# Proposta de Projeto

# Vigilância Solidária

## Sistema Colaborativo de Monitoramento Comunitário

**Vigilância Solidária** é uma aplicação web completa que implementa um sistema de monitoramento colaborativo entre vizinhos. O projeto simula uma rede inteligente onde câmeras IP podem ser conectadas, alertas de segurança são registrados automaticamente e informações são compartilhadas em tempo real entre os moradores de uma comunidade.

## Objetivo do Projeto

Desenvolver uma solução tecnológica para fortalecer a segurança comunitária através da colaboração entre vizinhos, utilizando conceitos modernos de desenvolvimento web, autenticação segura e interface responsiva. O sistema visa demonstrar a aplicação prática de tecnologias web em cenários de Internet das Coisas (IoT) e segurança urbana.

## Funcionalidades Principais

### Sistema de Autenticação

* Cadastro de usuários com validações robustas (email único, senha segura)
* Login protegido com sistema de sessões server-side
* Logout seguro com encerramento completo da sessão
* Proteção de rotas impedindo acesso não autorizado

### Dashboard Interativa

* Visualização em tempo real de alertas de segurança
* Simulação de câmeras ativas na comunidade
* Interface responsiva adaptável a diferentes dispositivos
* Alertas categorizados por tipo e prioridade

### Backend Robusto

* API RESTful com endpoints organizados
* Armazenamento persistente em arquivos JSON
* Middleware de segurança com Helmet.js
* Tratamento de erros centralizado

### Frontend Moderno

* Interface responsiva com Bootstrap
* Interatividade com jQuery
* Design intuitivo focado na experiência do usuário
* Atualizações dinâmicas sem reload da página

## Arquitetura do Sistema

vigilancia-solidaria/  
├── data/  
│ ├── users.json # Dados dos usuários cadastrados  
│ └── alerts.json # Dados dos alertas registrados  
├── public/  
│ ├── css/ # Arquivos CSS  
│ ├── js/ # Scripts JavaScript do front-end  
│ ├── index.html # Página inicial pública  
│ ├── login.html # Página de login  
│ ├── cadastro.html # Página de cadastro de usuários  
│ ├── dashboard.html # Página principal após login  
│ └── 404.html # Página de erro 404  
├── package.json # Dependências e scripts do projeto  
├── server.js # Servidor Node.js com Express  
└── README.md # Documentação do projeto

## Instalação e Execução

### Pré-requisitos

* Node.js versão 18 ou superior
* Navegador moderno (Chrome, Firefox, Safari, Edge)

### Instruções de Instalação

1. **Extraia o arquivo ZIP** enviado em uma pasta de sua escolha
2. **Navegue até a pasta do projeto** pelo terminal:

cd vigilancia-solidaria

1. **Instale as dependências** do projeto:

npm install

1. **Inicie o servidor** de desenvolvimento:

node server.js

1. **Acesse a aplicação** no navegador:

<http://localhost:3000>

## Documentação da API

|  |  |  |  |
| --- | --- | --- | --- |
| **Método** | **Endpoint** | **Descrição** | **Autenticação** |
| POST | /api/cadastro | Registra um novo usuário no sistema | Sim |
| POST | /api/login | Autentica usuário e cria sessão | Sim |
| POST | /api/logout | Encerra sessão do usuário | Sim |
| GET | /api/user | Retorna dados do usuário logado | Sim |
| POST | /api/simular-alerta | Registra novo alerta de segurança | Sim |

### Exemplos de Requisições

**Cadastro de usuário:**

fetch('/api/cadastro', {  
 method: 'POST',  
 headers: { 'Content-Type': 'application/json' },  
 body: JSON.stringify({  
 nome: 'João Silva',  
 email: '[joao@email.com](mailto:joao@email.com)',  
 senha: 'senha123'  
 })  
});

**Simulação de alerta:**

fetch('/api/simular-alerta', {  
 method: 'POST',  
 headers: { 'Content-Type': 'application/json' },  
 body: JSON.stringify({  
 tipo: 'movimento\_suspeito',  
 descricao: 'Movimento detectado no quintal'  
 })  
});

## Tecnologias Utilizadas

### Backend

* **Node.js** - Runtime JavaScript para servidor
* **Express.js** - Framework web minimalista e flexível
* **express-session** - Middleware para gerenciamento de sessões
* **helmet** - Middleware de segurança para aplicações Express
* **bcrypt** - Biblioteca para criptografia de senhas

### Frontend

* **HTML5** - Linguagem de marcação para estrutura das páginas
* **CSS3** - Folhas de estilo para design responsivo
* **JavaScript (ES6+)** - Linguagem de programação para lógica do frontend
* **jQuery** - Biblioteca JavaScript para manipulação do DOM
* **Bootstrap** - Framework CSS para desenvolvimento responsivo

## Recursos de Segurança Implementados

* **Criptografia de senhas** utilizando bcrypt com salt rounds
* **Proteção contra ataques XSS** através do middleware Helmet.js
* **Validação de dados** tanto no frontend quanto no backend
* **Sessões seguras** com cookies configurados como httpOnly
* **Sanitização de inputs** para prevenir injeções maliciosas
* **Controle de acesso** com middleware de autenticação

## Funcionalidades Técnicas Detalhadas

### Sistema de Sessões

O sistema utiliza express-session para gerenciar sessões de usuário de forma segura. Cada sessão é armazenada no servidor e identificada por um cookie único enviado ao cliente.

### Armazenamento de Dados

Os dados são persistidos em arquivos JSON locais, simulando um banco de dados. Esta abordagem facilita o desenvolvimento e demonstração sem necessidade de configuração de banco de dados externo.

### Interface Responsiva

A interface foi desenvolvida seguindo os princípios de Mobile First, garantindo usabilidade em dispositivos móveis, tablets e desktops.

## Possíveis Expansões Futuras

* Integração com câmeras IP reais via protocolos RTSP/HTTP
* Implementação de notificações push para alertas em tempo real
* Sistema de geolocalização para mapeamento de alertas
* Chat em tempo real entre vizinhos
* Desenvolvimento de aplicativo mobile nativo
* Análise de dados com dashboards estatísticos
* Integração com sistemas de automação residencial

## Objetivos Acadêmicos Alcançados

* **Desenvolvimento Full-Stack:** Implementação completa de frontend e backend
* **API RESTful:** Criação de endpoints seguindo padrões REST
* **Autenticação e Autorização:** Sistema completo de login e proteção de rotas
* **Interface Responsiva:** Design adaptável a diferentes dispositivos
* **Persistência de Dados:** Implementação de armazenamento local
* **Tratamento de Erros:** Validações robustas e feedback ao usuário
* **Organização de Código:** Estrutura clara e manutenível
* **Documentação Técnica:** Documentação completa do projeto

## Aspectos Pedagógicos

Este projeto demonstra a aplicação prática de conceitos fundamentais do desenvolvimento web moderno, incluindo:

* Arquitetura cliente-servidor
* Programação assíncrona com JavaScript
* Manipulação de APIs REST
* Gerenciamento de estado no frontend
* Segurança em aplicações web
* Design responsivo e experiência do usuário

## Metodologia de Desenvolvimento

O projeto foi desenvolvido seguindo boas práticas de engenharia de software:

* **Planejamento:** Definição de requisitos e arquitetura
* **Desenvolvimento Iterativo:** Implementação incremental de funcionalidades
* **Testes Manuais:** Validação contínua das funcionalidades
* **Documentação:** Registro detalhado do processo e decisões técnicas

## Considerações Finais

O sistema Vigilância Solidária representa uma solução completa de aplicação web, demonstrando domínio técnico em desenvolvimento full-stack. A aplicação está totalmente funcional e serve como base sólida para expansões futuras em cenários reais de monitoramento comunitário e cidades inteligentes.

**Tempo estimado de desenvolvimento:** 40 horas

**Nível de complexidade:** Intermediário a Avançado

**Padrões seguidos:** Clean Code, REST API, Responsive Design

**Desenvolvido por:**

João Moura

Estudante de Análise e Desenvolvimento de Sistemas

Especialização em Desenvolvimento Web e Segurança Digital