SGII – Sistema de Gestão Integrada Inteligente

Tema:

Desenvolvimento de uma plataforma web para gerenciamento inteligente de espaços físicos, controle de chaves, reservas e inventário de recursos compartilhados em ambientes institucionais diversos, como empresas, escolas e laboratórios.

Resumo do Escopo:

O SGII tem como objetivo permitir que qualquer instituição — pública ou privada — organize de forma centralizada a reserva de salas e equipamentos, o empréstimo e rastreamento de chaves, e o controle de inventário de recursos físicos. A plataforma pode ser aplicada com pouca ou nenhuma customização em diferentes tipos de organizações, trazendo economia de tempo, redução de erros humanos e maior transparência na gestão de infraestrutura.

Funcionalidades Principais:

- Reserva de ambientes: salas, laboratórios, auditórios, veículos, etc.
- Controle de chaves e acessos físicos: com histórico e relatórios.
- Gestão de inventário: equipamentos, móveis, materiais permanentes.
- Painel administrativo: com indicadores de uso, notificações e alertas.
- Sistema multiusuário com permissões: adequado para empresas, escolas, universidades, clínicas ou qualquer organização com estrutura física.

Tecnologias Sugeridas:

Frontend: React

Backend: Node.js com Express

Banco de dados: PostgreSQL ou MongoDB

Extras:

- Autenticação com JWT
- Upload de arquivos (comprovantes, laudos, etc.)
- Relatórios PDF e gráficos interativos

-

Estrutura do Banco de Dados:

A seguir, a descrição das principais tabelas do sistema:

instituicoes

- id (PK)
- nome
- tipo (empresa, escola, laboratório, etc.)
- cnpj_ou_codigo
- endereco
- telefone
- email

usuarios

- id (PK)
- nome
- email
- senha_hash
- cargo (admin, técnico, usuário, etc.)
- instituicao_id (FK)
- nivel_acesso

espacos

- id (PK)
- nome
- tipo (sala, laboratório, auditório, etc.)
- capacidade
- recursos (texto ou JSON: projetor, AC, etc.)
- instituicao_id (FK)

reservas

- id (PK)
- espaco_id (FK)
- usuario_id (FK)
- data_inicio
- data_fim
- finalidade
- status (pendente, confirmada, cancelada)

chaves

- id (PK)
- codigo_identificador
- espaco_id (FK)

- disponivel (boolean)
- observacoes

movimentacoes_chaves

- id (PK)
- chave_id (FK)
- usuario_id (FK)
- data_retirada
- data_devolucao
- responsavel_entrega
- responsavel_recebimento
- observacoes

tens_inventario

- id (PK)
- nome_item
- descricao
- numero_patrimonio (único)
- localizacao_atual
- quantidade
- estado_conservacao (novo, bom, danificado, etc.)
- data_aquisicao
- instituicao_id (FK)

logs_uso

- id (PK)
- usuario_id (FK)
- acao (reserva criada, chave retirada, item adicionado, etc.)
- entidade (reservas, chaves, inventario, etc.)
- entidade_id
- timestamp

Extras:

- Tabela notificacoes (para alertas de vencimento, reservas próximas)
- Tabela fotos_documentos (para anexar arquivos ou fotos aos registros)
- Tabela permissoes_customizadas (para controle granular de acessos)

Tabela A	Tabela B	Cardinalidade A → B	Cardinalidade B → A	Tipo de Relação
instituicoes	usuarios	(1, N)	(1, 1)	Cada usuário pertence a uma instituição.
instituicoes	espacos	(1, N)	(1, 1)	Cada espaço pertence a uma instituição.
instituicoes	itens_inventario	(1, N)	(1, 1)	Cada item pertence a uma instituição.
usuarios	reservas	(0, N)	(1, 1)	Usuários fazem váresreservas.
espacos	reservas	(0, N)	(1, 1)	Espaços podem ser reservados.
espacos	chaves	(0, 1)	(0, 1)	Cada chave está ligada a um espaço.
chaves	movimentacoes_chaves	(0, N)	(1, 1)	Uma chave pode ter várias movimentações.
usuarios	movimentacoes_chaves	(0, N)	(1, 1)	Um usuário pode movimentar várias chaves.
usuarios	logs_uso	(0, N)	(1, 1)	Cada log é gerado por um usuário.

Entidade	Atributos
instituicoes	id (PK), nome, tipo, cnpj_ou_codigo, endereco, telefone, email
usuarios	id (PK), nome, email, senha_hash, cargo, nivel_acesso, instituicao_id (FK)
espacos	id (PK), nome, tipo, capacidade, recursos, instituicao_id (FK)
reservas	id (PK), espaco_id (FK), usuario_id (FK), data_inicio, data_fim, finalidade, status
chaves	id (PK), codigo_identificador, espaco_id (FK - opcional), disponivel, observacoes
movimentacoes_chaves	id (PK), chave_id (FK), usuario_id (FK - opcional), data_retirada, data_devolucao, responsavel_entrega (FK), responsavel_recebimento (FK), observacoes
itens_inventario	id (PK), nome_item, descricao, numero_patrimonio, localizacao_atual, quantidade, estado_conservacao, data_aquisicao, instituicao_id (FK)

logs_uso	id (PK), usuario_id (FK - opcional), acao, entidade, entidade_id, timestamp

```
-- 🏛️ INSTITUIÇÕES
CREATE TABLE instituicoes (
  id SERIAL PRIMARY KEY,
  nome VARCHAR(100) NOT NULL,
  tipo VARCHAR(50), -- empresa, escola, laboratório, etc.
  cnpj_ou_codigo VARCHAR(30) UNIQUE,
  endereco TEXT,
  telefone VARCHAR(20),
  email VARCHAR(100)
);

-- 👥 USUÁRIOS
CREATE TABLE usuarios (
  id SERIAL PRIMARY KEY,
  nome VARCHAR(100) NOT NULL,
  email VARCHAR(100) UNIQUE NOT NULL,
  senha_hash TEXT NOT NULL,
  cargo VARCHAR(50), -- admin, técnico, usuário, etc.
  nivel_acesso INTEGER DEFAULT 1, -- 1=comum, 2=técnico, 3=admin
  instituicao_id INTEGER NOT NULL REFERENCES instituicoes(id) ON DELETE CASCADE
);
```

```
-- 🏠 ESPAÇOS
CREATE TABLE espacos (
  id SERIAL PRIMARY KEY,
  nome VARCHAR(100) NOT NULL,
  tipo VARCHAR(50), -- sala, laboratório, auditório, etc.
  capacidade INTEGER,
  recursos JSONB, -- exemplo: {"projetor": true, "AC": false}
  instituicao_id INTEGER NOT NULL REFERENCES instituicoes(id) ON DELETE CASCADE
);

-- 🗓️ RESERVAS
CREATE TABLE reservas (
  id SERIAL PRIMARY KEY,
  espaco_id INTEGER NOT NULL REFERENCES espacos(id) ON DELETE CASCADE,
  usuario_id INTEGER NOT NULL REFERENCES usuarios(id) ON DELETE CASCADE,
  data_inicio TIMESTAMP NOT NULL,
  data_fim TIMESTAMP NOT NULL,
  finalidade TEXT,
  status VARCHAR(20) DEFAULT 'pendente' -- pendente, confirmada, cancelada
);

-- 🔑 CHAVES
CREATE TABLE chaves (
  id SERIAL PRIMARY KEY,
  codigo_identificador VARCHAR(50) UNIQUE NOT NULL,
  espaco_id INTEGER REFERENCES espacos(id), -- opcional: (0,1) cardinalidade
```

```
    disponivel BOOLEAN DEFAULT TRUE,
    observacoes TEXT
);

-- 🔄 MOVIMENTAÇÕES DE CHAVES
CREATE TABLE movimentacoes_chaves (
    id SERIAL PRIMARY KEY,
    chave_id INTEGER NOT NULL REFERENCES chaves(id) ON DELETE CASCADE,
    usuario_id INTEGER REFERENCES usuarios(id) ON DELETE SET NULL,
    data_retirada TIMESTAMP NOT NULL,
    data_devolucao TIMESTAMP,
    responsavel_entrega INTEGER REFERENCES usuarios(id) ON DELETE SET NULL,
    responsavel_recebimento INTEGER REFERENCES usuarios(id) ON DELETE SET NULL,
    observacoes TEXT
);

-- 📦 ITENS DE INVENTÁRIO
CREATE TABLE itens_inventario (
    id SERIAL PRIMARY KEY,
    nome_item VARCHAR(100) NOT NULL,
    descricao TEXT,
    numero_patrimonio VARCHAR(50) UNIQUE,
    localizacao_atual VARCHAR(100),
    quantidade INTEGER DEFAULT 1,
    estado_conservacao VARCHAR(30), -- novo, bom, danificado, etc.
    data_aquisicao DATE,
```

```
    instituicao_id INTEGER NOT NULL REFERENCES instituicoes(id) ON DELETE CASCADE
);

-- 📜 LOGS DE USO
CREATE TABLE logs_uso (
    id SERIAL PRIMARY KEY,
    usuario_id INTEGER REFERENCES usuarios(id) ON DELETE SET NULL,
    acao TEXT NOT NULL, -- ex: reserva criada, chave retirada, etc.
    entidade VARCHAR(50) NOT NULL, -- ex: reservas, chaves, inventario, etc.
    entidade_id INTEGER NOT NULL,
    timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```