PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA POLITÉCNICA E DE ARTES
GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

SOLIDARIZE: APLICAÇÃO WEB PARA BUSCA DE ORGANIZAÇÕES NÃO-GOVERNAMENTAIS

GABRIEL TORRES BAHIA

GOIÂNIA
2023

GABRIEL TORRES BAHIA

SOLIDARIZE: APLICAÇÃO WEB PARA BUSCA DE ORGANIZAÇÕES NÃO-GOVERNAMENTAIS

Trabalho de Conclusão de Curso apresentado à Escola Politécnica e de Artes, da Pontifícia Universidade Católica de Goiás, como parte dos requisitos para a obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Me. Fernando Gonçalves Abadia

GOIÂNIA

2023

GABRIEL TORRES BAHIA

SOLIDARIZE: APLICAÇÃO WEB PARA BUSCA DE ORGANIZAÇÕES NÃO-GOVERNAMENTAIS

Trabalho de Conclusão de Curso aprovado em sua forma final pela Escola Politécnica e de Artes, da Pontifícia Universidade Católica de Goiás, para obtenção do título de Bacharel em Ciência da Computação, em _____/_____/_____.

_____

Orientador: Prof. Me. Fernando Gonçalves Abadia

_____

Prof. Me. Rafael Leal Martins

_____

Prof. Me. Anibal Santos Jukemura

GOIÂNIA

2023

RESUMO

Este trabalho de pesquisa foca no desenvolvimento de um protótipo de aplicação web, projetado para facilitar a busca por Organizações Não-Governamentais (ONGs). O projeto é moldado pelas dificuldades comumente enfrentadas pelos doadores na localização de ONGs apropriadas, e pelos desafios dos gestores em promover suas organizações. A fase de desenvolvimento envolveu um estudo bibliográfico sobre a pilha tecnológica, levantamento de requisitos e planejamento da arquitetura do sistema. Antes da implementação, foram criados protótipos das interfaces de usuário. O desenvolvimento do protótipo foi realizado utilizando as tecnologias Vue.js, Node.js, MongoDB e Express. A fase final do projeto apresenta as telas desenvolvidas do protótipo, detalhando suas funcionalidades e aspectos específicos da implementação, culminando com as conclusões sobre o desenvolvimento do projeto.

Palavras-Chave: Desenvolvimento Web; Vue.js; Node.js; ONG

ABSTRACT

This research work focuses on the development of a web application prototype, designed to facilitate the search for Non-Governmental Organizations (NGOs). The project is shaped by the difficulties commonly faced by donors in finding appropriate NGOs, and by the challenges of managers in promoting their organizations. The development phase involved a bibliographic study on the technology stack, requirements gathering, and system architecture planning. Before implementation, user interface prototypes were created. The prototype development was carried out using Vue.js, Node.js, MongoDB, and Express technologies. The final phase of the project presents the developed prototype screens, detailing their functionalities and specific aspects of the implementation, culminating in the conclusions about the project development.

Keywords: Web Development; Vue.js; Node.js; NGO.

LISTA DE FIGURAS

Figura 1 - Painel de elementos do Balsamiq Cloud 20
Figura 2 - Exemplo de requisição no Postman 21
Figura 3 - Visualização de Collections e Documents no MongoDB Atlas 31
Figura 4 - Representação da comunicação entre partes do sistema 35
Figura 5 - Protótipo da tela de Início 36
Figura 6 - Protótipo da Tela de Cadastro 37
Figura 7 - Protótipo da Tela de Sucesso 37
Figura 8 - Protótipo da Tela de Cadastro da ONG 38
Figura 9 - Protótipo da Tela de Login 39
Figura 10 - Protótipo da Tela de Busca de ONG 40
Figura 11 - Protótipo da Tela de Busca de ONG com listagem 40
Figura 12 - Protótipo Modal de Perfil da ONG 41
Figura 13 - Protótipo da Tela de Dados da ONG 42
Figura 14 - Protótipo de Tela de Edição da ONG 43
Figura 15 - Protótipo de Tela de Página Não Encontrada 44
Figura 16 - Tela de Início 46
Figura 17 - Tela de Cadastro de Usuário 47
Figura 18 - Tela de Sucesso 48
Figura 19 - Tela de Cadastro da ONG 50
Figura 20 - Tela de Login 52
Figura 21 - Tela de Busca de ONG 53
Figura 22 - Listagem de ONGs e paginação 54
Figura 23 - Alerta padrão do SweetAlert2 55
Figura 24 - Modal Perfil da ONG 56
Figura 25 - Imagem de perfil padrão 57
Figura 26 - Tela de Dados da ONG 58
Figura 27 - Barra de navegação do Dashboard 59
Figura 28 - Tela de Edição da ONG 60

LISTA DE SIGLAS

ONG	Organizações Não-Governamentais
NGO	*Non-Governmental Organizations*
ABONG	Associação Nacional Brasileira de ONGs
API	*Application Programming Interface*
SPA	*Single Page Applications*
MVVM	*Model-View-ViewModel*
HTTP	*Hypertext Transfer Protocol*
JSON	*JavaScript Object Notation*
JWT	JSON *Web Tokens*
UI	*User Interface*
URL	*Uniform Resource Locator*

SUMÁRIO

1. INTRODUÇÃO **10**
1.1 Objetivo Geral 11
1.2 Objetivos Específicos 11
1.3 Justificativa 12
1.4 Resultado esperados 12
1.5 Organização do trabalho 13

2. FUNDAMENTAÇÃO TEÓRICA **14**
2.1 Vue.js 14
2.2 MongoDB 15
2.3 Node.js 16
2.4 Express 16
2.5 JWT 17
2.6 Bootstrap 17
2.7 Jest 18

3. MATERIAIS E PROCEDIMENTOS METODOLÓGICOS **19**
3.1 Métodos 19
3.2 Materiais 19
3.2.1 Visual Studio Code *19*
3.2.2 Balsamiq Cloud *20*
3.2.3 Postman *20*
3.2.4 MongoDB Atlas 21

4. DESENVOLVIMENTO **23**
4.1 Descrição Geral do Produto 23
4.2 Organização das Tecnologias 24
4.3 Descrição de Usuários 25
4.4 Requisitos Funcionais 25
4.5 Requisitos Não Funcionais 27
4.6 Modelo Conceitual 29
4.6.1 Entidades e atributos *29*
4.6.2 Modelo de Dados do MongoDB *30*
4.7 Documentação da API 32
4.8 Arquitetura 34
4.9 Protótipos das Telas 35

5. RESULTADOS ESPERADOS **45**
5.1 Página de Início 45

5.2 Página de Cadastro de Usuário 46
5.3 Página de Sucesso 47
5.4 Página de Cadastro de ONG 48
5.5 Dashboard 52
5.5.1 Página Buscar ONG 53
5.5.2 Modal Perfil da ONG 55
5.5.3 Página de Dados da ONG 57
5.5.4 Página de Editar Dados 59

6. CONSIDERAÇÕES FINAIS 61

REFERÊNCIAS 62

1. INTRODUÇÃO

O termo ONG (Organização Não-Governamental), segundo Landim (1993) e Gohn (2000), foi criado na década de 1940, pela Organização das Nações Unidas - ONU para descrever organizações não-governamentais que recebiam recursos financeiros de entidades públicas, com o propósito de implementar projetos de cunho social, seguindo uma abordagem conhecida como desenvolvimento de comunidade.

Segundo a Associação Nacional Brasileira de ONGs (ABONG) são classificadas como ONGs “as entidades que, juridicamente constituídas sob a forma de fundação, associação e sociedade civil, todas sem fins lucrativos, notadamente autônomas e pluralistas, tenham compromisso com a construção de uma sociedade democrática, participativa e com o fortalecimento dos movimentos sociais de caráter democrático, condições estas, atestadas pelas suas trajetórias institucionais e pelos termos dos seus estatutos”.

No Brasil, as organizações não governamentais possuem uma trajetória mais atual. Durante os períodos dos anos 1960 e 1970, começaram a emergir entidades de ajuda e caridade que não se identificavam como ONGs. Contudo, de acordo com as observações de Landim (1993), é importante ressaltar que muitas dessas instituições passaram a adotar o termo "organização não-governamental" a partir da década seguinte.

As organizações não governamentais atuam em diversos setores sociais, priorizando especialmente as populações que possuem menor acesso aos serviços públicos. Seu objetivo é encontrar soluções para questões sociais, culturais e suas consequências, abrangendo áreas como educação, saúde, segurança, moradia, direitos humanos, entre outras. As ONGs apresentam características distintas entre si e desempenham múltiplos papéis na sociedade. Sua principal característica é a defesa dos direitos dos cidadãos e a promoção do desenvolvimento, trabalhando para melhorar e transformar a vida das pessoas menos privilegiadas. Elas operam de maneira formal, autônoma e sem fins lucrativos, conforme mencionado por Álvares (1999) e Landim (2002a), citados por Alvim e Teodósio (2004).

O trabalho em questão tem como foco a elaboração de uma proposta para criar um protótipo de uma aplicação web com finalidade social, nomeado de Solidarize, que visa facilitar a busca de ONGs, beneficiando tanto aqueles que

desejam fazer doações como os gestores dessas organizações. De acordo com as observações de Moresi (2017), os aplicativos sociais são considerados como ferramentas da tecnologia social, buscando oferecer uma alternativa ao desenvolvimento tecnológico voltado para a produção, consumo e vigilância do indivíduo.

A referida aplicação foi concebida com o intuito de oferecer uma solução prática e conveniente tanto para os doadores quanto para os gestores das ONGs, levando em consideração as limitações e dificuldades enfrentadas pelas ONGs. Por meio do *website* Solidarize, os doadores terão a facilidade de encontrar e contatar diretamente as organizações de caridade, proporcionando-lhes uma experiência simplificada e cômoda. Da mesma forma, os gestores das ONGs serão beneficiados com uma ferramenta eficiente que otimiza os processos de gerenciamento e comunicação, permitindo-lhes direcionar seus esforços de forma mais eficaz e maximizar o impacto positivo de suas atividades.

Diante do contexto, este projeto visa responder a questão de pesquisa: Como uma aplicação web pode estimular os cidadãos a realizarem doações?

1.1 Objetivo Geral

O objetivo geral deste trabalho é desenvolver uma aplicação web que constitua um acervo de ONGs, permitindo aos usuários selecionar o tipo de doação desejada e exibindo as ONGs que recebem o(s) tipo(s) de doação(es) selecionado(s).

1.2 Objetivos Específicos

Os objetivos específicos deste trabalho são:

1. Apresentar uma solução para facilitar a busca de ONGs.
2. Expor linguagens e ferramentas que compõem o desenvolvimento web.
3. Desenvolver uma aplicação utilizando as linguagens e ferramentas apresentadas, mostrando as etapas do desenvolvimento.
4. Gerar bibliografia sobre desenvolvimento de *websites* para agregar na causa das ONGs.

1.3 Justificativa

A escolha do assunto abordado neste trabalho se justifica pela importância das organizações não governamentais (ONGs) no contexto socioeconômico atual e pela necessidade de promover a visibilidade e interação entre as ONGs e a sociedade, especialmente em meio ao cenário pós-pandemia da COVID-19. De acordo com Wu & Zhu (2020), durante a pandemia, incentivar a participação pública em doações para instituições de caridade tornou-se uma questão social urgente. Essa demanda ressalta a importância do desenvolvimento de ferramentas que facilitem e auxiliem o processo de doação.

Conforme mencionado por Martone e Gil (2006), a captação e gestão de recursos são considerados, de forma unânime, os maiores desafios enfrentados pelos gestores das ONGs. Mesmo aquelas organizações que possuem vínculos com o setor público enfrentam dificuldades em obter recursos suficientes para manter suas operações em pleno funcionamento e para atender aos projetos futuros. Diante desse contexto, este trabalho se revela relevante ao apresentar uma proposta de desenvolvimento de um protótipo de uma aplicação web que visa facilitar o processo de aproximação da sociedade ao Terceiro Setor.

1.4 Resultado esperados

Do ponto de vista acadêmico, este trabalho contribui ao gerar uma bibliografia por meio da demonstração prática do desenvolvimento de uma aplicação web que evidencia de forma concreta e palpável como a tecnologia pode ser empregada para simplificar e agilizar a busca por instituições beneficentes. A bibliografia gerada pode ser um recurso valioso para pesquisadores e profissionais interessados em explorar o potencial das tecnologias móveis no contexto do Terceiro Setor.

Além disso, busca estimular o engajamento ativo dos cidadãos e fomentar uma cultura de doação solidária em nível local, fortalecendo assim os laços de solidariedade dentro da comunidade.

1.5 Organização do trabalho

Este trabalho está organizado em cinco capítulos. O primeiro capítulo, que é o atual, faz uma contextualização geral sobre as ONGs e esboça a ideia inicial do desenvolvimento da aplicação.

No segundo capítulo, será apresentada a fundamentação teórica, definindo os conceitos teóricos que embasam a proposta deste trabalho de conclusão de curso.

O terceiro capítulo, dedicado a materiais e métodos, detalha os meios e materiais empregados para alcançar os objetivos deste trabalho.

No quarto capítulo, o foco constitui no desenvolvimento: será descrito o produto, a organização das tecnologias, os usuários do sistema, os requisitos funcionais e não funcionais, o modelo conceitual, a documentação da *Application Programming Interface* (API), traduzido como Interface de Programação de Aplicativos, a arquitetura do sistema e a criação dos protótipos de tela.

Por fim, no quinto capítulo, serão apresentadas as considerações finais, refletindo sobre como o trabalho atendeu aos objetivos gerais e específicos e propondo possíveis direções para trabalhos subsequentes.

2. FUNDAMENTAÇÃO TEÓRICA

Este capítulo tem como objetivo introduzir os conceitos fundamentais das tecnologias utilizadas neste estudo, fornecendo uma base teórica essencial para entender as definições e aplicações específicas dessas ferramentas no desenvolvimento da aplicação, facilitando assim a compreensão de como cada componente contribui para a arquitetura e o funcionamento geral da aplicação.

2.1 Vue.js

Vue.js é um *framework* JavaScript utilizado para construir interfaces de usuário e aplicações de página única (*Single Page Applications* - SPAs). Sua criação foi inspirada na necessidade de uma ferramenta que fosse leve e adaptável, capaz de ser integrada em projetos de diferentes escalas, desde a adição de interatividade em páginas *web* simples até a base para o desenvolvimento de aplicações empresariais complexas (HANCHETT E LISTWON, 2018).

Com base no histórico dos *frameworks* de JavaScript e a evolução das aplicações web, Vue.js se destaca por adotar o padrão *Model-View-ViewModel* (MVVM), que facilita a manutenção do código. Segundo Hanchett e Listwon, Vue.js simplifica a implementação da vinculação entre a camada de visualização e os dados, promovendo uma atualização automática e coerente da interface do usuário à medida que o estado da aplicação se altera, oferecendo uma experiência de usuário responsiva.

De acordo com a documentação oficial do Vue.js, o framework utiliza um conceito chamado *databinding* que conecta os dados da aplicação às interfaces de usuário, permitindo que mudanças nos dados reflitam automaticamente na interface do usuário, sem necessidade de intervenção manual. Outro aspecto é a reatividade, que garante que quando os dados mudam, a parte da interface que depende desses dados é atualizada imediatamente.

Os componentes no Vue.js são blocos que encapsulam o HTML, CSS e JavaScript para uma parte da interface, tornando-os independentes e reutilizáveis. Além disso, o Vue.js oferece uma série de ganchos de ciclo de vida, permitindo que os desenvolvedores executem código em momentos específicos da instância do Vue, como na criação, atualização ou destruição de componentes. Para Hanchett e

Listwon, esta abordagem contribui para a gestão eficaz de recursos e uma melhor experiência no desenvolvimento.

Conforme sua documentação, o framework adota um modelo de estruturação chamado diretivas que permite adicionar comportamento ao HTML com simples prefixos, como "v-if" para renderizar um elemento condicionalmente, ou "v-for" para listar itens de uma coleção. Outra funcionalidade, o sistema de *event handling*, facilita a captura e o gerenciamento de ações do usuário, como cliques e entradas de teclado.

Por fim, outro ponto a se levar em consideração é o sistema de *plugins*. *Plugins* podem adicionar bibliotecas globais, adicionar métodos ou propriedades, ou inserir componentes que podem ser reutilizados em várias partes da aplicação. Isso torna o Vue.js muito extensível e personalizável.

2.2 MongoDB

MongoDB consiste em uma base de dados não relacional que possui uma arquitetura orientada a documentos. Essa estrutura facilita a representação de relações hierárquicas com registros únicos e elimina a necessidade de esquemas fixos, proporcionando aos desenvolvedores uma maior agilidade para evolução dos seus projetos (CHODOROW, 2013).

No MongoDB, as coleções funcionam como estruturas organizacionais para os dados, semelhantes às tabelas em um banco de dados SQL. Dentro destas coleções, os documentos representam os dados individuais, análogos aos registros em tabelas SQL. A principal diferença é a flexibilidade dos documentos, que, diferentemente dos registros em tabelas SQL, podem ter estruturas variadas com campos distintos uns dos outros. Isso permite um armazenamento de dados de maneira mais adaptável.

O projeto do MongoDB foca na escalabilidade horizontal, permitindo distribuir os dados por vários servidores, o que é mais viável economicamente e tecnicamente do que o escalonamento vertical. Isso simplifica o gerenciamento de grandes volumes de dados e, como Chodorow expõe, o sistema "cuida do balanceamento de dados e carga entre os clusters automaticamente".

Em relação aos recursos, o MongoDB oferece funcionalidades como índices secundários e agregações. No entanto, optou por não incluir operações de junção e

transações complexas para manter um desempenho elevado e escalabilidade. Chodorow articula que sua performance é uma de suas prioridades, o que reflete em sua capacidade de gerenciar grandes quantidades de dados de maneira eficiente, usando técnicas como o preenchimento dinâmico de documentos e a pré-alocação de arquivos de dados.

O MongoDB utiliza um formato de armazenamento de dados chamado BSON, uma variação binária do JSON, que permite o armazenamento de tipos de dados mais complexos e uma serialização mais eficiente. Essa abordagem é útil para armazenar e combinar dados com estruturas variadas e complexas sem a necessidade de um esquema pré-definido. Outra característica do MongoDB é a sua capacidade de realizar consultas flexíveis, buscando e filtrando dados com base em qualquer campo dos documentos.

2.3 Node.js

Node.js, conforme descrito no site da Oracle, é um ambiente de execução JavaScript de código aberto e multiplataforma, com foco em aplicações de servidor e rede. Criado por Ryan Dahl em 2009, o Node.js começou como um experimento e rapidamente ganhou popularidade entre os engenheiros pela sua capacidade de executar programas de alto desempenho sem a complexidade de linguagens tradicionais.

O Node.js se tornou uma parte integrante dos ambientes de desenvolvimento web em diversos setores, devido à sua escalabilidade e capacidades de alto desempenho. Ele é otimizado para aplicações web em tempo real, superando outras linguagens e frameworks em conexões simultâneas e tarefas intensivas de I/O (*Input*/*Output*), como foi dito ainda no site da Oracle. Exemplos de uso do Node.js incluem aplicações de chat, web apps, jogos e outras aplicações em tempo real.

2.4 Express

Express é um *framework* de aplicação web para Node.js, com um design minimalista que surgiu como uma resposta ao excesso encontrado em outros frameworks, permitindo um desenvolvimento mais ágil e focado (BROWN, 2019).

Ele opera de maneira simples ao processar requisições *Hypertext Transfer Protocol* (HTTP) de diferentes clientes e retornar respostas HTTP. Essa flexibilidade o torna altamente versátil, atendendo servidores de páginas web tradicionais ou APIs para aplicações mais complexas. Mesmo com a popularidade de *frameworks* para aplicações de página única, como React ou Angular, o Express, para Brown, se mantém relevante ao fornecer uma base sólida para o lado do servidor dessas aplicações.

O *framework* possui um ecossistema de pacotes comunitários, permitindo aos desenvolvedores escolherem as ferramentas e componentes que melhor se encaixam em seus projetos. Esta abordagem, ainda segundo Brown, facilita a integração com diversas tecnologias e necessidades de negócios.

2.5 JWT

JSON *Web Tokens* (JWT) é um método eficiente para a transmissão de informações seguras entre partes em uma aplicação web. De acordo com a especificação oficial (RFC 7519), JWT é um padrão para a criação de *tokens* de acesso que permitem a transmissão de declarações entre duas partes de maneira segura e confiável.

Os JWTs são úteis para autenticação e autorização em aplicações web. Eles funcionam permitindo que um servidor gere um *token* que certifica a identidade do usuário e o envie para o cliente. Posteriormente, o cliente envia o *token* de volta para o servidor para acessar recursos protegidos. Sua compactação permite uma comunicação eficiente em ambientes onde a largura de banda é uma preocupação.

2.6 Bootstrap

Bootstrap, segundo seu próprio site, é um *framework* de código aberto utilizado para o desenvolvimento de *front-end*, proporcionando aos desenvolvedores uma maneira eficiente e menos manual de criar aplicações web responsivas. O Bootstrap é projetado para simplificar e acelerar o processo de desenvolvimento web, oferecendo componentes reutilizáveis e pré-estilizados, como botões, cartões, navegação, etc.

2.7 Jest

Testes unitários, conforme explicado por Martin Fowler, são uma parte fundamental do desenvolvimento de software, embora sejam frequentemente mal definidos. Eles são considerados testes de baixo nível, focados em garantir a implementação de uma pequena parte do sistema de software. Geralmente, são escritos pelos próprios programadores usando suas ferramentas regulares, com a única diferença sendo o uso de algum *framework* de teste unitário.

Um exemplo de ferramenta de testes unitários é o Jest, que é um *framework* de código aberto para JavaScript. Segundo o portal JavaTpoint, ele é muito usado em aplicações web, focando na simplicidade durante a realização de testes unitários. O Jest possui uma biblioteca de simulação (*mocking*), uma biblioteca de asserções e um executor de testes, facilitando a validação de código JavaScript.

Além disso, o Jest requer pouca configuração e possui uma integração eficiente com *frameworks* como Vue.js e React.

3. MATERIAIS E PROCEDIMENTOS METODOLÓGICOS

O objetivo deste capítulo é expor as metodologias e estratégias adotadas para a realização deste projeto, assim como será feita uma introdução sobre as ferramentas utilizadas na implementação do aplicativo.

3.1 Métodos

Esta pesquisa é classificada como bibliográfica em relação aos procedimentos técnicos, pois é fundamentada em monografias, artigos e teses relacionados ao tema escolhido. A pesquisa bibliográfica desempenha um papel fundamental e preliminar em qualquer trabalho científico, fornecendo ao pesquisador informações públicas que ele ainda não possui. No entanto, ela não gera conhecimento novo por si só (WAZLAWICK, 2020).

Para executar o projeto, adotou-se uma metodologia composta por quatro fases:

1. Levantamento dos requisitos da aplicação.
2. Realização de uma pesquisa bibliográfica sobre as linguagens e ferramentas empregadas no desenvolvimento da aplicação.
3. Definição da arquitetura da aplicação e da comunicação entre os entre os sistemas que compõem a solução.
4. Implementação da solução concebida.
5. Realização de testes na aplicação a fim de verificar e confirmar o funcionamento adequado das funcionalidades.

3.2 Materiais

3.2.1 Visual Studio Code

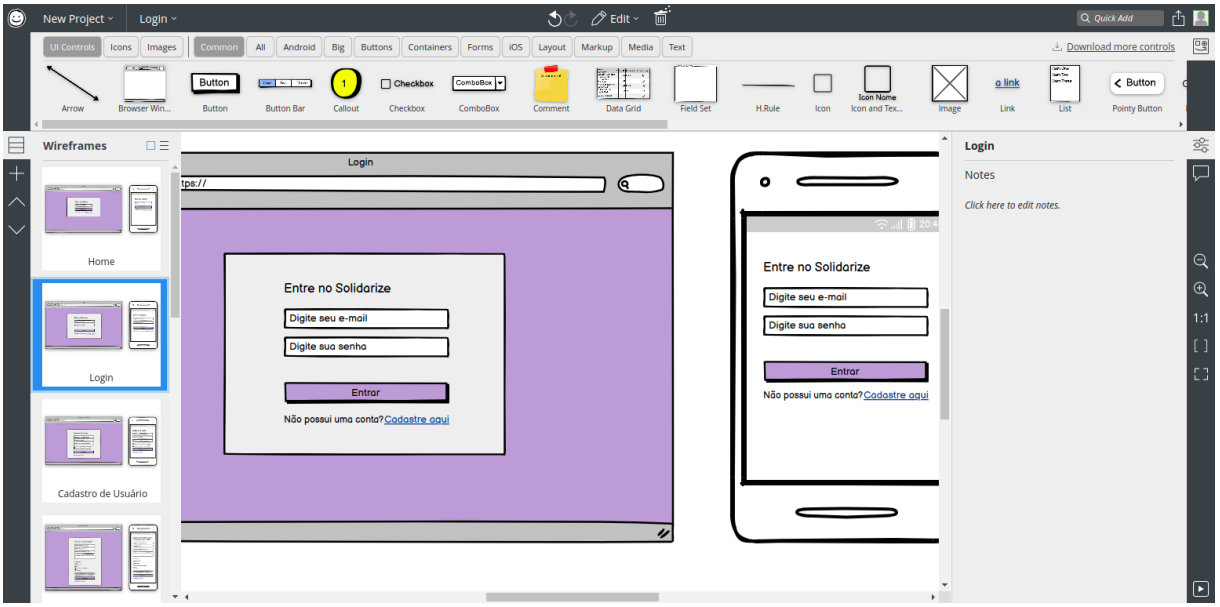
Visual Studio Code é um editor de código-fonte gratuito desenvolvido pela Microsoft.Segundo sua documentação oficial, é uma ferramenta disponível para Windows, Linux e macOS, o editor possui suporte a uma grande diversidade de linguagens de programação como Python, Java, JavaScript, C++, entre outras. Além disso, oferece integração com ferramentas de controle de versão Git, destacando-se

pelo seu suporte a depuração, *snippets* de código e refatoração de código inteligente. Uma das principais características do VS Code é a sua extensibilidade, permitindo aos usuários personalizar o ambiente de desenvolvimento com seleção de extensões e temas disponíveis em seu *marketplace*.

3.2.2 Balsamiq Cloud

A Balsamiq Cloud, conforme sua documentação oficial, é uma ferramenta web que permite aos usuários criar *wireframes* e protótipos com uma baixa curva de aprendizagem. Foi projetada para facilitar o esboço de ideias para aplicações e websites, provendo elementos, Figura 1, como barra de ferramentas, painel de navegação e inspetor de propriedades.

Figura 1 - Painel de edição do Balsamiq Cloud.



Fonte: Autoria própria.

3.2.3 Postman

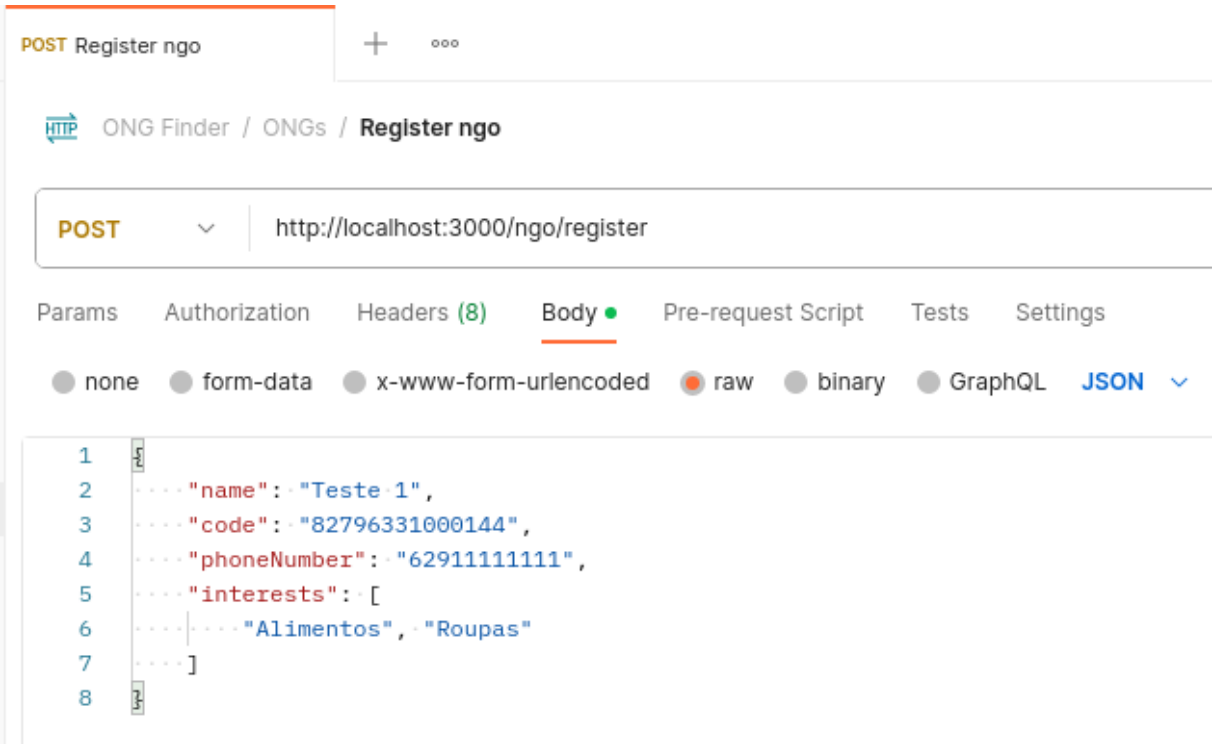
Postman é uma plataforma de API utilizada para construir e utilizar APIs. Segundo a seção do Postman do site JavaTpoint, esta ferramenta permite aos desenvolvedores enviar e receber solicitações de API para e de um servidor web,

facilitando significativamente o processo de teste de APIs. Sua interface gráfica é simples, e não requer a escrita de código de rede cliente HTTP para fins de teste.

A funcionalidade de envio de requisições para servidores pode ser vista na Figura 2. Cada parte da URL, que se encontra na primeira entrada de texto, tem uma finalidade: o protocolo (http), o endereço do servidor (*localhost*), a porta (3000) e o caminho ou *endpoint* (/ngo/register). Com estes dados, é possível fazer a solicitação para um servidor específico e o serviço desejado.

O corpo da requisição, localizado na caixa de texto, contém informações necessárias para a operação desejada, que no caso da Figura 2 são os dados da ONG a ser cadastrada. Isso exemplifica como o Postman facilita a configuração e o teste de diferentes requisições HTTP sem a necessidade de codificação adicional.

Figura 2 - Exemplo de requisição no Postman.



Fonte: Autoria própria.

3.2.4 MongoDB Atlas

O MongoDB Atlas, segundo sua documentação oficial, é um serviço de banco de dados na nuvem que cuida de todo o trabalho de configurar, gerenciar e consertar seus bancos de dados na nuvem. Ele permite que você use vários

serviços de nuvem diferentes e inclui várias ferramentas para trabalhar com dados. O Atlas torna mais fácil usar o MongoDB, um tipo de banco de dados que não usa tabelas tradicionais, mantendo-o disponível, rápido e seguro à medida que as necessidades mudam.

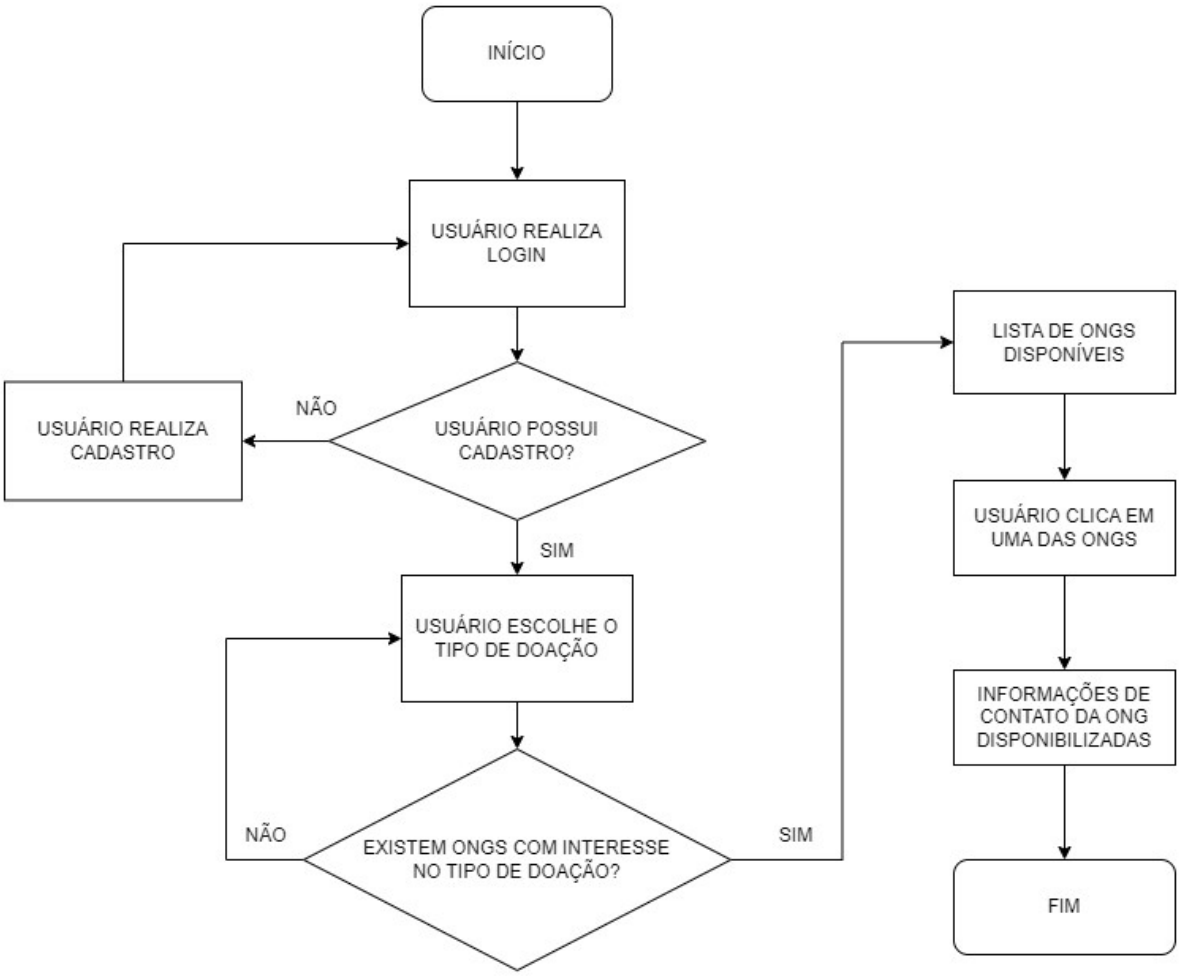
4. DESENVOLVIMENTO

4.1 Descrição Geral do Produto

O Solidarize é formado por duas partes: uma API, que se liga a um banco de dados para guardar informações de ONGs; e uma aplicação web destinada aos doadores e também gestores das ONGs. Os usuários doadores podem procurar informações sobre as ONGs disponibilizadas pela API. Já os gestores podem tanto efetuar o cadastro de suas organizações quanto procurar outras ONGs. O Datagrama 1 ilustra o caminho que o usuário realiza do Login até a busca da ONG.

Nesta parte do fluxo, o usuário pode realizar o Login. Caso ele não possua um cadastro, ele tem a opção de se cadastrar. Com o cadastro realizado, o usuário pode fazer o Login para ser redirecionado ao *Dashboard*. Nesta página, o usuário pode selecionar uma categoria de doação. Caso não existam ONGs com interesse na categoria selecionada, uma janela *pop-up* é exibida alertando o usuário. Caso exista pelo menos uma ONG com o interesse no tipo de doação escolhido, é exibida uma lista das ONGs disponíveis. O usuário pode clicar em qualquer ONG da lista, e uma janela de sobreposição (modal) contendo as informações da ONG é exibida.

Datagrama 1 - Caminho principal na busca de ONG.



Fonte: Autoria própria.

4.2 Organização das Tecnologias

No capítulo de Referencial Teórico foram apresentadas algumas tecnologias utilizadas no projeto. Nesta seção, será discutido o uso prático destas tecnologias no sistema desenvolvido.

Iniciando pela API, ela foi desenvolvida utilizando o Node.js e o banco de dados MongoDB. Para facilitar a conexão da aplicação com o banco de dados, foi utilizado o Express. Neste processo, a biblioteca Mongoose do Node.js também foi empregada, oferecendo uma maneira mais estruturada e modelada de interagir com o MongoDB. Por fim, o JWT foi implementado para a autenticação dos usuários.

A aplicação web, por sua vez, foi desenvolvida com o *framework* Vue.js. Nesta etapa, além da elaboração do layout da aplicação, também foi implementada

a conexão com a API. Para a construção da estrutura das páginas e estilização da interface, foi utilizado o Bootstrap, priorizando o uso dos componentes padrões e minimizando customizações.

O Jest foi escolhido devido à sua eficácia em garantir a confiabilidade e o cumprimento dos requisitos do sistema. Sua utilização abrangeu tanto o *back-end* quanto o *front-end*. Nem todos os arquivos foram testados, pois muitos não envolvem lógica, como arquivos de configuração.

4.3 Descrição de Usuários

O sistema desenvolvido possui dois tipos de usuários: gestores de ONGs e doadores. Para os gestores, a aplicação web serve como uma ferramenta para cadastrar e gerenciar informações sobre suas entidades, bem como buscar outras ONGs. Por outro lado, os doadores têm acesso limitado, sendo autorizados apenas a buscar ONGs na plataforma.

O registro na aplicação é um pré-requisito para ambos os grupos acessarem suas funcionalidades específicas. Os gestores, após se cadastrarem como usuários, devem adicionar informações sobre a ONG que pretendem associar à sua conta. Já os doadores passam por um processo de cadastro semelhante ao dos gestores, mas sem a necessidade de registrar uma ONG. Este procedimento de cadastro foi projetado para assegurar uma camada adicional de segurança.

4.4 Requisitos Funcionais

Nesta seção, será detalhado os requisitos funcionais do sistema, localizados na Tabela 1.

Tabela 1 - Lista de requisitos funcionais.

(continua)

Identificador	**Requisito Funcional**	**Descrição Detalhada**
RF-01	Cadastro de usuário	O usuário deve ter a capacidade de se registrar no sistema.
RF-02	Escolha de tipo de usuário	O usuário deve ter a capacidade de escolher entre ser um doador ou gestor de ONG.
RF-03	Registro de ONG	O usuário gestor deve ter a capacidade de registrar sua ONG.
RF-04	Autenticação de usuário	O usuário deve ter a capacidade de efetuar login.
RF-05	Pesquisa de ONGs	O usuário deve ter a capacidade de buscar ONGs baseados no tipo de doação.
RF-06	Visualização da listagem de ONGs	O usuário deve ter a capacidade visualizar uma lista com as ONGs de acordo com o tipo de doação escolhido.

Tabela 1 - Lista de requisitos funcionais.

(conclusão)

Identificador	Requisito Funcional	Descrição Detalhada
RF-07	Visualizar informações de uma ONG	O usuário deve ter a capacidade de ver os detalhes de qualquer ONG listada.
RF-08	Visualizar informações da própria ONG	O usuário gestor deve ter a capacidade de visualizar dados de sua própria ONG.
RF-09	Alteração de dados da ONG	O usuário gestor deve ter a capacidade de alterar dados de sua ONG.
RF-10	Deslogar o usuário	O usuário deve ter a capacidade de deslogar do sistema.

Fonte: Autoria própria.

4.5 Requisitos Não Funcionais

Nesta seção, será detalhado os requisitos não funcionais do sistema, localizados na Tabela 2.

Tabela 2 - Lista de requisitos não funcionais.

Identificador	Requisito Não Funcional	Descrição Detalhada
RNF-01	Usabilidade	As interfaces de usuário das aplicações devem ser intuitivas e fáceis de usar.
RNF-02	Usabilidade	Os caracteres digitados nos campos de senha devem ser ocultados.
RNF-03	Usabilidade	A aplicação deve passar mensagens de sucesso ou falha de uma ação na aplicação a partir de uma pequena janela de *pop-up*.
RNF-04	Integração e Consistência de Dados	Deve haver integração eficiente e consistência de dados entre as aplicações móvel e web.
RNF-05	Segurança de Dados	Os dados de usuários devem ser protegidos contra acessos não autorizados.
RNF-06	Segurança de Dados	Os usuários devem ter suas senhas criptografadas antes de serem persistidas no banco de dados.

Fonte: Autoria Própria.

4.6 Modelo Conceitual

4.6.1 Entidades e atributos

A aplicação é composta por três entidades principais, cada uma com seus atributos, conforme descrito a seguir:

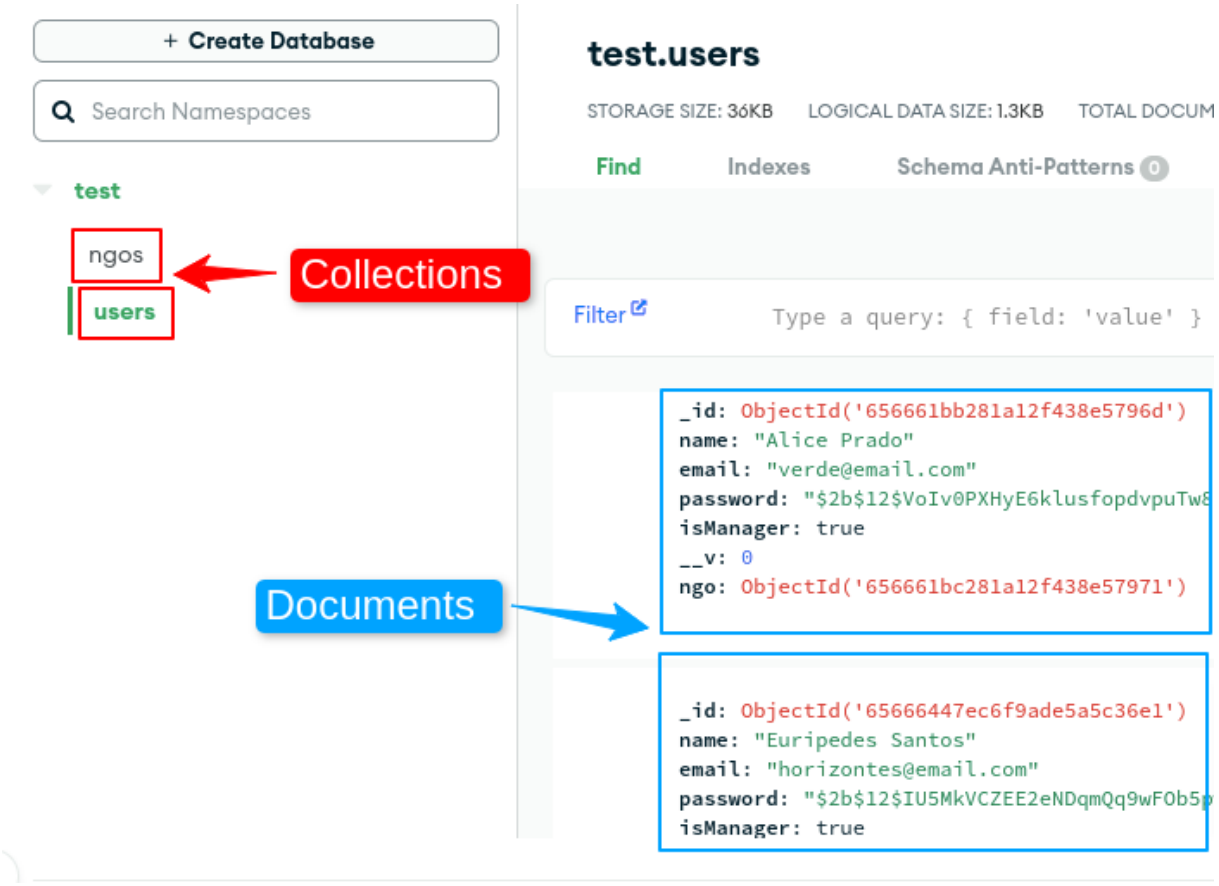
1. Doador: representa um usuário do tipo doador. Possui os atributos:
    a. ID do Usuário: Um identificador único para cada usuário.
    b. Nome: O nome do doador.
    c. E-mail: Endereço de e-mail do usuário.
    d. Senha: Senha criptografada para acesso à conta.
    e. Tipo: Identifica o usuário como um doador.
2. Gestor: representa um usuário do tipo gestor de ONG. Possui os atributos:
    a. ID do Usuário: Um identificador único para cada usuário.
    b. Nome: O nome do gestor.
    c. E-mail: Endereço de e-mail do gestor.
    d. Senha: Senha criptografada para acesso à conta.
    e. Tipo: Identifica o usuário como um gestor de ONG.
    f. ONG gerenciada: Contém referência à ONG que está sob a gestão do usuário.
3. ONG: representa uma Organização Não Governamental. Possui os atributos:
    a. ID da ONG: Um identificador único para cada ONG.
    b. CNPJ: O número de documento que pertence à ONG.
    c. Nome da ONG: O nome da organização.
    d. Descrição: Uma descrição detalhada da ONG.
    e. Contato: Representa o telefone.
    f. Interesses: Uma lista dos tipos de doações que a ONG aceita, como financeira, roupas, alimentos, etc.
    g. Imagem: A imagem de perfil da ONG, que é mostrada na busca de ONGs.

4.6.2 Modelo de Dados do MongoDB

Em bancos de dados NoSQL como o MongoDB, que são orientados a documentos, os "documentos" são as unidades básicas para armazenar dados, similares às linhas em bancos de dados relacionais, mas com maior flexibilidade. Eles são compostos por campos e valores e são armazenados em "coleções", que são equivalentes às tabelas em bancos de dados relacionais.

Na Figura 3, a coleção *users* dentro do banco de dados *test* exemplifica o conceito de documentos em MongoDB. Cada documento, identificado por um campo "_id" único, armazena dados de usuários, incluindo nome, email, senha e um status indicando o tipo de usuário, indicado pelo campo booleano *isManager*. Além disso, pode-se observar uma referência ao campo "ngo", que representa a ligação entre o documento de usuário e o documento de ONG da coleção de ONGs ("ngos", acima da aba "users" selecionada). Isto demonstra como as relações entre diferentes conjuntos de dados são gerenciadas em um sistema NoSQL.

Figura 3 - Visualização de *Collections* e *Documents* no MongoDB Atlas.



Fonte: Autoria própria.

A aplicação conta com as seguintes coleções:

1. Coleção de Usuários (Doadores e Gestores):
    a. São diferenciados entre “Doadores” e “Gestores” pelo atributo “Tipo”.
    b. Usuários do tipo “Gestores” incluem o campo adicional “ONG”, que faz referência à ONG que o gestor é responsável.
    c. Os documentos dos doadores são mais simplificados, focados na funcionalidade de busca e visualização das ONGs.
2. Coleção de ONGs:
    a. Contém documentos detalhando cada ONG.
    b. Acessível por doadores para busca e visualização.
    c. Acessível por gestores para gestão e atualização das informações das ONGs.

4.7 Documentação da API

Nesta seção, será detalhada as rotas da API, apontando o *endpoint*, método, funcionalidade do *endpoint*, corpo da requisição e resposta esperada do servidor.

Tabela 3 - Rotas da API.

(continua)

Endpoint	**Método**	**Descrição/Funcionalidade**	**Parâmetros/Corpo da Requisição**	**Resposta Esperada**
/login	POST	Autenticar usuário	email: String, senha: String	token: JWT Token, tipoUsuario: String
/logout	POST	Desautenticar usuário	N/A	N/A
/user/register	POST	Registrar novo usuário	nome: String, email: String, senha: String, tipoUsuario: String	message: String
/user/check-user	GET	Verificar existência de usuário	email: String	usuario: Object

Tabela 3 - Rotas da API.

(continuação)

Endpoint	**Método**	**Descrição/Funcionalidade**	**Parâmetros/Corpo da Requisição**	**Resposta Esperada**
/ngo/register	POST	Registrar nova ONG	nome: String, cnpj: String, telefone: String, descricao: String, interesses: Array, imagem: String	mensagem: String
/ngo/check-ngo-cnpj	GET	Verificar existência de ONG pelo CNPJ	cnpj: String	mensagem: String
/ngo/check-ngo-id	GET	Verificar existência de ONG pelo ID	id: String	mensagem: String, ong: Object
/ngo/check-ngo-phone	GET	Verificar existência de ONG pelo telefone	telefone: String	mensagem: String
/ngo/delete	DELETE	Remover ONG	cnpj: String	mensagem: String

Tabela 3 - Rotas da API.

(conclusão)

Endpoint	**Método**	**Descrição/Funcionalidade**	**Parâmetros/Corpo da Requisição**	**Resposta Esperada**
/protected/update	PUT	Atualizar dados da ONG	nome: String, cnpj: String, telefone: String, descricao: String, interesses: Array, imagem: String	mensagem: String
/protected/list-category	GET	Listar ONGs por interesses	pagina: numeric, tamanhoDaPagina: numeric, interesse: String	mensagem: String, ongs: Array

Fonte: Autoria própria.

4.8 Arquitetura

Nesta seção, será descrita a arquitetura do sistema, assim como a comunicação entre suas partes. A arquitetura empregada no projeto foi a Arquitetura Limpa, em inglês *Clean Architecture*, uma metodologia de *design de software* desenvolvida por Robert C. Martin, que enfatiza a separação de responsabilidades e a independência de *frameworks* (MARTIN, 2017). Essa abordagem permite criar um sistema flexível, ideal para um projeto que envolve múltiplas plataformas e tecnologias, como uma aplicação web em Vue.js e uma API REST Node.js.

A estrutura proposta separa a lógica de negócios, o armazenamento de dados e as interfaces de usuário. Seguindo as diretrizes de Martin (2017), o projeto foi organizado em entidades, casos de uso, adaptadores e componentes externos,

garantindo a organização necessária para ter uma manutenção eficiente em cada componente do sistema.

O fluxo geral de comunicação entre os sistemas, que também está representado na Figura 4, é o seguinte:

1. Usuário interage com a Aplicação Web: realiza uma ação que requer dados ou alterações (por exemplo, visualizar listas de ONGs).
2. Aplicação Web envia Requisição para a API: a requisição é processada pelos controladores na API.
3. API Executa Lógica de Negócios e Acessa Banco de Dados: retorna a resposta para a aplicação cliente.
4. Aplicação Cliente Recebe e Processa a Resposta: Atualiza a *User Interface* (UI) com base nos dados recebidos.

Figura 4 - Representação da comunicação entre partes do sistema.

Fonte: Autoria própria.

4.9 Protótipos das Telas

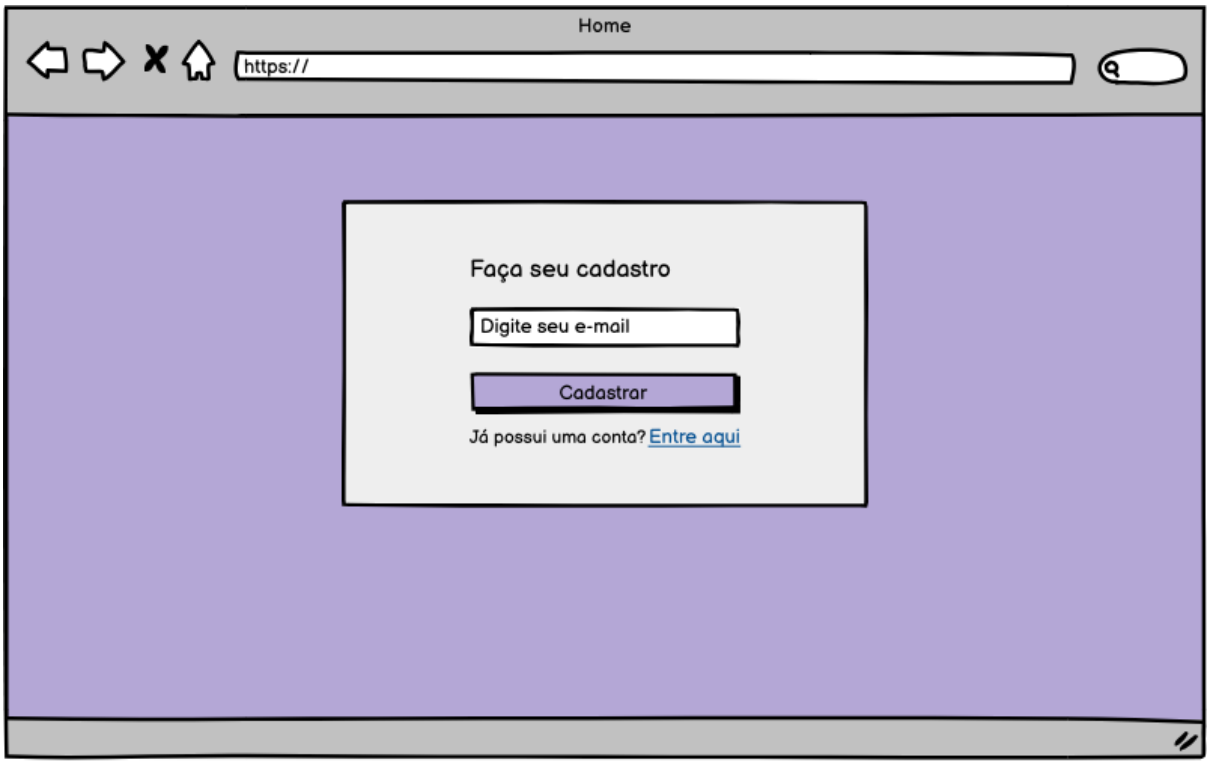
Como discutido previamente neste trabalho, o protótipo foi desenvolvido utilizando o Balsamiq Cloud, uma ferramenta focada na prototipagem de interfaces e no design de experiência do usuário. O Balsamiq permite criar protótipos interativos que podem ser navegados por meio de um link, oferecendo a oportunidade de explorar as telas da aplicação antes mesmo de sua implementação.

Embora o fluxograma da aplicação já tenha sido mostrado anteriormente, nesta seção, o fluxo será descrito com imagens ilustrativas. Durante a prototipagem, foram desenvolvidos modelos tanto para *desktop* quanto para *mobile*. No entanto,

para melhorar a legibilidade no trabalho, serão exibidas somente as versões para *desktop*.

A tela inicial, Figura 5, possui um campo de entrada para o usuário digitar um e-mail caso queira se cadastrar. No clique do botão “Continuar”, o usuário é levado para outra tela que dá continuidade ao cadastro. Caso o usuário já tenha cadastro, o *link* “Entre aqui” prossegue para a tela de Login.

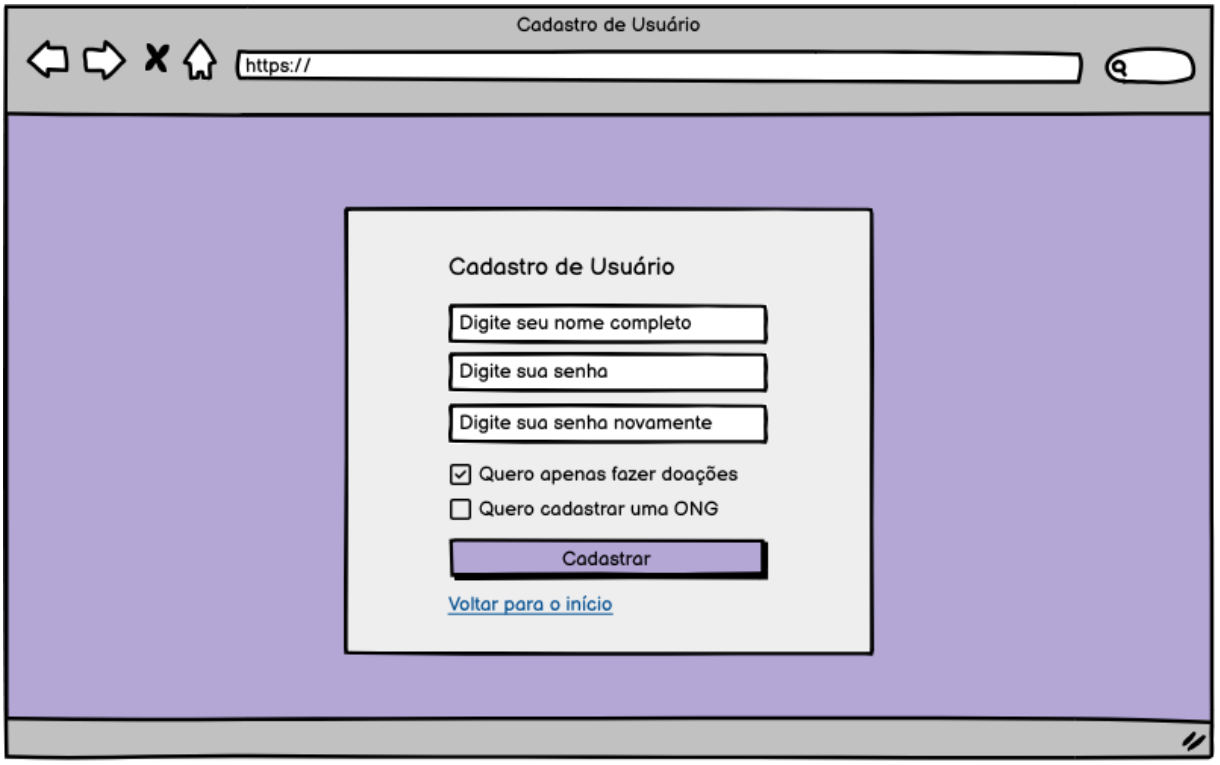
Figura 5 - Protótipo da tela de Início.



Fonte: Autoria própria.

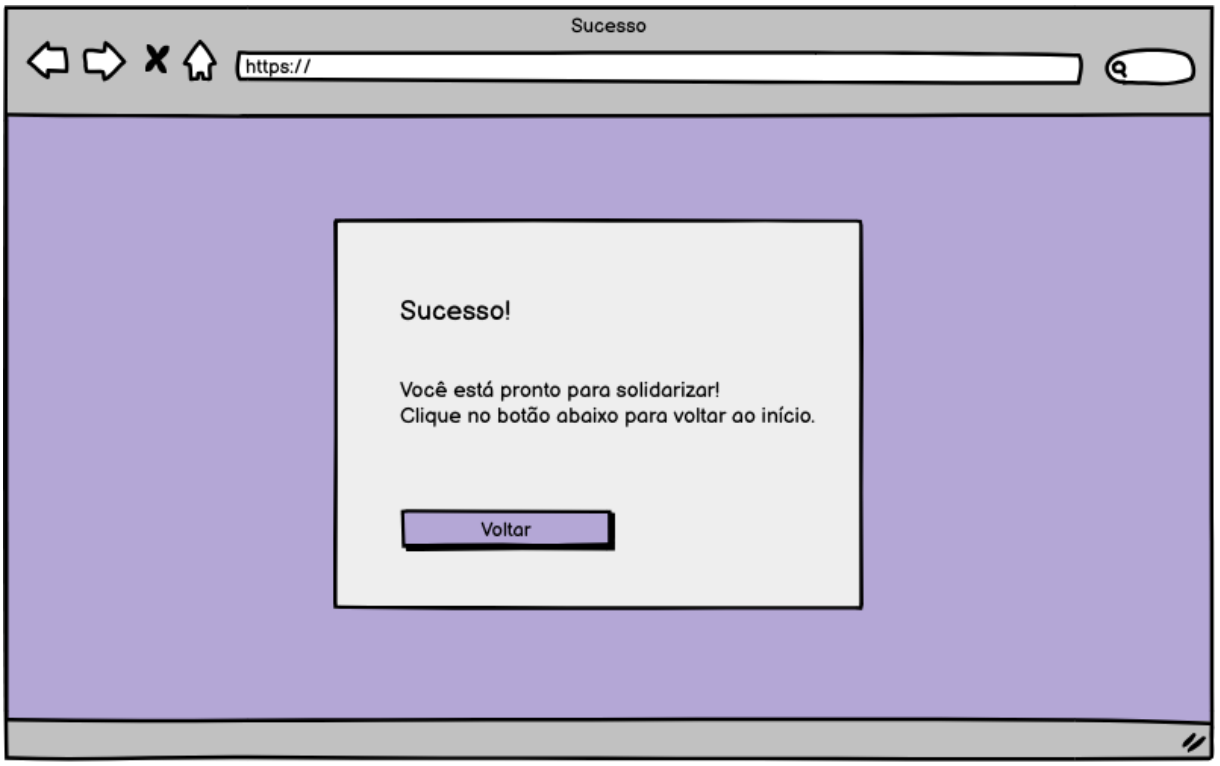
Na tela de Cadastro, Figura 6, o usuário é instruído a preencher três campos: nome completo, senha e confirmação da senha. Nesta etapa, o usuário pode escolher se será um doador ou gestor de ONG. Caso o usuário escolha a opção de doador, sua jornada de cadastro finaliza e ele é direcionado para a tela de Sucesso, Figura 7.

Figura 6 - Protótipo da Tela de Cadastro.



Fonte: Autoria própria.

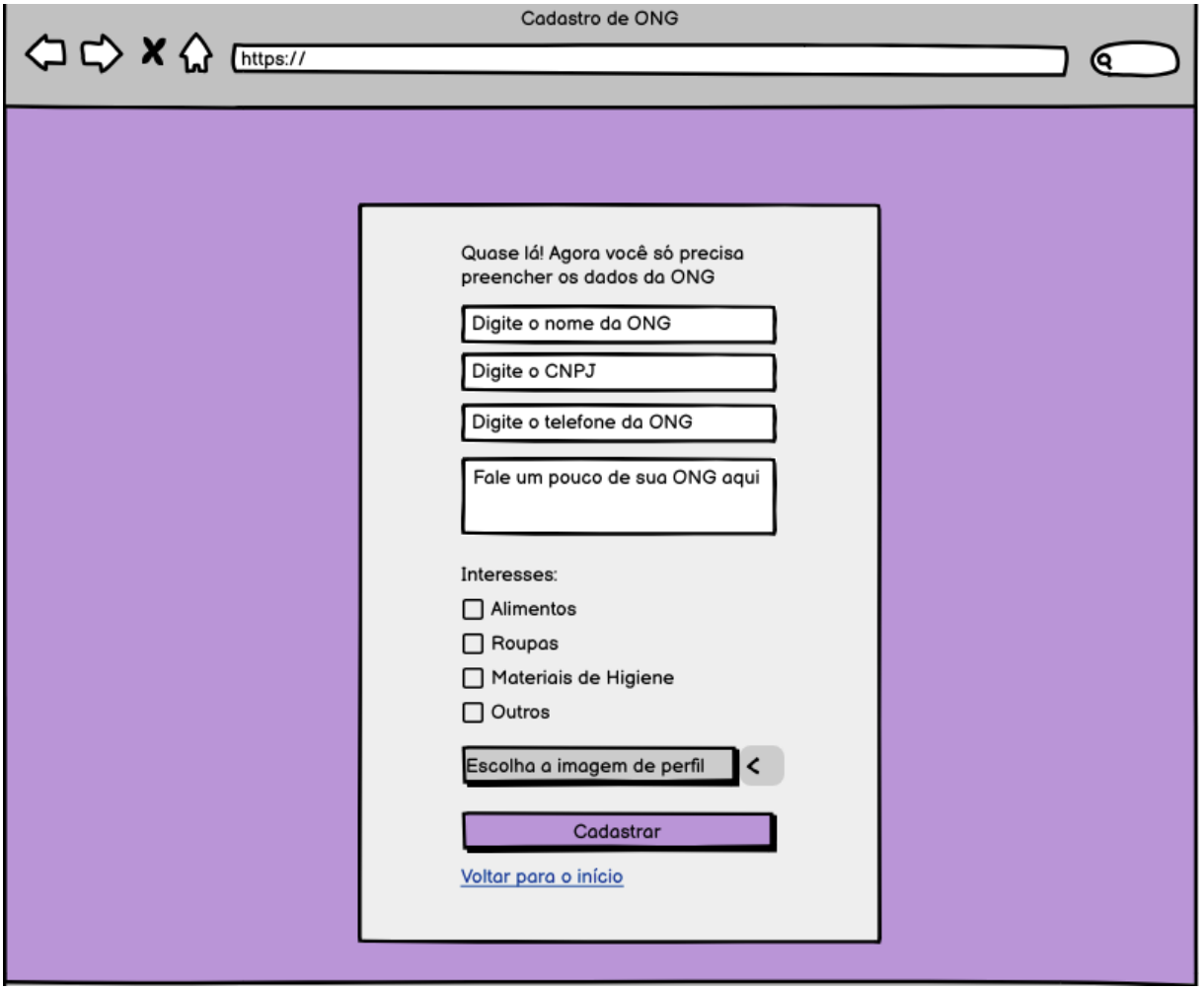
Figura 7 - Protótipo da Tela de Sucesso.



Fonte: Autoria própria.

Já se o usuário selecionar a opção de gestor, o direcionamento ocorre para a tela de Cadastro da ONG, ilustrada na Figura 8. Nela, o usuário deve preencher os dados da ONG, nos quais todos são obrigatórios, com exceção da imagem utilizada para ilustrar o perfil. Após o preenchimento correto dos dados, o usuário é redirecionado para a tela de Sucesso citada previamente.

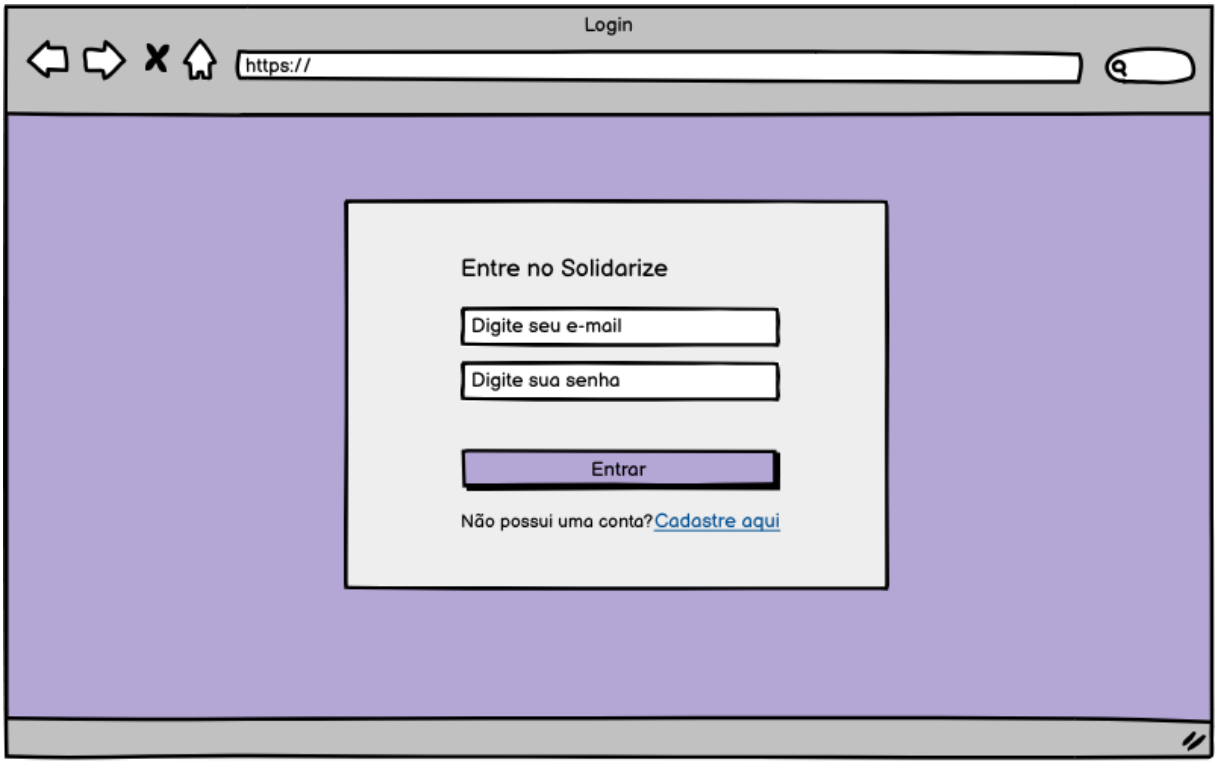
Figura 8 - Protótipo da Tela de Cadastro da ONG.



Fonte: Autoria própria.

Já cadastrado, o usuário pode realizar o login, como mostra a Figura 9. Caso o usuário tenha entrado na tela de Login acidentalmente, o link “Cadastre aqui” o leva de volta para a página inicial, onde se encontra o primeiro passo do Cadastro.

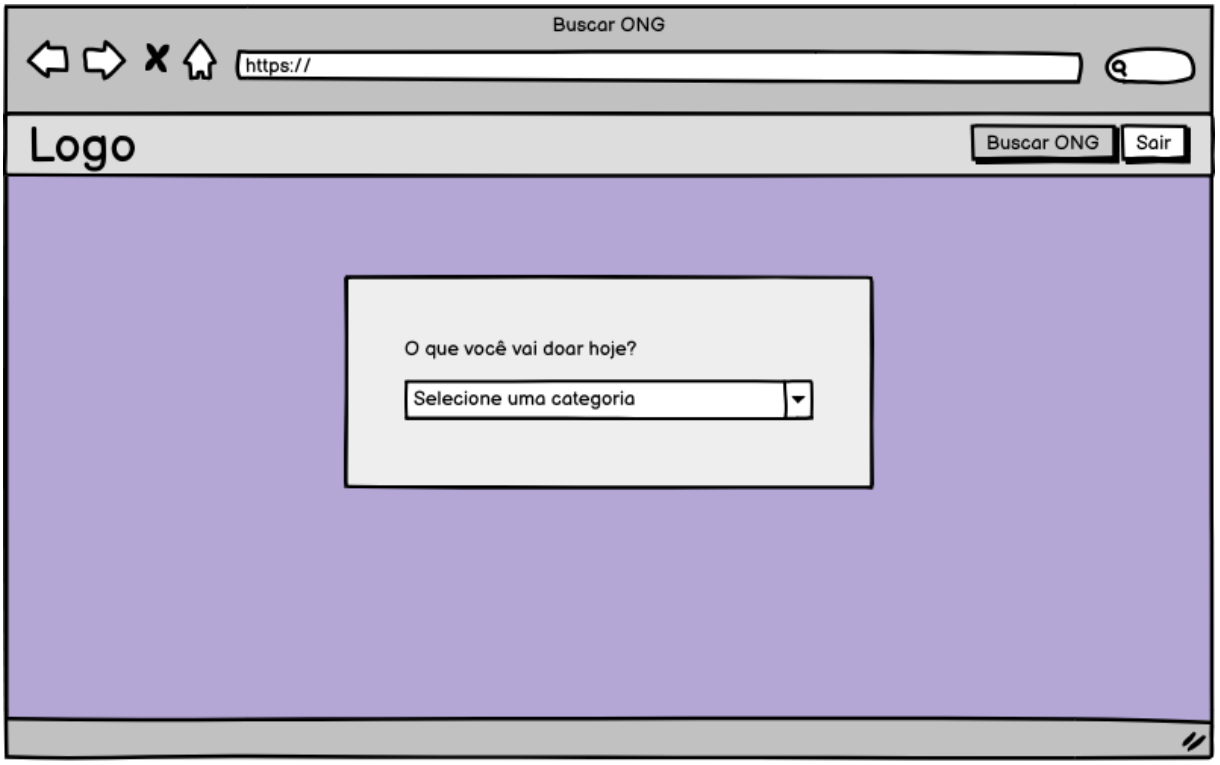
Figura 9 - Protótipo da Tela de Login.



Fonte: Autoria própria.

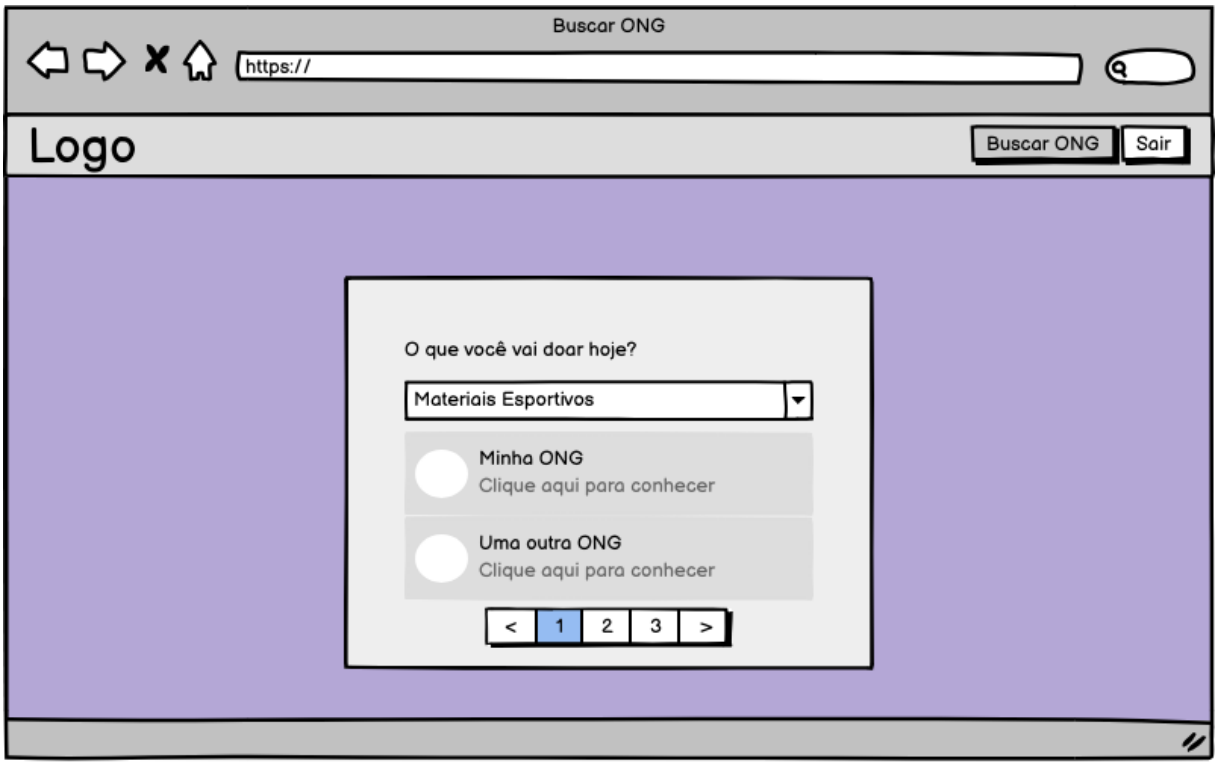
Após o usuário realizar o login e se autenticar no sistema, ele é direcionado para o *Dashboard*. É nesta tela que os usuários podem realizar as ações designadas para cada tipo de usuário. Para usuários doadores, o Dashboard é iniciado com a tela de “Buscar ONG”, da Figura 10. É nela que o usuário pode selecionar a categoria para filtrar e listar as ONGs que têm interesse naquele tipo de doação. A Figura 11 demonstra o protótipo da listagem das ONGs.

Figura 10 - Protótipo da Tela de Busca de ONG.



Fonte: Autoria própria.

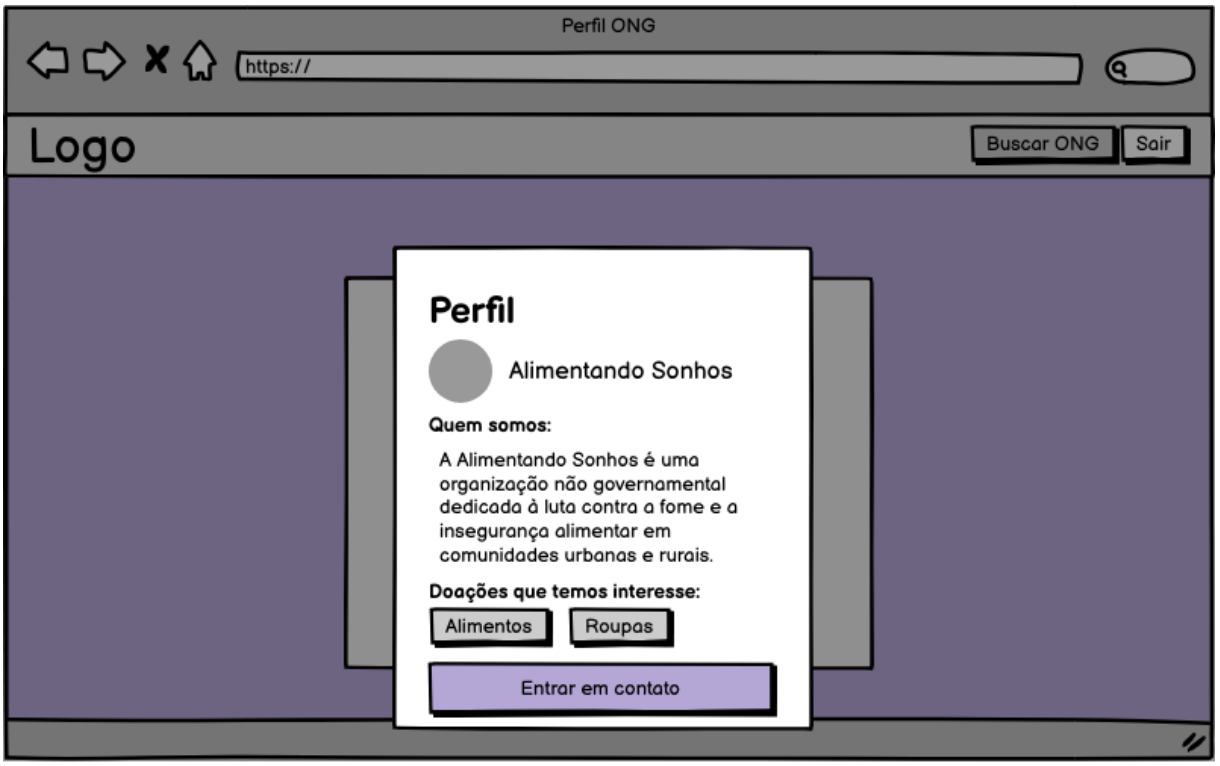
Figura 11 - Protótipo da Tela de Busca de ONG com listagem.



Fonte: Autoria própria.

Quando um usuário seleciona um item da lista, uma janela modal com detalhes sobre a ONG aparece, como na Figura 12. Nesta janela, além de ler as informações, o usuário pode iniciar uma conversa no WhatsApp com o telefone que foi cadastrado pelo gestor da ONG.

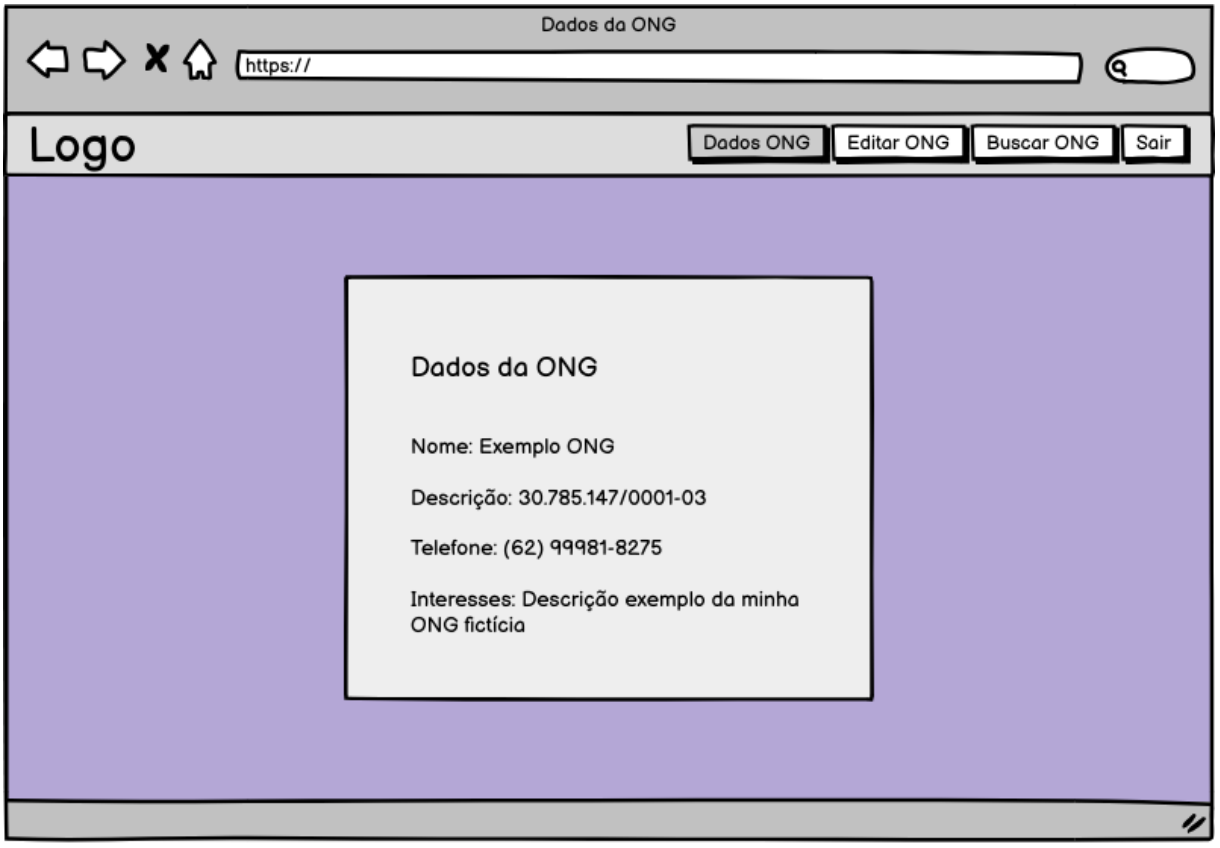
Figura 12 - Protótipo Modal de Perfil da ONG.



Fonte: Autoria própria.

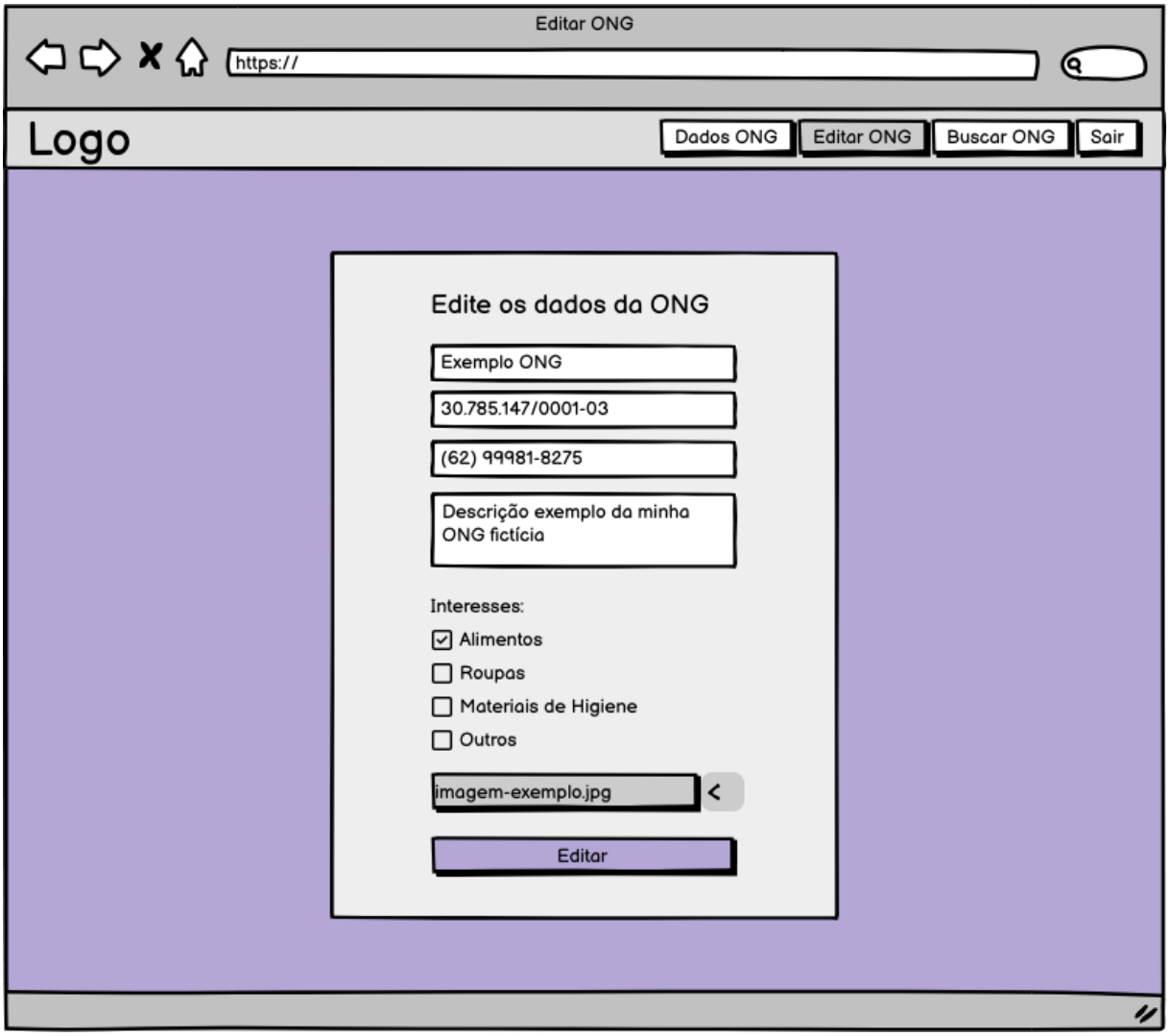
A funcionalidade de busca de ONGs, como citado anteriormente, é disposta a ambos os tipos de usuários. No entanto, somente os gestores têm acesso ao painel de controle onde podem ver e editar as informações das ONGs. Os campos da tela de edição de dados da ONG (Figura 14) são os mesmos campos da tela de cadastro e, ao término da atualização feita pelo gestor, os dados são refletidos na tela de “Dados da ONG”, Figura 13.

Figura 13 - Protótipo da Tela de Dados da ONG.



Fonte: Autoria própria.

Figura 14 - Protótipo de Tela de Edição da ONG.



Fonte: Autoria própria.

Por último, foi prototipada a tela de página não encontrada, vista na Figura 15. Esta tela é apresentada caso o usuário solicite uma rota inexistente. Nela, o usuário pode acessar a tela inicial pelo clique do botão.

Figura 15 - Protótipo de Tela de Página Não Encontrada.

Fonte: Autoria própria.

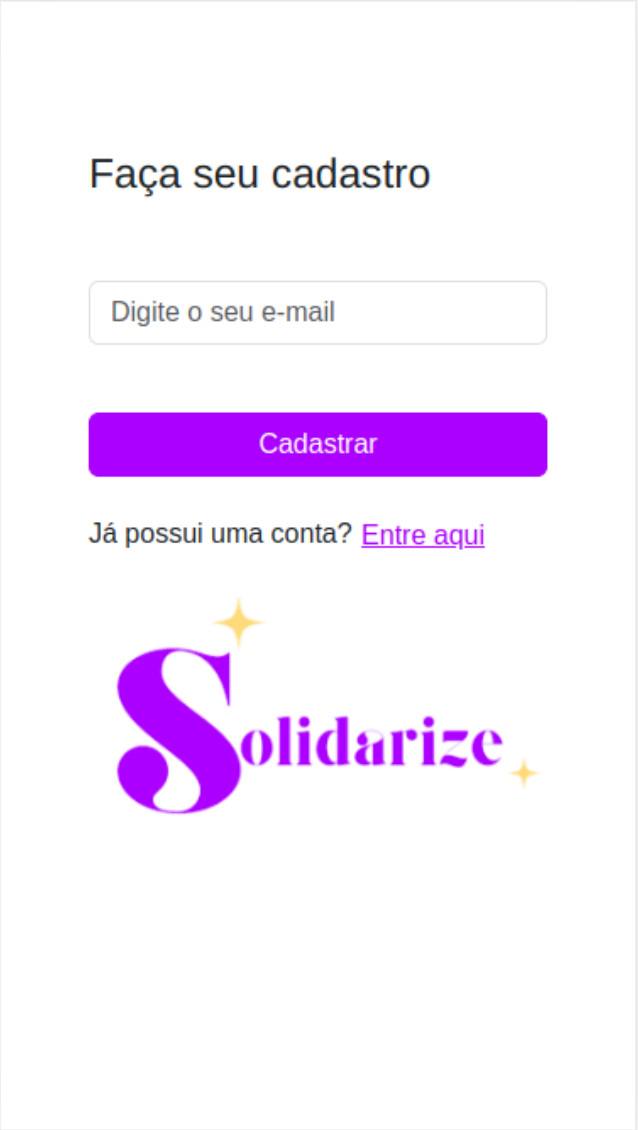
5. RESULTADOS ESPERADOS

Neste capítulo, será apresentada a aplicação implementada e operante, conforme os requisitos funcionais e não funcionais do sistema, detalhando o desenvolvimento de cada uma delas. Tendo em vista que os protótipos foram apresentados no modo *desktop*, a implementação será exposta em modo responsivo adaptado para dispositivos móveis.

5.1 Página de Início

Como foi constatado anteriormente, a tela de início, Figura 16, começa com a primeira etapa do cadastro. Ao inserir um e-mail já cadastrado, é exibida uma mensagem de erro abaixo da entrada de texto. A validação foi feita com uma biblioteca do Vue.js chamada VeeValidate e uma biblioteca do Node.js chamada Yup. Ambas facilitam o processo de validação de campos de formulários. Outra validação deste campo é a do formato do e-mail fornecido.

Figura 16 - Tela de Início.



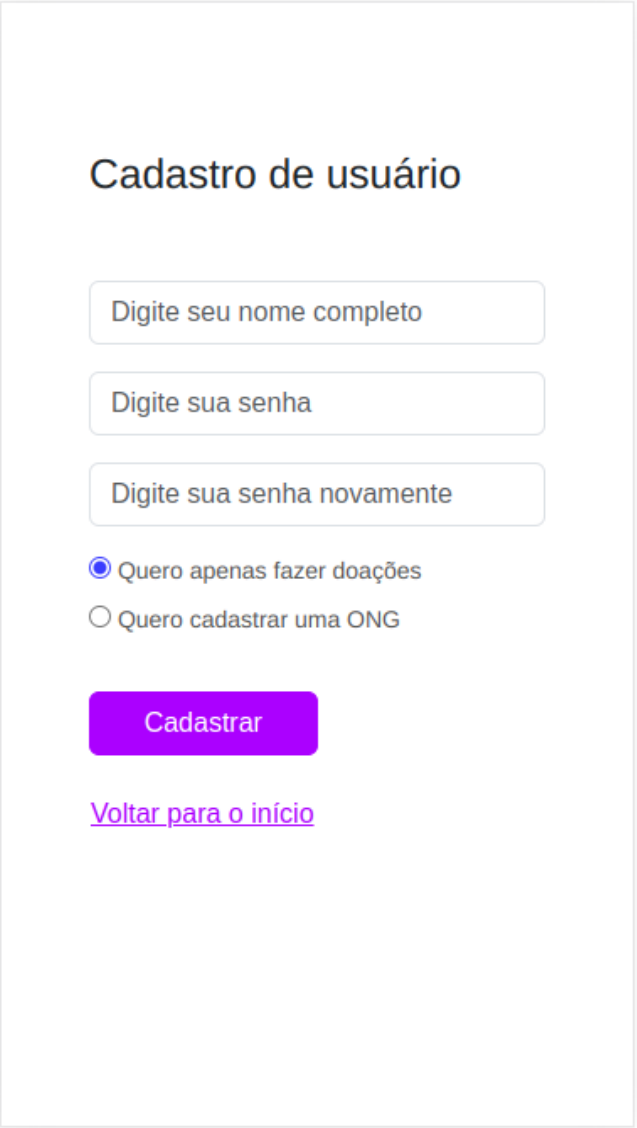
Fonte: Autoria própria.

5.2 Página de Cadastro de Usuário

Nesta página, ilustrada pela Figura 17, o e-mail digitado na tela de Início, é recuperado a partir do Vuex, um pacote do Vue que consiste em um gerenciador de estado. Ele funciona como uma *store* disponível e acessível por todos os componentes Vue. Caso o usuário selecione a opção “Quero apenas fazer doações”, a chamada à rota da API de criação de usuário é realizada. Caso contrário, os dados de e-mail e os solicitados na tela de cadastro são salvos na *store* para serem usados

posteriormente. Para diferenciar os usuários, há um campo "*isManager*", que consiste em um booleano, na *Collection* de *users* do MongoDB.

Figura 17 - Tela de Cadastro de Usuário.



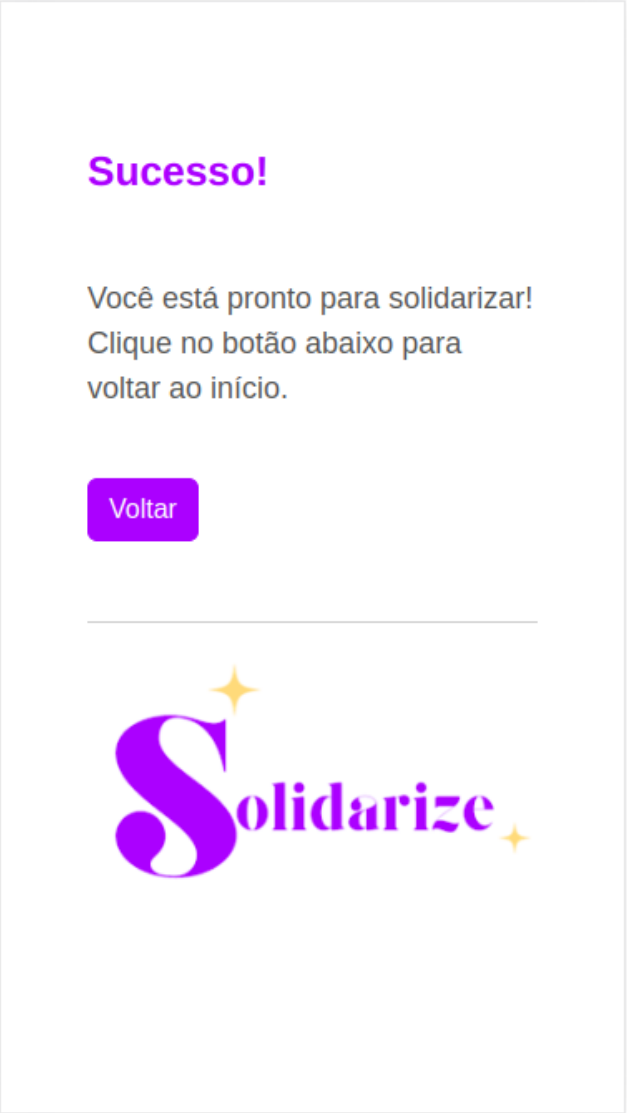
Fonte: Autoria própria.

5.3 Página de Sucesso

A tela de sucesso, Figura 18, é acessada ao fim do cadastro de usuário doador e gestor. Clicando no botão "Voltar", o usuário é direcionado à Página de Início e sua desautenticação é realizada. Para isso, a rota de *logout* da API é

chamada e a *store* do Vuex tem seus valores restaurados. O processo de autenticação será discutido mais à frente neste capítulo.

Figura 18 - Tela de Sucesso.



Fonte: Autoria própria.

5.4 Página de Cadastro de ONG

O usuário chega a esta página se na Página de Cadastro de Usuário ele selecionar a opção “Quero cadastrar uma ONG”. Neste ponto, o objeto salvo na *store* referente aos dados do usuário são acessados. Isto ocorre porque, o cadastro de usuário só ocorre de fato na finalização do cadastro da ONG. Ao finalizar o cadastro da ONG, a instância de ONG gerada é adicionada à instância do Usuário

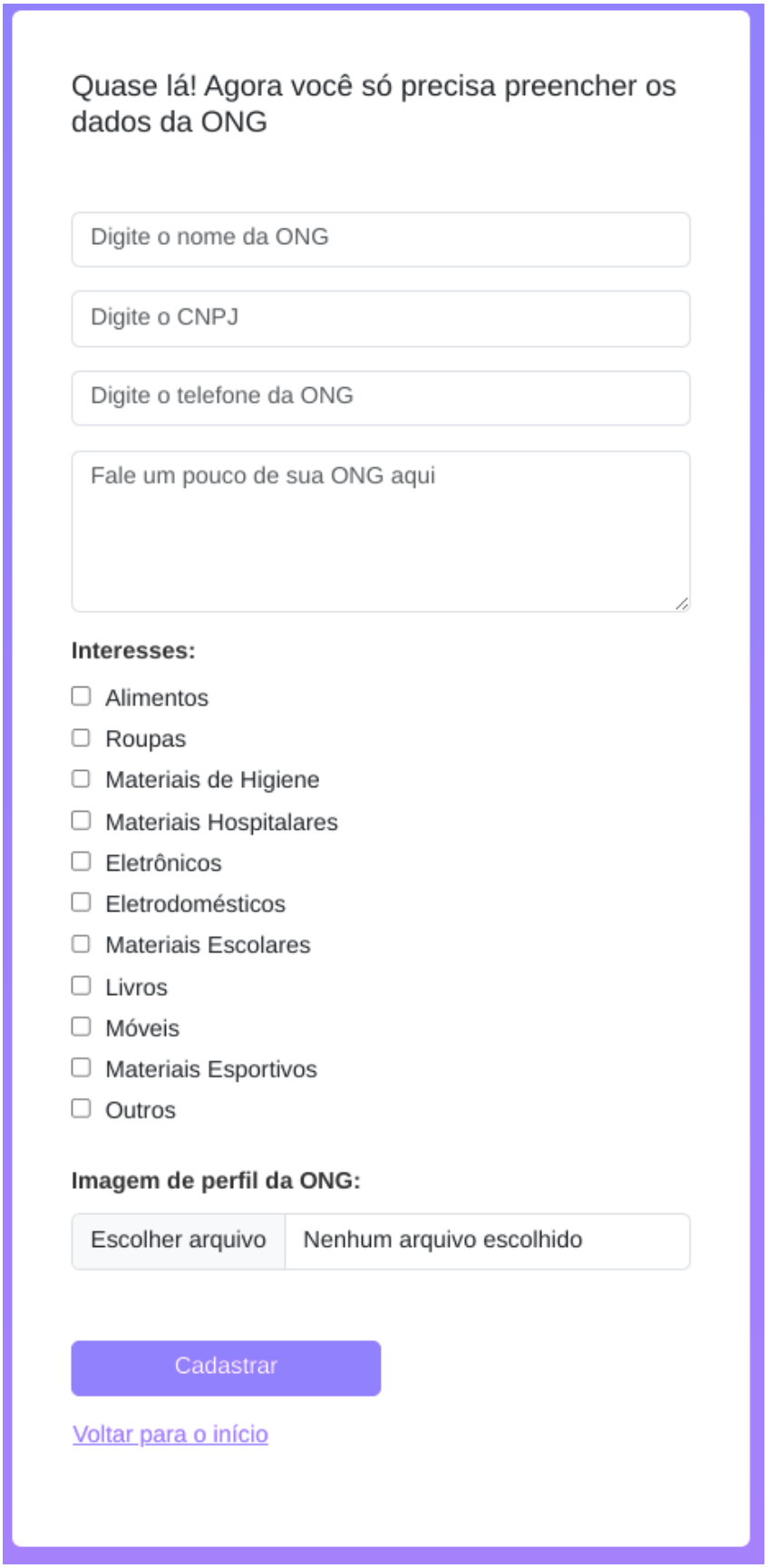
no MongoDB. Portanto, se o cadastro da ONG falhar por alguma intermitência, o usuário também não é criado.

Durante o registro da ONG, processo que ocorre na tela demonstrada pela Figura 19, o usuário pode anexar uma imagem utilizada para o perfil da organização. Essa imagem é transformada em Base64, um método de codificação que transforma dados binários, como imagens, em texto legível. Uma vez convertida, a imagem em formato de texto é incorporada ao documento correspondente no banco de dados MongoDB.

Antes de completar o cadastro, realiza-se a validação de dados específicos inseridos. Utilizando o pacote “cpf-cnpj-validator” do Node.js, o CNPJ passa por uma verificação de seus dígitos, conforme o algoritmo exigido para sua validação. Adicionalmente, verifica-se a existência prévia do CNPJ e do telefone da ONG no banco de dados; caso já estejam registrados, uma mensagem é exibida abaixo do campo de entrada correspondente.

Após a validação dos dados, o usuário e a ONG são persistidos e o usuário é redirecionado à página de Sucesso.

Figura 19 - Tela de Cadastro da ONG.



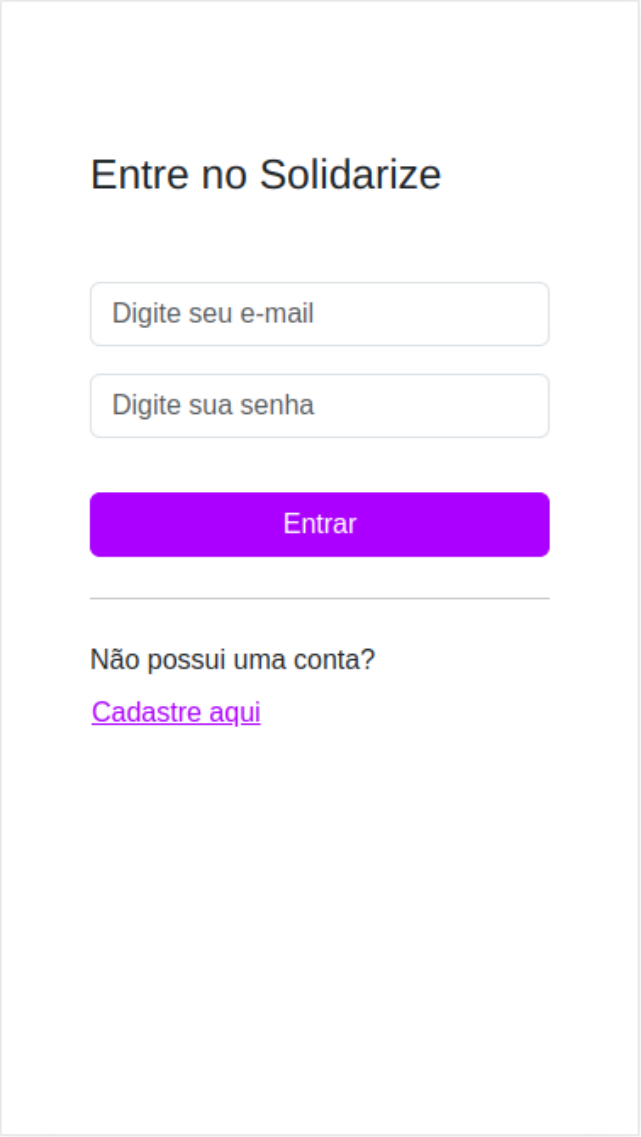
Fonte: Autoria própria.

5.5 Página de Login

Quando o usuário submete suas credenciais de login, Figura 20, o sistema faz uma solicitação à API para confirmar se o usuário está registrado. Uma vez confirmada a existência do usuário, inicia-se o processo de autenticação. Isso envolve o envio de uma requisição para a rota de login da API, que, por sua vez, responde com um *token* gerado com o JWT. Esse *token* é armazenado nos *cookies* do navegador utilizando o pacote “js-cookie” do Node.js. Com o *token* salvo, as informações do usuário obtidas na resposta da primeira chamada são armazenadas na *store* do Vuex, disponibilizando-as para uso no Dashboard de ações, cujos detalhes serão explicados posteriormente.

Com o *token* armazenado, a aplicação pode restringir o acesso a quaisquer rotas que requeiram segurança adicional. Especificamente, o *token* é usado para restringir o acesso ao *Dashboard*, que contém informações delicadas do usuário. Além disso, esse *token* é necessário para realizar solicitações a rotas protegidas no servidor. Existem arquivos específicos no *back-end* que gerenciam essas rotas seguras. Qualquer pedido a essas rotas protegidas é primeiro inspecionado por um *middleware*, que verifica a presença e a validade do *token* no cabeçalho da solicitação. Isso assegura uma camada adicional de proteção às páginas que contêm dados confidenciais dos usuários.

Figura 20 - Tela de Login.



Fonte: Autoria própria.

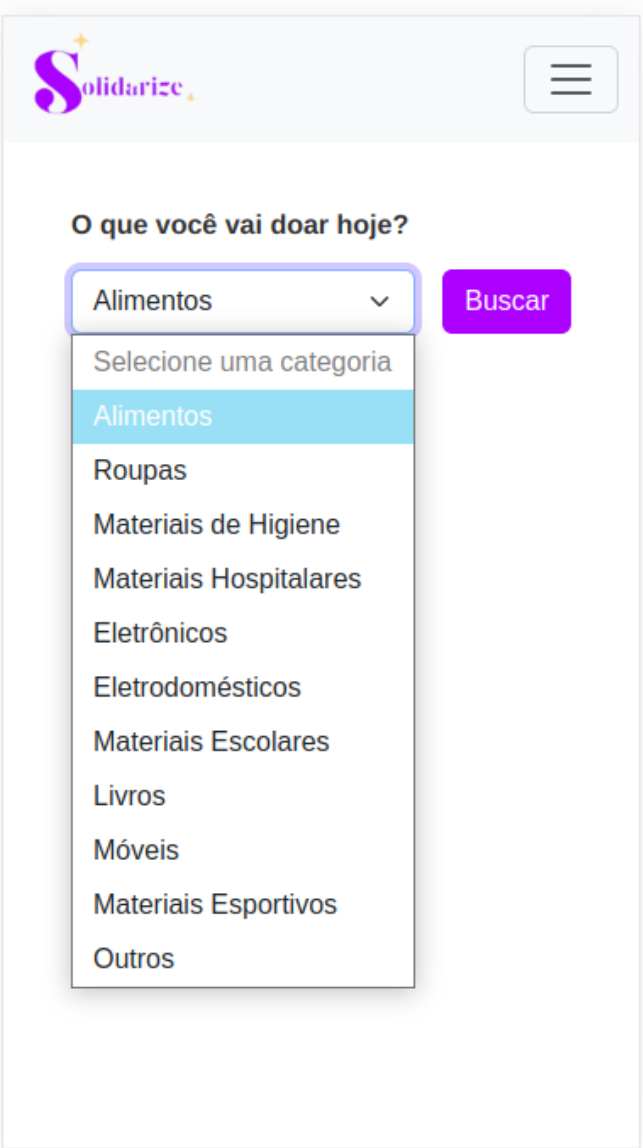
5.5 Dashboard

O *Dashboard* consiste numa página que engloba funcionalidades distintas e serve como a tela principal após o Login para todos os tipos de usuários. No caso de gestores, assim que o *Dashboard* é acessado, as informações do usuário logado são recuperadas na *store* para serem utilizadas nas páginas exclusivas de gestores.

5.5.1 Página Buscar ONG

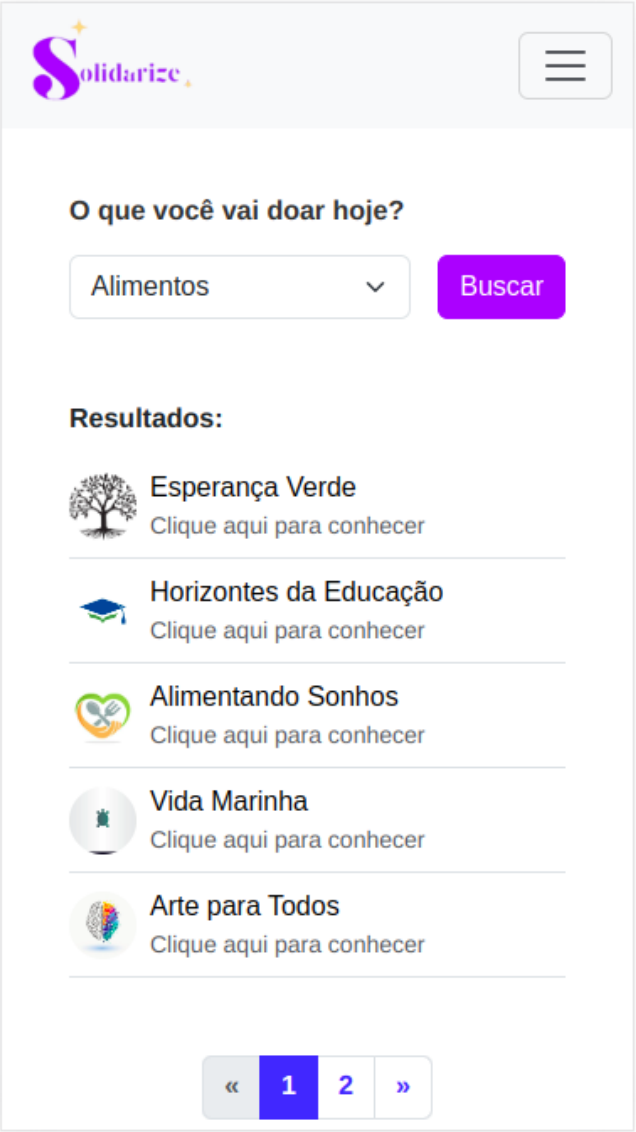
Para doadores, o *Dashboard* é aberto na página de “Buscar ONG”, ilustrada pela Figura 21. Visando a otimização da busca das organizações, foi utilizada a técnica de paginação, que seu visual está representado na Figura 22. A parte lógica foi implementada no *back-end*, que retorna um subconjunto das ONGs com base no interesse especificado e na paginação. A resposta inclui as ONGs da página solicitada, juntamente com informações sobre o total de páginas e ONGs disponíveis.

Figura 21 - Tela de Busca de ONG.



Fonte: Autoria própria.

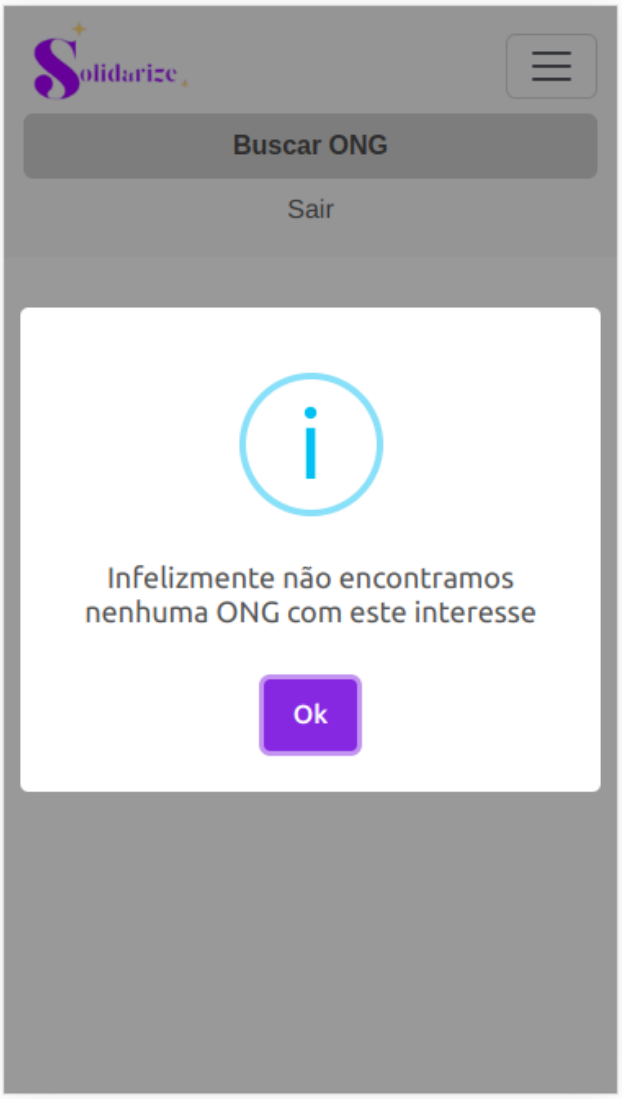
Figura 22 - Listagem de ONGs e paginação.



Fonte: Autoria própria.

Um detalhe que não foi comentado até aqui se trata da utilização de alertas para alguns erros específicos, principalmente aqueles envolvendo intermitências no serviço API. Para isso, foi utilizado o SweetAlert2, um pacote Node.js que possibilita utilizar componentes pré-definidos de alertas. No contexto da página de “Buscar ONG”, foi utilizado um alerta para notificar o usuário de que não há ONGs registradas com o interesse selecionado (Figura 23).

Figura 23 - Alerta padrão do SweetAlert2.

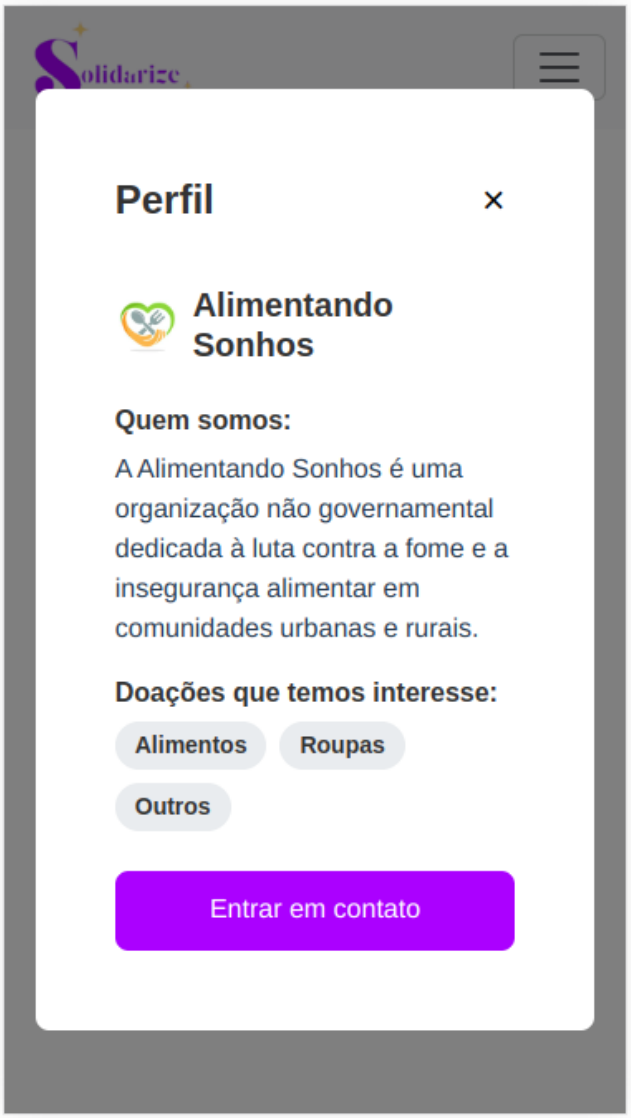


Fonte: Autoria própria.

5.5.2 Modal Perfil da ONG

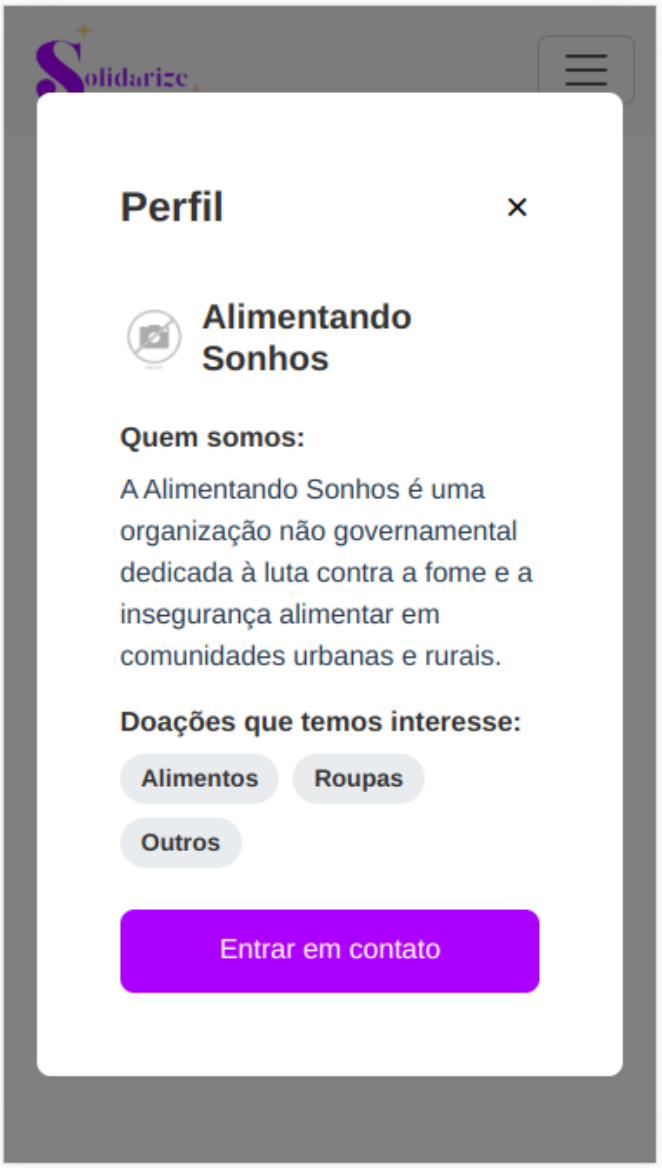
Na tela de sobreposição (modal) de perfil, Figura 24, a única particularidade não discutida é a não obrigatoriedade da imagem de perfil no momento do cadastro da ONG. Sendo assim, foi utilizada uma imagem padrão caso o usuário não cadastre uma imagem para sua ONG, como na Figura 25.

Figura 24 - Tela de sobreposição Perfil da ONG.



Fonte: Autoria própria.

Figura 25 - Imagem de perfil padrão.

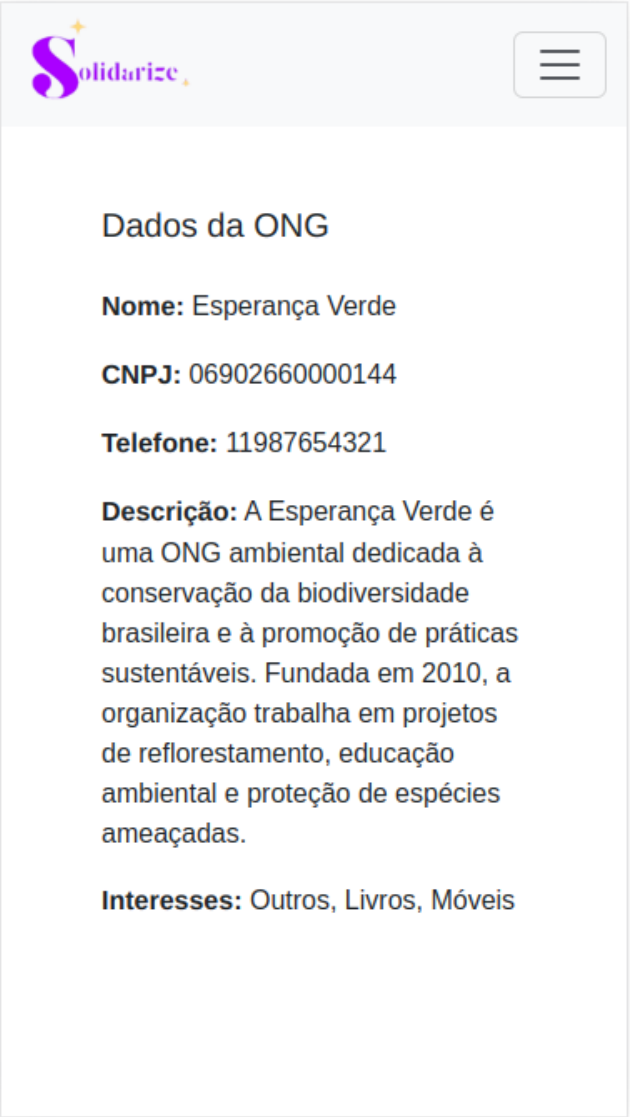


Fonte: Autoria própria.

5.5.3 Página de Dados da ONG

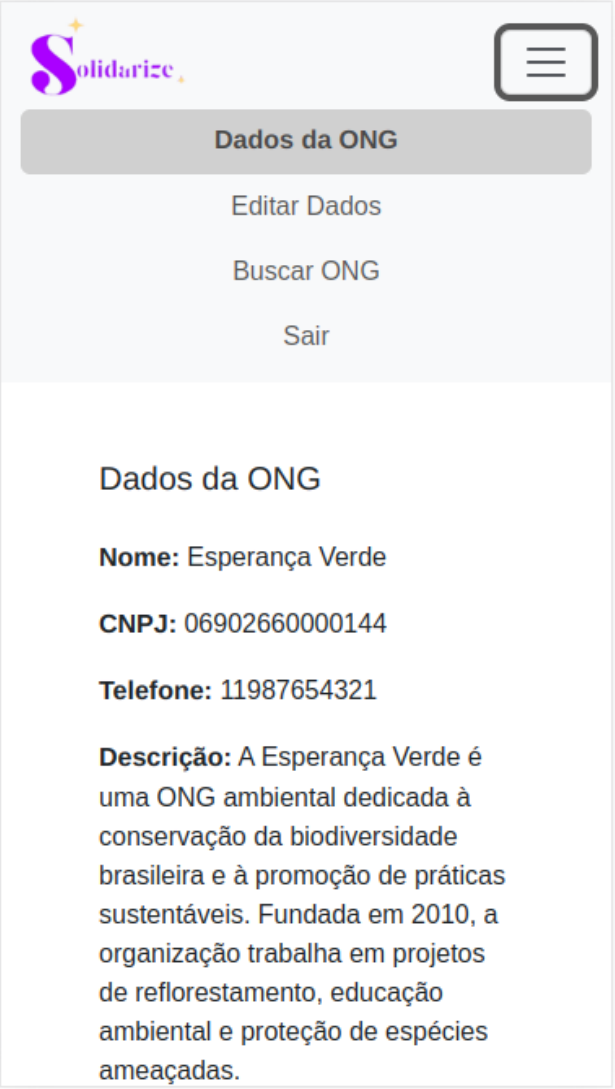
Para os usuários gestores, a tela de visualização de dados da própria ONG mostrada na Figura 26 é a primeira a ser renderizada; ela não possui nenhuma ação. A barra de navegador foi feita com um componente Bootstrap e ajustada conforme o desejado. Note na Figura 27 que na barra, há o botão “Sair”. Ele executa a desautenticação do usuário da mesma maneira que na página de Sucesso, que inclusive, utiliza o mesmo *controller* para fazer o *logout* do usuário.

Figura 26 - Tela de Dados da ONG.



Fonte: Autoria própria.

Figura 27 - Barra de navegação do Dashboard.



Fonte: Autoria própria.

5.5.4 Página de Editar Dados

Nesta página, conforme a Figura 28, os campos de entrada são preenchidos com os campos de usuário recuperados no início do *Dashboard*. Apesar da interface ser quase idêntica à página de cadastro de ONGs, esta utiliza uma rota diferente para fazer a atualização dos dados.

Figura 28 - Tela de Edição da ONG.

Fonte: Autoria própria.

6. CONSIDERAÇÕES FINAIS

O presente trabalho teve como objetivo criar uma plataforma que simplifica a interação entre doadores e gestores de ONGs. Sua fase inicial consistiu em um estudo aprofundado sobre as tecnologias Vue.js, Node.js, MongoDB e Express, as quais foram as tecnologias principais na implementação da aplicação.

Este aplicativo não só atende aos seus objetivos iniciais de criar uma aplicação de um acervo de ONGs, mas também estabelece uma nova alternativa para a interação entre doadores e ONGs, demonstrando ser uma possível ferramenta auxiliar no setor de doações.

Além disso, o projeto contribui para o campo acadêmico, fornecendo uma visão sobre o desenvolvimento web e a integração de várias tecnologias na criação de um aplicativo social. O conhecimento adquirido e documentado neste trabalho pode servir como um recurso útil para futuros desenvolvedores e pesquisadores na área.

Em conclusão, o "Solidarize" é um exemplo de como a tecnologia pode ser utilizada para promover a solidariedade e o engajamento social. Enquanto o projeto atual atingiu seus objetivos, há espaço para expansão e melhorias futuras, com a implementação de recursos adicionais. Uma proposta válida seria a implementação de um chat dentro do aplicativo, ao invés de depender de aplicativos de terceiros. Outra funcionalidade poderia ser um *feed* em que cada ONG pudesse fazer *posts* que usuários pudessem interagir.

REFERÊNCIAS

ABONG. **Associação Brasileira de Organizações Não Governamentais**. Brasil 2014. Disponível em: <www.abong.org.br>

ALVIM, F., TEODÓSIO, A. S. S. **Gestão da Cooperação Internacional: perspectivas e desafios para as ONGs** In: Anais do XV Encontro Nacional dos Cursos de Graduação em Administração. Florianópolis: ANGRAD, 2004. P. 93 – 102.

Balsamiq. **Balsamiq Wireframes**. Disponível em: https://balsamiq.com/wireframes/. Acesso em: 25 nov. 2023.

Bootstrap. **Bootstrap**. Disponível em: https://getbootstrap.com/. Acesso em: 25 nov. 2023.

BROWN, Ethan. **Web Development with Node and Express: Leveraging the JavaScript Stack**. 2ª ed. O'Reilly Media, 2019.

CHODOROW, Kristina. **MongoDB: The Definitive Guide**, 3rd Edition. O'Reilly Media, Inc., 2019.

FOWLER, Martin. **UnitTest**. Disponível em: https://martinfowler.com/bliki/UnitTest.html. Acesso em: 25 nov. 2023.

GOHN, Maria da G. **Os sem-terra, ONGs e cidadania**. 2 ed. São Paulo: Cortez, 2000.

HANCHETT, Erik; LISTWON, Ben. **Vue.js in Action**. Manning Publications, 2018.

JavaTpoint. **Jest Framework**. Disponível em: https://www.javatpoint.com/jest-framework. Acesso em: 25 nov. 2023.

JavaTpoint. **Postman**. Disponível em: https://www.javatpoint.com/postman. Acesso em: 25 nov. 2023.

LANDIM, Leilah. **A invenção das ONGs: do serviço invisível à profissão impossível**. Tese de doutorado. Universidade Federal do Rio de Janeiro. Programa de Pós-graduação em Antropologia Social. Rio de Janeiro, 1993.

MARTONE, L. M. C.; GIL, A. C. **Desafios à gestão de ONGs: OSCIPS do grande ABC**. In: Anais… III SEGET, Rio de Janeiro, 2006.

MongoDB. **MongoDB Atlas Tutorial**. Disponível em: https://www.mongodb.com/basics/mongodb-atlas-tutorial. Acesso em: 25 nov. 2023.

MORESI, E.A., GODINHO, S.G., MARIZ, R.S., FILHO, M.D., BARBOSA, J.A., LOPES, M.C., JÚNIOR, W.A., & MORAIS, M.A. **Tecnologia Social: a doação na perspectiva do aplicativo Solidarius**. RISTI: Revista Ibérica de Sistemas e Tecnologias de Informação, 1-16, 2017.

Oracle. **Node.js: The Basics**. Oracle Developer. Disponível em: https://developer.oracle.com/learn/technical-articles/what-is-node-js. Acesso em: 25 nov. 2023.

Visual Studio Code. **Why Visual Studio Code?**. Disponível em: https://code.visualstudio.com/docs/editor/whyvscode. Acesso em: 25 nov. 2023.

WU, H., ZHU X., **Developing a Reliable Service System of Charity Donation During the 49 Covid-19 Outbreak**. IEEE Access, vol. 8, pp. 154848-154860, 2020.