📊 Sistema Empresarial - Plataforma de Gestión Integral

🚀 Descripción General

Sistema empresarial moderno desarrollado en React.js con backend en Node.js/Express, diseñado para la gestión integral de procesos empresariales incluyendo contabilidad, recursos humanos y CRM. La aplicación utiliza PostgreSQL como base de datos y está construida con arquitectura modular y escalable.

🏗️ Arquitectura del Sistema

Frontend (React.js)

- **Framework**: React 18 con Vite
- **UI Library**: Radix UI + Tailwind CSS + Material UI
- **Routing**: React Router DOM v6
- **State Management**: React Hooks
- **Iconografía**: Lucide React + Material Icons
- **Animaciones**: Framer Motion
- **Gráficos**: Recharts

Backend (Node.js/Express)

- **Framework**: Express.js 5
- **ORM**: Sequelize
- **Base de Datos**: PostgreSQL
- **Autenticación**: JWT (preparado)
- **CORS**: Habilitado para desarrollo
- **Procesamiento XML**: xml2js, xmlbuilder2
- **Exportación Excel**: XLSX

📁 Estructura del Proyecto

```
sistema-empresarial/
├── 📁 public/                    # Archivos estáticos
├── 📁 src/
│   ├── 📁 components/            # Componentes reutilizables
│   │   ├── 📁 ui/                # Componentes base UI
│   │   └── 📁 forms/             # Componentes de formularios
│   ├── 📁 pages/                 # Páginas principales
│   │   ├── 📁 contabilidad/      # Módulo de contabilidad
│   │   ├── 📁 rrhh/              # Módulo de recursos humanos
│   │   ├── 📁 crm/               # Módulo CRM
│   │   ├── Dashboard.jsx         # Dashboard principal
│   │   └── Login.jsx             # Página de login
│   ├── 📁 layouts/               # Layouts de aplicación
│   ├── 📁 hooks/                 # Custom hooks
│   ├── 📁 utils/                 # Utilidades y helpers
│   ├── 📁 services/              # Servicios API
│   ├── 📁 lib/                   # Configuraciones de librerías
```

```
│   ├── 📁 styles/                # Estilos globales
│   ├── 📁 config/                # Configuración de la aplicación
│   │   └── database.js           # Configuración de base de datos
│   ├── 📁 models/                # Modelos Sequelize
│   ├── 📁 controllers/           # Controladores del backend
│   ├── 📁 routes/                # Rutas del API
│   ├── 📁 scripts/               # Scripts de utilidad
│   ├── server.js                 # Servidor Express
│   ├── App.jsx                   # Componente principal
│   └── main.jsx                  # Punto de entrada
├── 📁 xmls-facturas-electronicas/ # Archivos XML de facturas
├── package.json                  # Dependencias del proyecto
├── vite.config.js                # Configuración de Vite
├── tailwind.config.js            # Configuración de Tailwind
└── postcss.config.js             # Configuración de PostCSS
```

🗄️ Modelos de Base de Datos

Módulo de Contabilidad

📋 **Facturas (`Factura`)**

```
{
  id_factura: INTEGER (PK),
  numero_factura: STRING,
  estatus_factura: ENUM ['PENDIENTE', 'PAGADA', 'VENCIDA'],
  fecha_radicado: DATE,
  fecha_estimada_pago: DATE,
  subtotal_facturado_moneda: DECIMAL,
  id_contrato: INTEGER (FK),
  id_moneda: INTEGER (FK)
}
```

💰 **Transacciones (`Transaccion`)**

```
{
  id_transaccion: INTEGER (PK),
  fecha_transaccion: DATE,
  valor_total_transaccion: DECIMAL,
  tipo_transaccion: ENUM ['VENTA', 'COMPRA', 'PAGO', 'REEMBOLSO'],
  descripcion_transaccion: TEXT,
  id_cuenta: INTEGER (FK),
  id_tipo_transaccion: INTEGER (FK),
  id_moneda: INTEGER (FK)
}
```

🤝 **Terceros (`Tercero`)**

```
{
  id_tercero: INTEGER (PK),
  nombre_tercero: STRING,
  tipo_personalidad: ENUM ['NATURAL', 'JURIDICA'],
  numero_identificacion: STRING,
  estado_tercero: ENUM ['ACTIVO', 'INACTIVO'],
  telefono_tercero: STRING,
  email_tercero: STRING,
  direccion_tercero: TEXT
}
```

📄 **Contratos (** `Contrato` **)**

```
{
  id_contrato: INTEGER (PK),
  nombre_contrato: STRING,
  fecha_inicio_contrato: DATE,
  fecha_fin_contrato: DATE,
  valor_cotizado: DECIMAL,
  estado_contrato: ENUM ['ACTIVO', 'FINALIZADO', 'SUSPENDIDO'],
  descripcion_contrato: TEXT,
  id_tercero: INTEGER (FK),
  id_moneda: INTEGER (FK)
}
```

🏷️ **Impuestos (** `Tax` **)**

```
{
  id_impuesto: INTEGER (PK),
  nombre_impuesto: STRING,
  porcentaje_impuesto: DECIMAL,
  estado_impuesto: ENUM ['ACTIVO', 'INACTIVO'],
  fecha_inicio: DATE,
  fecha_fin: DATE,
  descripcion_impuesto: TEXT
}
```

Módulos de Soporte

🏢 **Líneas de Servicios (** `LineaServicio` **)**

🏷️ **Etiquetas Contables (** `EtiquetaContable` **)**

💱 **Monedas (** `Moneda` **)**

🏬 **Centros de Costos (** `CentroCosto` **)**

📊 **Conceptos de Transacciones (** `ConceptoTransaccion` **)**

🔄 **Tipos de Transacciones (** `TipoTransaccion` **)**

🔗 APIs y Endpoints

Base URL: `http://localhost:5000/api`

📋 Facturas

- `GET /facturas` - Listar todas las facturas
- `GET /facturas/:id` - Obtener factura por ID
- `POST /facturas` - Crear nueva factura
- `POST /facturas/import` - Importar facturas desde Excel

💰 Transacciones

- `GET /transacciones` - Listar todas las transacciones
- `GET /transacciones/:id` - Obtener transacción por ID
- `POST /transacciones` - Crear nueva transacción

🤝 Terceros

- `GET /terceros` - Listar todos los terceros
- `GET /terceros/:id` - Obtener tercero por ID
- `POST /terceros` - Crear nuevo tercero
- `POST /terceros/bulk-create` - Creación masiva

📄 Contratos

- `GET /contratos` - Listar todos los contratos
- `GET /contratos/:id` - Obtener contrato por ID
- `POST /contratos` - Crear nuevo contrato

🏷️ Impuestos

- `GET /impuestos` - Listar todos los impuestos
- `GET /impuestos/:id` - Obtener impuesto por ID
- `POST /impuestos` - Crear nuevo impuesto
- `POST /impuestos/bulk-create` - Creación masiva

🏢 Líneas de Servicios

- `GET /lineas-servicios` - Listar todas las líneas
- `GET /lineas-servicios/:id` - Obtener línea por ID
- `POST /lineas-servicios` - Crear nueva línea
- `POST /lineas-servicios/bulk-create` - Creación masiva

📊 Catálogos

- `GET /catalogos/cuentas` - Catálogo de cuentas
- `GET /catalogos/tipos-transaccion` - Tipos de transacciones
- `GET /catalogos/monedas` - Catálogo de monedas
- `GET /catalogos/etiquetas-contables` - Etiquetas contables
- `GET /catalogos/terceros` - Catálogo de terceros
- `GET /catalogos/conceptos` - Conceptos DIAN
- `GET /catalogos/contratos` - Catálogo de contratos
- `GET /catalogos/taxes` - Catálogo de impuestos

🛠️ Instalación y Configuración

Prerrequisitos

- Node.js 18+
- PostgreSQL 12+
- npm o yarn

1. Clonar el Repositorio

```
git clone <repository-url>
cd sistema-empresarial
```

2. Instalar Dependencias

```
npm install
```

3. Configurar Base de Datos

1. Crear base de datos PostgreSQL llamada `SQL_DDL_ADMCOT`
2. Configurar credenciales en `src/config/database.js`:

```
const sequelize = new Sequelize('SQL_DDL_ADMCOT', 'postgres', '12345', {
  host: 'localhost',
  port: 5432,
  dialect: 'postgres'
});
```

4. Ejecutar Migraciones (si existen)

```
npm run migrate
```

🚀 Ejecución

Modo Desarrollo

```
# Terminal 1: Frontend (Puerto 5173)
npm run dev

# Terminal 2: Backend (Puerto 5000)
npm run server
```

Modo Producción

```
npm run build
npm run preview
```

🎨 Características del Frontend

📊 Dashboard Principal

- Resumen ejecutivo de todos los módulos
- Métricas en tiempo real
- Gráficos y barras de progreso
- Navegación rápida a módulos

🧮 Módulo de Contabilidad

- **Dashboard de Contabilidad**: Métricas específicas del módulo
- **Gestión de Facturas**: CRUD completo con importación Excel
- **Gestión de Transacciones**: Registro y seguimiento
- **Gestión de Terceros**: Administración de clientes/proveedores
- **Gestión de Contratos**: Seguimiento de contratos
- **Configuración de Impuestos**: Gestión tributaria
- **Líneas de Servicios**: Catálogo de servicios
- **Clasificaciones Contables**: Centros de costos, etiquetas, conceptos

🎯 Características Técnicas

- **Responsive Design**: Adaptable a todos los dispositivos
- **Dark/Light Mode**: Tema personalizable
- **Lazy Loading**: Carga optimizada de componentes
- **Error Boundaries**: Manejo robusto de errores
- **Formularios Validados**: React Hook Form + validaciones
- **Exportación Excel**: Descarga de plantillas y datos
- **Importación Masiva**: Carga de datos desde Excel
- **Búsqueda y Filtros**: Funcionalidad avanzada de filtrado
- **Paginación**: Manejo eficiente de grandes volúmenes
- **Toast Notifications**: Notificaciones de usuario

🔧 Tecnologías y Librerías

Frontend

```
{
  "react": "^18.2.0",
  "react-router-dom": "^6.16.0",
  "@radix-ui/react-*": "Componentes UI",
  "tailwindcss": "^3.3.3",
  "@mui/material": "^7.1.0",
  "lucide-react": "^0.285.0",
  "framer-motion": "^10.16.4",
  "recharts": "^2.15.3",
  "react-hook-form": "^7.56.3",
  "react-hot-toast": "^2.5.2",
  "xlsx": "^0.18.5",
  "date-fns": "^4.1.0"
}
```

Backend

```
{
  "express": "^5.1.0",
  "sequelize": "^6.37.7",
  "pg": "^8.16.0",
  "cors": "^2.8.5",
  "xml2js": "^0.6.2",
  "xmlbuilder2": "^3.1.1",
  "uuid": "^11.1.0"
}
```

📋 Scripts Disponibles

```
{
  "dev": "vite",                    // Inicia frontend en desarrollo
  "build": "vite build",            // Construye para producción
  "preview": "vite preview",        // Preview de build
  "server": "node src/server.js",   // Inicia backend
  "start": "npm run dev"            // Alias para desarrollo
}
```

🧪 Archivos de Prueba

El proyecto incluye varios archivos de prueba y scripts de utilidad:

- `test-api.js` - Pruebas de endpoints API
- `test-frontend.html` - Pruebas de componentes frontend
- `test_facturas.js` - Pruebas específicas de facturas
- `debug-factura.js` - Debug de procesamiento de facturas
- `verificar-factura.js` - Validación de facturas XML

📝 Funcionalidades Implementadas

✅ Completado

- [x] Estructura base del proyecto
- [x] Configuración de base de datos PostgreSQL
- [x] Modelos Sequelize completos
- [x] APIs REST para todos los módulos
- [x] Dashboard principal con métricas reales
- [x] Dashboard de contabilidad con indicadores
- [x] CRUD completo para todas las entidades
- [x] Sistema de importación/exportación Excel
- [x] Interfaz responsive y moderna
- [x] Manejo de errores robusto
- [x] Sistema de notificaciones

- [x] Validación de formularios

🔄 En Desarrollo

- [ ] Sistema de autenticación completo
- [ ] Módulo de RRHH
- [ ] Módulo de CRM
- [ ] Reportes avanzados
- [ ] API de facturación electrónica
- [ ] Integración con DIAN

🤝 Contribución

1. Fork el proyecto
2. Crea una rama para tu feature (`git checkout -b feature/AmazingFeature`)
3. Commit tus cambios (`git commit -m 'Add some AmazingFeature'`)
4. Push a la rama (`git push origin feature/AmazingFeature`)
5. Abre un Pull Request

📄 Licencia

Este proyecto está bajo la Licencia MIT - ver el archivo LICENSE.md para detalles.

👥 Equipo de Desarrollo

- **Desarrollador Principal**: [Tu Nombre]
- **Arquitecto de Software**: [Nombre]
- **UI/UX Designer**: [Nombre]

📞 Soporte

Para soporte técnico o consultas:

- Email: soporte@empresa.com
- Documentación: [Enlace a docs]
- Issues: [GitHub Issues]

⭐ **¡No olvides dar una estrella al proyecto si te parece útil!**