SISTEMA AGROPECUÁRIO

Documentação Técnica Completa

Versão 1.0.0 - Julho 2025

Desenvolvido por: Sistema Agro Team
Data: 12 de Julho de 2025
Versão: 1.0.0
Repositório: https://github.com/m-marianaM/sistema-agropecuario

---

SUMÁRIO

---

1. VISÃO GERAL DO PROJETO

1.1 Descrição

O **Sistema Agropecuário** é uma aplicação web moderna e completa para gestão de fazendas, desenvolvida com tecnologias de ponta para oferecer uma experiência intuitiva e

eficiente na administração de propriedades rurais.

1.2 Objetivos

- **Otimizar** a gestão de fazendas e cultivos
- **Centralizar** informações agropecuárias
- **Automatizar** processos de controle e monitoramento
- **Fornecer** insights através de dashboards BI
- **Facilitar** a tomada de decisões baseada em dados

1.3 Público-Alvo

- **Produtores rurais** de pequeno, médio e grande porte
- **Gestores agrícolas** e consultores
- **Cooperativas** e associações rurais
- **Empresas do agronegócio**

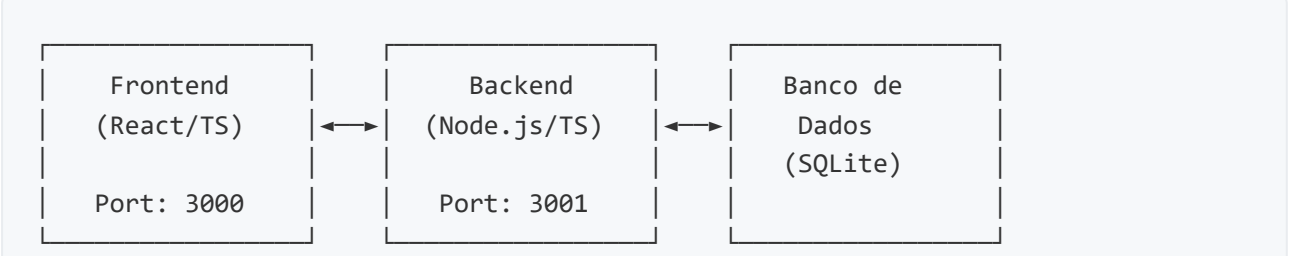
1.4 Principais Benefícios

- ✅ **Controle completo** de cultivos e produção
- ✅ **Dashboard BI** com métricas em tempo real
- ✅ **Interface responsiva** para mobile e desktop
- ✅ **Sistema de autenticação** seguro
- ✅ **Importação** de dados via Excel
- ✅ **Gestão integrada** de fazendas, vendas e estoque

---

2. ARQUITETURA DO SISTEMA

2.1 Arquitetura Geral



2.2 Frontend (React + TypeScript)

- **Framework:** React 18 com TypeScript
- **Estilização:** TailwindCSS
- **Gráficos:** Recharts
- **Ícones:** Lucide React
- **Estado:** Context API
- **Requisições:** Axios

2.3 Backend (Node.js + Express)

- **Runtime:** Node.js 18+
- **Framework:** Express com TypeScript
- **ORM:** Prisma
- **Autenticação:** JWT + Bcrypt
- **Validação:** Express Validator
- **Documentação:** Swagger UI

2.4 Banco de Dados

- **Desenvolvimento:** SQLite
- **Produção:** PostgreSQL (recomendado)
- **ORM:** Prisma para modelagem e queries

---

3. TECNOLOGIAS UTILIZADAS

3.1 Frontend

Tecnologia	Versão	Propósito
React	18.x	Framework principal
TypeScript	5.x	Tipagem estática
TailwindCSS	3.x	Estilização
Recharts	2.x	Gráficos e dashboards
Lucide React	Latest	Ícones

Axios	Latest	Requisições HTTP

3.2 Backend

Tecnologia	Versão	Propósito
Node.js	18.x	Runtime JavaScript
Express	4.x	Framework web
TypeScript	5.x	Tipagem estática
Prisma	5.x	ORM e database toolkit
JWT	Latest	Autenticação
Bcrypt	Latest	Criptografia de senhas

3.3 DevOps

Tecnologia	Versão	Propósito
Docker	Latest	Containerização
Git	Latest	Controle de versão
ESLint	Latest	Linting de código
Prettier	Latest	Formatação de código

---

4. FUNCIONALIDADES

4.1 Dashboard BI

- **Métricas em tempo real** de produção
- **Gráficos interativos** por tipo de cultivo
- **Cards informativos** com dados consolidados
- **Análise de tendências** e performance
- **Filtros** por fazenda e período

4.2 Gestão de Fazendas

- **Cadastro completo** de propriedades rurais

- **Informações detalhadas:** área, localização, proprietário
- **Status** de atividade e produção
- **Histórico** de operações

4.3 Controle de Cultivos

- **Cadastro** de plantios com variedades
- **Controle de status:** plantado → crescimento → colhido/perdido
- **Associação** com fazendas
- **Cálculo automático** de áreas
- **Estimativas** de produção e custos
- **Edição inline** para atualizações rápidas

4.4 Sistema de Autenticação

- **Login seguro** com JWT
- **Diferentes níveis** de acesso
- **Sessões persistentes**
- **Recuperação** de senha

4.5 Interface Responsiva

- **Design mobile-first**
- **Componentes adaptativos**
- **Tema claro/escuro**
- **Navegação intuitiva**

4.6 Importação de Dados

- **Upload** de planilhas Excel
- **Validação** automática de dados
- **Templates** para padronização
- **Feedback** detalhado de erros

---

5. INSTALAÇÃO E CONFIGURAÇÃO

5.1 Pré-requisitos

- **Node.js** 18.0 ou superior
- **NPM** ou Yarn
- **Git**

5.2 Instalação

Passo 1: Clone o Repositório

```
git clone https://github.com/m-marianaM/sistema-agropecuario.git
cd sistema-agropecuario
```

Passo 2: Instale as Dependências

```
# Dependências do projeto principal
npm install

# Backend
cd backend
npm install

# Frontend
cd ../frontend
npm install
```

Passo 3: Configure o Banco de Dados

```
cd ../backend

# Gerar cliente Prisma
npx prisma generate

# Executar migrações
npx prisma db push

# Popular com dados iniciais
npx prisma db seed
```

Passo 4: Execute o Sistema

```
# Voltar para raiz
cd ..

# Executar backend e frontend
npm run dev
```

5.3 Configuração de Ambiente

O sistema estará disponível em:

- **Frontend:** http://localhost:3000
- **Backend:** http://localhost:3001
- **API Docs:** http://localhost:3001/api-docs

5.4 Credenciais Padrão

```
Email: admin@systemagro.com
Senha: admin123
```

---

6. GUIA DO USUÁRIO

6.1 Primeiro Acesso

1. **Acesse** http://localhost:3000
2. **Faça login** com as credenciais padrão
3. **Navegue** pelas seções do menu lateral

6.2 Dashboard Principal

- **Visualize** métricas consolidadas
- **Analise** gráficos de produção
- **Monitore** status dos cultivos
- **Acompanhe** tendências

6.3 Gestão de Fazendas

1. **Acesse** a seção "Fazendas"
2. **Cadastre** novas propriedades
3. **Edite** informações existentes
4. **Visualize** dados consolidados

6.4 Controle de Cultivos

1. **Acesse** a seção "Cultivos"
2. **Cadastre** novos plantios
3. **Atualize** status conforme necessário
4. **Monitore** progresso e produção

6.5 Funcionalidades Avançadas

- **Filtros** por período e fazenda
- **Exportação** de relatórios
- **Importação** de dados via Excel
- **Alternância** de tema claro/escuro

---

7. API DOCUMENTATION

7.1 Endpoints Principais

Autenticação

```
POST /api/auth/login
POST /api/auth/register
GET  /api/auth/me
```

Fazendas

```
GET    /api/fazendas
POST   /api/fazendas
GET    /api/fazendas/:id
```

```
PUT    /api/fazendas/:id
DELETE /api/fazendas/:id
```

Cultivos

```
GET    /api/cultivos
POST   /api/cultivos
GET    /api/cultivos/:id
PUT    /api/cultivos/:id
DELETE /api/cultivos/:id
```

Dashboard

```
GET /api/dashboard/metrics
GET /api/dashboard/charts
```

7.2 Autenticação

Todas as rotas protegidas requerem header:

```
Authorization: Bearer <JWT_TOKEN>
```

7.3 Formato de Resposta

```
{
  "success": true,
  "data": {},
  "message": "Operação realizada com sucesso"
}
```

---

8. BANCO DE DADOS

8.1 Modelo de Dados

Entidade: Fazenda

```
CREATE TABLE fazendas (
  id INTEGER PRIMARY KEY,
  nome TEXT NOT NULL,
  proprietario TEXT NOT NULL,
  area REAL NOT NULL,
  endereco TEXT,
  cidade TEXT,
  estado TEXT,
  cep TEXT,
  telefone TEXT,
  email TEXT,
  criadoEm DATETIME DEFAULT CURRENT_TIMESTAMP,
  atualizadoEm DATETIME DEFAULT CURRENT_TIMESTAMP
);
```

Entidade: Cultivo

```
CREATE TABLE cultivos (
  id INTEGER PRIMARY KEY,
  nome TEXT NOT NULL,
  variedade TEXT,
  areaPlantada REAL NOT NULL,
  dataPlantio DATETIME,
  dataColheita DATETIME,
  status TEXT DEFAULT 'plantado',
  producaoTotal REAL,
  custoTotal REAL,
  fazendaId INTEGER,
  criadoEm DATETIME DEFAULT CURRENT_TIMESTAMP,
  atualizadoEm DATETIME DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (fazendaId) REFERENCES fazendas(id)
);
```

Entidade: Usuario

```
CREATE TABLE usuarios (
  id INTEGER PRIMARY KEY,
  nome TEXT NOT NULL,
  email TEXT UNIQUE NOT NULL,
  senha TEXT NOT NULL,
  cargo TEXT DEFAULT 'Peão',
```

```
  status TEXT DEFAULT 'ativo',
  fazendaId INTEGER,
  criadoEm DATETIME DEFAULT CURRENT_TIMESTAMP,
  atualizadoEm DATETIME DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (fazendaId) REFERENCES fazendas(id)
);
```

8.2 Relacionamentos

- **Fazenda** → **Cultivos** (1:N)
- **Fazenda** → **Usuários** (1:N)
- **Fazenda** → **Vendas** (1:N)
- **Fazenda** → **Estoque** (1:N)

---

9. SEGURANÇA

9.1 Autenticação

- **JWT** com expiração configurável
- **Senhas** criptografadas com bcrypt
- **Sessões** persistentes e seguras

9.2 Autorização

- **Níveis de acesso:** Administrador, Supervisor, Peão
- **Controle por módulo** e ação
- **Validação** de permissões

9.3 Validação de Dados

- **Validação rigorosa** de entrada
- **Sanitização** de dados
- **Prevenção** de ataques XSS e SQL Injection

9.4 Headers de Segurança

- **CORS** configurado
- **Headers** de segurança implementados

- **Rate limiting** para APIs

---

10. MANUTENÇÃO E SUPORTE

10.1 Logs do Sistema

- **Backend:** Logs de requisições e erros
- **Frontend:** Console errors e warnings
- **Banco:** Query logs via Prisma

10.2 Backup

- **Banco de dados:** Backup automático diário
- **Arquivos:** Versionamento via Git
- **Configurações:** Documentadas e versionadas

10.3 Monitoramento

- **Performance:** Métricas de resposta
- **Disponibilidade:** Health checks
- **Recursos:** CPU, memória, disco

10.4 Suporte Técnico

- **Email:** suporte@systemagro.com
- **Documentação:** GitHub Wiki
- **Issues:** GitHub Issues

---

11. ROADMAP

11.1 Versão 1.1 (Agosto 2025)

- [ ] **Relatórios** em PDF
- [ ] **Notificações** push
- [ ] **API** mobile
- [ ] **Integração** com sensores IoT

11.2 Versão 1.2 (Setembro 2025)

- ☐ **Machine Learning** para previsões
- ☐ **Integração** com drones
- ☐ **Análise** de solo
- ☐ **Marketplace** de produtos

11.3 Versão 2.0 (Q4 2025)

- ☐ **Migração** para microserviços
- ☐ **PWA** completo
- ☐ **Multi-tenancy**
- ☐ **Analytics** avançados

📞 CONTATOS E SUPORTE

Equipe de Desenvolvimento:

- **Email:** admin@systemagro.com
- **GitHub:** https://github.com/m-marianaM/sistema-agropecuario

Suporte Técnico:

- **Email:** suporte@systemagro.com
- **Horário:** Segunda a Sexta, 8h às 18h

Documentação Online:

- **GitHub Wiki:** Documentação atualizada
- **API Docs:** http://localhost:3001/api-docs

📄 LICENÇA

🏆 AGRADECIMENTOS

Agradecemos a todos que contribuíram para tornar este projeto uma realidade, especialmente à comunidade open source que fornece as ferramentas e bibliotecas que utilizamos.

Documentação gerada automaticamente em 12 de Julho de 2025