MARKETPLACE DE SERVICIOS

Una aplicación móvil desarrollada con React Native y Expo que conecta proveedores de servicios con clientes. Los usuarios pueden registrarse como clientes o proveedores, publicar servicios, buscar y reservar servicios de manera segura y eficiente.

gerardo david ramírez villalta

## 🚀 Características Principales

* **Autenticación segura** con Supabase Auth.
* **Perfiles diferenciados**: Clientes y proveedores con interfaces personalizadas.
* **Gestión de servicios**: Proveedores pueden crear, editar y eliminar sus servicios.
* **Búsqueda y reservas**: Clientes pueden buscar servicios por categoría y reservar con validaciones automáticas.
* **Estados de reservas**: Seguimiento completo desde pendiente hasta completada.
* **Interfaz moderna** con Tailwind CSS y navegación intuitiva.

## 📋 Requisitos Previos

* **Node.js** (versión 18 o superior)
* **npm** o **yarn**
* **Expo CLI** (instalado globalmente: npm install -g @expo/cli)
* **Cuenta en Supabase** para el backend
* **Dispositivo o emulador** para probar la app (Android/iOS)

## 🛠️ Instalación

1. **Clona el repositorio**:
2. git clone <url-del-repositorio>
3. cd marketplace-services
4. **Instala las dependencias**:
5. npm install
6. **Configura Supabase** (ver sección de Configuración de Base de Datos).
7. **Inicia la aplicación**:
8. npm start

Esto abrirá el menú de Expo. Puedes escanear el código QR con la app Expo Go en tu dispositivo o usar un emulador.

## ⚙️ Configuración de Base de Datos

Este proyecto utiliza **Supabase** como backend. Sigue estos pasos para configurarlo:

### 1. Crear Proyecto en Supabase

* Ve a [supabase.com](https://supabase.com/) y crea una cuenta.
* Crea un nuevo proyecto.
* Anota tu **Project URL** y **anon/public key**.

### 2. Ejecutar el Script de Setup

* En el dashboard de Supabase, ve a **SQL Editor**.
* Copia y pega el contenido completo de database\_setup.sql.
* Ejecuta el script para crear tablas, políticas y funciones.

### 3. Configurar Variables de Entorno

Las claves de Supabase ya están configuradas en app.json:

{

"expo": {

"extra": {

"supabaseUrl": "https://tu-project-url.supabase.co",

"supabaseAnonKey": "tu-anon-key-aqui"

}

}

}

Reemplaza con tus propias claves.

## 🚀 Ejecución

* **Desarrollo**: npm start o expo start
* **Android**: npm run android o expo run:android
* **iOS**: npm run ios o expo run:ios
* **Web**: npm run web o expo start --web
* **Linting**: npm run lint

# Texto El contenido generado por IA puede ser incorrecto.📁 Estructura del Proyecto

## 🔧 Funcionalidades

### Autenticación y Perfiles

* Registro y login con email y contraseña.
* Setup inicial de perfil: nombre, teléfono y tipo (cliente/proveedor).
* Perfiles automáticos creados al registrarse.

### Para Proveedores

* **Publicar servicios**: Título, descripción, categoría, precio, ubicación e imagen.
* **Gestionar servicios**: Editar o eliminar servicios propios.
* **Ver reservas**: Acceso a reservas de sus servicios.

### Para Clientes

* **Buscar servicios**: Por nombre o categoría con barra de búsqueda.
* **Reservar servicios**: Seleccionar fecha y confirmar reserva.
* **Gestionar reservas**: Ver historial y estados de reservas.

### Estados de Reservas

* **Pendiente**: Esperando confirmación.
* **Confirmada**: Aceptada y lista para ejecutar.
* **En proceso**: Servicio en ejecución.
* **Completada**: Finalizada exitosamente.
* **Cancelada**: Cancelada por alguna de las partes.

**Validaciones**: No se permiten reservas activas duplicadas para el mismo cliente y servicio.

## 🗄️ Base de Datos

### Tablas Principales

* **profiles**: Información de usuarios (nombre, teléfono, tipo).
* **servicios**: Servicios ofrecidos (título, descripción, precio, etc.).
* **reservas**: Reservas con estados y fechas.

### Seguridad

* **Row Level Security (RLS)** habilitado en todas las tablas.
* Políticas para acceso basado en roles (usuarios ven solo sus datos).
* Validaciones automáticas para transiciones de estados y reservas duplicadas.

### Funciones Útiles

* get\_provider\_services(provider\_id): Obtiene servicios de un proveedor.
* get\_client\_bookings(client\_id): Obtiene reservas de un cliente.
* Validaciones para reservas activas y transiciones de estados.

Para más detalles, consulta DATABASE\_README.md.

## 🎨 Tecnologías Utilizadas

* **Frontend**: React Native, Expo, TypeScript, Tailwind CSS
* **Backend**: Supabase (PostgreSQL, Auth, RLS)
* **Navegación**: React Navigation (Stack y Bottom Tabs)
* **Estilos**: NativeWind (Tailwind para React Native)
* **Autenticación**: Supabase Auth con persistencia en AsyncStorage