Plataforma SaaS Jurídica com Automação e IA para Advogados

■■ Plataforma SaaS Jurídica com Automação e IA para Advogados

Uma plataforma completa de Software como Serviço (SaaS) desenvolvida especificamente para escritórios de advocacia e profissionais do direito, oferecendo automação de processos, inteligência artificial e gestão completa de documentos jurídicos.

■ Índice

■ O que é esta aplicação?

Imagine um **assistente digital completo** para advogados e escritórios de advocacia. Esta plataforma SaaS (Software as a Service) funciona como uma **central de comando** que ajuda profissionais do direito a:

• **Automatizar tarefas repetitivas** como criação de documentos

• **Organizar e gerenciar** todos os casos e clientes

• **Usar inteligência artificial** para análise de documentos e pesquisa jurídica

• **Controlar prazos** e compromissos importantes

• **Gerar relatórios** e estatísticas do escritório

É como ter um **escritório digital completo** que funciona 24/7, acessível de qualquer lugar do mundo através do navegador.

■ Principais Funcionalidades

■ **Inteligência Artificial Integrada**

• **Análise automática de contratos** e documentos jurídicos

• **Geração de peças processuais** baseada em modelos e IA

• **Pesquisa jurídica inteligente** com sugestões contextuais

• **Revisão automática** de documentos para identificar inconsistências

■ **Gestão de Documentos**

• **Upload e organização** de documentos por cliente/caso

• **Controle de versões** automático

• **Busca avançada** por conteúdo, tags ou metadados

• **Exportação** em múltiplos formatos (PDF, DOCX)

■ **Gestão de Clientes e Casos**

• **Cadastro completo** de clientes e casos

• **Histórico detalhado** de todas as interações

• **Sistema de tags** para categorização

• **Dashboard** com visão geral de todos os casos

■ **Controle de Prazos e Agenda**

• **Calendário integrado** com compromissos jurídicos

• **Alertas automáticos** para prazos processuais

• **Notificações** por email e na plataforma

• **Sincronização** com calendários externos

■ **Relatórios e Analytics**

• **Dashboards interativos** com métricas do escritório
• **Relatórios personalizados** de produtividade
• **Análise de tempo** gasto por cliente/caso
• **Estatísticas financeiras** e de performance

■ **Segurança e Compliance**

• **Autenticação JWT** com tokens seguros
• **Controle de acesso** baseado em papéis (admin, user)
• **Logs de auditoria** de todas as ações
• **Backup automático** dos dados
• **Conformidade com LGPD** (Lei Geral de Proteção de Dados)

■■ **Sistema de Configurações Avançado**

• **Ambientes separados** para teste e produção
• **Feature flags** para habilitar/desabilitar funcionalidades
• **Configurações personalizáveis** por usuário/escritório
• **Limites dinâmicos** de documentos e usuários

■■ Tecnologias Utilizadas

Backend (Servidor)

• **Python 3.10+** - Linguagem de programação principal
• **Flask** - Framework web minimalista e poderoso
• **SQLAlchemy** - ORM para gerenciamento do banco de dados
• **Flask-JWT-Extended** - Autenticação e autorização
• **SQLite** - Banco de dados (desenvolvimento)
• **PostgreSQL** - Banco de dados (produção)

Frontend (Interface)

• **React 18** - Biblioteca para interfaces dinâmicas
• **TypeScript** - JavaScript com tipagem estática
• **Vite** - Ferramenta de build ultra-rápida

• **Tailwind CSS** - Framework CSS utilitário

• **Axios** - Cliente HTTP para API

Inteligência Artificial

• **OpenAI GPT** - Modelos de linguagem avançados

• **Processamento de Linguagem Natural** (NLP)

• **Machine Learning** para análise de documentos

Infraestrutura

• **Docker** - Containerização da aplicação

• **Nginx** - Servidor web e proxy reverso

• **Redis** - Cache e sessões

• **AWS/GCP** - Cloud hosting (produção)

■ Pré-requisitos

Antes de começar, você precisa ter instalado em sua máquina:

Obrigatórios:

• **Python 3.10 ou superior** (Download aqui)

• **Node.js 18 ou superior** (Download aqui)

• **Git** (Download aqui)

Opcionais (mas recomendados):

• **Visual Studio Code** - Editor de código

• **Docker** - Para containerização

• **Postman** - Para testar APIs

Como verificar se está tudo instalado:

```bash

# Verificar Python
```

```
python --version # ou python3 --version
```

Verificar Node.js

node --version

Verificar npm

npm --version

Verificar Git

git --version
```

---

## ■ Como Rodar Localmente

### **Passo 1: Clonar o Repositório**

```bash

## Clone o projeto

git clone

## Entre na pasta do projeto

cd "Plataforma SaaS Jurídica com Automação e IA para Advogados"
```

Passo 2: Configurar o Backend (Servidor)

```bash

## Criar ambiente virtual Python

```
python -m venv venv
```

Ativar o ambiente virtual

No macOS/Linux:

source venv/bin/activate

No Windows:

venv\Scripts\activate

Instalar dependências Python

pip install --registry=https://registry.npmjs.org/ -r requirements.txt

```

### **Passo 3: Configurar o Banco de Dados**

    ```bash

## Configurar variável de ambiente do banco

export DATABASE_URL="sqlite:///$(pwd)/src/jurissaas.db"

## Inicializar o banco de dados

cd src

    python -c "

from main import create_app

from extensions import db

app = create_app()

with app.app_context():

db.create_all()

print(' Banco de dados criado!')

"

```
```

Passo 4: Configurar o Frontend (Interface)

```bash

# Voltar para a raiz do projeto

cd ..

# Entrar na pasta do frontend

cd frontend

# Instalar dependências Node.js

npm install --registry=https://registry.npmjs.org/

# Configurar variáveis de ambiente

cp .env.example .env.local

# Edite o arquivo .env.local com suas configurações

```

Passo 5: Iniciar a Aplicação

Terminal 1 - Backend:

```bash

# Na pasta raiz do projeto

source venv/bin/activate

export DATABASE_URL="sqlite:///$(pwd)/src/jurissaas.db"

cd src

python main.py

```

Terminal 2 - Frontend:

```bash
```

Na pasta frontend

cd frontend

npm run dev

```

### **Passo 6: Acessar a Aplicação**

• **Frontend (Interface):** http://localhost:3000

• **Backend (API):** http://localhost:5005

• **Documentação da API:** http://localhost:5005/docs

### **Passo 7: Criar Usuário Administrador**

```bash

# Execute este comando para criar um usuário admin

cd src

```
python -c "
```

from main import create_app

from extensions import db

from models.user import User

app = create_app()

with app.app_context():

admin = User(

nome='Administrador',

email='admin@jurissaas.com',

senha='admin123',

papel='admin'

)

db.session.add(admin)

db.session.commit()

print(' Usuário admin criado!')

print(' Email: admin@jurissaas.com')

print(' Senha: admin123')

"

```

---

## ■ Estrutura do Projeto

```

Plataforma SaaS Jurídica/

■■■ src/ # Backend (Servidor Python)

■ ■■■ models/ # Modelos do banco de dados

■ ■ ■■■ user.py # Modelo de usuários

■ ■ ■■■ subscription.py # Modelo de assinaturas

■ ■ ■■■ config.py # Modelo de configurações

■ ■ ■■■ ...

■ ■■■ routes/ # Rotas da API

■ ■ ■■■ auth.py # Autenticação

■ ■ ■■■ documents.py # Gestão de documentos

■ ■ ■■■ clients.py # Gestão de clientes

■ ■ ■■■ ...

■ ■■■ services/ # Serviços e lógica de negócio

■ ■ ■■■ ai_service.py # Serviços de IA

■ ■ ■■■ document_service.py # Processamento de documentos

■ ■ ■■■ ...

■ ■■■ main.py # Arquivo principal do servidor

■ ■■■ config.py # Configurações gerais

■ ■■■ jurissaas.db # Banco de dados SQLite

■

■■■ frontend/ # Frontend (Interface React)

■ ■■■ src/

■ ■ ■■■ components/ # Componentes React

■ ■ ■■■ pages/ # Páginas da aplicação

■ ■ ■■■ services/ # Comunicação com API

■ ■ ■ ■■■ api.ts # Configuração base da API

■ ■ ■ ■■■ authService.ts # Serviço de autenticação

■ ■ ■ ■■■ ...

■ ■ ■■■ types/ # Tipos TypeScript

■ ■ ■■■ App.tsx # Componente principal

■ ■■■ package.json # Dependências Node.js

■ ■■■ vite.config.ts # Configuração do Vite

■

■■■ docs/ # Documentação

■ ■■■ config-api.md # API de configurações

■ ■■■ ...

■

■■■ migrations/ # Migrações do banco

■■■ requirements.txt # Dependências Python

■■■ .env.example # Exemplo de variáveis de ambiente

■■■ README.md # Este arquivo

```

---

## ■■ Configurações de Ambiente

A aplicação suporta configurações diferenciadas para **desenvolvimento** (test) e **produção** (prod):

### **Ambiente de Desenvolvimento (test):**

• Debug mode ativado

• Limites menores de documentos (100)

• Funcionalidades experimentais habilitadas

• Logs detalhados

### **Ambiente de Produção (prod):**

• Debug mode desativado

• Limites maiores de documentos (1000)

• Funcionalidades estáveis

• Rate limiting ativado

• Sessões com timeout configurado

### **Configurar via Interface:**

```typescript

// Exemplo de uso no frontend

import { setFlagsFromString, getFlags } from './services/authService';

// Configurar flags de desenvolvimento

await setFlagsFromString(

'debug_mode=true,api_version=v2,max_users=50',

'test'

);

// Obter configurações atuais

const flags = await getFlags('prod');

```

■ API e Endpoints

Autenticação

• `POST /api/auth/register` - Registrar novo usuário

• `POST /api/auth/login` - Fazer login

• `GET /api/auth/me` - Obter dados do usuário logado

• `POST /api/auth/refresh` - Renovar token

Configurações

• `POST /api/auth/set-flags` - Configurar flags de ambiente

• `GET /api/auth/flags` - Obter configurações atuais

Documentos (em desenvolvimento)

• `GET /api/documents` - Listar documentos

• `POST /api/documents` - Upload de documento

• `PUT /api/documents/:id` - Atualizar documento

• `DELETE /api/documents/:id` - Excluir documento

Exemplo de Uso da API:

```bash

## Login

curl -X POST http://localhost:5005/api/auth/login \

-H "Content-Type: application/json" \

-d '{"email":"admin@jurissaas.com","senha":"admin123"}'

## Configurar flags (requer token)

curl -X POST http://localhost:5005/api/auth/set-flags \

-H "Authorization: Bearer SEU_TOKEN" \

-H "Content-Type: application/json" \

-d '{"flags":"max_users=100,debug=true","environment":"test"}'

```

■ Interface do Usuário

Dashboard Principal

• Visão geral de casos ativos

• Prazos próximos do vencimento

• Estatísticas de produtividade

• Acesso rápido às funcionalidades

Gestão de Clientes

• Lista completa de clientes

• Formulários de cadastro intuitivos

• Histórico de interações

• Dados de contato organizados

Gestão de Documentos

• Upload por drag-and-drop

• Visualização de documentos

• Sistema de tags e categorias

• Busca avançada por conteúdo
```

Configurações

• Preferências do usuário

• Configurações do escritório

• Gerenciamento de usuários

• Configurações de ambiente

■ Funcionalidades de IA

Análise de Documentos

```python

# Exemplo de análise automática

def analyze_contract(document_content):

"""

Analisa um contrato e extrai informações importantes

"""

ai_analysis = ai_service.analyze_document(document_content)

return {

'contract_type': ai_analysis.type,

'key_clauses': ai_analysis.clauses,

'risks': ai_analysis.risks,

'suggestions': ai_analysis.suggestions

}

```

Geração de Peças

• Templates inteligentes para petições

• Preenchimento automático com dados do caso

• Revisão gramatical e jurídica

• Formatação automática segundo normas

Pesquisa Jurisprudencial

• Busca em bases de dados jurídicas

• Sugestões baseadas no contexto do caso

• Análise de precedentes

• Resumo automático de decisões

■ Sistema de Autenticação

Níveis de Acesso:

• **Admin:** Acesso total ao sistema

• **Superuser:** Acesso avançado com algumas restrições

• **User:** Acesso básico às funcionalidades

Segurança:

• Tokens JWT com expiração automática

• Refresh tokens para renovação segura

• Rate limiting para prevenir ataques

• Logs de auditoria de todas as ações

Exemplo de Uso:

```typescript
// Login
const response = await login('email@exemplo.com', 'senha123');
localStorage.setItem('token', response.access_token);
// Verificar se o usuário está logado
const isValid = await verifyToken(token);
```

■ Planos e Assinaturas

Plano Básico (Trial)

• 10 documentos por mês

• Funcionalidades básicas

• Suporte por email

• 1 usuário

Plano Editor IA

• 100 documentos por mês

• IA para análise de documentos

• Geração automática de peças

• Até 5 usuários

Plano Completo

• Documentos ilimitados

• Todas as funcionalidades de IA

• API para integrações

• Suporte prioritário

• Usuários ilimitados

■ Solução de Problemas Comuns

Erro de Conexão com o Banco:

```bash

# Verificar se o banco foi criado

ls -la src/jurissaas.db

# Recriar o banco se necessário

rm src/jurissaas.db
```

```
cd src && python -c "from main import create_app; from extensions import db; app
= create_app(); app.app_context().push(); db.create_all()"
```

```

**Erro de Porta em Uso:**

```bash
# Verificar processos na porta 5005
lsof -i :5005

# Matar processo se necessário
kill -9
```

Erro de Dependências:

```bash
# Reinstalar dependências Python
pip install --upgrade -r requirements.txt

# Reinstalar dependências Node.js
cd frontend && rm -rf node_modules && npm install
```

■ Próximas Funcionalidades

- [ ] **Integração com tribunais** para consulta processual
- [ ] **App mobile** para iOS e Android
- [ ] **Integração com WhatsApp** para atendimento
- [ ] **Sistema de faturamento** integrado
- [ ] **Relatórios avançados** com BI
```

• [ ] **API pública** para integrações

• [ ] **Backup na nuvem** automático

• [ ] **Assinatura digital** de documentos

■ Contribuição

Quer contribuir com o projeto? Siga estes passos:

• **Fork** o repositório

• **Crie uma branch** para sua funcionalidade (`git checkout -b feature/nova-funcionalidade`)

• **Commit** suas mudanças (`git commit -m 'Adiciona nova funcionalidade'`)

• **Push** para a branch (`git push origin feature/nova-funcionalidade`)

• **Abra um Pull Request**

Diretrizes:

• Escreva testes para novas funcionalidades

• Mantenha o código bem documentado

• Siga os padrões de codificação do projeto

• Atualize a documentação quando necessário

■ Licença

Este projeto está licenciado sob a MIT License - veja o arquivo LICENSE para detalhes.

■ Suporte e Contato

• **Email:** suporte@jurissaas.com

• **Chat:** Disponível na plataforma

• **Documentação:** https://docs.jurissaas.com

• **Bugs:** Abra uma issue no GitHub

■ Agradecimentos

Agradecemos a todos os advogados e profissionais do direito que contribuíram com feedback e sugestões para tornar esta plataforma uma realidade!

Feito com para revolucionar o mundo jurídico através da tecnologia!

Informações do Documento

Este documento foi gerado automaticamente a partir do arquivo README.md

Plataforma SaaS Jurídica com Automação e IA para Advogados

Documento técnico para desenvolvedores e usuários