

Stack Tecnológico - Sistema de Gestión Integral

Información General del Proyecto

Nombre: Sistema Update

Tipo: Aplicación web de gestión empresarial integral

Versión: 0.0.0

Arquitectura: Single Page Application (SPA)

Arquitectura del Sistema

Frontend Framework

- **React 19.1.0** - Framework principal para la interfaz de usuario
- **React DOM 19.1.0** - Renderizado del DOM virtual
- **Vite 6.3.5** - Build tool y servidor de desarrollo

Lenguajes y Sintaxis

- **JavaScript (ES6+)** - Lenguaje principal
- **JSX** - Sintaxis para componentes React
- **CSS3** - Estilos complementarios

Styling Framework

- **Tailwind CSS 4.1.7** - Framework de utilidades CSS
- **PostCSS 8.5.3** - Procesador de CSS
- **Autoprefixer 10.4.21** - Prefijos automáticos para CSS

Base de Datos y Backend

Base de Datos como Servicio

- **Supabase 2.49.8** - Backend-as-a-Service (BaaS)
 - PostgreSQL como base de datos principal
 - Autenticación integrada
 - Real-time subscriptions
 - REST API automática
 - Row Level Security (RLS)

Configuración del Cliente

```
// Cliente Supabase configurado con variables de entorno
const supabaseUrl = import.meta.env.VITE_SUPABASE_URL
const supabaseKey = import.meta.env.VITE_SUPABASE_ANON_KEY
export const supabase = createClient(supabaseUrl, supabaseKey)
```

Herramientas de Desarrollo

Linting y Formateo

- **ESLint 9.25.0** - Linter de JavaScript

- **@eslint/js** - Configuración base de ESLint
- **eslint-plugin-react-hooks** - Reglas específicas para React Hooks
- **eslint-plugin-react-refresh** - Soporte para React Fast Refresh

TypeScript Support

- **@types/react 19.1.2** - Definiciones de tipos para React
- **@types/react-dom 19.1.2** - Definiciones de tipos para React DOM

Build y Bundling

- **@vitejs/plugin-react 4.4.1** - Plugin de React para Vite

Librerías y Dependencias Principales

Generación de Documentos

- **@react-pdf/renderer 4.3.0** - Generación de PDFs con React
- **jsPDF 3.0.1** - Librería alternativa para PDFs
- **jspdf-autotable 5.0.2** - Tablas automáticas en PDFs
- **JSZip 3.10.1** - Compresión y manejo de archivos ZIP

Comunicación por Email

- **@emailjs/browser 4.4.1** - Envío de emails desde el frontend

Iconografía y UI

- **Lucide React 0.511.0** - Librería de iconos moderna
- **Recharts 2.15.3** - Gráficos y visualizaciones de datos

Inteligencia Artificial

- **@anthropic-ai/sdk 0.53.0** - SDK de Anthropic Claude
- **claude 0.0.6** - Cliente adicional para Claude

Tipografía

- **Roboto Font Family** - Familia tipográfica completa incluida

Arquitectura de la Aplicación

Estructura Modular

El proyecto sigue una arquitectura modular organizada por dominio:

```
src/
├── modules/                # Módulos de negocio
│   ├── administracion/    # Gestión administrativa
│   ├── contabilidad/      # Sistema contable
│   ├── importaciones/     # Gestión de importaciones
│   ├── inventario/        # Control de inventario
│   ├── reportes/          # Reportes y análisis
│   ├── soporte/           # Soporte técnico y reparaciones
│   └── ventas/             # Gestión de ventas
├── shared/                 # Componentes compartidos
├── components/             # Componentes globales
└── context/                # Contextos de React
```

```
|— hooks/                # Custom hooks
|— lib/                  # Librerías y configuraciones
|— services/             # Servicios de negocio
|— utils/                # Utilidades
```

Patrones de Diseño Implementados

1. Custom Hooks Pattern

Cada módulo implementa hooks personalizados para la lógica de negocio:

- `useInventario()` - Gestión de computadoras
- `useCelulares()` - Gestión de dispositivos móviles
- `useVentas()` - Procesamiento de ventas
- `useCarrito()` - Carrito de compras
- `useAuth()` - Autenticación

2. Provider Pattern

Utilización de React Context para estado global:

```
<AuthProvider>
  <AppWithAuth />
</AuthProvider>
```

3. Module Pattern

Cada módulo exporta sus componentes de forma centralizada:

```
// modules/ventas/components/index.js
export { default as VentasSection } from './VentasSection.jsx'
export { default as ClientesSection } from './ClientesSection.jsx'
```



Sistema de Autenticación y Seguridad

Autenticación

- Sistema basado en Supabase Auth
- Contexto de autenticación global
- Protección de rutas por roles
- Verificación de permisos por sección

Control de Acceso

```
const handleSectionChange = (newSection) => {
  if (hasAccess(newSection)) {
    setActiveSection(newSection);
  } else {
    console.warn('Acceso denegado a la sección:', newSection);
  }
};
```

Configuración del Servidor de Desarrollo

Vite Configuration

```
export default defineConfig({
  plugins: [tailwindcss(), react()],
  server: {
    host: '0.0.0.0',
    port: 5173,
    allowedHosts: [
      'localhost',
      '.ngrok-free.app',
      '.ngrok.io'
    ]
  }
})
```

Características:

- Servidor accesible desde cualquier IP (0.0.0.0)
- Soporte para túneles ngrok
- Hot Module Replacement (HMR)
- Fast Refresh para React

Módulos de Negocio

1. Módulo de Ventas

- Gestión de inventario (computadoras, celulares, otros)
- Procesamiento de ventas
- Sistema de carrito de compras
- Gestión de clientes
- Cuentas corrientes

2. Módulo de Contabilidad

- Plan de cuentas
- Libro diario y mayor
- Gastos operativos
- Conciliación de caja
- Reportes de movimientos

3. Módulo de Soporte

- Gestión de reparaciones
- Control de repuestos
- Testeo de equipos
- Presupuestos y garantías

4. Módulo de Importaciones

- Cotizaciones de proveedores
- Seguimiento de pedidos

- Productos en tránsito
- Historial de importaciones

5. Módulo de Administración

- Dashboard de reportes
- Análisis de ventas
- Gestión de comisiones
- Control de stock



Gestión de Estado

Estado Local

- **useState** para estado de componentes
- **useEffect** para efectos secundarios
- **Custom Hooks** para lógica reutilizable

Estado Global

- **React Context** para autenticación
- **Supabase** como fuente de verdad para datos

Patrón de Datos

```
const {
  data,
  loading,
  error,
  fetchData,
  createData,
  updateData,
  deleteData
} = useCustomHook();
```



Sistema de Diseño

Tema de Colores (Tailwind)

```
colors: {
  primary: {
    50: '#eff6ff',
    500: '#3b82f6',
    900: '#1e3a8a',
  }
}
```

Tipografía

- **Familia principal:** Inter, Roboto
- **Fallbacks:** ui-sans-serif, system-ui, sans-serif

Componentes de Layout

- **Sidebar** - Navegación lateral
- **Header** - Cabecera de la aplicación
- **CarritoWidget** - Widget flotante del carrito

Flujo de Datos

Arquitectura de Datos

1. **UI Components** → Dispatchers (handlers)
2. **Custom Hooks** → Supabase Client
3. **Supabase** → PostgreSQL Database
4. **Real-time Updates** → UI Re-render

Ejemplo de Flujo:











```
Usuario hace clic → handleAddToCart() →  
useCarrito() → setCarrito() →  
UI actualizada automáticamente
```

Scripts de Desarrollo





```
{  
  "dev": "vite --host 0.0.0.0",  
  "build": "vite build",  
  "lint": "eslint .",  
  "preview": "vite preview"  
}
```




Características de la Aplicación

Funcionalidades Principales

-  Sistema de autenticación robusto
-  Gestión integral de inventarios
-  Procesamiento de ventas en tiempo real
-  Generación automática de PDFs
-  Sistema contable completo
-  Gestión de reparaciones y soporte
-  Control de importaciones
-  Dashboard de reportes y análisis
-  Sistema de cuentas corrientes
-  Gestión de fotos de productos

Características Técnicas

-  Responsive design con Tailwind CSS
-  Componentes modulares y reutilizables
-  Real-time updates con Supabase
-  Manejo robusto de errores

-  Validación de permisos por rol
-  Hot reload en desarrollo
-  Build optimizado para producción



Variables de Entorno

```
VITE_SUPABASE_URL=your_supabase_url
VITE_SUPABASE_ANON_KEY=your_supabase_anon_key
```



Escalabilidad y Mantenibilidad

Puntos Fuertes

- **Arquitectura modular** permite fácil extensión
- **Custom hooks** centralizan lógica de negocio
- **Supabase** proporciona escalabilidad de base de datos
- **Tailwind** permite estilos consistentes
- **TypeScript support** mejora la robustez del código

Patrones de Escalabilidad

- Separación clara entre UI y lógica de negocio
- Hooks reutilizables entre módulos
- Componentes de presentación puros
- Estado inmutable con React patterns



Herramientas de Productividad

Desarrollo

- **Vite** - Build tool ultra-rápido
- **ESLint** - Calidad de código
- **Hot Module Replacement** - Desarrollo ágil

Producción

- **Build optimizado** con tree-shaking
- **Lazy loading** de componentes
- **Bundle splitting** automático



Conclusión

Este sistema representa una aplicación empresarial moderna construida con tecnologías de vanguardia. La combinación de React 19, Vite, Tailwind CSS y Supabase proporciona una base sólida para un sistema de gestión integral, escalable y mantenible.

La arquitectura modular permite el crecimiento orgánico del sistema, mientras que las herramientas seleccionadas garantizan una experiencia de desarrollo productiva y un producto final robusto.

Fecha de documentación: 18 de Junio, 2025

Versión del sistema: 0.0.0

Autor: Sistema automático de documentación