

Sistema de Agendamentos - Documentação Completa

Índice

1. [Visão Geral](#)
 2. [Funcionalidades](#)
 3. [Arquitetura do Sistema](#)
 4. [Instalação e Configuração](#)
 5. [Manual de Uso](#)
 6. [Documentação Técnica](#)
 7. [APIs e Endpoints](#)
 8. [Estrutura de Arquivos](#)
 9. [Troubleshooting](#)
 10. [Changelog](#)
-

Visão Geral

O Sistema de Agendamentos é uma aplicação web completa desenvolvida para gerenciar clientes, agendamentos e fila de atendimento em escritórios e consultórios. O sistema oferece uma interface moderna e intuitiva, com funcionalidades avançadas como painel de chamada para monitor externo, síntese de voz e integração em tempo real.

URL de Produção

<https://w5hni7c7l7n0.manus.space>

Credenciais de Acesso

- **Assistente:** assistente@escritorio.com / assistente123
 - **Profissional:** profissional@escritorio.com / profissional123
-

Funcionalidades

Sistema de Autenticação

- Login seguro com diferentes perfis de usuário
- Sessões persistentes
- Controle de acesso por funcionalidade

Gestão de Clientes

- **Cadastro completo** com todos os dados necessários
- **Campos obrigatórios:** Nome, CPF, Telefone
- **Campos opcionais:** Data nascimento, Email, Endereço, Observações
- **Máscaras de entrada** para CPF e telefone
- **Operações CRUD** completas (Criar, Ler, Atualizar, Deletar)
- **Interface responsiva** com modal profissional

Sistema de Agendamentos

- **Agendamento por data e horário**
- **Seleção de cliente** via dropdown
- **Controle de status:** AGENDADO → PRESENTE → EM_ATENDIMENTO → ATENDIDO
- **Filtros por data** e status

- **Visualização do dia** com estatísticas

Check-in de Clientes

- **Interface para recepção**
- **Busca por nome ou CPF**
- **Estatísticas em tempo real**
- **Status visual** com badges coloridos
- **Processo de check-in** com um clique

Fila de Atendimento

- **Visualização da fila** ordenada por prioridade e horário
- **Sistema de priorização** para casos urgentes
- **Botão "Chamar Próximo"** com integração automática
- **Controle de atendimento** em tempo real
- **Finalização de atendimento**

Painel de Chamada (Monitor Externo)

- **Tela em fullscreen** otimizada para monitores externos
- **Nome do cliente em MAIÚSCULO** com fonte gigante
- **Instrução clara:** "FAVOR DIRIGIR-SE AO CONSULTÓRIO"
- **Informações completas:** CPF, horário, prioridade
- **Sinal sonoro automático** (3 bips)
- **Síntese de voz** chamando o cliente pelo nome
- **Atualização automática** a cada 3 segundos

- **Abertura em nova aba** para posicionamento no monitor



Recursos de Áudio

- **Sinal sonoro** gerado via Web Audio API
- **Síntese de voz** em português brasileiro
- **Chamada automática** quando cliente é chamado
- **Múltiplos formatos** de áudio para compatibilidade



Dashboard e Estatísticas

- **Visão geral do dia** com números atualizados
 - **Contadores em tempo real:** Total, Presentes, Em Atendimento, Atendidos
 - **Próximos na fila** sempre visíveis
 - **Interface intuitiva** com cards informativos
-



Arquitetura do Sistema



Stack Tecnológico

Backend

- **Framework:** Flask (Python)
- **Banco de Dados:** SQLite
- **ORM:** SQLAlchemy
- **CORS:** Flask-CORS
- **Autenticação:** Sessions

Frontend

- **Tecnologia:** HTML5, CSS3, JavaScript (Vanilla)
- **Design:** Responsivo com CSS Grid e Flexbox
- **Comunicação:** Fetch API para requisições AJAX
- **Áudio:** Web Audio API e Speech Synthesis API

Deploy

- **Plataforma:** Manus Cloud
- **URL Permanente:** <https://w5hni7c7l7n0.manus.space>
- **SSL:** Certificado automático
- **CDN:** Distribuição global



Fluxo de Dados

Plain Text

```
Frontend (JavaScript) ↔ Backend (Flask) ↔ Database (SQLite)
      ↓
    Painel de Chamada (Tempo Real)
      ↓
    Áudio + Síntese de Voz
```



Comunicação em Tempo Real

1. **Frontend** envia dados via POST para `/api/fila/chamar-proximo`
 2. **Backend** atualiza estado global com timestamp único
 3. **Painel** busca atualizações via GET em `/api/painel/status` a cada 3s
 4. **Detecção de mudança** por comparação de timestamp
 5. **Ativação automática** de som e voz no painel
-

Instalação e Configuração

Pré-requisitos

- Python 3.11+
- Flask
- SQLAlchemy
- Flask-CORS

Instalação Local

Bash

```
# 1. Clone ou baixe os arquivos do projeto
mkdir sistema-agendamentos
cd sistema-agendamentos
```

```
# 2. Instale as dependências
pip install flask flask-cors sqlalchemy
```

```
# 3. Estrutura de arquivos necessária
sistema_agendamentos_novo/
```

```
├─ src/
│   ├── main.py
│   ├── static/
│   │   ├── index.html
│   │   └── painel.html
│   └── models/
│       └── user.py
```

```
# 4. Execute o servidor
cd src
python main.py
```

```
# 5. Acesse no navegador
http://localhost:5000
```

Deploy em Produção

O sistema está configurado para deploy automático na plataforma Manus Cloud:

Bash

```
# Deploy via Manus CLI  
manus-deploy-backend flask sistema_agendamentos_novo/
```

Configurações

Variáveis de Ambiente

Python

```
SECRET_KEY = 'sistema-agendamentos-2024'  
SQLALCHEMY_DATABASE_URI = 'sqlite:///database/app.db'  
SQLALCHEMY_TRACK_MODIFICATIONS = False
```

CORS

Python

```
CORS(app, origins="*", supports_credentials=True)
```



Manual de Uso

Acesso ao Sistema

1. **Acesse:** <https://w5hni7c7l7n0.manus.space>
2. **Faça login** com uma das credenciais:
 - Assistente: assistente@escritorio.com / assistente123
 - Profissional: profissional@escritorio.com / profissional123

Gestão de Clientes

Cadastrar Novo Cliente

1. Clique em " **Clientes**" no menu lateral
2. Clique no botão " **Novo Cliente**"
3. Preencha os campos obrigatórios (marcados com *)
4. Clique em "**Salvar Cliente**"

Editar Cliente



1. Na lista de clientes, clique no botão "**Editar**" (azul)
2. Modifique os dados necessários
3. Clique em "**Salvar Cliente**"

Excluir Cliente

1. Na lista de clientes, clique no botão "**Excluir**" (vermelho)
2. Confirme a exclusão

Agendamentos

Criar Agendamento


1. Clique em " **Agendamentos**" no menu lateral
2. Clique no botão " **Novo Agendamento**"
3. Selecione o cliente, data e horário
4. Adicione observações se necessário
5. Clique em "**Salvar Agendamento**"

Visualizar Agendamentos

- **Dashboard:** Mostra agendamentos do dia atual
- **Página de Agendamentos:** Lista completa com filtros
- **Status:** AGENDADO, PRESENTE, EM_ATENDIMENTO, ATENDIDO



Processo de Check-in

Para a Recepção


1. Clique em " **Check-in**" no menu lateral
2. Use a busca para encontrar o cliente (nome ou CPF)
3. Clique no botão "**Check-in**" quando o cliente chegar
4. Status muda automaticamente para **PRESENTE**

Fila de Atendimento

Gerenciar a Fila


1. Clique em " **Fila de Atendimento**" no menu lateral
2. Visualize clientes aguardando (status PRESENTE)
3. Use "**Priorizar**" para casos urgentes
4. Clique " **Chamar Próximo**" para chamar o primeiro da fila

Chamar Cliente

1. Clique em " **Chamar Próximo**"
2. O sistema automaticamente:
 - Move cliente para EM_ATENDIMENTO
 - Atualiza o painel de chamada
 - Toca sinal sonoro
 - Chama cliente por voz
3. Clique "**Finalizar Atendimento**" quando concluir

Painel de Chamada

Configuração do Monitor Externo

1. Clique em " **Painel de Chamada**" no menu lateral
2. O painel abre automaticamente em nova aba/janela
3. **Posicione esta janela no monitor da recepção**
4. Deixe em fullscreen para melhor visualização

Funcionamento Automático

- **Atualização:** A cada 3 segundos
- **Chamada:** Automática quando "Chamar Próximo" é clicado
- **Som:** 3 bips + síntese de voz
- **Visual:** Nome em MAIÚSCULO + instruções



Recursos de Áudio

Configuração do Navegador

1. **Permita áudio** quando solicitado pelo navegador
2. **Volume adequado** para que clientes ouçam
3. **Teste inicial** clicando na tela do painel (ativa áudio)

Síntese de Voz

- **Idioma:** Português brasileiro
 - **Frase:** "[Nome do Cliente], favor dirigir-se ao consultório"
 - **Velocidade:** 0.8x (mais clara)
 - **Volume:** 0.8 (adequado para ambiente)
-

Documentação Técnica

Estrutura de Arquivos

Plain Text

```
sistema_agendamentos_novo/
├── src/
│   ├── main.py                # Servidor Flask principal
│   ├── static/
│   │   ├── index.html        # Interface principal
│   │   └── painel.html       # Painel de chamada
│   ├── models/
│   │   └── user.py           # Modelos de dados
│   └── database/
│       └── app.db            # Banco SQLite
```

Modelo de Dados

Usuários

Python

```
class User:
    id: Integer (Primary Key)
    email: String(120) (Unique)
    password: String(255)
    perfil: PerfilEnum (ASSISTENTE, PROFISSIONAL)
    ativo: Boolean
```

Clientes (Frontend)

JavaScript

```
{
  id: Number,
  nome: String,
  cpf: String,
  telefone: String,
  email: String,
  dataNascimento: String,
  endereco: String,
```

```
observacoes: String
}
```

Agendamentos (Frontend)

JavaScript

```
{
  id: Number,
  clienteId: Number,
  clienteNome: String,
  data: String (YYYY-MM-DD),
  horario: String (HH:MM),
  status: String (AGENDADO|PRESENTE|EM_ATENDIMENTO|ATENDIDO),
  observacoes: String,
  prioridade: Boolean,
  chamadoEm: String
}
```

Estado Global (Backend)

Python

```
estado_chamada = {
    'cliente_atual': {
        'nome': String,
        'cpf': String,
        'horario': String,
        'prioridade': Boolean
    },
    'timestamp_chamada': String (ISO format)
}
```

CSS Classes Principais

Layout

- `.container` - Container principal
- `.sidebar` - Menu lateral
- `.content` - Área de conteúdo

- `.modal` - Modais do sistema

Componentes

- `.btn` - Botões padrão
- `.btn-primary` - Botão principal (azul)
- `.btn-success` - Botão de sucesso (verde)
- `.btn-danger` - Botão de perigo (vermelho)
- `.badge` - Badges de status
- `.card` - Cards informativos

Painel de Chamada

- `.painel-container` - Container do painel
- `.chamada-ativa` - Estado de chamada ativa
- `.nome-cliente` - Nome em destaque
- `.instrucao` - Instrução para o cliente
- `.aguardando` - Estado de aguardando

Responsividade

Breakpoints

CSS

```
/* Desktop */
@media (min-width: 1200px) { ... }

/* Tablet */
@media (max-width: 1200px) { ... }

/* Mobile */
@media (max-width: 768px) { ... }
```

Adaptações Mobile

- Menu lateral colapsável
 - Tabelas com scroll horizontal
 - Botões maiores para touch
 - Fonte ajustada para legibilidade
-

APIs e Endpoints

Autenticação

POST /api/login

Realiza login no sistema.

Request:

JSON

```
{
  "email": "assistente@escritorio.com",
  "password": "assistente123"
}
```

Response:

JSON

```
{
  "success": true,
  "message": "Login realizado com sucesso",
  "user": {
    "email": "assistente@escritorio.com",
    "perfil": "ASSISTENTE"
  }
}
```

POST /api/logout

Realiza logout do sistema.

Response:

JSON

```
{
  "success": true,
  "message": "Logout realizado com sucesso"
}
```

Clientes

GET /api/clientes

Lista todos os clientes.

Response:

JSON

```
{
  "success": true,
  "clientes": [
    {
      "id": 1,
      "nome": "João Silva",
      "cpf": "123.456.789-00",
      "telefone": "(11) 99999-9999",
      "email": "joao@email.com"
    }
  ]
}
```

POST /api/clientes

Cria novo cliente.

Request:

JSON

```
{
  "nome": "Maria Santos",
  "cpf": "987.654.321-00",
  "telefone": "(11) 88888-8888",
  "email": "maria@email.com"
}
```

PUT /api/clientes/{id}

Atualiza cliente existente.

DELETE /api/clientes/{id}

Remove cliente.

17 Agendamentos

GET /api/agendamentos

Lista agendamentos com filtros opcionais.

Query Parameters:

- `data` - Filtrar por data (YYYY-MM-DD)
- `status` - Filtrar por status

Response:

JSON

```
{
  "success": true,
  "agendamentos": [
    {
      "id": 1,
      "cliente": {
        "nome": "João Silva",
        "cpf": "123.456.789-00"
      },
      "data": "2025-07-13",
      "horario": "09:00",
      "status": "AGENDADO"
    }
  ]
}
```



```
}  
]  
}
```

POST /api/agendamentos

Cria novo agendamento.

PUT /api/agendamentos/{id}

Atualiza agendamento.

DELETE /api/agendamentos/{id}

Remove agendamento.

✓ Check-in

POST /api/checkin/{agendamento_id}

Realiza check-in do cliente.

Response:

JSON

```
{  
  "success": true,  
  "message": "Check-in realizado com sucesso"  
}
```

🎯 Fila de Atendimento

GET /api/fila

Obtém estado atual da fila.

Response:

JSON

```
{
  "success": true,
  "fila": [
    {
      "id": 1,
      "cliente": {
        "nome": "João Silva",
        "cpf": "123.456.789-00"
      },
      "horario": "09:00",
      "status": "PRESENTE",
      "posicao": 1,
      "prioridade": false
    }
  ],
  "em_atendimento": null,
  "estatisticas": {
    "total_agendados": 2,
    "presentes": 1,
    "em_atendimento": 0,
    "atendidos": 0
  }
}
```

POST /api/fila/chamar-proximo

Chama próximo cliente da fila.

Request:

JSON

```
{
  "cliente": {
    "nome": "João Silva",
    "cpf": "123.456.789-00",
    "horario": "09:00",
    "prioridade": false
  }
}
```

Response:

JSON

```
{
  "success": true,
  "message": "Próximo cliente chamado",
  "cliente_chamado": {
    "nome": "João Silva",
    "cpf": "123.456.789-00",
    "horario": "09:00",
    "prioridade": false
  },
  "timestamp": "2025-07-13T15:02:57.211508"
}
```

POST /api/fila/priorizar/{agendamento_id}

Prioriza cliente na fila.

Painel de Chamada

GET /api/painel/status

Obtém status atual do painel.

Response:

JSON

```
{
  "success": true,
  "cliente_atual": {
    "nome": "João Silva",
    "cpf": "123.456.789-00",
    "horario": "09:00",
    "prioridade": false
  },
  "timestamp_chamada": "2025-07-13T15:02:57.211508",
  "proximos": [
    {
      "nome": "Maria Santos",
      "horario": "10:00",
      "posicao": 1,
      "prioridade": false
    }
  ],
  "estatisticas": {
```

```
    "total_agendados": 2,  
    "presentes": 1,  
    "em_atendimento": 1,  
    "atendidos": 0  
  }  
}
```

Horários

GET /api/horarios-disponiveis

Lista horários disponíveis para agendamento.

Query Parameters:

- `data` - Data para verificar disponibilidade

Response:

JSON

```
{  
  "success": true,  
  "horarios": [  
    {"horario": "09:00", "disponivel": true},  
    {"horario": "10:00", "disponivel": false},  
    {"horario": "11:00", "disponivel": true}  
  ]  
}
```

Troubleshooting

Problemas Comuns

1. Painel não atualiza automaticamente

Sintomas:

- Nome do cliente não aparece no painel

- Sem som ou voz quando chamado

Soluções:

1. Verifique se o painel está aberto em: `/painel.html`
2. Abra o console do navegador (F12) e verifique logs
3. Certifique-se que não há bloqueadores de pop-up
4. Teste a conectividade: `fetch('/api/painel/status')`

Debug:

JavaScript

```
// No console do painel
console.log('Testando API...');
fetch('/api/painel/status')
  .then(r => r.json())
  .then(d => console.log('Dados:', d))
  .catch(e => console.error('Erro:', e));
```

2. Áudio não funciona

Sintomas:

- Sem sinal sonoro
- Síntese de voz não fala

Soluções:

1. **Clique na tela do painel** para ativar áudio
2. Verifique permissões de áudio no navegador
3. Teste volume do sistema
4. Verifique se há bloqueadores de áudio

Teste manual:

JavaScript

```
// No console do painel  
speechSynthesis.speak(new SpeechSynthesisUtterance('Teste de voz'));
```

3. Login não funciona

Sintomas:

- Erro ao fazer login
- Redirecionamento não acontece

Soluções:

1. Verifique credenciais:
 - assistente@escritorio.com / assistente123
 - profissional@escritorio.com / profissional123
2. Limpe cache do navegador
3. Verifique conexão com servidor

4. Dados não salvam

Sintomas:

- Clientes/agendamentos não persistem
- Erro ao salvar

Soluções:

1. Verifique conexão com backend
2. Verifique logs do servidor
3. Confirme que banco de dados está acessível

Logs e Debug

Frontend (Console do Navegador)

JavaScript

```
// Ativar logs detalhados
localStorage.setItem('debug', 'true');

// Ver logs da fila
console.log('=== CHAMAR PRÓXIMO INICIADO ===');

// Ver logs do painel
console.log('Buscando dados do painel...');
```

Backend (Terminal)

Python

```
# Logs automáticos no terminal
print(f"Cliente chamado: {proximo_cliente['nome']}")
print(f"Timestamp: {estado_chamada['timestamp_chamada']}")
```



Problemas de Conectividade

Verificar Status do Servidor

Bash

```
curl -I https://w5hni7c7l7n0.manus.space
```

Testar APIs

JavaScript

```
// Testar login
fetch('/api/login', {
  method: 'POST',
  headers: {'Content-Type': 'application/json'},
  body: JSON.stringify({
    email: 'assistente@escritorio.com',
    password: 'assistente123'
  })
}).then(r => r.json()).then(console.log);
```

```
// Testar painel
fetch('/api/painel/status')
  .then(r => r.json())
  .then(console.log);
```

Changelog

Versão 2.0 (Atual) - 13/07/2025

✨ Novas Funcionalidades

- **Painel de chamada para monitor externo**
- **Integração automática** entre fila e painel
- **Síntese de voz** em português brasileiro
- **Sinal sonoro** automático via Web Audio API
- **Atualização em tempo real** com timestamps
- **Sistema de priorização** de clientes
- **Logs de debug** detalhados

🔧 Melhorias

- **Interface responsiva** aprimorada
- **Comunicação backend-frontend** otimizada
- **Deteção de mudanças** mais robusta
- **Tratamento de erros** melhorado
- **Performance** otimizada

🐛 Correções

- Corrigido problema de sincronização do painel
- Corrigido detecção de novas chamadas
- Corrigido áudio em diferentes navegadores
- Corrigido responsividade em mobile

Versão 1.0 - 12/07/2025

✨ Funcionalidades Iniciais

- **Sistema de login** com perfis
 - **Gestão completa de clientes** (CRUD)
 - **Sistema de agendamentos**
 - **Check-in de clientes**
 - **Fila de atendimento** básica
 - **Dashboard** com estatísticas
 - **Interface responsiva**
-

Suporte

Links Úteis

- **Sistema em Produção:** <https://w5hni7c7l7n0.manus.space>
- **Painel de Chamada:** <https://w5hni7c7l7n0.manus.space/painel.html>

Contato

Para suporte técnico ou dúvidas sobre o sistema, consulte a documentação ou entre em contato com a equipe de desenvolvimento.



Próximas Funcionalidades

- Relatórios de atendimento
- Integração com WhatsApp
- Notificações push
- Backup automático
- Multi-tenancy