



















Wayne Industries

Sistema de Gerenciamento e Segurança

Documentação Técnica Completa

Projeto Full Stack | Next.js 14 | TypeScript | Supabase

 Links do Projeto	4
 Sobre o Projeto	4
 Objetivos do Projeto	4
 Tecnologias Utilizadas	4
Frontend.....	4
Backend	4
 Funcionalidades Principais.....	5
 Autenticação.....	5
 CRUD de Recursos	5
 Dashboard.....	5
 Segurança.....	5
 Scripts SQL do Banco de Dados.....	6
1. Script de Criação do Banco	6
2. Inserção de Usuários de Teste	7
3. Inserção de Recursos de Exemplo	7
 Observações Importantes.....	7
 Estrutura do Banco de Dados	9
user_profiles	9
resources	9
restricted_areas	9
access_logs	9
activities	9
 Segurança Implementada	10
1. Row Level Security (RLS).....	10
2. Autenticação JWT	10
3. Validação	10
 Credenciais de Teste	10
Admin	10
Manager	10
Employee.....	10
 Guia de Instalação	11
Pré-requisitos	11
1. Clone.....	11
2. Instale.....	11
3. Configure .env.local.....	11
4. Execute	11
 Conclusão.....	11
Conquistas	11

Links do Projeto

Repositório GitHub: <https://github.com/Rhuan12/wayne-industries>

Deploy (Vercel): <https://wayne-industries-sigma.vercel.app/dashboard>

Sobre o Projeto

O Wayne Industries System é uma aplicação web completa para gerenciamento de recursos e controle de segurança empresarial. Desenvolvido como projeto acadêmico, o sistema demonstra a implementação de um stack moderno de desenvolvimento web com foco em segurança, usabilidade e escalabilidade.

Objetivos do Projeto

- Implementar autenticação segura com múltiplos níveis de acesso
- Desenvolver CRUD completo de recursos
- Criar dashboard com visualizações interativas
- Implementar sistema de controle de segurança e logs
- Aplicar boas práticas de desenvolvimento Full Stack

Tecnologias Utilizadas

Frontend

- **Next.js 14** - Framework React com App Router
- **TypeScript** - Tipagem estática
- **Tailwind CSS** - Estilização responsiva
- **Recharts** - Gráficos interativos
- **React Hook Form + Zod** - Validação
- **Sonner + Lucide** - UI/UX

Backend

Supabase - Backend as a Service

- PostgreSQL
- Authentication (JWT)
- Row Level Security (RLS)
- Real-time subscriptions

Funcionalidades Principais

Autenticação

- Login seguro com email/senha
- 3 níveis: Admin, Manager, Employee
- Middleware de proteção
- RLS no banco de dados

CRUD de Recursos

- Create, Read, Update, Delete
- Tipos: Equipamentos, Veículos, Dispositivos
- Status: Disponível, Em Uso, Manutenção, Aposentado
- Filtros por tipo e status

Dashboard

- Cards de estatísticas
- Gráficos (barras e linhas)
- Atividades recentes

Segurança

- 5 áreas restritas
- Logs de acesso
- Controle por nível

Scripts SQL do Banco de Dados

Scripts SQL completos para configuração do banco de dados PostgreSQL no Supabase.

1. Script de Criação do Banco

Crie as tabelas e configurações de segurança:

```
-- Habilitar extensão UUID
CREATE EXTENSION IF NOT EXISTS "uuid-oss";

-- Tabela: user_profiles
CREATE TABLE IF NOT EXISTS public.user_profiles (
    id UUID PRIMARY KEY REFERENCES auth.users(id),
    email TEXT NOT NULL UNIQUE,
    full_name TEXT NOT NULL,
    role TEXT NOT NULL CHECK (role IN ('employee', 'manager', 'admin')),
    department TEXT,
    created_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);

-- Tabela: resources
CREATE TABLE IF NOT EXISTS public.resources (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    type TEXT NOT NULL CHECK (type IN ('equipment', 'vehicle',
'security_device')),
    name TEXT NOT NULL,
    description TEXT,
    status TEXT NOT NULL CHECK (status IN ('available', 'in_use', 'maintenance',
'retired')),
    location TEXT,
    created_by UUID NOT NULL REFERENCES public.user_profiles(id),
    created_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);

-- Habilitar Row Level Security
ALTER TABLE public.user_profiles ENABLE ROW LEVEL SECURITY;
ALTER TABLE public.resources ENABLE ROW LEVEL SECURITY;
```

 **Nota:** O script completo está disponível no arquivo *schema.sql* do repositório.

2. Inserção de Usuários de Teste

Insira os perfis dos 3 usuários de teste:

```
-- Substitua os UUIDs pelos IDs copiados do Supabase Auth
INSERT INTO public.user_profiles (id, email, full_name, role, department) VALUES
    ('4323e1e9-210a-4b13-ad6b-e4b842779a9a', 'admin@wayne.com',
    'Bruce Wayne', 'admin', 'Diretoria'),
    ('dbc7e3ac-5947-4d4d-8b84-078c599aa558', 'manager@wayne.com',
    'Alfred Pennyworth', 'manager', 'Operações'),
    ('fc23b83b-64db-4294-a55b-df488727cf7c', 'employee@wayne.com',
    'Lucius Fox', 'employee', 'P&D');
```

3. Inserção de Recursos de Exemplo

Popule o banco com recursos de teste:

```
-- Equipamentos
INSERT INTO public.resources (type, name, description, status, location,
created_by)
VALUES
    ('equipment', 'Batcomputer', 'Supercomputador da Batcaverna',
    'available', 'Batacaverna', '4323e1e9-210a-4b13-ad6b-e4b842779a9a'),
    ('equipment', 'Traje do Batman', 'Armadura kevlar',
    'in_use', 'Armário', '4323e1e9-210a-4b13-ad6b-e4b842779a9a'),

-- Veículos
    ('vehicle', 'Batmóvel', 'Veículo blindado principal',
    'available', 'Garagem', '4323e1e9-210a-4b13-ad6b-e4b842779a9a'),
    ('vehicle', 'Batmoto', 'Motocicleta alta velocidade',
    'in_use', 'Garagem', '4323e1e9-210a-4b13-ad6b-e4b842779a9a'),
    ('vehicle', 'Batwing', 'Aeronave de combate',
    'maintenance', 'Hangar', '4323e1e9-210a-4b13-ad6b-e4b842779a9a'),

-- Dispositivos de Segurança
    ('security_device', 'Sistema de Vigilância', 'Câmeras de segurança',
    'available', 'Sala de Controle', '4323e1e9-210a-4b13-ad6b-e4b842779a9a'),
    ('security_device', 'Scanner de Retina', 'ID biométrica',
    'available', 'Entrada', '4323e1e9-210a-4b13-ad6b-e4b842779a9a');
```

Observações Importantes

1. **UUIDs:** Substitua os UUIDs pelos IDs reais dos usuários criados no Supabase Auth
2. **Ordem:** Execute os scripts na ordem: 1) Criação, 2) Usuários, 3) Recursos
3. **RLS:** As políticas RLS são criadas automaticamente no script completo
4. **Índices:** Índices são criados para melhorar a performance das queries

Estrutura do Banco de Dados

user_profiles

- id (UUID) - PK
- role - employee, manager, admin
- department - Departamento

resources

- type - equipment, vehicle, security_device
- status - available, in_use, maintenance, retired
- created_by - FK para user_profiles

restricted_areas

- required_access_level - Nível mínimo
- is_active - Status da área

access_logs

- action - entry, exit, denied
- timestamp - Data/hora do acesso

activities

- action_type - Tipo da ação
- description - Descrição detalhada

Segurança Implementada

1. Row Level Security (RLS)

- Políticas por tabela
- Controle de acesso granular
- Logs protegidos

2. Autenticação JWT

- Tokens seguros
- Refresh automático

3. Validação

- Client-side: Zod
- Server-side: Supabase

Credenciais de Teste

Admin

- admin@wayne.com / admin123
- Acesso total ao sistema

Manager

- manager@wayne.com / manager123
- Gerenciar recursos

Employee

- employee@wayne.com / employee123
- Visualizar recursos

Guia de Instalação

Pré-requisitos

- Node.js 18+
- npm ou yarn
- Conta Supabase

1. Clone

```
git clone https://github.com/Rhuan12/wayne-industries.git
cd wayne-industries
```

2. Instale

```
npm install
```

3. Configure .env.local

```
NEXT_PUBLIC_SUPABASE_URL=...
NEXT_PUBLIC_SUPABASE_ANON_KEY=...
NEXT_PUBLIC_APP_URL=http://localhost:3000
```

4. Execute

```
npm run dev
```

Acesse: <http://localhost:3000>

Conclusão

O Wayne Industries System demonstra com sucesso a implementação de um sistema full stack moderno e seguro.

Conquistas

- Sistema 100% funcional
- Arquitetura escalável
- Interface moderna
- Segurança robusta

Desenvolvido com  por Rhuan Mateus

Wayne Industries System © 2026