right; cursor: pointer; font-size: 1.5rem; color: #999; }

.form-actions { display: flex; gap: 10px; margin-top: 20px; }
.btn-cancel { flex: 1; padding: 10px; border: none; background: #dc3545; border-radius: 8px; cursor: pointer; font-weight: 600; }
.btn-confirm { flex: 1; padding: 10px; border: none; background: #0056b3; color: white; border-radius: 8px; cursor: pointer; font-weight: 600; }

.table-wrapper { overflow-x: auto; }
:global(.registro-table) { width: 100%; border-collapse: collapse; margin-top: 15px; }
:global(.registro-table th) { background: #eee; padding: 10px; text-align: left; }
:global(.registro-table td) { padding: 10px; border-bottom: 1px solid #eee; }
</style>
```





Finalmente, si llegamos a entrar con una cuenta que tenga asociado el rol de “admin”, podremos gestionar las solicitudes de actualización de información de los usuarios registrados, además claro de ver un registro detallado de todos los vehículos y sus respectivos dueños.

```
<script>
  import { onMount } from 'svelte';
  import { supabase } from '$lib/supabaseClient';
  import { goto } from '$app/navigation';

  let pestanaActual = 'inventario'; // 'inventario' / 'solicitudes'
  let vehiculos = [];
  let solicitudes = [];
  let cargando = false;
  let usuarioAdmin = null;

  onMount(async () => {
    await verificarAdmin();
    await cargarDatos();
  });

  async function verificarAdmin() {
    const { data: { session } } = await supabase.auth.getSession();
    if (!session) goto('/');

    // Verificar rol
    const { data: usuario } = await supabase
      .from('Usuarios')
      .select('rol, nombres')
      .eq('id', session.user.id)
      .single();

    if (usuario?.rol !== 'admin') goto('/home');
    usuarioAdmin = usuario;
  }

  async function cargarDatos() {
    cargando = true;

    // 1. Inventario
    const { data: datosVehiculos } = await supabase
      .from('Vehiculos')
      .select(`*, Usuarios ( nombres, apellidos, cedula )`)
      .order('created_at', { ascending: false });

    if (datosVehiculos) vehiculos = datosVehiculos;

    // 2. Solicitudes
    const { data: datosSolicitudes } = await supabase
      .from('Solicitudes')
      .select(`*, Usuarios ( nombres, apellidos, cedula ), Vehiculos ( placa )`)
      .eq('estado', 'pendiente');
```

```

        if (datosSolicitudes) solicitudes = datosSolicitudes;
        cargando = false;
    }

    async function procesarSolicitud(solicitud, accion) {
        if(!confirm(`¿Deseas ${accion.toUpperCase()} esta solicitud?`)) return;

        try {
            if (accion === 'aprobar') {
                const cambios = solicitud.datos_nuevos;
                const { error: errUpdate } = await supabase.from('Vehiculos').update({
                    marca: cambios.marca, modelo: cambios.modelo, color: cambios.color,
                    placa: cambios.placa
                }).eq('id', solicitud.vehiculo_id);
                if (errUpdate) throw errUpdate;
            }

            const { error: errEstado } = await supabase.from('Solicitudes')
                .update({ estado: accion === 'aprobar' ? 'aprobada' : 'rechazada' })
                .eq('id', solicitud.id);

            if (errEstado) throw errEstado;

            alert(`Solicitud ${accion === 'aprobar' ? 'APROBADA' : 'RECHAZADA'} correctamente.`);
            cargarDatos();
        } catch (error) {
            alert("Error: " + error.message);
        }
    }

    // --- CORRECCIÓN AQUÍ: Renombrada a cerrarSesion ---
    async function cerrarSesion() {
        await supabase.auth.signOut(); // Cierra sesión en Supabase
        localStorage.removeItem('usuarioActual'); // Limpia caché Local
        goto('/', { replaceState: true }); // Redirige al Login
    }
</script>

<div class="top">
    <div class="logo-container">
        
        {#if usuarioAdmin}
            <span class="admin-badge">Admin: {usuarioAdmin.nombres}</span>
        {/if}
    </div>
</div>

```

```
</div>
<button class="btn-logout" on:click={cerrarSesion}>
    <i class="fas fa-sign-out-alt"></i> Cerrar Sesión
</button>
</div>

<div class="main-container">

<div class="header-section">
    <div class="titulo-pagina">Panel de Control</div>
    <div class="tabs">
        <button
            class:activo={pestanaActual === 'inventario'}
            on:click={() => pestanaActual = 'inventario'}>
            🚙 Inventario
        </button>
        <button
            class:activo={pestanaActual === 'solicitudes'}
            on:click={() => pestanaActual = 'solicitudes'}>
            📋 Solicitudes
            {#if solicitudes.length > 0}
                <span class="contador">{solicitudes.length}</span>
            {/if}
        </button>
    </div>
</div>

<div class="contenido">
{#if cargando}
    <div class="loading">Cargando datos...</div>
{:#else}

{#if pestanaActual === 'inventario'}
    <div class="card-table">
        <table>
            <thead>
                <tr>
                    <th>Propietario</th>
                    <th>Cédula</th>
                    <th>Vehículo</th>
                    <th>Placa</th>
                    <th>Estado</th>
                </tr>
            </thead>
            <tbody>
                {#each vehiculos as v}
                    <tr>
```

```

        <td class="resaltado">{v.Usuarios?.nombres}
{v.Usuarios?.apellidos}</td>
        <td>{v.Usuarios?.cedula}</td>
        <td>{v.marca} {v.modelo} <span class="color-dot"
style="background-color: {v.color}"></span></td>
        <td><span class="placa-badge">{v.placa}</span></td>
        <td>
            {#if v.en_estacionamiento}
                <span class="status dentro">En Parking
(P{v.plaza_actual})</span>
            {:else}
                <span class="status fuera">Fuera</span>
           {/if}
        </td>
        </tr>
    {:else}
        <tr><td colspan="5" class="vacio">No hay vehículos
registrados</td></tr>
    {:each}
        </tbody>
    </table>
</div>
{/if}

{#if pestanaActual === 'solicitudes'}
<div class="grid-cards">
    {#each solicitudes as s}
        <div class="card-solicitud">
            <div class="card-header">
                <span class="fecha">{new
Date(s.created_at).toLocaleDateString()}</span>
                <strong>{s.Usuarios?.nombres} {s.Usuarios?.apellidos}</strong>
            </div>
            <div class="card-body">
                <div class="cambio-row">
                    <div class="col">
                        <small>Actual</small>
                        <div class="dato viejo">{s.Vehiculos?.placa || 'Sin
Datos'}</div>
                    </div>
                    <div class="arrow">→</div>
                    <div class="col">
                        <small>Solicitado</small>
                        <ul class="lista-cambios">
                            <li><strong>Placa:</strong> {s.datos_nuevos?.placa}</li>
                            <li><strong>Auto:</strong> {s.datos_nuevos?.marca}
{s.datos_nuevos?.modelo}</li>
                        </ul>
                    </div>
                </div>
            </div>
        </div>
    </each>
</div>

```

```
          <li><strong>Color:</strong> {s.datos_nuevos?.color}</li>
        </ul>
      </div>
    </div>
  </div>
  <div class="card-actions">
    <button class="boton-accion rechazar" on:click={() =>
procesarSolicitud(s, 'rechazar')}>Rechazar</button>
    <button class="boton-accion aprobar" on:click={() =>
procesarSolicitud(s, 'aprobar')}>Aprobar</button>
  </div>
</div>
{:else}
  <div class="empty-state">
    <p>✿ No hay solicitudes pendientes</p>
  </div>
{/each}
</div>
{/if}

{/if}
</div>
</div>

<style>
  @import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600;700;800&display=swap');

:global(body) { margin: 0; font-family: 'Poppins', sans-serif; background:
#f0f2f5; }

/* BARRA SUPERIOR (Estilo Original) */
.top {
  background: rgba(45, 45, 45, 0.95);
  backdrop-filter: blur(10px);
  position: fixed;
  top: 0; left: 0;
  width: 100%; height: 65px;
  display: flex; justify-content: space-between; align-items: center;
  z-index: 100;
  padding: 0 20px;
  box-sizing: border-box;
  border-bottom: 1px solid rgba(255, 255, 255, 0.1);
}

.logo-container { height: 100%; display: flex; align-items: center; gap: 15px; }
```

```
.top img { height: 40px; filter: drop-shadow(0 2px 4px rgba(0,0,0,0.5)); }

.admin-badge {
  color: #edf2f7;
  font-weight: 600;
}

.btn-logout { background: #e74c3c; color: white; border: none; padding: 8px 15px; border-radius: 6px; cursor: pointer; font-weight: 600; transition: 0.3s; }
.btn-logout:hover { background: #c0392b; }

/* CONTENEDOR PRINCIPAL */
.main-container {
  margin-top: 65px;
  padding: 40px 20px;
  max-width: 1200px;
  margin-left: auto; margin-right: auto;
}

.header-section {
  display: flex; justify-content: space-between; align-items: center;
  margin-bottom: 30px; flex-wrap: wrap; gap: 20px;
}

.titulo-pagina {
  font-size: 1.8rem; font-weight: 800; color: #1a202c;
}

/* PESTAÑAS ESTILO ULEAM */
.tabs {
  display: flex; background: white; padding: 5px; border-radius: 12px;
  box-shadow: 0 4px 6px -1px rgba(0,0,0,0.05);
}

.tabs button {
  background: transparent; border: none; padding: 10px 20px;
  font-family: 'Poppins', sans-serif; font-weight: 600; color: #718096;
  cursor: pointer; border-radius: 8px; transition: all 0.3s;
  display: flex; align-items: center; gap: 8px;
}

.tabs button.activo {
  background: linear-gradient(135deg, #007bff 0%, #0056b3 100%);
  color: white; shadow: 0 4px 12px rgba(0,123,255,0.3);
}

.contador {
```

```

background: #e53e3e; color: white; font-size: 0.7rem; padding: 2px 6px;
border-radius: 10px;
}

/* TABLA DE INVENTARIO */
.card-table {
background: white; border-radius: 20px; padding: 25px;
box-shadow: 0 10px 25px -5px rgba(0,0,0,0.05); overflow-x: auto;
}

table { width: 100%; border-collapse: collapse; }
th { text-align: left; color: #a0aec0; font-size: 0.85rem; text-transform:
uppercase; padding: 15px; border-bottom: 2px solid #edf2f7; }
td { padding: 15px; border-bottom: 1px solid #edf2f7; color: #4a5568; font-size:
0.95rem; }

.resaltado { font-weight: 700; color: #2d3748; }
.placa-badge {
background: #ffeb3b; color: #000; font-weight: 800;
padding: 5px 10px; border-radius: 6px; border: 2px solid #fdd835;
}

.status { padding: 5px 10px; border-radius: 20px; font-size: 0.8rem; font-weight:
600; }
.status.dentro { background: #c6f6d5; color: #22543d; }
.status.fuera { background: #edf2f7; color: #718096; }

/* CARDS SOLICITUDES */
.grid-cards { display: grid; grid-template-columns: repeat(auto-fill,
minmax(320px, 1fr)); gap: 25px; }

.card-solicitud {
background: white; border-radius: 20px; padding: 25px;
box-shadow: 0 10px 20px -5px rgba(0,0,0,0.05);
border: 1px solid rgba(0,0,0,0.02);
display: flex; flex-direction: column; justify-content: space-between;
}

.card-header { border-bottom: 1px solid #edf2f7; padding-bottom: 15px; margin-
bottom: 15px; }
.card-header strong { display: block; font-size: 1.1rem; color: #2d3748; }
.fecha { font-size: 0.8rem; color: #a0aec0; }

.cambio-row { display: flex; align-items: center; gap: 10px; margin-bottom: 20px;
}
.col small { display: block; text-transform: uppercase; font-size: 0.7rem; color:
#a0aec0; margin-bottom: 5px; font-weight: 700; }

```

```
.dato.viejo { background: #edf2f7; padding: 8px; border-radius: 8px; color: #718096; text-decoration: line-through; }
.arrow { color: #007bff; font-weight: 800; }

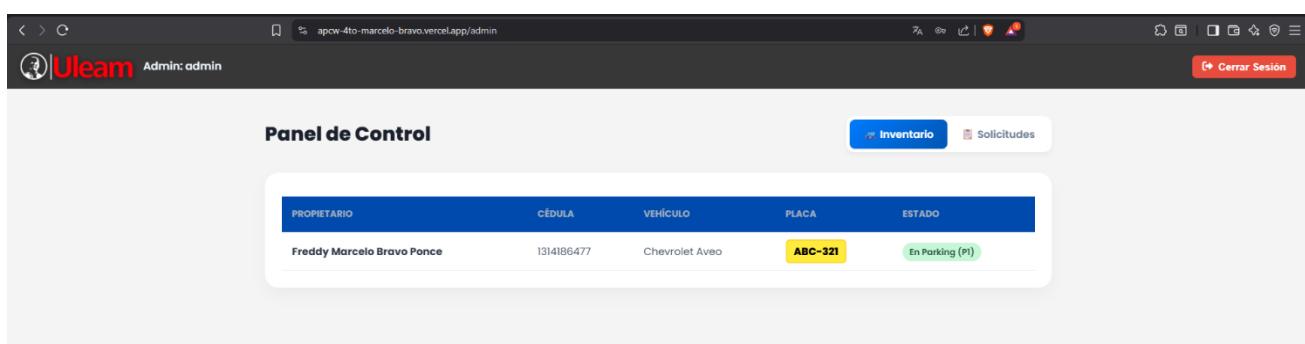
.lista-cambios { list-style: none; padding: 0; margin: 0; font-size: 0.9rem; }
.lista-cambios li { margin-bottom: 4px; color: #2d3748; }

.card-actions { display: flex; gap: 10px; }

.boton-accion {
  flex: 1; padding: 12px; border: none; border-radius: 12px;
  font-family: 'Poppins', sans-serif; font-weight: 700; cursor: pointer;
  transition: transform 0.2s;
}
.boton-accion:hover { transform: translateY(-2px); }

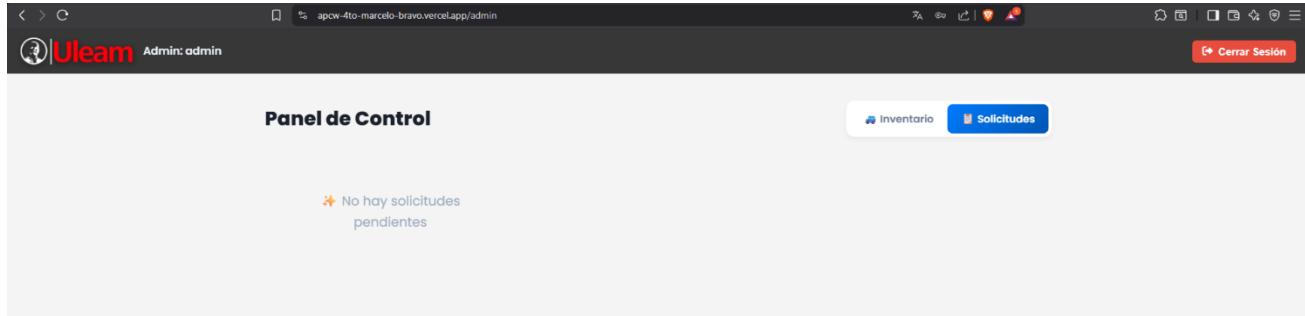
.boton-accion.aprobar {
  background: linear-gradient(135deg, #007bff 0%, #0056b3 100%); color: white;
  box-shadow: 0 4px 10px rgba(0,123,255,0.3);
}
.boton-accion.rechazar {
  background: #fff; border: 2px solid #fed7d7; color: #e53e3e;
}
.boton-accion.rechazar:hover { background: #fff5f5; border-color: #e53e3e; }

.empty-state { width: 100%; text-align: center; padding: 50px; color: #a0aec0;
font-size: 1.2rem; }
</style>
```



The screenshot shows the 'Panel de Control' (Admin Panel) interface. At the top, there are two tabs: 'Inventario' (Inventory) and 'Solicitudes' (Requests). Below the tabs, a table displays information for a specific vehicle:

PROPIETARIO	CÉDULA	VEHÍCULO	PLACA	ESTADO
Freddy Marcelo Bravo Ponce	1314186477	Chevrolet Aveo	ABC-321	En Parking (P)



The screenshot shows the 'Panel de Control' (Admin Panel) interface. At the top, there are two tabs: 'Inventario' (Inventory) and 'Solicitudes' (Requests). The 'Solicitudes' tab is active, showing a message: 'No hay solicitudes pendientes' (There are no pending requests).

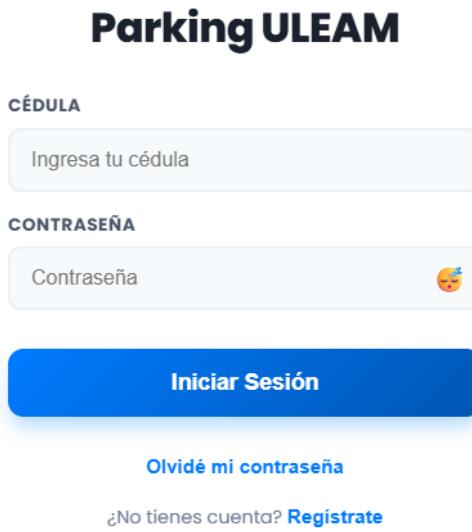
7. Manual de usuario del sistema web.

El siguiente manual describe el funcionamiento de la interfaz y los pasos que debe seguir cada actor para operar el sistema correctamente.

7.1. Acceso al Sistema (Login)

Al ingresar a la URL del proyecto, se presenta la pantalla de Inicio de Sesión.

- **Número de Cédula:** El usuario debe ingresar su identificación (solo números, 10 dígitos).
- **Contraseña:** Debe ingresar su clave. Puede pulsar en el emoji para visualizar los caracteres.
- **Botón Iniciar Sesión:** Valida los datos comunicándose con Supabase y redirige al panel correspondiente según el rol (Guardia, Admin o Alumno/Maestro).



The screenshot shows the login interface for the 'Parking ULEAM' system. At the top, it says 'Parking ULEAM'. Below that is a 'CÉDULA' field with the placeholder 'Ingresa tu cédula'. Below that is a 'CONTRASEÑA' field with the placeholder 'Contraseña' and a visibility icon (eye emoji). At the bottom is a large blue button labeled 'Iniciar Sesión'. Below the button are two links: 'Olvidé mi contraseña' and '¿No tienes cuenta? Regístrate'.

8.2. Registro de Nuevos Usuarios

Si el usuario no tiene cuenta, debe seleccionar "Regístrate".

- **Datos requeridos:** Nombres, Apellidos, Cédula (10 dígitos), Correo Institucional y Contraseña.
- **Validación de Correo:** El sistema solo acepta correos que terminen en “@live.uleam.edu.ec” (Alumnos) o “@uleam.edu.ec” (Maestros).
- **Contraseña:** Debe contener al menos 6 caracteres, una letra mayúscula y un número.

Registro

NOMBRES	<input type="text"/>
APELLIDOS	<input type="text"/>
CÉDULA	<input type="text"/>
CORREO INSTITUCIONAL	<input type="text"/>
CONTRASEÑA	<input type="password"/> Mín. 6 caracteres 

Enviar Confirmación

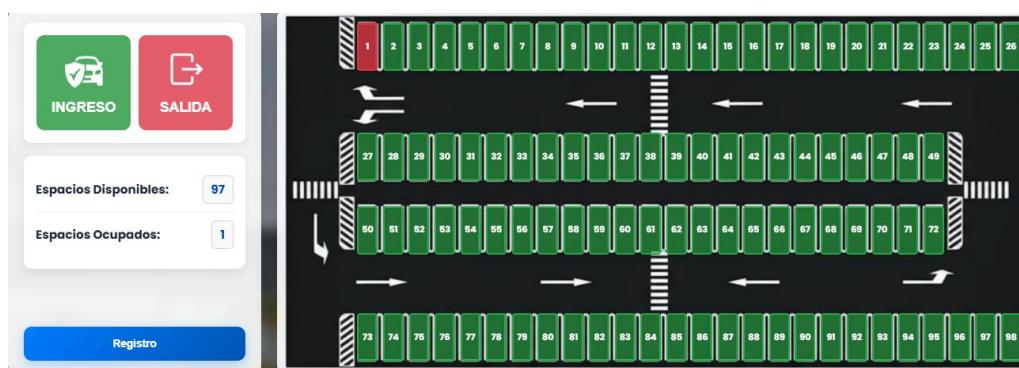
¿Ya tienes cuenta? [Inicia Sesión](#)

8.3. Panel del Guardia (Gestión de Parqueadero)

Es la interfaz principal para el control operativo. Se compone de un mapa interactivo con 98 plazas.

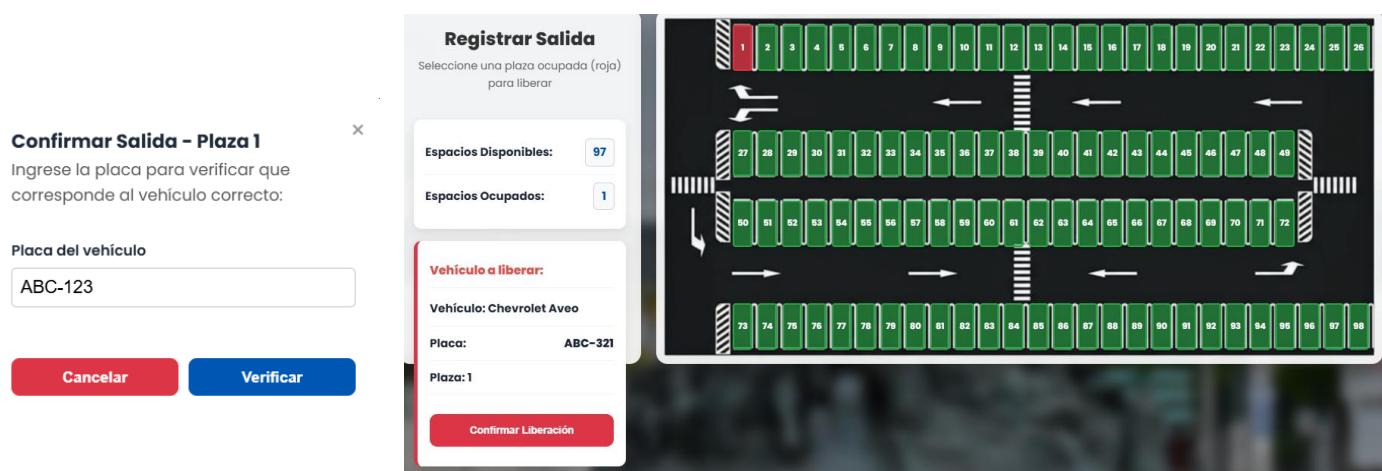
- **Registro de Ingreso:**

1. Presionar el botón **INGRESO**.
2. En el mapa, seleccionar una plaza de color **verde** (libre).
3. Se abrirá un cuadro de texto donde se debe ingresar la **Placa del Vehículo**.
4. Presionar **Validar e Ingresar**. Si el auto está registrado en la universidad, la plaza cambiará a **rojo**.



- **Registro de Salida:**

1. Presionar el botón **SALIDA**.
2. Seleccionar en el mapa la plaza de color **rojo** que desea liberar.
3. El sistema solicitará confirmar la placa del vehículo.
4. Al presionar **Confirmar Salida**, la plaza volverá a estar **verde** y se registrará el movimiento en el historial.



- **Consulta de Registro:** El botón **Registro** abre un listado con los últimos movimientos (Entradas en verde, Salidas en rojo) con su respectiva hora.

Registro de Vehículos			
Hora	Movimiento	Placa	Plaza
04:41 a. m.	ENTRADA	ABC-321	1

8.4. Panel de Administración (Control y Solicitudes)

El administrador gestiona la base de datos global.

- **Pestaña Inventario:** Permite visualizar todos los vehículos registrados en la ULEAM, indicando quién es el dueño, su cédula y si el auto se encuentra actualmente dentro o fuera del campus.

Panel de Control

Inventario
Solicitudes

PROPIETARIO	CÉDULA	VEHÍCULO	PLACA	ESTADO
Freddy Marcelo Bravo Ponce	1314186477	Chevrolet Aveo	ABC-321	En Parking (P)

- **Pestaña Solicitudes:** Aquí aparecen las peticiones de los alumnos que han cambiado de vehículo.
 - **Aprobar:** Actualiza automáticamente los datos del vehículo en la base de datos.
 - **Rechazar:** Elimina la solicitud sin realizar cambios en el inventario.

Panel de Control

Inventario
Solicitudes

No hay solicitudes pendientes

8.5. Panel de Usuario (Alumno / Maestro)

Interfaz simplificada para que el dueño del vehículo gestione su información.

- **Registro de Vehículo:** Si es la primera vez, el usuario ingresa Marca, Modelo, Color, y Placa.

Vehículo Registrado

Placa
ABC-321

Marca
Chevrolet

Modelo
Aveo

Color
Azul

 **Vehículo Activo**

- **Solicitud de Cambio:** Si el usuario adquiere un vehículo nuevo, llena el formulario de "Actualización". Estos datos no se cambian de inmediato; se envían al Administrador para su validación.

Solicitud de Actualización

Llena solo los campos que deseas cambiar:

Nueva Placa ABC-321	Nuevo Color Azul
Nueva Marca Chevrolet	Nuevo Modelo Aveo

Motivo del cambio (Obligatorio)

Ej: Vendí el auto anterior...

Enviar Solicitud

8. Conclusión

Concluyo este informe habiendo podido crear esta magnífica herramienta, sin duda ha sido un proyecto largo pero con una gran recompensa, ya que se ha podido desarrollar una web integral que sustituye registros físicos de forma manual por una base de datos virtual en tiempo real, mejorando en gran medida la seguridad en la ULEAM; además de esto la implementación de SvelteKit y de la plataforma de Supabase, permitió crear una aplicación ligera, segura y con una experiencia de usuario fluida; sin dejar atrás también a Vercel que garantiza que el personal de seguridad pueda acceder al sistema desde cualquier dispositivo en cualquier momento. Sin duda este conjunto de herramientas hace de esta página una combinación ganadora, además de que su potencial de crecimiento es evidente, pudiendo abrirse a otras áreas de trabajo bajo el mismo concepto.

9. Enlaces

- **Repositorio Github:**

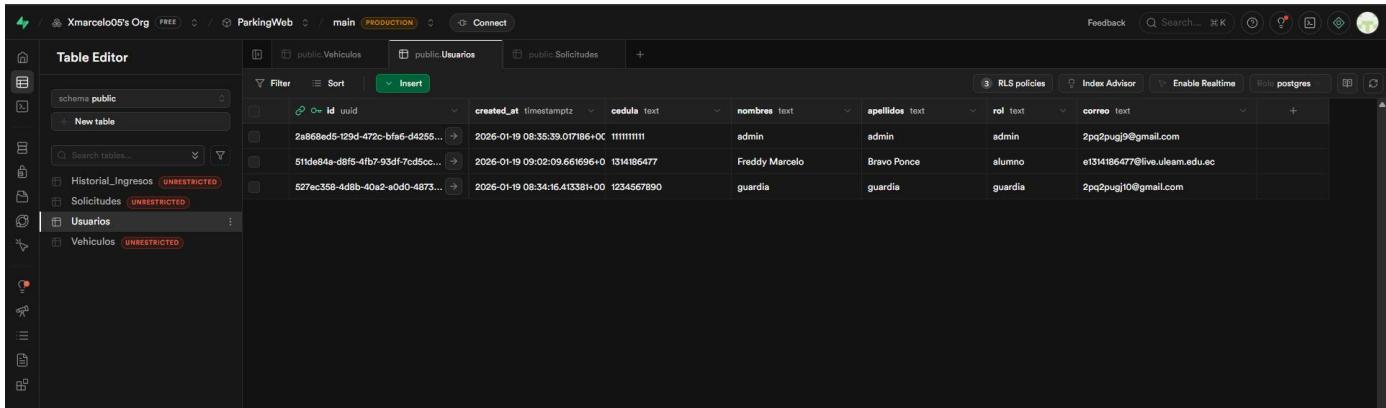
<https://github.com/Xmarcelo05/APCW-4to-Marcelo-Bravo/tree/master/ParkingWeb>

- **Página Web:**

<https://apcw-4to-marcelo-bravo.vercel.app/>

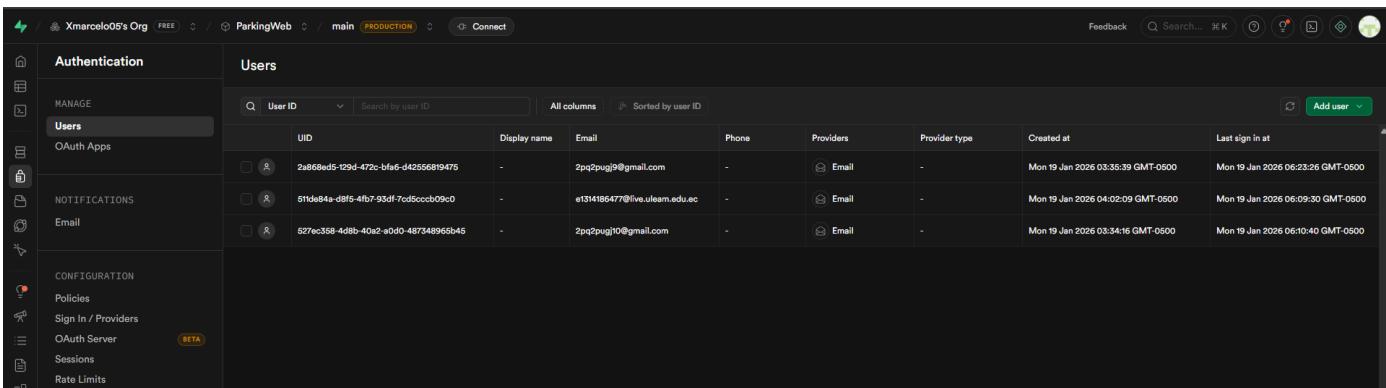
Capturas Adicionales

(Supabase):



The screenshot shows the Supabase Table Editor interface. On the left, there's a sidebar with a tree view of tables: 'Historial_Ingresos' (UNRESTRICTED), 'Solicitudes' (UNRESTRICTED), 'Usuarios', and 'Vehiculos' (UNRESTRICTED). The main area displays the 'public.Users' table with the following data:

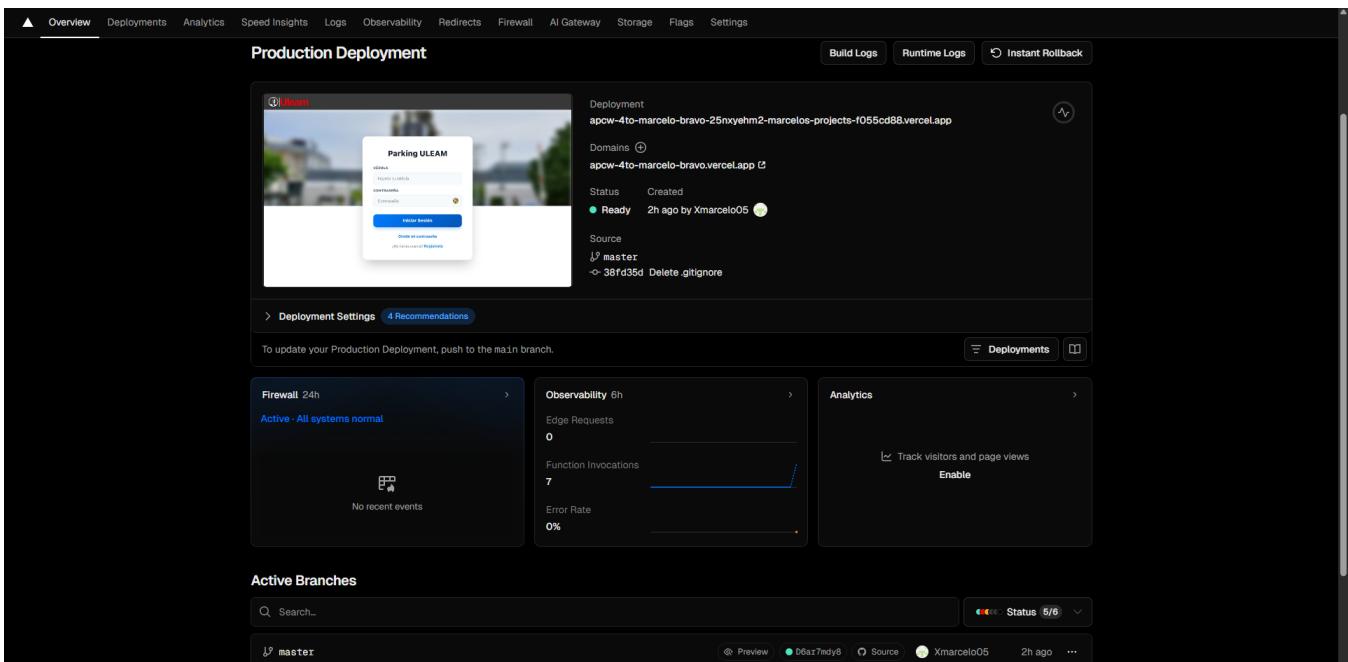
	id	created_at	cedula	nombres	apellidos	rol	correo
1	2a869ed5-129d-472c-bfa6-d4255...	2026-01-19 08:35:39.071B6+00	11111111	admin	admin	admin	2pq2pugj9@gmail.com
2	51de84a-d8f5-4fb7-93df-7cd5ccc09...	2026-01-19 09:02:09.661696+00	1314186477	Freddy Marcelo	Bravo Ponce	alumno	e1314186477@live.uleam.edu.ec
3	527ec358-4d8b-40a2-a0d0-4873...	2026-01-19 08:34:16.413381+00	1234567890	guardia	guardia	guardia	2pq2pugj10@gmail.com



The screenshot shows the Supabase Authentication section. The sidebar has sections for 'MANAGE' (Users, OAuth Apps, Notifications, Email) and 'CONFIGURATION' (Policies, Sign In / Providers, OAuth Server, Sessions, Rate Limits). The main area shows the 'Users' table with the following data:

User ID	UID	Display name	Email	Phone	Providers	Provider type	Created at	Last sign in at
2a869ed5-129d-472c-bfa6-d42556819475	-	-	2pq2pugj9@gmail.com	-	Email	-	Mon 19 Jan 2026 03:35:39 GMT-0500	Mon 19 Jan 2026 06:23:26 GMT-0500
51de84a-d8f5-4fb7-93df-7cd5ccc09c0	-	-	e1314186477@live.uleam.edu.ec	-	Email	-	Mon 19 Jan 2026 04:02:09 GMT-0500	Mon 19 Jan 2026 06:09:30 GMT-0500
527ec358-4d8b-40a2-a0d0-487348965b45	-	-	2pq2pugj10@gmail.com	-	Email	-	Mon 19 Jan 2026 03:34:16 GMT-0500	Mon 19 Jan 2026 06:10:40 GMT-0500

(Vercel)



The screenshot shows the Vercel Production Deployment dashboard. At the top, it says 'Production Deployment'. Below that, there's a preview of the app showing a parking form. To the right, there's a deployment card for 'apcw-4to-marcelo-bravo-25xyehm2-marcelos-projects-f055cd88.vercel.app'. The card shows the status is 'Ready' (2h ago by Xmarcelo05), source is 'master' (38fd35d Delete .gitignore), and domains are 'apcw-4to-marcelo-bravo.vercel.app'. Below the card, there are sections for 'Deployment Settings' and 'Recommendations'. Further down, there are sections for 'Firewall' (Active - All systems normal), 'Observability' (Edge Requests 0, Function Invocations 7, Error Rate 0%), and 'Analytics' (Track visitors and page views, Enable). At the bottom, there's a 'Active Branches' section with a search bar and a table showing the 'master' branch with a status of '5/6'.