

# Sistema de Gestão de Propostas - Documentação Completa

## Índice








1. Visão Geral
  2. Arquitetura do Sistema
  3. Módulos do Sistema
  4. Gerenciamento de Usuários
  5. Instalação e Configuração
  6. Guia de Uso
  7. APIs e Integrações
  8. Segurança
  9. Manutenção
- 

## 1. Visão Geral

O Sistema de Gestão de Propostas é uma plataforma integrada para gerenciamento completo de processos licitatórios, conectando três perfis principais:

- **Requisitantes:** Criam Termos de Referência (TRs)
- **Compradores:** Aprovam TRs e gerenciam processos de compra
- **Fornecedores:** Participam de processos e enviam propostas

### Características Principais

-  Sistema totalmente integrado e em produção
  -  Autenticação JWT com controle de sessão
  -  Banco de dados SQLAlchemy (SQLite/PostgreSQL)
  -  Interface responsiva com Bootstrap 5
  -  Sistema de notificações em tempo real
  -  Logs de auditoria completos
  -  Relatórios e exportação de dados
- 

## 2. Arquitetura do Sistema

### Stack Tecnológico

#### Backend:

- Python 3.8+
- Flask 2.0

- SQLAlchemy
- Flask-JWT-Extended
- Flask-CORS
- Bcrypt

### Frontend:

- HTML5/CSS3
- JavaScript ES6+
- Bootstrap 5.1.3
- Font Awesome 6.0
- Chart.js

### Banco de Dados:

- Desenvolvimento: SQLite
- Produção: PostgreSQL (Render)

## Estrutura de Arquivos

```
sistema-propostas/  
├── static/  
│   ├── index.html  
│   ├── dashboard-requisitante.html  
│   ├── dashboard-comprador.html  
│   ├── dashboard-fornecedor.html  
│   ├── admin-usuarios.html  
│   ├── login-admin.html  
│   ├── login-requisitante.html  
│   ├── login-comprador.html  
│   └── sistema-autenticacao-fornecedores.html  
├── backend_render_fix.py (API principal)  
├── models.py (Modelos do banco)  
├── auth.py (Sistema de autenticação)  
├── config.py (Configurações)  
├── user_management_system.py (Gerenciamento de usuários)  
├── migration_add_users.py (Script de migração)  
├── requirements.txt  
└── render.yaml
```

---

## 3. Módulos do Sistema

### 3.1 Módulo Requisitante

#### Funcionalidades:

- Criar e gerenciar Termos de Referência
- Acompanhar status de aprovação
- Visualizar pareceres do comprador
- Baixar TRs em PDF/Word
- Receber notificações

**Fluxo:**

1. Requisitante cria TR com todas as especificações
2. TR é enviado para aprovação do comprador
3. Requisitante recebe notificação do resultado
4. Se aprovado, TR pode ser usado em processo de compra

### **3.2 Módulo Comprador**

**Funcionalidades:**

- Aprovar/reprovar TRs com parecer
- Criar processos de compra
- Convidar fornecedores
- Analisar propostas (comparativo)
- Gerar relatórios de compras
- Gerenciar fornecedores

**Fluxo:**

1. Comprador analisa TRs pendentes
2. Aprova TR e cria processo de compra
3. Seleciona e convida fornecedores
4. Analisa propostas recebidas
5. Gera comparativo e relatórios

### **3.3 Módulo Fornecedor**

**Funcionalidades:**

- Auto-cadastro no sistema
- Visualizar processos disponíveis
- Enviar propostas técnicas e comerciais
- Acompanhar status das propostas
- Manter documentação atualizada

**Fluxo:**

1. Fornecedor se cadastra no sistema
  2. Aguarda aprovação do cadastro
  3. Recebe convites para processos
  4. Envia propostas completas
  5. Acompanha resultados
- 

## 4. Gerenciamento de Usuários

### 4.1 Tipos de Usuário

**Administrador do Sistema** (`admin_sistema`):

- Acesso total ao sistema
- Gerencia todos os usuários
- Aprova cadastros de fornecedores
- Visualiza logs de auditoria
- Gera relatórios gerenciais

**Comprador** (`comprador`):

- Aprova/reprova TRs
- Cria processos de compra
- Gerencia fornecedores
- Analisa propostas

**Requisitante** (`requisitante`):

- Cria Termos de Referência
- Acompanha aprovações
- Solicita compras

**Fornecedor:**

- Participa de processos
- Envia propostas
- Mantém cadastro atualizado

### 4.2 Credenciais de Produção

**Administradores:**

João Carlos Silva

E-mail: joao.silva@empresa.com

Senha: JoaoAdmin@2025

Maria Helena Santos

E-mail: maria.santos@empresa.com

Senha: MariaAdmin@2025

## Compradores:

Pedro Augusto Oliveira

E-mail: pedro.oliveira@empresa.com

Senha: PedroComp@2025

Ana Paula Ferreira

E-mail: ana.ferreira@empresa.com

Senha: AnaComp@2025

Carlos Eduardo Lima

E-mail: carlos.lima@empresa.com

Senha: CarlosComp@2025

## Requisitantes:

Fernanda Costa Silva

E-mail: fernanda.costa@empresa.com

Senha: FernandaReq@2025

Roberto Alves Pereira

E-mail: roberto.pereira@empresa.com

Senha: RobertoReq@2025

(+ 3 outros - ver arquivo de credenciais)

---

## 5. Instalação e Configuração

### 5.1 Requisitos

- Python 3.8 ou superior
- pip (gerenciador de pacotes Python)
- Git

### 5.2 Instalação Local

```
bash
```

```
# Clonar repositório
git clone https://github.com/seu-usuario/sistema-propostas.git
cd sistema-propostas

# Criar ambiente virtual
python -m venv venv

# Ativar ambiente (Windows)
venv\Scripts\activate

# Ativar ambiente (Linux/Mac)
source venv/bin/activate

# Instalar dependências
pip install -r requirements.txt

# Executar migrações
python migration_add_users.py

# Iniciar servidor
python backend_render_fix.py
```

## 5.3 Variáveis de Ambiente

Criar arquivo `.env`:

```
env

# Configurações gerais
FLASK_ENV=production
SECRET_KEY=sua-chave-secreta-aqui
JWT_SECRET_KEY=sua-jwt-secret-aqui

# Banco de dados
DATABASE_URL=postgresql://usuario:senha@host:porta/database

# E-mail (SMTP)
SMTP_HOST=smtp.gmail.com
SMTP_PORT=587
SMTP_USER=seu-email@gmail.com
SMTP_PASS=sua-senha-app

# Configurações adicionais
MAX_CONTENT_LENGTH=16777216
LOG_LEVEL=INFO
```

## 5.4 Deploy no Render

1. Criar conta no [Render.com](https://render.com)
  2. Conectar repositório GitHub
  3. Configurar Web Service:
    - Build Command: `pip install -r requirements.txt`
    - Start Command: `python backend_render_fix.py`
  4. Adicionar variáveis de ambiente
  5. Deploy automático
- 

## 6. Guia de Uso

### 6.1 Primeiro Acesso - Administrador

1. Acessar `/login-admin.html`
2. Login com credenciais fornecidas
3. Alterar senha no primeiro acesso
4. Acessar painel administrativo

### 6.2 Criar Novo Usuário

1. Login como administrador
2. Menu "Usuários" > "Novo Usuário"
3. Preencher dados obrigatórios
4. Sistema envia senha por e-mail
5. Usuário altera senha no primeiro acesso

### 6.3 Aprovar Fornecedor

1. Login como administrador ou comprador
2. Menu "Fornecedores" > "Pendentes"
3. Analisar documentação
4. Aprovar ou rejeitar com justificativa

### 6.4 Fluxo Completo de Compra

1. **Requisitante:** Criar TR detalhado
2. **Comprador:** Aprovar TR
3. **Comprador:** Criar processo de compra
4. **Comprador:** Convidar fornecedores
5. **Fornecedor:** Enviar proposta
6. **Comprador:** Analisar propostas
7. **Sistema:** Gerar comparativo

## 7. APIs e Integrações

### 7.1 Endpoints Principais

#### Autenticação:

POST /api/auth/login  
POST /api/auth/login-fornecedor  
POST /api/auth/alterar-senha  
GET /api/auth/verify

#### Usuários:

POST /api/usuarios/criar  
GET /api/usuarios  
PUT /api/usuarios/{id}/ativar  
POST /api/usuarios/{id}/resetar-senha

#### Fornecedores:

POST /api/fornecedores/cadastro  
GET /api/fornecedores/pendentes  
PUT /api/fornecedores/{id}/aprovar  
GET /api/fornecedores/aprovados

#### Termos de Referência:

GET /api/trs  
POST /api/trs  
GET /api/trs/{id}  
PUT /api/trs/{id}/aprovar  
PUT /api/trs/{id}/reprovar  
GET /api/trs/{id}/pdf

#### Processos:

GET /api/processos  
POST /api/processos  
GET /api/processos/{id}  
POST /api/processos/{id}/convidar  
GET /api/processos/disponiveis

#### Propostas:



```
GET /api/propostas
POST /api/propostas
GET /api/comparativo/{processo_id}
```

## 7.2 Autenticação JWT

Todas as requisições autenticadas devem incluir:

```
javascript

headers: {
  'Authorization': 'Bearer SEU_TOKEN_JWT',
  'Content-Type': 'application/json'
}
```

## 8. Segurança

### 8.1 Medidas Implementadas

- ☒ Senhas com hash bcrypt
- ☒ Tokens JWT com expiração
- ☒ Validação de entrada em todos os endpoints
- ☒ Proteção contra SQL Injection (SQLAlchemy)
- ☒ CORS configurado
- ☒ Rate limiting
- ☒ Logs de auditoria completos
- ☒ Bloqueio após tentativas de login falhadas

### 8.2 Política de Senhas

- Mínimo 8 caracteres
- Pelo menos 1 maiúscula
- Pelo menos 1 minúscula
- Pelo menos 1 número
- Pelo menos 1 caractere especial
- Alteração obrigatória no primeiro acesso

### 8.3 Permissões por Perfil

Ação	Admin	Comprador	Requisitante	Fornecedor
Criar usuários	✓	✗	✗	✗
Aprovar fornecedores	✓	✓	✗	✗
Criar TRs	✗	✗	✓	✗
Aprovar TRs	✗	✓	✗	✗
Criar processos	✗	✓	✗	✗
Enviar propostas	✗	✗	✗	✓
Ver logs auditoria	✓	✗	✗	✗

## 9. Manutenção

### 9.1 Backup do Banco

```
bash

# SQLite (desenvolvimento)
cp sistema_propostas.db backup_$(date +%Y%m%d).db

# PostgreSQL (produção)
pg_dump -U usuario -d database > backup_$(date +%Y%m%d).sql
```

### 9.2 Logs do Sistema

Localização dos logs:

- Desenvolvimento: `app.log`
- Produção: Render Dashboard > Logs

### 9.3 Monitoramento

Métricas importantes:

- Taxa de login bem-sucedido
- Tempo de resposta das APIs
- Número de usuários ativos
- Processos em andamento
- Propostas recebidas

### 9.4 Atualizações

1. Testar em ambiente de desenvolvimento
2. Backup do banco de produção
3. Deploy no Render (automático via Git)
4. Verificar logs pós-deploy

## Suporte

Para suporte técnico ou dúvidas:

- E-mail: [suporte@empresa.com](mailto:suporte@empresa.com)
  - Documentação: </docs>
  - Admin do sistema: [admin@sistema.com](mailto:admin@sistema.com)
- 

**Última atualização:** Dezembro 2024

**Versão:** 1.0.0