**FACULDADE DE INFORMAÇÃO E ADMINISTRAÇÃO PAULISTA (FIAP)**

JOÃO PEDRO DE SOUZA VIEIRA, GABRIEL RIQUETO REIS E LEONARDO NICASTRO MANSUR CASTILLO

**GLOBAL SOLUTION**

São Paulo

2025

**JOÃO PEDRO DE SOUZA VIEIRA, GABRIEL RIQUETO REIS E LEONARDO NICASTRO MANSUR CASTILLO**

**GLOBAL SOLUTION**

Documento descritivo para entrega da Global do quinto semestre de Engenharia de Software na FIAP.

Orientador: Vinícius Costa Santos

**SÃO PAULO**

**2025**

**SUMÁRIO**

[**INTRODUÇÃO**](#_4czibvb96m8g) **4**

**1.**[**OBJETIVO DO PROJETO**](#_1rw2ykkzo4mf) **4**

**2.**[**REQUISITOS DO SISTEMA**](#_6f9nq77wbwok) **5**

[2.1 Requisitos Funcionais](#_q0w7uerr7lu0) 5

[2.2 Requisitos Não Funcionais](#_8cmgu92p7agt) 6

**3.**[**FLUXOGRAMA DO SISTEMA**](#_ybe87mcxtzdi) **7**

**4.**[**TECNOLOGIAS UTILIZADAS**](#_jmx2rsuvyhog) **8**

**5.**[**REGRAS DE NEGÓCIO**](#_q51i55ugg2ml) **8**

**6.**[**ESTRUTURA DE PASTAS**](#_bf1sgm78g8l1) **9**

# **INTRODUÇÃO**

A ocorrência de apagões, principalmente causados por eventos climáticos extremos, aumenta significativamente o risco de **acidentes domésticos** como quedas, incêndios e outros incidentes que podem comprometer a segurança das pessoas em casa.

Pensando nisso, desenvolvemos o **SPADA — Sistema de Prevenção de Acidentes Domésticos durante Apagões**: uma solução integrada e multifuncional, que combina tecnologias móveis, desktop e detecção automática por câmeras.

Este documento apresenta o **algoritmo** do SPADA, seguindo a linguagem de programação **C#** para a disciplina de **C-Sharp Software Development**, conforme as orientações da Global Solution da FIAP.

Nossa equipe é composta por:

* **João Pedro de Souza VIeira, RM 99805**
* **Leonardo Nicastro Mansur Castillo, RM 551659**
* **Gabriel Riqueto Reis, RM 98685**

# **OBJETIVO DO PROJETO**

O nosso objetivo é desenvolver uma aplicação de console em C# capaz de prevenir acidentes domésticos relacionados a falhas de energia, como quedas, incêndios e outros riscos.

O SPADA permite:

* Registro de acidentes domésticos.
* Geração de relatórios de incidentes.
* Avaliação do nível de risco do usuário.
* Fornecimento de dicas personalizadas de segurança.
* Gerenciamento de um checklist de preparação para apagões.

A solução proporciona uma interface simples, mas eficiente, permitindo que qualquer usuário registre e consulte informações de segurança relacionadas a incidentes domésticos.

# **REQUISITOS DO SISTEMA**

O SPADA (Sistema de Prevenção de Acidentes Domésticos durante Apagões) visa oferecer um sistema inteligente e preventivo para minimizar acidentes domésticos durante apagões, por meio de um sistema de registro de acidentes sofridos pelo usuário, que gera uma lista de cuidados preventivos com base em acidentes sofridos anteriormente e o quão comum esse acidente pode ocorrer, precavendo assim acidentes mais prováveis para aquele determinado usuário.

## **2.1 Requisitos Funcionais**

1. O sistema deve permitir que usuários se **cadastrem** e **realizem login** com segurança.
2. O sistema deve permitir que o usuário **registre novos incidentes** informando:

* Tipo do acidente (Ex.: Queda, Incêndio, Outro).
* Descrição do incidente.

1. O sistema deve gerar um **relatório organizado** dos incidentes registrados, agrupando-os por tipo e frequência.
2. O sistema deve calcular e exibir o **nível de risco pessoal** com base no histórico de incidentes:

* Baixo
* Moderado
* Alto

1. O sistema deve sugerir **dicas de prevenção personalizadas**, conforme o histórico de acidentes.
2. O sistema deve fornecer e permitir a atualização de um **checklist de preparação** para apagões.
3. O sistema deve **persistir com todas as informações** em um arquivo JSON, garantindo que os dados sejam mantidos entre sessões.

## **2.2 Requisitos Não Funcionais**

1. O sistema deve ser implementado exclusivamente em **C#**, utilizando a plataforma **.NET 8.0**.
2. A interface deve ser **inteiramente baseada em console**, com navegação clara e objetiva.
3. O sistema deve **validar todas as entradas** do usuário, impedindo a gravação de dados inválidos (Ex.: campos vazios, senhas curtas).
4. O sistema deve possuir **tratamento de exceções** robustas, utilizando try-catch para evitar falhas críticas.
5. A aplicação deve seguir princípios de **código limpo**, com:

* Nomes de variáveis e métodos significativos.
* Boas práticas de indentação.
* Comentários XML para documentação automática.

1. O sistema deve armazenar os dados de usuários, incidentes e checklist em um **arquivo JSON**, permitindo a **persistência de dados** de forma simples e eficaz.

# **FLUXOGRAMA DO SISTEMA**

# **TECNOLOGIAS UTILIZADAS**

No desenvolvimento do SPADA foram utilizadas as seguintes tecnologias:

| Tecnologia | Descrição |
| --- | --- |
| C# | Linguagem de programação utilizada no desenvolvimento |
| .NET 8.0 | Framework para compilação e execução do projeto. |
| JSON | Formato de persistência de dados (users.json). |
| Visual Studio 2022 | Ambiente de desenvolvimento integrado (IDE). |

# **REGRAS DE NEGÓCIO**

1. O login é obrigatório para utilizar as funcionalidades do sistema.
2. O nome de usuário deve ser único no cadastro.
3. A senha deve conter no mínimo 6 caracteres.
4. Um incidente só pode ser registrado com tipo e descrição válidos.
5. A avaliação de risco considera a quantidade total de incidentes:

* Baixo → até 2 incidentes.
* Moderado → entre 3 e 5 incidentes.
* Alto → 6 ou mais incidentes.

1. As dicas de prevenção são exibidas com base no tipo de incidentes registrados:

* Queda → Sugestão de iluminação e retirada de obstáculos.
* Incêndio → Sugestão de evitar velas próximas a materiais inflamáveis.

1. O checklist é um conjunto de tarefas predefinidas que o usuário pode marcar como concluídas ou pendentes.
2. Todas as alterações no sistema (registro, checklist) são imediatamente persistidas no arquivo JSON, evitando a perda de dados.

# **ESTRUTURA DE PASTAS**

O SPADA foi estruturado de uma forma simples, mas funcional,conforme detalhado a seguir.

SpadaApp/

├── Program.cs → Classe principal (fluxo de execução)

├── UserService.cs → Classe de gerenciamento de usuários e persistência

├── InputHelper.cs → Classe de apoio para validação e leitura de inputs

├── User.cs → Modelo de dados do usuário

├── Incident.cs → Modelo de dados de incidente

├── ChecklistItem.cs → Modelo de item de checklist

├── users.json → Arquivo de dados persistentes

└── SpadaApp.sln → Solução Visual Studio