DESENVOLVIMENTO DE APLICAÇÃO WEB PARA IDENTIFICAÇÃO COLABORATIVA DE ÁREAS DE RISCOS UTILIZANDO API'S DE GEOLOCALIZAÇÃO E PREVISÃO METEOROLÓGICA

DEVELOPMENT OF A WEB APPLICATION FOR COLLABORATIVE IDENTIFICATION OF RISK AREAS USING GEOLOCATION AND WEATHER FORECASTING APIS

Lucas Souza Guimarães
lsg1@discente.ifpe.edu.br

Felipe Rodrigues Barbosa
frb@discente.ifpe.edu.br

Matheus Pereira de Souza Costa
mpsc1@discente.ifpe.edu.br

Pedro Henrique dos Santos Silva
phss21@discente.ifpe.edu.br

Bruno Wagner Lemos de Souza
IFPE – brunnosouza@jaboatao.ifpe.edu.br
UPE – brunno.souza@upe.br

RESUMO

Este trabalho tem como objetivo desenvolver uma aplicação web colaborativa para identificação e alerta de áreas sujeitas a alagamentos, fornecendo informações em tempo real aos usuários. O crescimento urbano desordenado, aliado à intensificação de eventos climáticos extremos, tem ampliado significativamente a ocorrência de alagamentos em áreas urbanas, expondo a população a riscos socioambientais relevantes. A aplicação foi desenvolvida utilizando HTML, CSS e JavaScript, com integração de APIs externas de geolocalização (Mapbox) e previsão meteorológica (OpenWeather). Foram aplicados princípios da Engenharia de Software, modelagem UML, Design Thinking e o modelo evolucionário de desenvolvimento. Os resultados demonstram a viabilidade de um sistema web leve, responsivo e acessível, capaz de exibir informações climáticas e áreas de risco por meio de mapas interativos. Conclui-se que a aplicação pode auxiliar a população durante eventos de chuvas intensas, embora apresente limitações relacionadas à ausência de um back-end estruturado, indicando possibilidades de aprimoramento em trabalhos futuros.

Palavras-chave: Aplicação Web; Alagamentos; Geolocalização; Monitoramento Climático.

ABSTRACT

This work aims to develop a collaborative web application for identifying and alerting users to areas prone to flooding, providing real-time information. Unplanned urban growth, coupled with the intensification of extreme weather events, has significantly increased the occurrence of flooding in urban areas, exposing the population to relevant socio-environmental risks. The application was developed using HTML, CSS, and JavaScript, with the integration of external APIs for geolocation (Mapbox) and weather forecasting (OpenWeather). Principles of Software Engineering, UML modeling, Design Thinking, and the evolutionary development model were

applied. The results demonstrate the viability of a lightweight, responsive, and accessible web system capable of displaying climate information and risk areas through interactive maps. It is concluded that the application can assist the population during intense rainfall events, although it presents limitations related to the absence of a structured back-end, indicating possibilities for improvement in future work.

Keywords: Web Application; Flooding; Geolocation; Climate Monitoring.

1 INTRODUÇÃO

Nas estações chuvosas do ano ou em localizações em que inundações e intempéries climáticas são comuns de ocorrer as pessoas ficam suscetíveis a sofrerem com alguns problemas e riscos aos bens materiais e mais grave ainda a sua própria integridade física. A partir dessa visão de problema idealizou-se alguma aplicação que pudesse sanar ou reduzir os riscos para os pedestres inicialmente que tinham necessidades de transitarem pelas ruas e estradas durante chuvas, alagamentos e inundações.

Os órgãos responsáveis por cuidar da segurança pública em relação aos desastres naturais possuem muitos desafios em relação ao mapeamento e à obtenção de informação em tempo real de desastres naturais e suas situações de riscos. Como meio de solucionar ou diminuir este déficit este projeto utiliza-se de um sistema colaborativo de informações do usuário juntamente com validações de informações através de API 's de terceiros que fornecem dados meteorológicos e de um sistema que avalia a quantidade de reporte feita pelos usuários.

Neste projeto o objetivo principal é a divulgação de informações sobre locais com riscos e perigos de alagamentos ou choques elétricos pela própria comunidade auxiliando a locomoção dos usuários por rotas seguras tendo como cidade modelo Recife, tendo em vista seus inúmeros casos de inundações, afogamentos, choques elétricos em ruas alagadas e acidentes com ‘bocas de lobo’ abertas e submersas pela água.

Diante desse cenário, surge a seguinte questão de pesquisa: **como uma aplicação web colaborativa, baseada em APIs de geolocalização e dados meteorológicos, pode auxiliar na identificação de áreas urbanas sujeitas a alagamentos e na redução de riscos à população?**

Assim, o objetivo geral deste trabalho é desenvolver uma aplicação web colaborativa para identificação e alerta de áreas de risco de alagamentos, utilizando dados de geolocalização e previsão meteorológica. A relevância do estudo está associada tanto à contribuição acadêmica, ao explorar a integração de APIs em sistemas web, quanto ao impacto social potencial, ao

oferecer informações que podem auxiliar a população e gestores públicos na prevenção de acidentes urbanos.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Engenharia de Software e seu papel no desenvolvimento de sistemas

A Engenharia de Software é a área responsável por estudar, projetar, desenvolver e manter sistemas computacionais de maneira sistemática e controlada.

De acordo com Pressman (2016), a Engenharia de Software busca aplicar princípios, métodos e boas práticas com o objetivo de melhorar a qualidade do produto final, reduzir custos e minimizar falhas ao longo do ciclo de vida do sistema.

Nesse sentido, a Engenharia de Software atua como base para a construção de sistemas confiáveis e eficientes, garantindo que o software atenda às necessidades do usuário, mantenha boa manutenibilidade e seja escalável. Sommerville (2019), reforça que essa disciplina fornece uma estrutura organizada para gerenciar o desenvolvimento, permitindo que equipes executem atividades de planejamento, modelagem, implementação e testes de forma integrada.

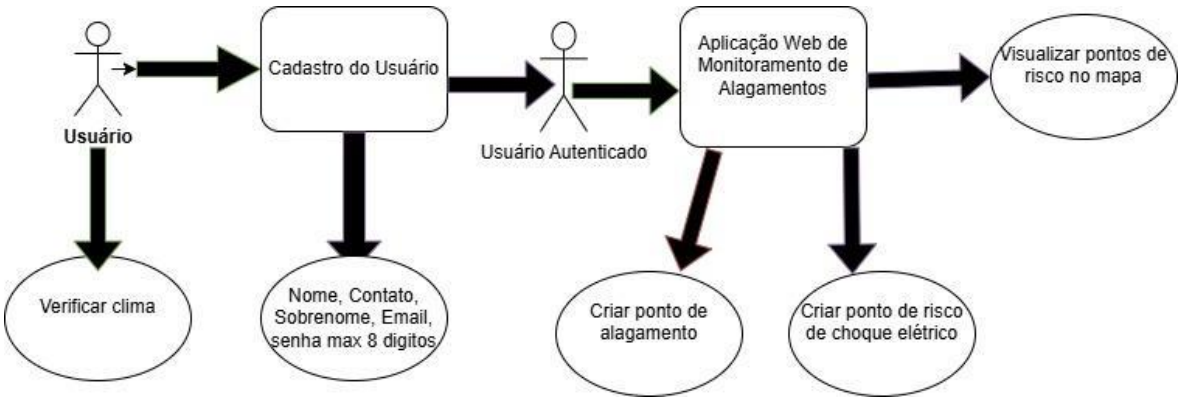
No contexto deste projeto, os princípios da engenharia foram fundamentais para orientar as etapas de análise, modelagem e implementação, incluindo a integração de APIs externas, que ampliam a capacidade do sistema e permitem o consumo de dados em tempo real.

2.2 Modelagem UML

A UML (Unified Modeling Language) é uma linguagem padronizada para especificar, visualizar e documentar sistemas de software. Booch, Rumbaugh e Jacobson (2005), criadores da UML, afirmam que a modelagem auxilia na compreensão do problema e permite representar a estrutura e o comportamento do sistema antes da codificação.

2.2.1 Diagrama de Casos de Uso

Figura 1



Fonte: Próprio Autor

A Figura 1 apresenta o diagrama de casos de uso da aplicação web colaborativa, evidenciando as funcionalidades disponíveis aos usuários, bem como as restrições de acesso impostas pelo sistema. Observa-se que a criação de eventos de alagamento e de risco de choque elétrico é permitida apenas a usuários previamente cadastrados e autenticados por meio de login e senha, enquanto a consulta ao clima permanece acessível a todos os usuários.

Segundo Sommerville (2019), casos de uso ajudam a identificar as interações entre os usuários e o sistema, descrevendo funcionalidades e objetivos principais do software. No presente projeto, foi modelado o fluxo de acesso às APIs externas, permitindo representar a interação do usuário com serviços de mapa, meteorologia e dados fornecidos pela API Layer.

2.2.2 Diagrama de Classes

O diagrama de classes representa a estrutura estática do sistema. Para Pressman (2016), esse diagrama permite visualizar atributos, métodos e relacionamentos entre objetos. Neste projeto, classes específicas foram definidas para consumir, tratar e armazenar dados provenientes da API de mapa, da API de meteorologia e da API Layer, garantindo modularidade e reuso de código.

2.2.3 Diagrama de Sequência

Conforme Booch, Rumbaugh e Jacobson (2005), o diagrama de sequência mostra a ordem temporal das interações entre objetos durante a execução de uma funcionalidade. Ele foi essencial para validar o comportamento esperado das requisições às APIs, descrevendo o envio da solicitação, o processamento e o retorno dos dados.

2.3 Arquitetura Web e Padrão MVC

A arquitetura web organiza a aplicação em camadas responsáveis por processar requisições, armazenar dados e entregar conteúdo ao usuário. De acordo com Sommerville (2019), sistemas Web devem seguir princípios de modularidade e separação de responsabilidades para garantir escalabilidade e fácil manutenção. O padrão MVC (Model-View-Controller) divide o software em três componentes principais (PRESSMAN, 2016): Model: gerencia dados e lógica de negócio; View: exibe informações; Controller: realiza a ponte entre usuário e sistema. Neste projeto, o padrão MVC foi utilizado para integrar APIs externas de forma organizada. O Controller é responsável por solicitar os dados às APIs; o Model armazena, válida e processa as informações; e a View exibe os resultados ao usuário de maneira dinâmica. O sistema foi estruturado em três camadas principais:

- Camada de apresentação (front-end): composta por HTML, CSS e elementos de interface.
- Camada lógica: implementada em JavaScript, responsável pelos fluxos internos, rotas e manipulação do mapa.
- Camada de integração: utilizada para comunicação com serviços externos, como APIs de mapa e clima.
- A organização dos arquivos segue o padrão apresentado no Quadro abaixo:
- Estrutura de diretórios da aplicação
- /index.html – Interface principal
- /style.css – Estilos e responsividade
- /script.js – Lógica e rotas
- /LICENSE – Chaves das APIs

2.4 Tecnologias Utilizadas: HTML, CSS e JavaScript

2.4.1 HTML

O HTML (Hypertext Markup Language) estrutura o conteúdo exibido no navegador (SILVA, 2015). Os elementos foram utilizados para construir a interface que apresenta os dados consumidos pelas APIs externas, para a construção semântica da interface, contemplando a definição do layout principal, menus, contêineres do mapa, formulários de cadastro e login, além de elementos destinados à exibição de alertas e dados climáticos.

2.4.2 CSS

O CSS controla a aparência visual das páginas, permitindo padronização e responsividade (MEYER, 2018). Ele define a aparência visual da aplicação, garantindo responsividade para diferentes tamanhos de tela. Inclui a definição de paletas de cores, estilização de modais, botões e painéis, além da aplicação de regras de acessibilidade visual.

2.4.3 JavaScript

O JavaScript adiciona dinamismo às páginas (FLANAGAN, 2020). Nesta aplicação, ele foi a tecnologia responsável pela comunicação assíncrona com as APIs, utilizando requisições HTTP para obter dados de mapas, clima e outros serviços externos. O JavaScript é responsável pelo comportamento dinâmico da interface, permitindo a manipulação do DOM, abertura e fechamento de modais, atualizações em tempo real da barra climática e validações iniciais dos formulários de autenticação.

2.5 Integração de APIs no Sistema

As APIs (Application Programming Interfaces) desempenham papel crucial no desenvolvimento de aplicações modernas, pois permitem integrar recursos externos sem a necessidade de manter grandes bancos de dados ou lógica complexa localmente.

2.5.1 API de Mapa

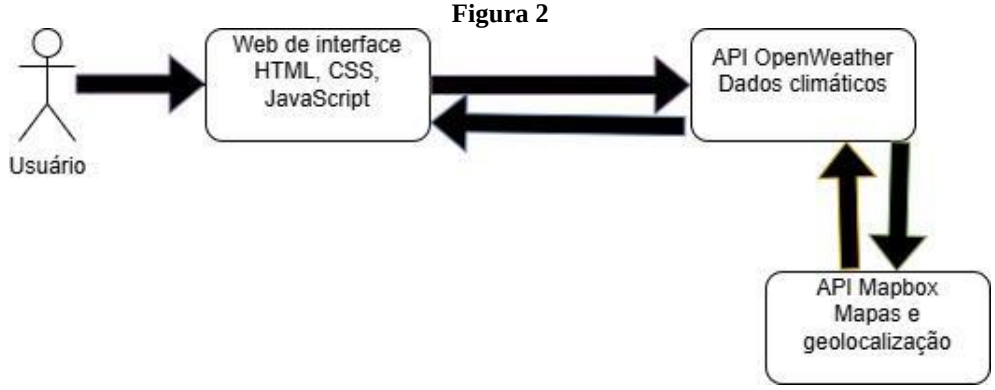
A integração de uma API de mapas possibilita exibir localizações geográficas, rotas e coordenadas. De acordo com Richardson e Ruby (2007), APIs REST permitem consumir dados através de requisições padronizadas, tornando a integração mais simples e eficiente. No projeto, a API de mapa é responsável pela renderização do mapa, pela personalização visual, pela manipulação de marcadores e pela integração com dados geógrafos. Também fornece recursos de geocodificação e navegação.

2.5.2 API de Meteorologia

A API de meteorologia fornece dados climáticos atualizados, como temperatura, umidade e condições atmosféricas. Essa integração possibilita consultas em tempo real,

aumentando a utilidade e relevância do sistema, atualizando a interface e apoiando a tomada de decisão do usuário. O uso desse tipo de API segue o princípio de reutilização de serviços externos, defendido por Sommerville (2019) ao tratar de arquiteturas orientadas a serviços.

Figura 2



Fonte: Próprio Autor

2.5.3 API Layer

A API Layer é um serviço especializado em fornecer dados estruturados sobre diferentes domínios, como moedas, localização, validação de informações e estatísticas. Por operar seguindo padrões REST, ela facilita a obtenção e manipulação dos dados diretamente no backend ou frontend da aplicação. Sua utilização no projeto permitiu complementar informações do sistema com dados confiáveis disponibilizados por provedores externos.

2.6 Alagamentos urbanos e identificação de áreas de risco

Os alagamentos urbanos configuram-se como um dos principais problemas socioambientais em cidades brasileiras de grande porte. Estudos apontam que a combinação entre chuvas intensas, impermeabilização do solo e sistemas de drenagem insuficientes resulta em inundações recorrentes, especialmente em áreas densamente urbanizadas (ANJOS et al., 2024).

Pesquisas baseadas em geoprocessamento indicam que regiões de baixa altitude apresentam maior susceptibilidade a alagamentos, sendo o mapeamento dessas áreas fundamental para ações preventivas e para o planejamento urbano (LIMA et al., 2025). Modelagens hidrológicas aplicadas ao contexto do Recife demonstram que a infraestrutura existente é insuficiente para suportar eventos extremos de precipitação, reforçando a necessidade de soluções complementares baseadas em tecnologia da informação (SILVA; SOARES; HOLANDA, 2023).

3 METODOLOGIA

3.1 Design Thinking: How Might We HMW

A técnica How Might We (HMW) é amplamente utilizada no Design Thinking como forma de transformar problemas gerais em oportunidades de inovação. Segundo Brown (2009), perguntas do tipo “Como poderíamos…” auxiliam no processo de explorar soluções criativas sem restringir o escopo do projeto. No contexto do desenvolvimento de software, essa abordagem contribui para alinhar a identificação do problema às necessidades reais do usuário, favorecendo processos iterativos e centrados no usuário.

Aplicando o método HMW ao presente projeto, foram formuladas questões que orientaram o desenvolvimento do sistema, tais como:

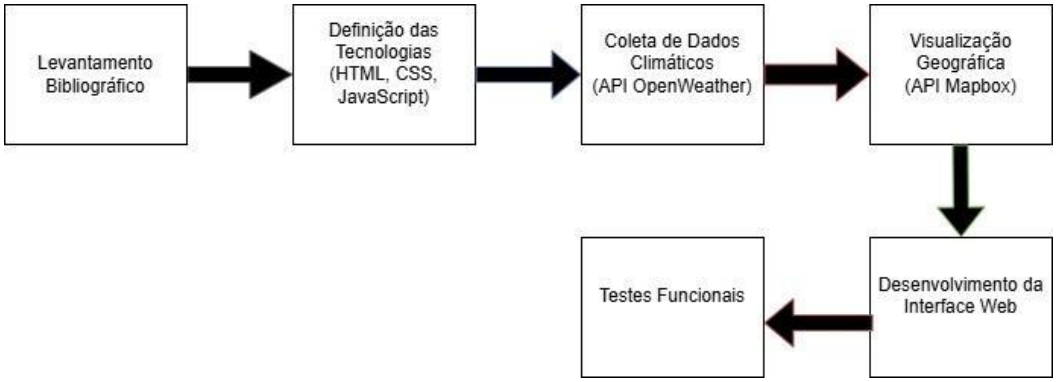
- Como poderíamos integrar APIs externas de forma que o usuário obtenha informações confiáveis em tempo real?
- Como poderíamos facilitar a visualização de mapas e dados meteorológicos dentro da aplicação?
- Como poderíamos garantir que o sistema permaneça simples, rápido e acessível mesmo consumindo múltiplas APIs?

Essas perguntas nortearam as etapas de modelagem, arquitetura e implementação, alinhando-se ao modelo evolucionário adotado e contribuindo para decisões mais claras e direcionadas.

3.2 Modelo Evolucionário

O modelo evolucionário é uma abordagem iterativa de desenvolvimento que permite criar versões incrementais do sistema. Segundo Sommerville (2019), essa abordagem favorece adaptações contínuas conforme o feedback do usuário. Pressman (2016) afirma que o modelo reduz riscos, especialmente quando há integração com APIs externas, pois testes rápidos podem revelar incompatibilidades e permitir ajustes precoces. No presente projeto, cada ciclo evolutivo permitiu aprimorar a integração com a API de mapa, a API de meteorologia e a API Layer, resultando em um sistema mais robusto, funcional e alinhado aos requisitos iniciais.

Figura 3



Fonte: Próprio Autor

De acordo com a Figura 3, o processo metodológico adotado neste estudo foi organizado em etapas sequenciais, iniciando-se pelo levantamento bibliográfico, seguido da definição das tecnologias, integração das APIs de previsão meteorológica e geolocalização, desenvolvimento da interface web e, por fim, a realização de testes funcionais da aplicação.

4 RESULTADOS E ANÁLISE

Os resultados obtidos no desenvolvimento da aplicação web, estão descritos em seguida através das funcionalidades implementadas, telas finais do sistema, a comparação entre o que foi planejado e o que foi efetivamente entregue, bem como os impactos positivos, limitações identificadas e possíveis melhorias futuras. O sistema desenvolvido, foi construído com as tecnologias HTML, CSS e JavaScript, aliado às APIs Mapbox e OpenWeather, contempla um conjunto de funcionalidades que garantem a operação básica de um sistema de alerta, com foco na visualização de áreas afetadas e na atualização contínua de dados climáticos. Dentre essas funcionalidades, destacam-se:

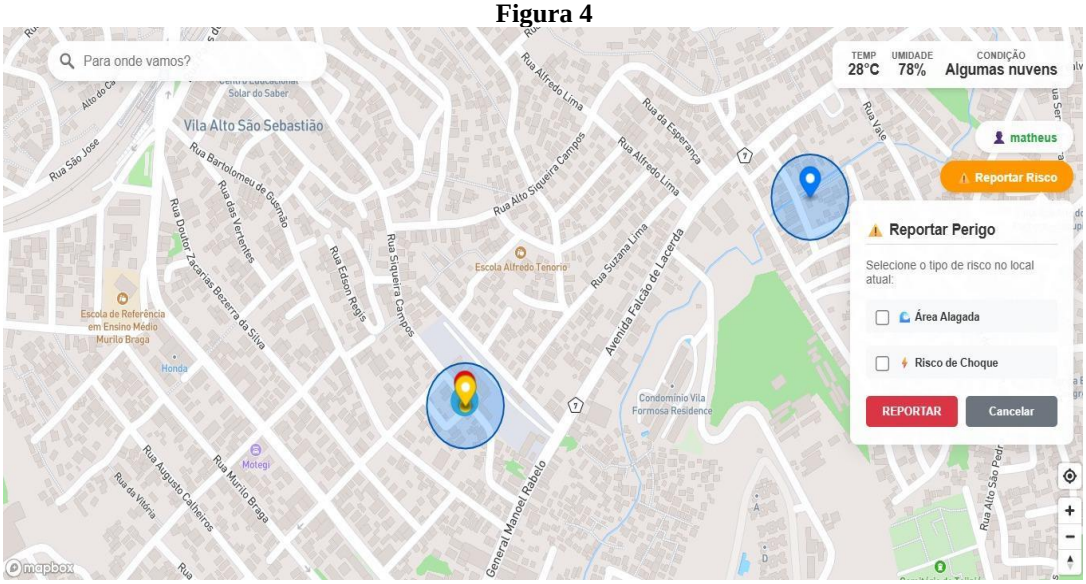
- Tela inicial responsiva, adaptada a diferentes resoluções de tela, garantindo uso adequado em dispositivos móveis e computadores;
- Barra de informações climáticas em tempo real, obtidas por meio da API OpenWeather, exibindo dados como temperatura, condições atmosféricas e localização aproximada;
- Modal inicial de alerta, apresentado ao usuário ao carregar a aplicação, contendo orientações e avisos de segurança;
- Sistema de cadastro e login, utilizando o recurso de LocalStorage para armazenar dados de forma local no navegador;
- Mapa interativo estilizado, fornecido pela API Mapbox, com destaque para marcações de áreas alagadas e pontos críticos;

- Notificações visuais sobre regiões de risco, exibidas diretamente no mapa;
- Interface de uso intuitivo, favorecendo a navegação e a interpretação das informações pelo usuário.
- O conjunto das telas foi desenvolvido conforme princípios de usabilidade, com foco na clareza visual e na organização dos elementos informativos. resultando assim nas seguintes telas principais:
- Tela inicial responsiva, contendo cabeçalho, informações climáticas e acesso ao mapa;
- Modal de alerta e autenticação, utilizado para exibição de mensagens iniciais, bem como formulários de cadastro e login;
- Mapa interativo, que constitui a área central do sistema, com camadas customizadas e marcações de alerta;
- Painel de informações adicionais, onde são exibidos dados complementares relativos à situação climática e às áreas de risco.

As principais vantagens são a facilidade de implantação, o carregamento ágil das interfaces e a redução de pontos de falha, aspectos que favorecem seu uso em contextos de demonstração e em ambientes acadêmicos. Mesmo com escopo reduzido, a aplicação cumpre o objetivo de fornecer informações sobre áreas de risco e condições climáticas, demonstrando potencial para apoiar decisões durante eventos de chuva intensa, reduzir acidentes em locais vulneráveis, ampliar o acesso a dados meteorológicos atualizados e promover maior conscientização sobre segurança urbana. A proposta apresentada está alinhada a estudos que evidenciam a necessidade de sistemas de alerta acessíveis para mitigar riscos associados a eventos extremos de precipitação, especialmente em cidades como o Recife, onde a recorrência de alagamentos expõe a população a situações de perigo (ANJOS et al., 2024).

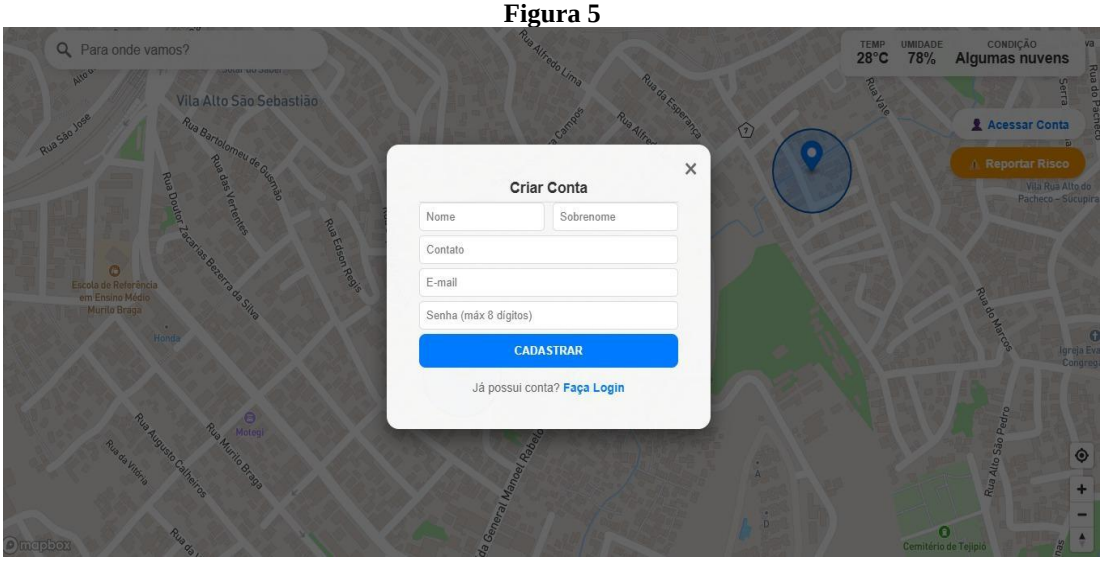
Apesar desses resultados, o sistema apresenta limitações decorrentes da ausência de back-end, como a falta de persistência de dados, a impossibilidade de compartilhamento remoto de informações, restrições de segurança, dependência do armazenamento local e inexistência de integração com bases oficiais. Para aprimorar sua robustez e ampliar sua aplicabilidade, são sugeridas melhorias como a implementação de um servidor back-end, o desenvolvimento de uma API própria, a adoção de mecanismos de autenticação segura, a criação de um módulo administrativo, a integração com serviços oficiais de monitoramento, a utilização de notificações em tempo real, o suporte a aplicações móveis e o emprego de técnicas de aprendizado de máquina.

Figura 4



Fonte: Próprio Autor

Figura 5



Fonte: Próprio Autor

Conforme apresentado nas Figuras 4 e 5, a interface da aplicação web exibe o mapa interativo com áreas de risco, informações climáticas em tempo real e os recursos de autenticação necessários para o cadastro e registro colaborativo de eventos de alagamento e risco de choque elétrico.

5 CONSIDERAÇÕES FINAIS/ CONCLUSÕES

O desenvolvimento da aplicação web para alerta de áreas alagadas permitiu demonstrar a viabilidade de uma solução simples, acessível e funcional para apoiar a população em situações de risco relacionadas a eventos climáticos. A proposta alcançou seu objetivo central ao integrar informações fornecidas por APIs externas e recursos de geolocalização, oferecendo

ao usuário um conjunto de dados atualizados que favorecem escolhas mais seguras durante deslocamentos urbanos. A adoção de tecnologias front-end e a organização da arquitetura em camadas contribuíram para a construção de uma aplicação leve, de fácil implantação e adequada a ambientes acadêmicos. O uso de princípios da Engenharia de Software e de métodos como o Design Thinking e o modelo evolucionário orientou o processo de elaboração, permitindo avanços contínuos e ajustes progressivos ao longo do desenvolvimento.

Embora tenha atendido aos requisitos essenciais, o sistema apresenta limitações significativas decorrentes da ausência de um back-end estruturado, como a impossibilidade de persistência de dados, a inexistência de mecanismos robustos de autenticação e a dificuldade de compartilhamento remoto de informações. Tais restrições não comprometem a funcionalidade básica do protótipo, mas indicam caminhos claros para versões futuras. Diante disso, recomenda-se a expansão da aplicação por meio da implementação de uma API própria, adoção de práticas de segurança adequadas, integração com bancos de dados oficiais e utilização de técnicas de análise preditiva. Essas melhorias têm potencial para ampliar o impacto social da solução, elevando sua confiabilidade, escalabilidade e capacidade de auxiliar políticas públicas relacionadas à prevenção de desastres.

Assim, o projeto contribui tanto para o campo acadêmico, ao demonstrar a aplicação prática de conceitos de engenharia e design de software, quanto para a sociedade, ao propor um recurso tecnológico voltado à redução de riscos e à promoção de segurança em ambientes urbanos sujeitos a alagamentos.

REFERÊNCIAS

ANJOS, L. S. et al. Resgate histórico dos eventos extremos de precipitação e seus impactos no município do Recife-PE. **Revista Brasileira de Climatologia**, v. 34, p. 335–359, 2024.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR 14724**: trabalhos acadêmicos. Rio de Janeiro: 2011.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **UML: Guia do Usuário**. 2. ed. Rio de Janeiro: Elsevier, 2005.

BROWN, T. **Change by Design**. New York: Harper Business, 2009.

FLANAGAN, D. **JavaScript:** O Guia Definitivo. 7. ed. Porto Alegre: Bookman, 2020.

GUEDES, R. P.; ARAÚJO, M. P. S.; ANDRADE, A. P. G. **Necessidade do gerenciamento dos recursos hídricos em grandes cidades como Recife**. Architecton, 2023.

LIMA, L. D. B. et al. **Mapeamento de áreas suscetíveis a inundações na cidade do Recife-PE**. Espaço em Revista, 2025.

MAPBOX. **Mapbox GL JS Documentation**. Disponível em: https://docs.mapbox.com. Acesso em: 26 de setembro de 2025.

MEYER, E. **CSS: The Definitive Guide**. 4. ed. Sebastopol: O'Reilly, 2018.

OPENWEATHER. **Weather API Documentation.** Disponível em: https://openweathermap.org/api. Acesso em: 05 de outubro de 2025.

PRESSMAN, R. S. **Engenharia de Software: Uma Abordagem Profissional**. 8. ed. Porto Alegre: AMGH, 2016.

RICHARDSON, L.; RUBY, S. **RESTful Web Services**. Sebastopol: O'Reilly, 2007.

SILVA, M. A. V.; SOARES, W. A.; HOLANDA, M. A. C. R. **Application of the SWMM model for the analysis of urban flooding in the city of Recife – PE**. Fórum Ambiental da Alta Paulista, 2023.

SILVA, S. **HTML5 e CSS3: Guia Prático**. São Paulo: Novatec, 2015.

SOMMERVILLE, I. **Engenharia de Software**. 10. ed. São Paulo: Pearson, 2019.