

Documentação: Trabalho Prático
Desenvolvimento Web 2023/01
Sistema de Aviso de Chuva

1. Introdução

Em meio a desafios climáticos, como chuvas intensas, surge a necessidade de criar soluções inovadoras que possam auxiliar comunidades vulneráveis de forma rápida e eficiente. Nesse contexto, apresentamos a proposta de uma aplicação web dedicada à notificação e coordenação de doações e necessidades de ajuda durante eventos climáticos adversos, como chuvas intensas na cidade de Apucarana.

Desastres naturais, como fortes chuvas, frequentemente desencadeiam situações de emergência, demandando uma resposta ágil e coordenada para atender às necessidades das comunidades afetadas. A falta de uma plataforma específica para gerenciar e canalizar doações, juntamente com a ausência de alertas personalizados, contribui para a complexidade dessas situações, limitando a eficácia da ajuda prestada.

Este projeto tem como principal objetivo fornecer uma solução abrangente para a notificação de chuvas intensas, visando melhorar a gestão de situações climáticas adversas em Apucarana. A ausência de um sistema especializado para tal notificação pode levar a desafios significativos, como falta de preparação, resposta inadequada e dificuldades na coordenação de ações durante eventos climáticos extremos. Buscamos, assim, mitigar esses problemas, promovendo uma abordagem centralizada e eficiente para alertas de chuva na cidade.

A aplicação conta com funcionalidades essenciais para a gestão eficaz de situações climáticas adversas. Através da integração com uma API externa de dados meteorológicos, a aplicação é capaz de avaliar as condições climáticas em tempo real. Essa funcionalidade possibilita a identificação de chuvas críticas, desencadeando alertas personalizados para os usuários, promovendo assim uma resposta rápida diante de eventos meteorológicos intensos.

Com o intuito de garantir a segurança e organização das informações, foi implementado um sistema de login restrito para administradores. Este acesso exclusivo permite a administração eficiente das doações e pedidos de ajuda, proporcionando uma visão centralizada e autenticada das operações realizadas na plataforma.

O banco de dados da aplicação é estruturado de forma a facilitar a incorporação de novas funcionalidades no futuro. Essa abordagem visa garantir a adaptabilidade e expansibilidade da aplicação, possibilitando a integração de recursos adicionais conforme as necessidades evoluam, sem comprometer a integridade do sistema.

Essas funcionalidades fundamentais, combinadas, visam não apenas fornecer alertas precisos sobre condições meteorológicas críticas, mas também estabelecer uma base sólida para a gestão eficiente de doações e pedidos de ajuda durante eventos climáticos adversos, promovendo assim uma resposta rápida e coordenada da comunidade.

Atualmente existem alguns softwares parecidos com a nossa aplicação. Dentre os sistemas existentes, destacam-se soluções como "Community Aid" e "Emergency Relief Hub". No entanto, esses softwares muitas vezes carecem de uma abordagem holística, não integrando efetivamente alertas meteorológicos com o gerenciamento de doações.

Nossa aplicação, em contrapartida, se destaca pela abordagem integrada, conectando alertas em tempo real com um sistema eficiente de coordenação de doações. O design modular permite expansibilidade fácil para enfrentar diferentes tipos de emergências, sem sacrificar a simplicidade de uso. Além disso, a administração centralizada promove a legitimidade das doações, construindo confiança entre doadores, necessitados e organizações locais de ajuda.

Alguns objetivos específicos estão elencados abaixo:

Implementar a Integração da API Meteorológica: Garantir a correta integração com a API externa de dados meteorológicos para o cálculo preciso de chuvas críticas.

Desenvolver Módulo de Alerta: Criar um sistema de alerta que seja rápido, confiável e forneça informações claras aos usuários em caso de chuvas críticas, incentivando a tomada de ações preventivas.

Configurar Sistema de Login Administrativo: Estabelecer um sistema seguro de login administrativo, assegurando que apenas usuários autorizados possam acessar e gerenciar informações relacionadas a doações e pedidos de ajuda.

Projetar Banco de Dados Flexível: Desenvolver um banco de dados que permita fácil expansão para futuras funcionalidades, mantendo a consistência e integridade das informações existentes.

Criar Páginas de Interface Amigável: Projetar e implementar páginas de interface do usuário intuitivas e amigáveis, promovendo uma experiência positiva para doadores, necessitados e administradores.

Estabelecer Sistema de Coordenação de Doações: Criar um sistema eficiente que permita a coordenação transparente entre doadores e necessitados, garantindo a autenticidade e distribuição adequada das doações.

Realizar Testes de Usabilidade: Conduzir testes de usabilidade para identificar e corrigir possíveis pontos de dificuldade na interação dos usuários com a aplicação.

Implementar Recursos de Segurança: Reforçar medidas de segurança, como criptografia de dados e validação adequada, para proteger a integridade das informações dos usuários.

Estabelecer Comunicação com Organizações Locais: Facilitar a comunicação e cooperação com organizações locais de ajuda, incentivando sua participação na plataforma para uma resposta mais abrangente e eficaz.

Monitorar e Atualizar Regularmente: Implementar um sistema de monitoramento contínuo para garantir o bom funcionamento da aplicação e realizar atualizações regulares para aprimorar e corrigir eventuais falhas.

2. Escopo do Produto de Software

Definir um escopo é essencial para estabelecer uma base clara e comum de entendimento entre a equipe de desenvolvimento, os clientes e outros interessados no projeto. O escopo

deste projeto contempla o desenvolvimento de uma App Web, pensando principalmente em cidadãos e trazendo um sistema de aviso de chuvas fortes que melhore a organização e preparação dos mesmos para conseguir passar por essas questões da natureza. Algumas funcionalidades e seus usuários serão melhor descritas abaixo.

2.1. Usuários e Principais Usuários

A aplicação em questão atende a diferentes tipos de atores, cada um com papéis específicos e necessidades distintas. Abaixo temos todos os atores da aplicação.

Usuário Comum: Representa o cidadão médio que acessa a aplicação para obter informações sobre as condições meteorológicas, alertas de chuvas intensas e procedimentos de segurança. Além disso, pode participar ativamente, oferecendo doações ou solicitando ajuda em caso de necessidade.

Administrador da Aplicação: São os responsáveis pela gestão e manutenção da aplicação. Têm acesso a funcionalidades avançadas, como verificação e moderação de pedidos de ajuda e doações. Podem realizar atualizações de emergência nas informações meteorológicas.

Organizações Locais de Ajuda: Representantes de organizações locais, como ONGs e serviços de emergência, que colaboram com a aplicação para oferecer assistência direta às pessoas afetadas. Participam ativamente na coordenação de doações e na prestação de auxílio direto.

Parceiros de Logística: Empresas ou organizações envolvidas na logística e distribuição de doações. Utilizam a aplicação para coordenar pontos de coleta e entrega de itens doados.

Desenvolvedores: Profissionais responsáveis pela manutenção e aprimoramento contínuo da aplicação. Podem ser necessários para integrar novos recursos, corrigir problemas e realizar atualizações de segurança.

Serviços Meteorológicos: Fornecedores de serviços meteorológicos cujos dados são integrados à aplicação. Esses dados são essenciais para calcular se as condições meteorológicas representam uma chuva crítica ou não.

Os principais usuários são os Usuários Comuns, pois representam a base da comunidade que utiliza a aplicação para se manter informada, tomar ações preventivas e participar ativamente em situações de emergência.

2.1. Principais Funcionalidades

Como funcionalidade principal temos a notificação de chuva crítica no qual permitirá que os usuários consigam se programar antecipadamente e sofrer menos com possíveis desastres. Claro que fora dessa funcionalidade principal temos outras.

Cálculo de Chuva Crítica: Utilização de uma API externa para avaliar as condições meteorológicas em tempo real e determinar se a chuva atinge níveis críticos. Isso permite a

emissão de alertas personalizados para os usuários, contribuindo para a preparação e segurança da comunidade.

Login Administrativo: Implementação de um sistema de login restrito para administradores, assegurando o acesso exclusivo a funcionalidades avançadas. Os administradores têm a responsabilidade de organizar e verificar doações, bem como gerenciar pedidos de ajuda.

Facilidade de Integração:

Estruturação do banco de dados de maneira flexível para facilitar a integração de novas funcionalidades. Isso permite que o sistema seja expandido e adaptado conforme as necessidades evoluem, sem prejudicar a integridade dos dados existentes.

Páginas de Interface Amigável: Desenvolvimento de páginas de interface do usuário intuitivas e amigáveis, seguindo um padrão visual agradável. Isso promove uma experiência positiva para os usuários, tornando a interação com a aplicação mais eficiente.

Coordenação de Doações: Criação de um sistema eficiente que permite a coordenação transparente entre doadores e aqueles que necessitam de ajuda. Essa funcionalidade visa garantir a autenticidade das doações e uma distribuição adequada dos recursos.

Alerta em Tempo Real: Implementação de um sistema de alerta em tempo real que fornece informações imediatas sobre condições meteorológicas críticas. Essa funcionalidade é essencial para manter os usuários informados e tomar ações preventivas diante de situações de emergência.

Comunicação com Organizações Locais: Facilitação da comunicação e cooperação com organizações locais de ajuda. Isso incentiva a participação ativa dessas organizações na plataforma, fortalecendo a resposta comunitária durante eventos climáticos adversos.

Sistema de Monitoramento e Atualização: Implementação de um sistema de monitoramento contínuo para garantir o funcionamento adequado da aplicação. Além disso, a realização de atualizações regulares visa aprimorar a aplicação, corrigir falhas e integrar melhorias conforme necessário.

3. Processo de Desenvolvimento

Dentro de um processo de desenvolvimento temos várias etapas a serem seguidas, sendo elas as histórias dos usuários e levantamento de requisitos no qual montamos as histórias de acordo com a necessidade dos usuários, o projeto de software definindo arquitetura, modelagem e interfaces, implementação do projeto no qual é o desenvolvendo em si do software, testes e verificações no qual seria todas as validações para poder finalmente colocar o software para funcionar e claro a manutenção que é uma parte após a app já está rodando na Web nesta parte corrigimos e melhoramos o software, abaixo nesse documento citamos como cada etapa funcionou. Para uma melhor organização do projeto, todas as etapas foram descritas e suas atividades disponibilizadas na plataforma Trello.

Uma das principais vantagens do Trello é a sua interface intuitiva e flexível, que permite personalizar o quadro e as listas de acordo com as necessidades de cada projeto ou equipe. Além disso, o Trello é altamente colaborativo, permitindo que várias pessoas

trabalhem no mesmo quadro, atribuam cartões a membros da equipe, adicione comentários e realizem discussões sobre as tarefas.

Para intensificar ainda mais a otimização de tempo e produtividade do grupo, todas as etapas foram realizadas utilizando a metodologia ágil Scrum, sem nenhuma alteração ou modificação seguindo todos os conceitos e regras dessa metodologia. O Scrum promove transparência, colaboração e adaptação, permitindo que a equipe responda a mudanças de requisitos e prioridades de forma ágil.

A metodologia ágil Scrum é um framework de gerenciamento de projetos que visa facilitar a entrega de produtos de forma iterativa e incremental. Ele é baseado em uma abordagem colaborativa e adaptativa, onde equipes auto-organizadas trabalham em sprints (iterações) para entregar valor de forma contínua. Alguns pontos-chaves dessa metodologia são:

- O backlog do produto: É uma lista priorizada de requisitos, funcionalidades e itens que precisam ser desenvolvidos para o produto. É mantido e gerenciado pelo Product Owner.
- O sprint backlog: É uma lista de tarefas ou itens selecionados do backlog do produto para serem desenvolvidos durante o sprint. A equipe de desenvolvimento estima o esforço necessário para realizar essas tarefas e as organiza de acordo com sua capacidade.
- O sprint: É um período de tempo fixo (tipicamente de 1 a 4 semanas) durante o qual a equipe trabalha para entregar um conjunto de funcionalidades. Antes de cada sprint, a equipe realiza uma reunião de planejamento do sprint para selecionar itens do backlog do produto que serão incluídos no sprint e definir os objetivos do sprint. Ao final de cada sprint, a equipe deve ter um produto funcional e potencialmente entregável. Isso significa que o incremento deve estar em um estado que possa ser apresentado e utilizado pelo cliente ou usuário final.
No final de cada sprint, a equipe realiza uma reunião de revisão do sprint para apresentar o incremento desenvolvido durante o sprint ao Product Owner e a outros stakeholders relevantes. Feedback é coletado e o backlog do produto é atualizado de acordo.
Também no final de cada sprint, a equipe realiza uma retrospectiva do sprint para analisar o trabalho realizado, identificar pontos fortes e áreas de melhoria, e definir ações para aprimorar o processo no próximo sprint.
- Daily Scrum: É uma reunião diária curta (geralmente de 15 minutos) onde a equipe se reúne para sincronizar o trabalho. Cada membro compartilha o que fez desde a última reunião, o que planeja fazer até a próxima reunião e quaisquer impedimentos ou obstáculos que estejam enfrentando.

Abaixo temos uma representação gráfica do board utilizado contendo as tarefas.

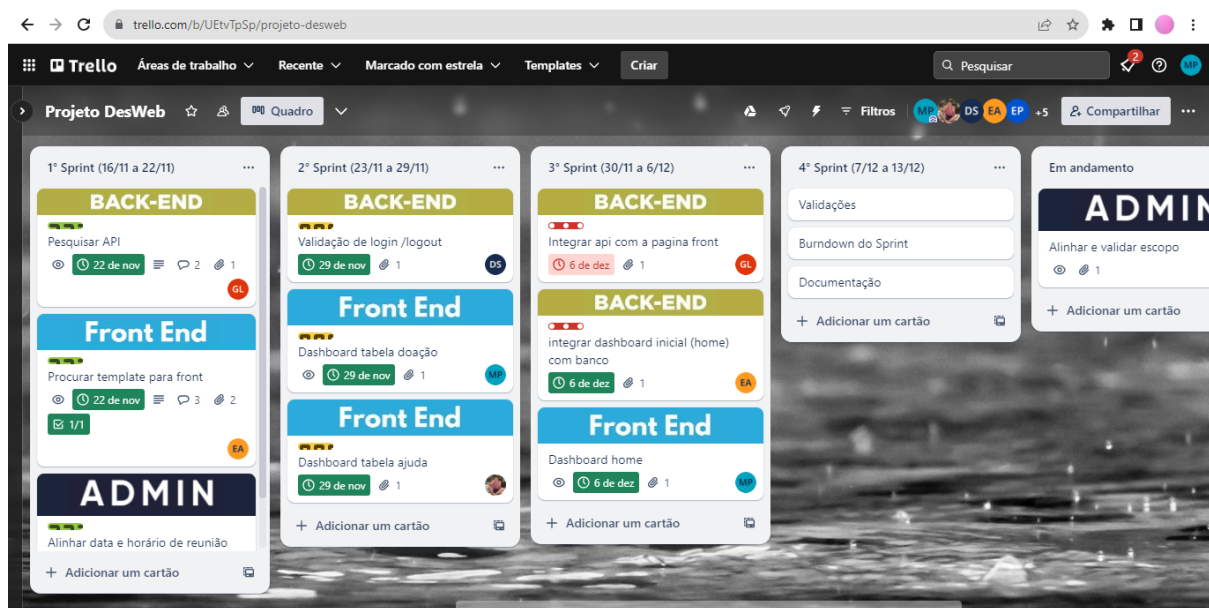


Figura 1. Exemplo de aplicação do Trello. Autoria própria (2023).

4. Engenharia de Requisitos

A engenharia de requisitos desempenha um papel crucial no desenvolvimento de sistemas de alta qualidade, pois ajuda a evitar problemas de compreensão e expectativas dos stakeholders, reduzindo os riscos de falhas no sistema e melhorando a comunicação entre os envolvidos no projeto.

Os requisitos são as especificações que descrevem o que o sistema deve fazer, suas funcionalidades, restrições e características desejadas. Esses requisitos foram coletados através de um processo sistemático para coletar informações sobre as necessidades e expectativas dos usuários e todos os atores envolvidos no software, os requisitos precisam ser documentados e validados corretamente para assim serem compreendidos da melhor maneira.

4.1. Histórias de Usuário (User Stories): Requisitos Funcionais

Para elencar os requisitos funcionais, primeiramente especificamos as histórias do usuário. A ideia por trás das histórias do usuário é fornecer uma descrição compreensível e orientada para o usuário das funcionalidades desejadas, em vez de documentos extensos e detalhados de requisitos. Elas são escritas em linguagem simples e podem ser facilmente compreendidas por desenvolvedores, testadores e outros membros da equipe de projeto. Abaixo temos as histórias desse projeto.

Como usuário comum, eu quero receber alertas imediatos sobre chuvas críticas para que eu possa tomar medidas preventivas e me manter seguro durante eventos climáticos adversos.

Como administrador da aplicação, eu quero ter acesso restrito para organizar e verificar doações e pedidos de ajuda, garantindo a autenticidade das informações e a eficiência na distribuição de recursos.

Como usuário comum, eu quero poder se cadastrar na aplicação para oferecer doações, para que eu possa contribuir ativamente durante situações de emergência e ajudar aqueles que necessitam.

Como organização local de ajuda, eu quero ser capaz de colaborar diretamente com a aplicação para oferecer assistência às pessoas afetadas, coordenando esforços de auxílio e participando da distribuição de recursos.

Como usuário comum, eu quero visualizar informações claras e atualizadas sobre a quantidade de doações disponíveis e as necessidades da comunidade, para que eu possa tomar decisões informadas sobre como contribuir.

Como parceiro de logística, eu quero utilizar a aplicação para coordenar pontos de coleta e entrega de doações, otimizando a logística e garantindo uma distribuição eficiente.

Como desenvolvedor, eu quero ser capaz de integrar novas funcionalidades na aplicação de forma fácil, garantindo que a plataforma possa evoluir para atender a diferentes cenários de emergência e demandas futuras.

Como usuário comum, eu quero poder solicitar ajuda através da aplicação, fornecendo informações sobre as minhas necessidades, para que eu possa receber assistência adequada durante eventos climáticos adversos.

Como administrador da aplicação, eu quero poder realizar atualizações de emergência nas informações meteorológicas, para que os alertas sejam precisos e reflitam com precisão as condições climáticas em tempo real.

Como usuário comum, eu quero que a aplicação tenha uma interface intuitiva e amigável, para que eu possa navegar facilmente, contribuir com doações ou solicitar ajuda de maneira eficiente.

E através dessas histórias conseguimos elencar os requisitos funcionais da nossa aplicação, com elas podemos ver quem terá o privilégio, como será feito e o porquê do requisito existir.

- RF1 - Alertas de Chuvas Críticas: O sistema deve ser capaz de fornecer alertas imediatos aos usuários comuns sobre condições meteorológicas críticas, utilizando dados de uma API externa.
- RF2 - Login Administrativo: Deve existir um sistema de login para administradores, garantindo acesso restrito a funcionalidades avançadas, como organização e verificação de doações e pedidos de ajuda.
- RF3 - Cadastro de Usuários: Os usuários comuns e administradores devem ser capazes de se cadastrar na aplicação, fornecendo informações básicas para participar da plataforma.

- RF4 - Coordenação de Doações: A aplicação deve permitir a coordenação eficiente entre doadores e necessitados, garantindo transparência, autenticidade e distribuição adequada dos recursos.
- RF5 - Informações Atualizadas: Deve haver uma exibição clara e atualizada das informações sobre a quantidade de doações disponíveis, necessidades da comunidade e quantas pessoas já foram ajudadas.
- RF6 - Colaboração com Organizações Locais: A aplicação deve facilitar a comunicação e cooperação entre a plataforma e organizações locais de ajuda para uma resposta mais abrangente.
- RF7 - Logística de Doações: Parceiros de logística devem ser capazes de utilizar a aplicação para coordenar pontos de coleta e entrega de doações, otimizando a distribuição.
- RF8 - Integração de Funcionalidades Futuras: O banco de dados deve ser projetado para permitir a fácil integração de novas funcionalidades, assegurando a escalabilidade da aplicação.
- RF9 - Solicitação de Ajuda: Usuários comuns devem ter a capacidade de solicitar ajuda através da aplicação, fornecendo informações sobre suas necessidades durante eventos climáticos adversos.
- RF10 - Interface Intuitiva: A interface do usuário deve ser intuitiva e amigável, proporcionando uma experiência positiva para todos os usuários.
- RF11 - Atualizações de Emergência: Administradores devem ser capazes de realizar atualizações de emergência nas informações meteorológicas para garantir alertas precisos.

4.2. Requisitos Não-Funcionais

Os requisitos não-funcionais são atributos ou características do sistema que não estão diretamente relacionados às funcionalidades específicas, mas impactam sua qualidade, desempenho, segurança e usabilidade. No caso do nosso software de notificação de chuva, teremos alguns requisitos no qual vamos nos atentar elencados abaixo.

- RNF1 - Desempenho: A aplicação deve ser capaz de fornecer alertas em tempo real e responder rapidamente mesmo em situações de tráfego intenso, garantindo eficiência na comunicação de eventos críticos.
- RNF2 - Segurança: O sistema deve adotar medidas de segurança, como criptografia de dados, para proteger as informações dos usuários e garantir a integridade das transações, evitando acesso não autorizado.
- RNF3 - Escalabilidade: A aplicação deve ser projetada para lidar com um aumento significativo no número de usuários, doadores e necessitados, mantendo a eficiência e a qualidade do serviço durante eventos climáticos extremos.

- RNF4 - Disponibilidade: A plataforma deve estar disponível de forma contínua, mesmo em situações de alta demanda, para garantir que os usuários possam acessar informações cruciais durante eventos meteorológicos adversos.
- RNF5 - Usabilidade: A interface do usuário deve ser intuitiva e fácil de usar, promovendo a participação ativa de usuários comuns, doadores e administradores, independentemente de sua familiaridade com a tecnologia.
- RNF6 - Manutenibilidade: O sistema deve ser facilmente mantido e atualizado, permitindo que desenvolvedores realizem correções, atualizações e integrem novas funcionalidades de maneira eficiente.
- RNF7 - Confiabilidade: A aplicação deve ser confiável, garantindo que os alertas meteorológicos sejam precisos e que as transações entre doadores e necessitados ocorram de maneira confiável e segura.
- RNF8 - Compatibilidade: A aplicação deve ser compatível com diferentes dispositivos e navegadores, garantindo que os usuários possam acessar a plataforma independentemente do dispositivo que estão utilizando.
- RNF9 - Integração com API Meteorológica: A integração com a API externa de dados meteorológicos deve ser eficiente e estável, assegurando que as informações sobre chuvas críticas sejam precisas e atualizadas em tempo real.
- RNF10 - Documentação: Deve haver documentação clara e abrangente para desenvolvedores, administradores e usuários finais, facilitando o entendimento do sistema e seu correto uso.

5. Interface Gráfica do Usuário (User Interface / UI)

5.1. Protótipos Visuais e Funcionais

Os protótipos funcionais são representações interativas e testáveis de um produto ou sistema em desenvolvimento. Eles desempenham um papel crucial no processo de design e desenvolvimento de software, permitindo que os designers, desenvolvedores e partes interessadas (stakeholders) visualizem e interajam com as funcionalidades propostas antes da implementação final, conseguindo assim ter a capacidade de simular o comportamento real do sistema. Eles podem ser criados usando uma variedade de ferramentas, como ferramentas de design de interface do usuário (UI), frameworks de desenvolvimento front-end ou plataformas específicas de prototipagem.

No nosso caso, a modelagem foi realizada por meio de desenhos manuais, fornecendo uma base para a realização da interface gráfica do FrontEnd. Os protótipos mostrados abaixo juntamente com sua legenda são do nosso MVP (mínimo produto viável) em um nível de baixa fidelidade, ou seja uma representação básica das funcionalidades.

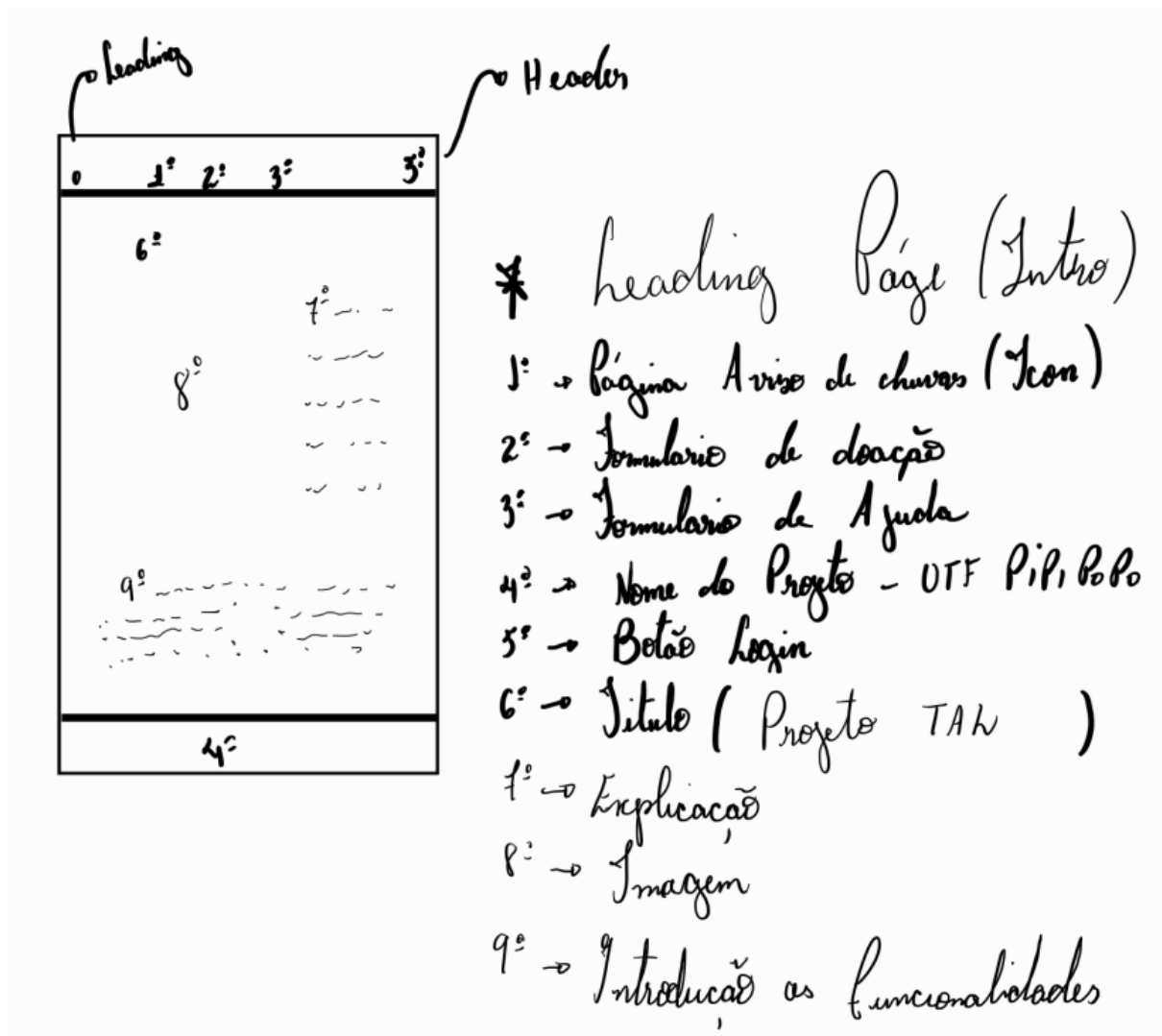


Figura 2. Esquema para a página inicial. Autoria própria (2023).

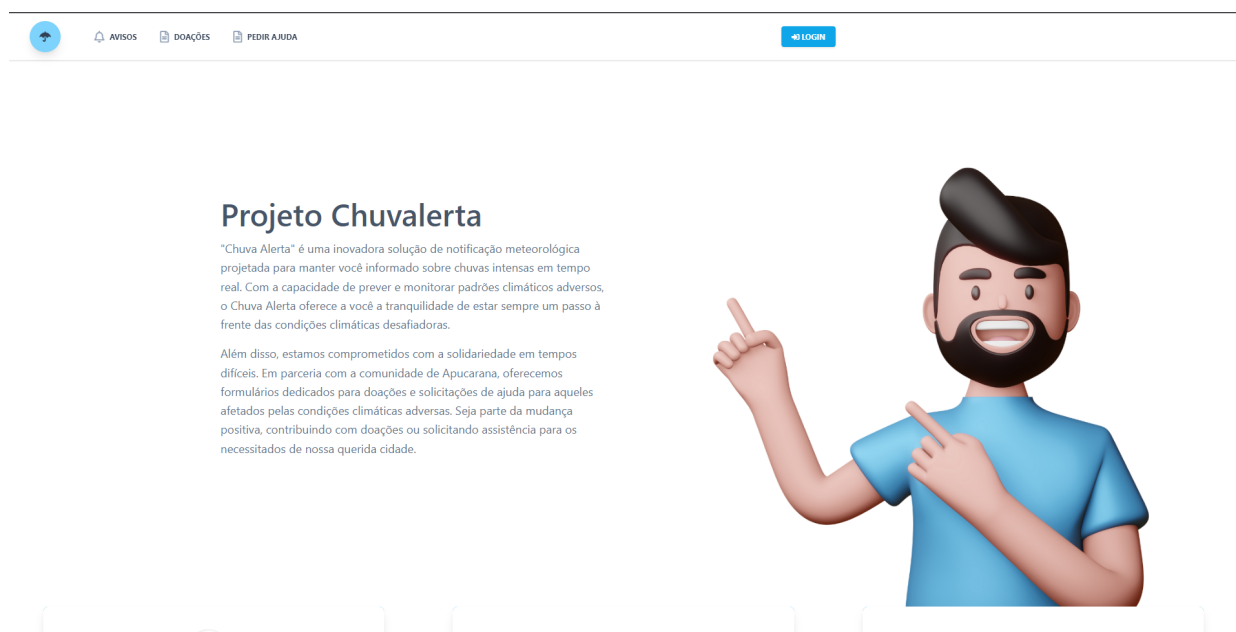


Figura 3. Página inicial - topo. Autoria própria (2023).

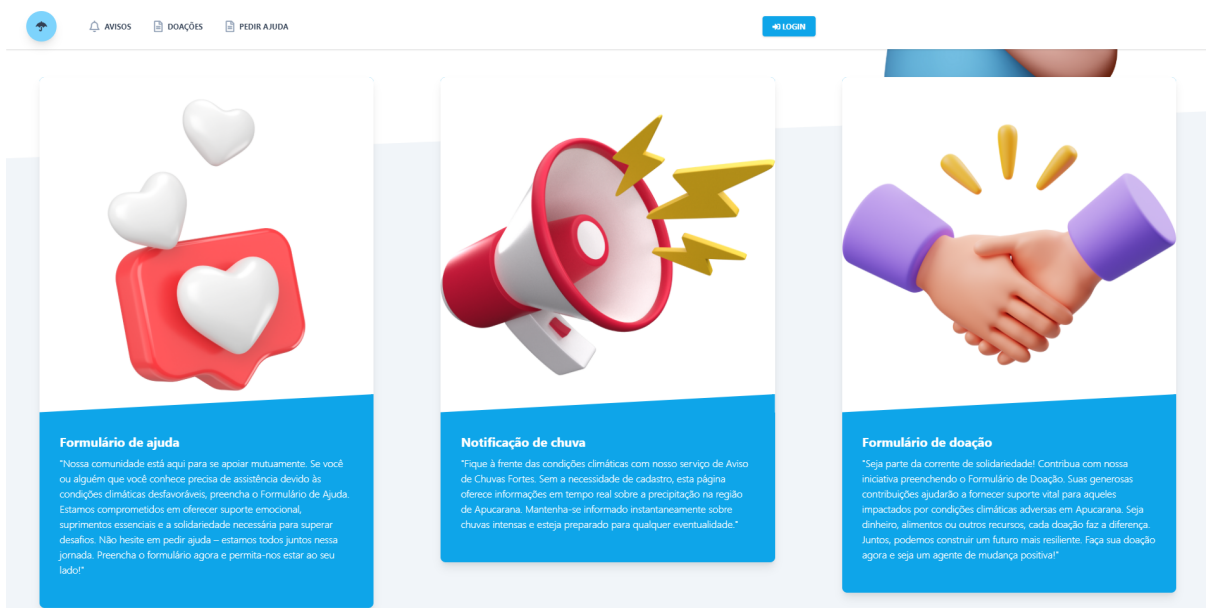


Figura 4. Página inicial - meio. Autoria própria (2023).

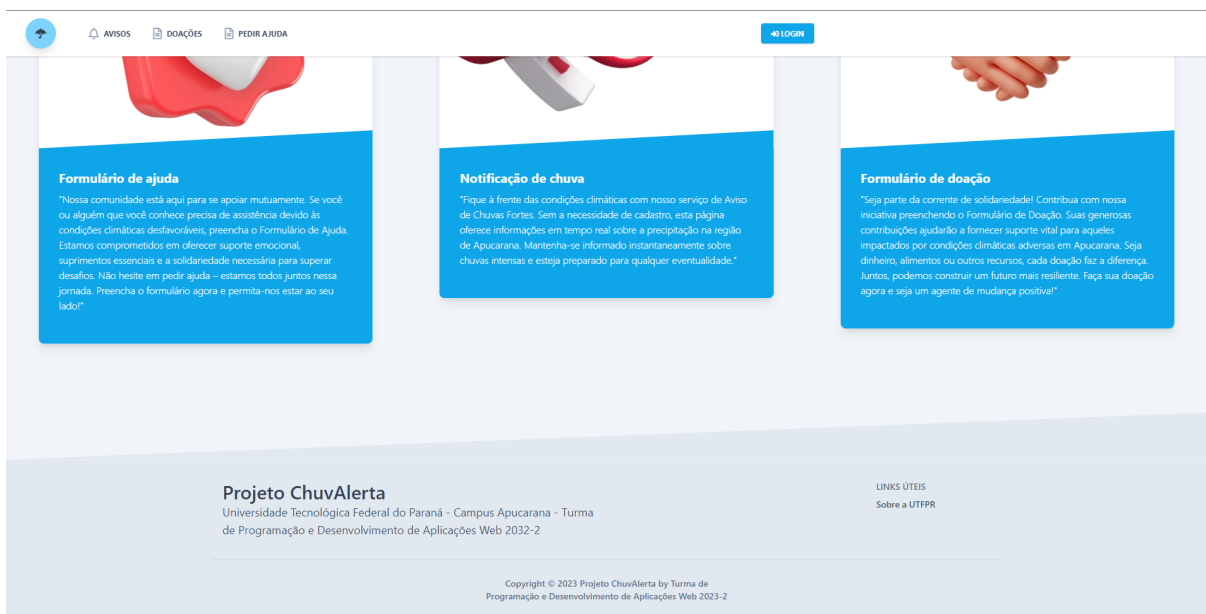


Figura 5. Página inicial - rodapé. Autoria própria (2023).

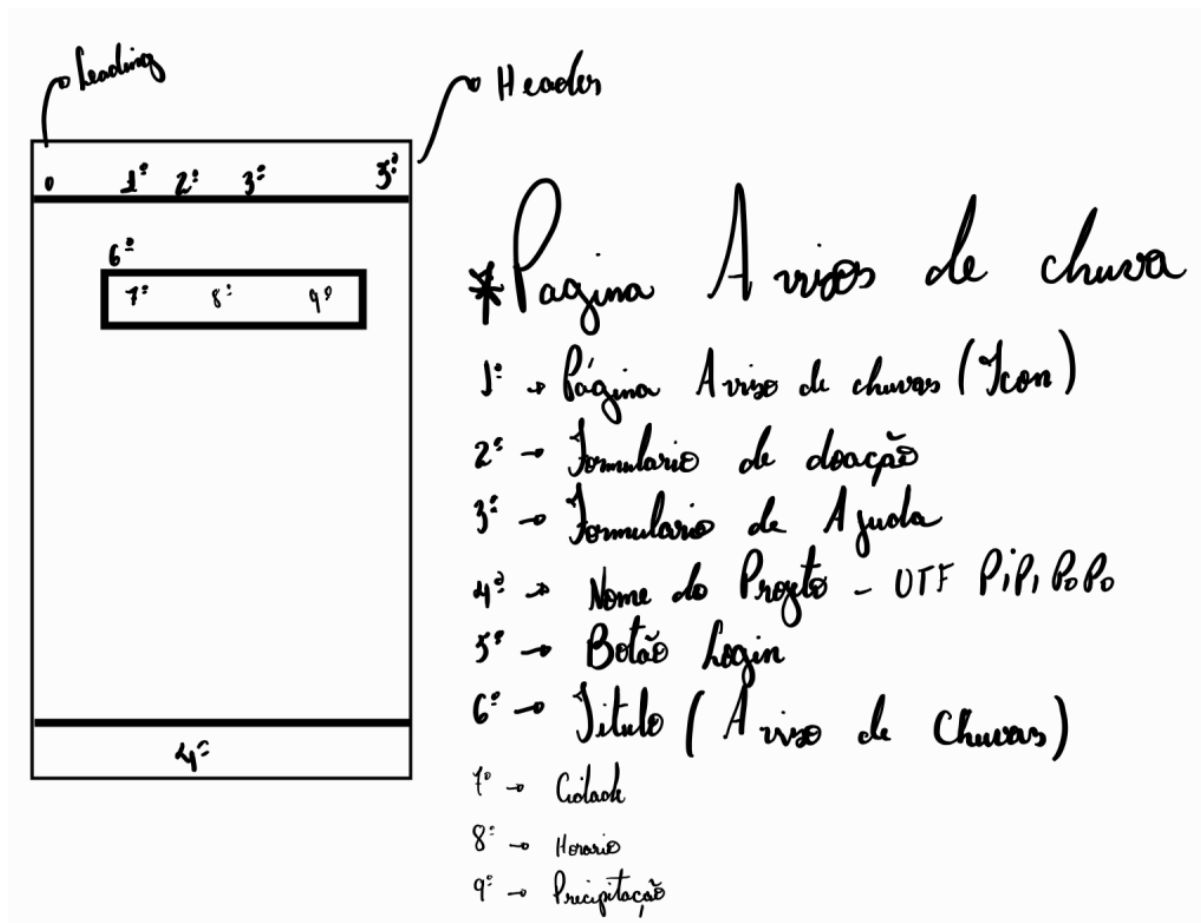


Figura 6. Esquema para a página de aviso de chuvas. Autoria própria (2023).

The screenshot shows a web page titled 'Aviso de Chuvas'. At the top, there is a navigation bar with links: AVISOS, DOAÇÕES, PEDIR AJUDA, and a LOGIN button. The main content area contains a form with the following fields:

- ESCOLHA A CIDADE: São José dos Pinhais (dropdown menu)
- DIA PREVISTO: 13/12 - Qua
- PRECIPITAÇÃO APROXIMADA: 0mm
- ALERTA DE ENCHENTE: Sem alerta

Below the form is a button labeled VERIFICAR. At the bottom of the page, there is a footer with the text: Copyright © 2023. UTFPR and Sobre a UTFPR.

Figura 7. Página de aviso de chuvas. Autoria própria (2023).

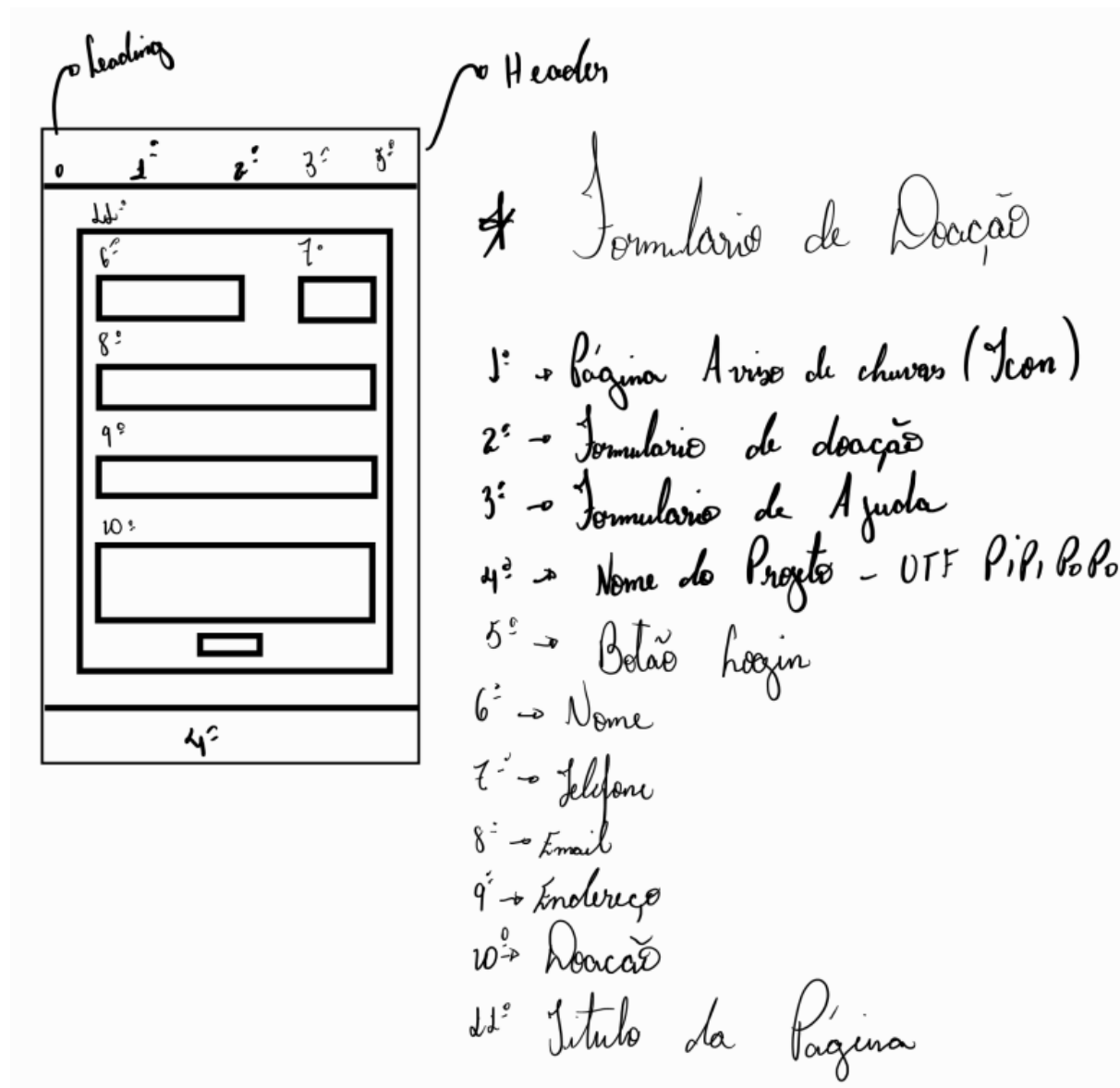






Figura 8. Esquema para a página com o formulário para efetuar doações. Autoria própria (2023).



 AVISOS

 DOAÇÕES

 PEDIR AJUDA

LOGIN

Formulário de Doação

INFORMAÇÕES DO DOADOR

NOME

Nome

TELEFONE

(00) 00000-0000

EMAIL

email@exemplo.com

ENDEREÇO

RUA

Rua/Avenida

CIDADE

Cidade

ESTADO

Paraná

CEP

00000-000

DOAÇÃO

ITENS QUE VOCÊ PODERÁ DOAR

Digite aqui os itens que você gostaria de doar

ENVIAR

Figura 9. Página com o formulário para efetuar doações. Autoria própria (2023).

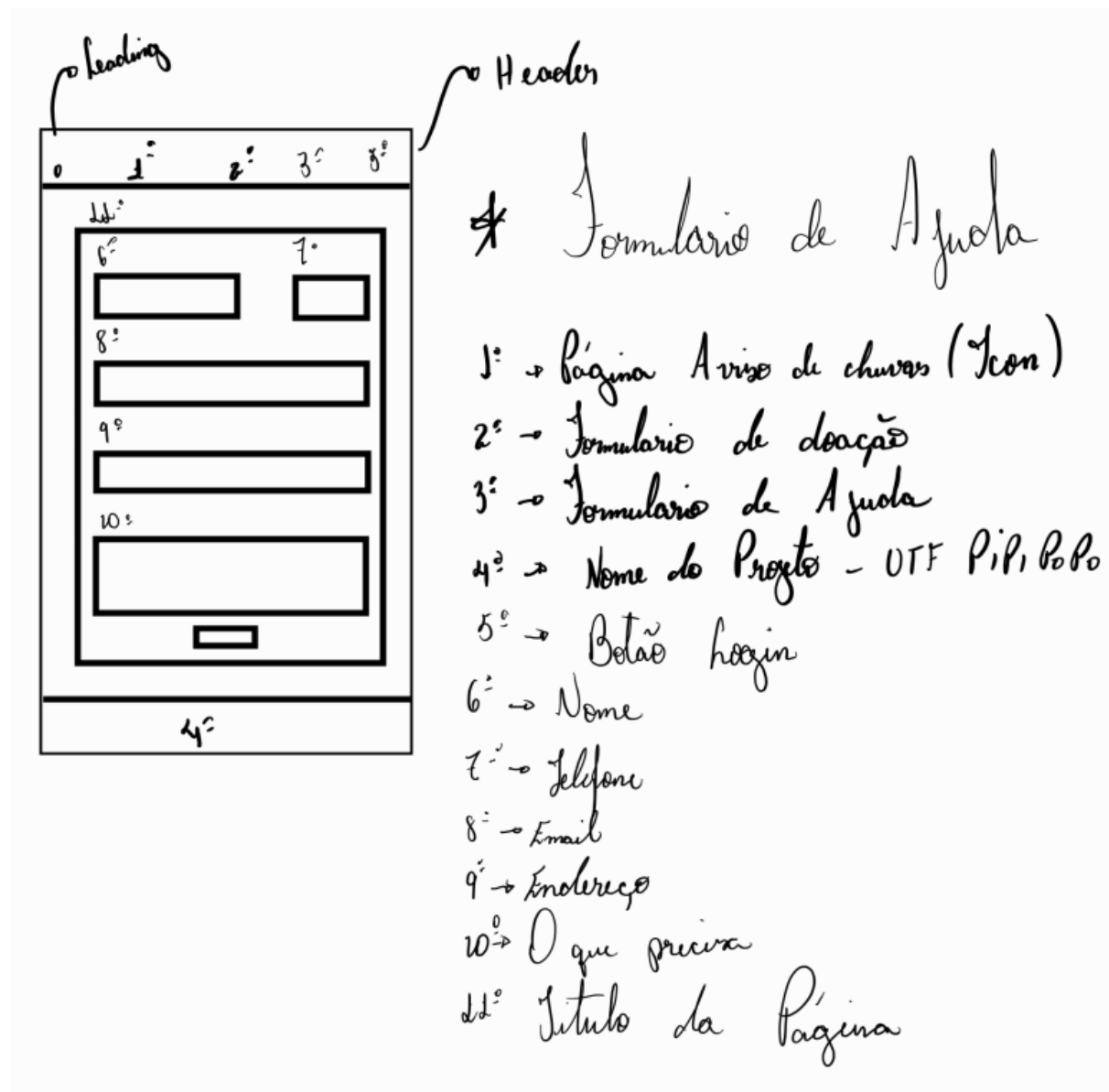


Figura 10. Esquema para a página com formulário para pedir ajuda. Autoria própria (2023).

Formulário de Requisição de Ajuda

INFORMAÇÕES DO RECEPTOR

NOME

TELEFONE

EMAIL

ENDEREÇO

RUA

CIDADE

ESTADO

CEP

AJUDA

ITEMS QUE VOCÊ PRECISA

Digite aqui os itens que você gostaria de doar

Figura 11. Página com formulário para pedir ajuda. Autoria própria (2023).

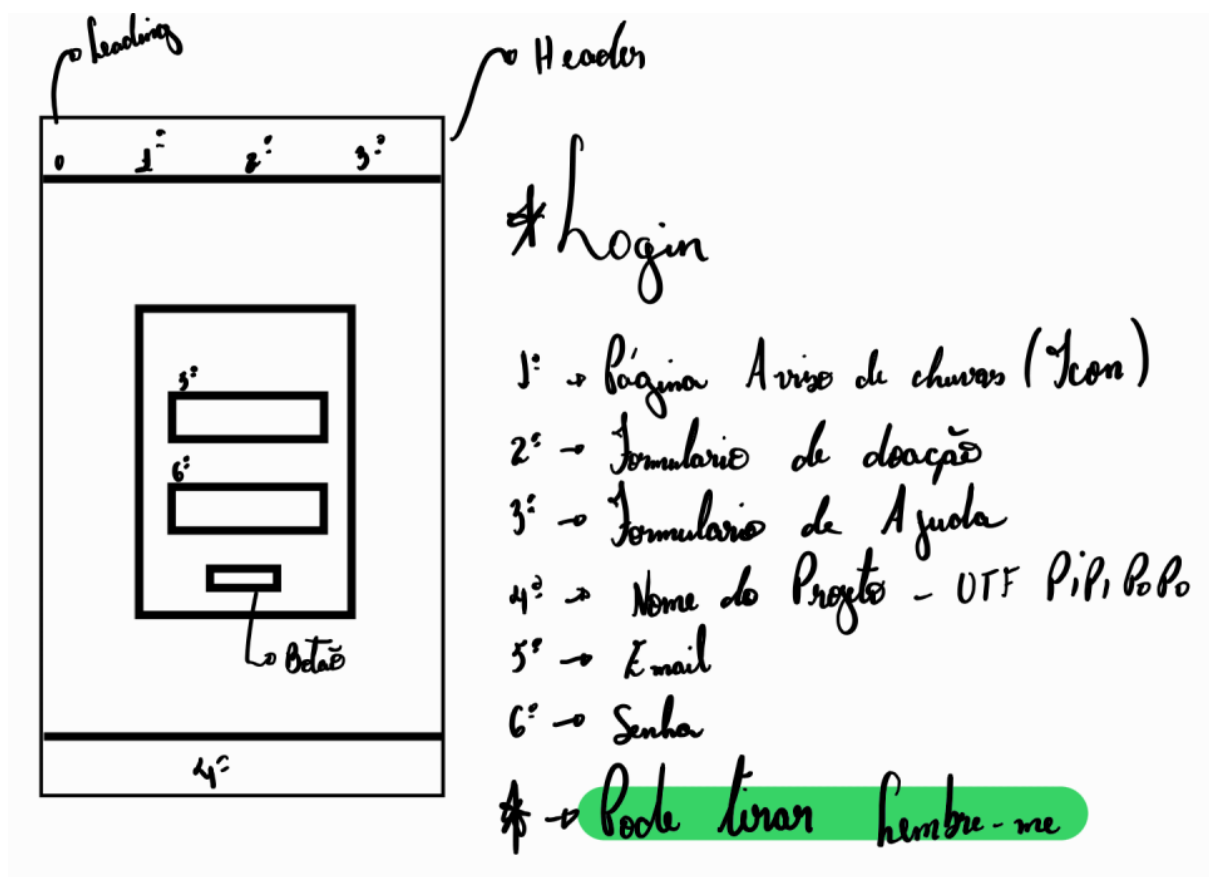


Figura 12. Esquema para a página de login dos administradores. Autoria própria (2023).

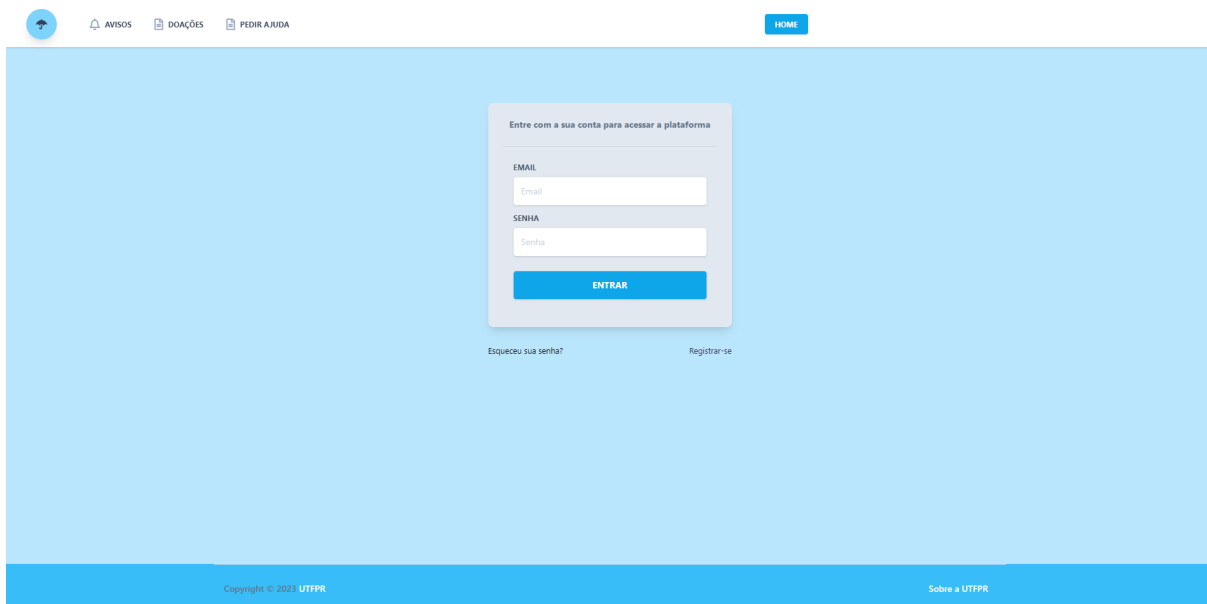


Figura 13. Página de login dos administradores. Autoria própria (2023).

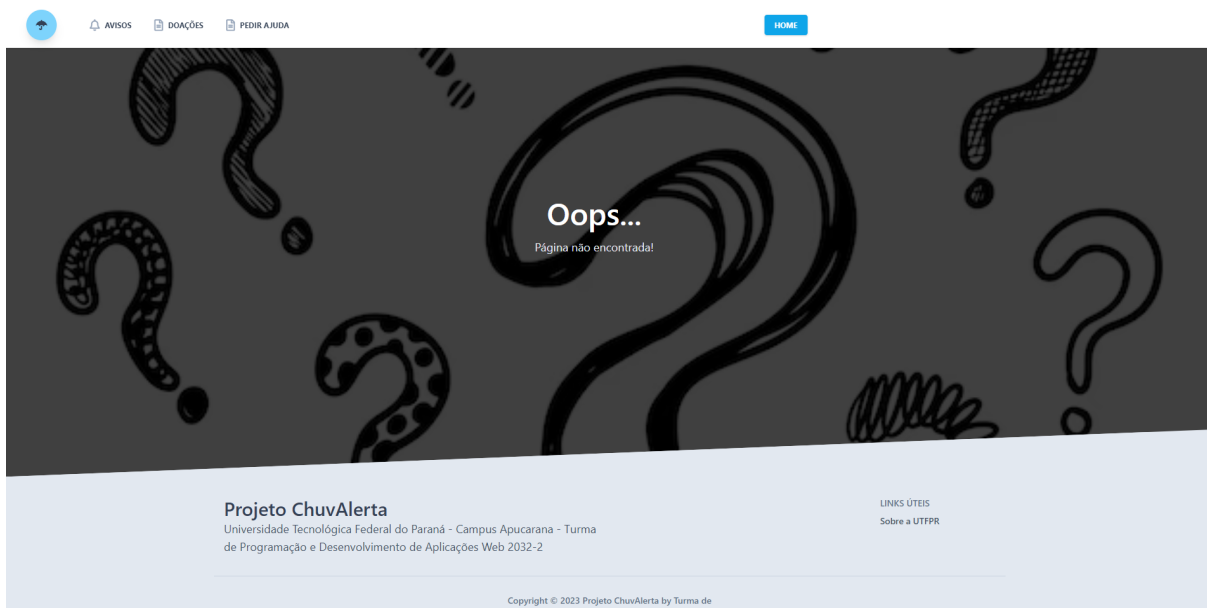


Figura 14. Página caso não encontre a desejada. Autoria própria (2023).

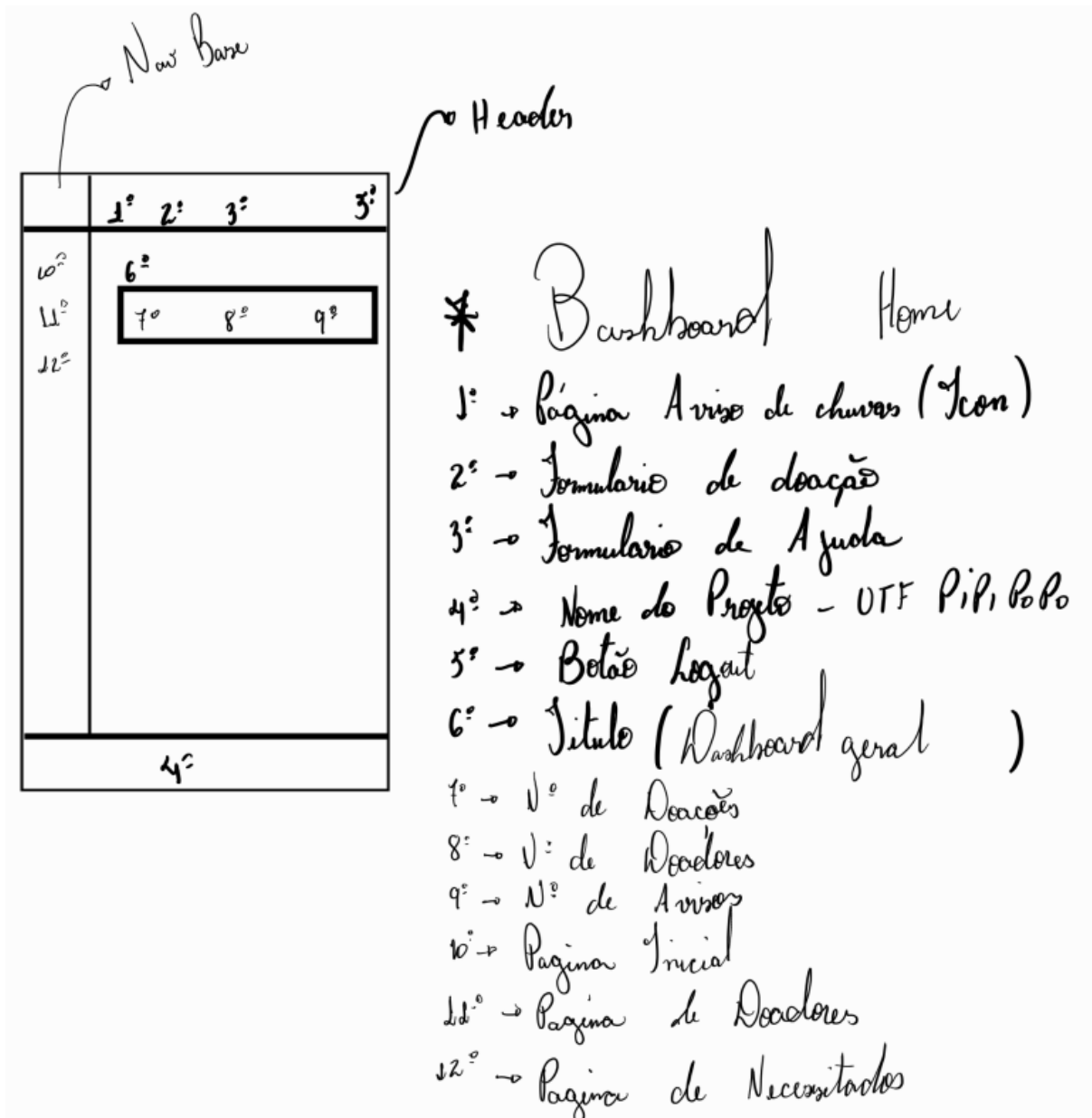


Figura 15. Esquema para o dashboard dos administradores. Autoria própria (2023).

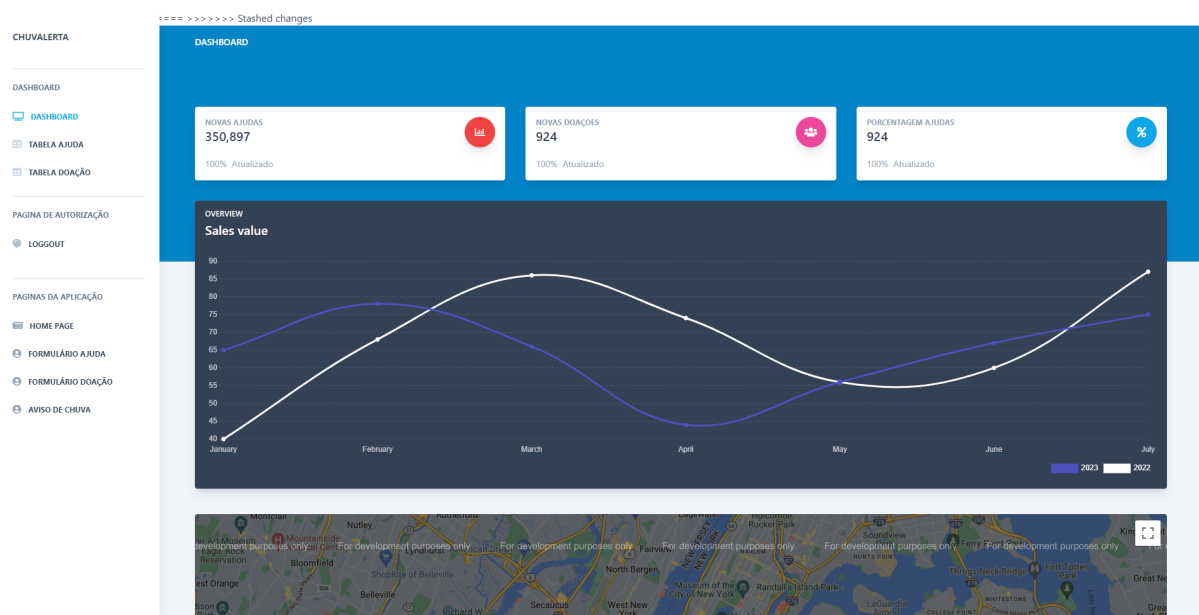


Figura 16. Dashboard dos administradores. Autoria própria (2023).

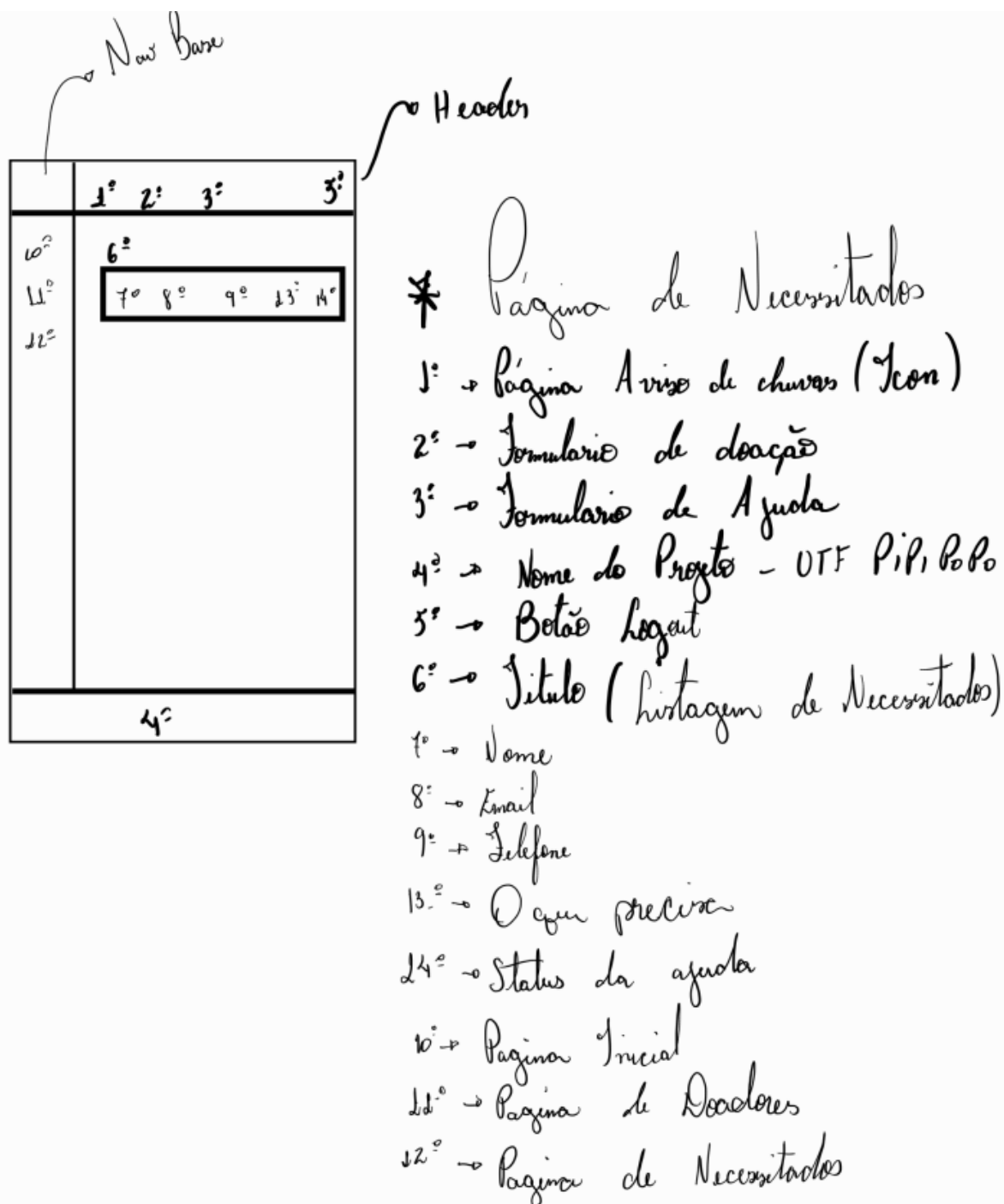


Figura 17. Esquema da página para exibir todas as ajudas solicitadas. Autoria própria (2023).

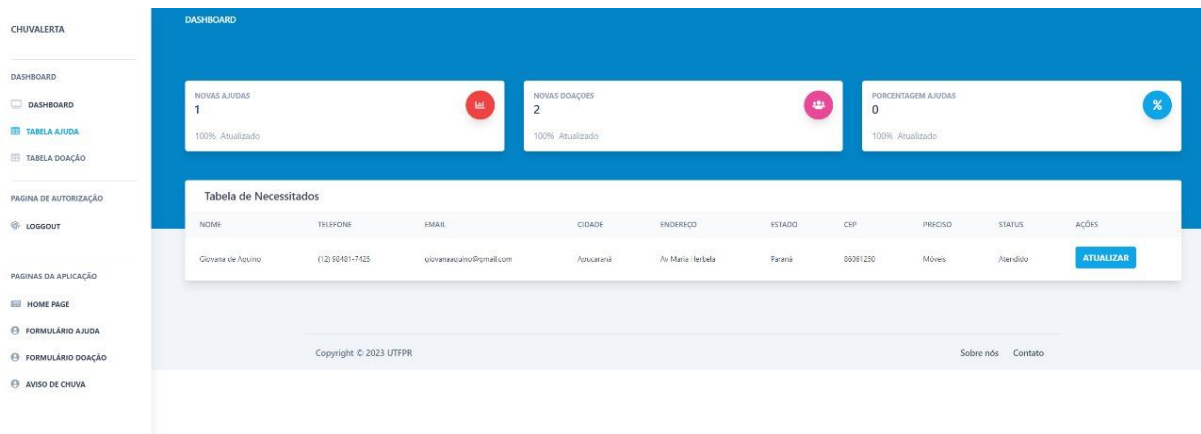


Figura 18. Página para exibir todas as ajudas solicitadas. Autoria própria (2023).

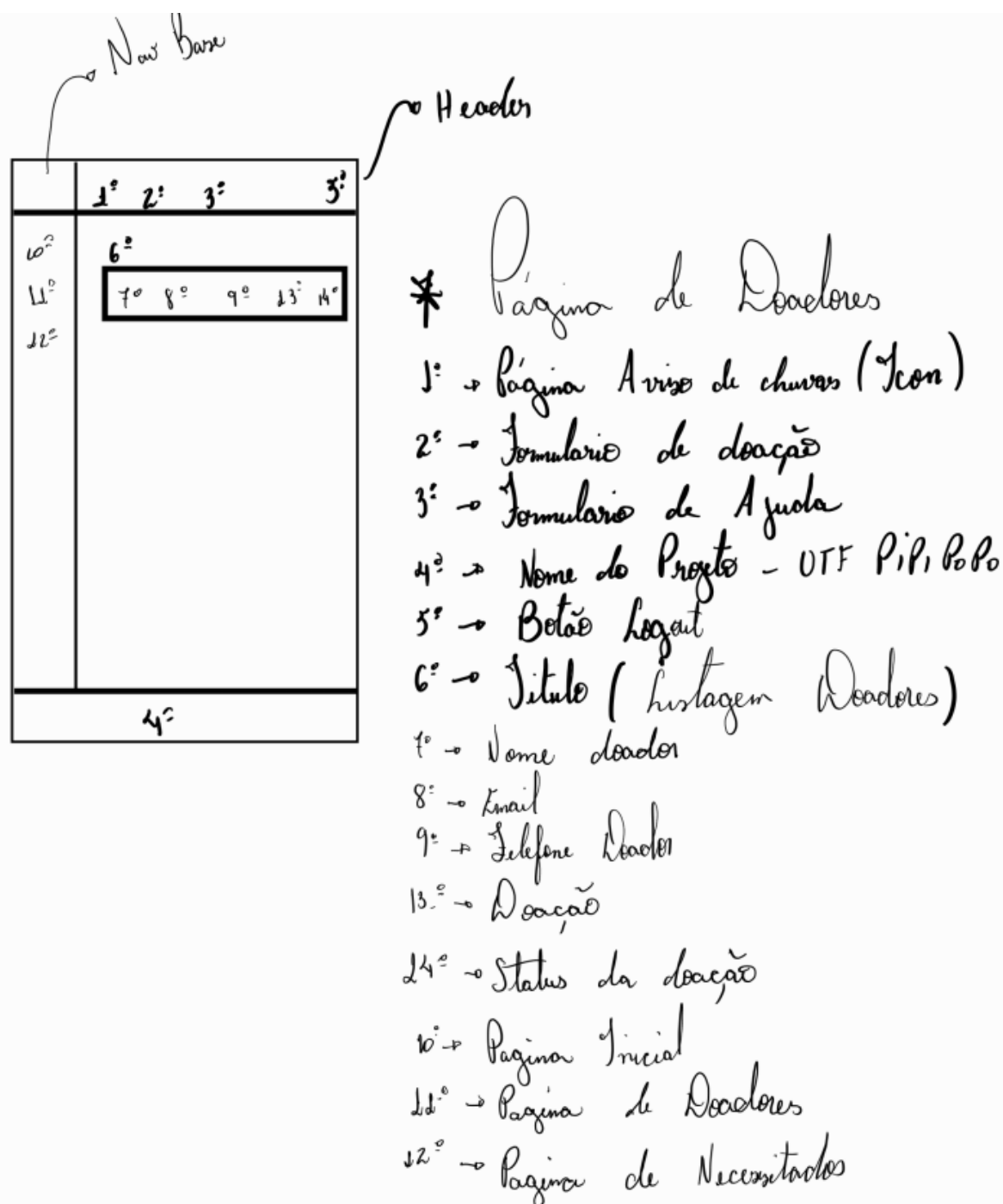


Figura 19. Esquema para exibir todas as doações efetuadas. Autoria própria (2023).

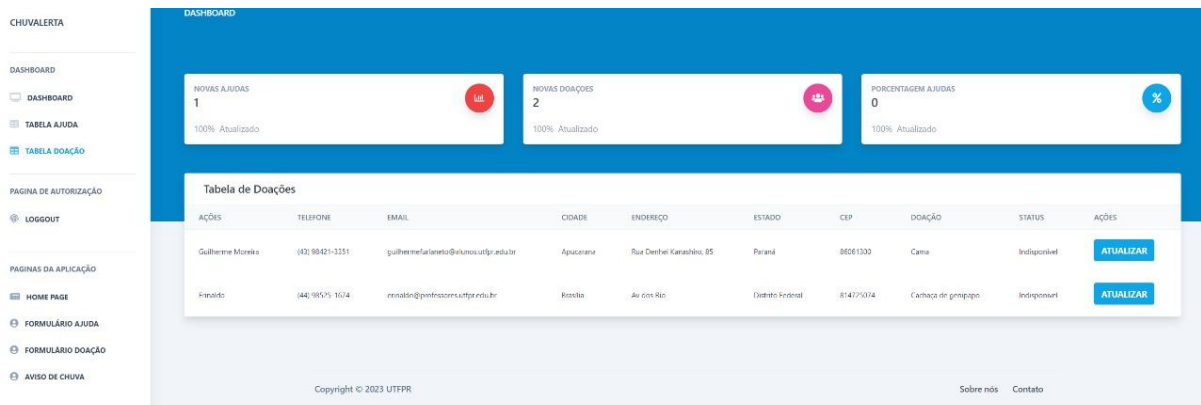


Figura 20. Página para exibir todas as doações efetuadas. Autoria própria (2023).

6. Arquitetura do Produto de Software

6.1. Padrões Arquiteturais

Como padrão arquitetural utilizamos microsserviços, nessa abordagem, dividimos o software em serviços independentes, cada um sendo uma aplicação autônoma e autocontida. Cada microsserviço é responsável por uma funcionalidade específica do sistema e se comunica com outros microsserviços por meio de APIs. Visto que teremos um front em react e um back em node esse padrão será bem útil para fazer essa conexão. Além do mais, a arquitetura em microsserviços permite escalabilidade, desacoplamento, manutenção e implantação independentes de cada serviço.

6.2. Linguagens e Frameworks

Para implementação do nosso software foi utilizado no front React combinada com o framework Tailwind, usaremos a flexibilidade do React para gerar componentes reutilizáveis e a capacidade do Tailwind CSS de estilizar rapidamente esses componentes com suas classes utilitárias. Essa combinação oferece um fluxo de desenvolvimento ágil e eficiente para criar interfaces de usuário bonitas e responsivas.

O React é uma biblioteca JavaScript amplamente utilizada para desenvolvimento de interfaces de usuário (UI). Utilizamos o React principalmente por utilizar uma abordagem baseada em componentes, permitindo que você divida a interface do usuário em partes independentes e reutilizáveis. Isso facilita a manutenção, o teste e a atualização de componentes individuais, além de melhorar o desempenho do aplicativo, pois o React é capaz de atualizar somente os componentes que foram modificados, evitando atualizações desnecessárias.

Podemos com essa biblioteca criar componentes que serão reutilizados em diferentes partes do software. Isso economizou tempo e esforço de desenvolvimento, ele também é acompanhado por um ecossistema rico de ferramentas e bibliotecas complementares, como o React Router para roteamento, o Redux para gerenciamento de estado, o Axios para requisições HTTP, entre outros.

Além de ser usado para desenvolvimento web, o React possui o React Native, que permite o desenvolvimento de aplicativos móveis nativos para iOS e Android usando JavaScript. Isso significa que podemos compartilhar grande parte do código entre o aplicativo web e o aplicativo móvel, reduzindo a duplicação de esforços e podendo assim com facilidade deixar nosso software disponível na Web e também em formato de aplicativo.

O backend de nossa aplicação é desenvolvido utilizando o Node.js, uma plataforma de código aberto construída sobre o motor V8 do Google Chrome. Optamos por utilizar o Node.js devido às suas características que proporcionam eficiência e versatilidade no desenvolvimento de aplicações web.

Node.js permite a execução de código JavaScript no servidor, o que simplifica a sincronização entre o frontend e o backend, facilitando a comunicação entre as camadas da aplicação. Além disso, o modelo de I/O não bloqueante do Node.js contribui para um desempenho otimizado, permitindo que a aplicação manipule várias operações simultaneamente, sem esperar pela conclusão de uma para iniciar outra.

A escalabilidade é outro benefício notável do Node.js, que se destaca na manipulação eficiente de um grande número de conexões concorrentes. Isso é especialmente crucial em ambientes onde a aplicação precisa lidar com uma alta carga de solicitações simultâneas, como em aplicações em tempo real, chats e streaming.

O ecossistema do Node.js é rico em módulos e pacotes disponíveis no npm (Node Package Manager), o que facilita a integração de diversas funcionalidades e bibliotecas de terceiros. Isso acelera o desenvolvimento, uma vez que muitas soluções já estão prontas e podem ser incorporadas com facilidade.

Em resumo, o Node.js é uma escolha estratégica para o desenvolvimento do backend de nossa aplicação devido à sua eficiência, capacidade de lidar com operações assíncronas, escalabilidade e rica comunidade de desenvolvedores. Essas características combinadas ajudam a criar uma base sólida e eficiente para garantir o desempenho e a robustez de nossa aplicação.

6.3. Base de Dados

Para a nossa base de dados foi escolhido o SGBD (Sistema de Gerenciamento de Base de Dados) PostgreSQL visto que ele é um SGBD código aberto, o que significa que podemos usá-lo, modificá-lo e distribuí-lo gratuitamente, sendo bem mais rentável para a implementação do software.

Trazendo um pouco da história, o PostgreSQL é um dos SGBDs relacionais mais antigos e estabelecidos disponíveis. Ele tem sido desenvolvido e aprimorado há mais de 30 anos e é conhecido por sua estabilidade, confiabilidade, robustez e flexibilidade, resumindo esse SGBD possui uma ampla gama de recursos avançados. Ele suporta consultas complexas, índices, gatilhos, visões, transações ACID (Atomicidade, Consistência, Isolamento e Durabilidade) e muito mais.

Além disso, o Postgres é projetado para lidar com grandes volumes de dados e cargas de trabalho intensivas. Ele possui otimizadores de consultas sofisticados que podem melhorar o desempenho das consultas e suporta a replicação e a partitioning (divisão) de dados para

escalabilidade horizontal. Ele oferece também recursos avançados de segurança, incluindo autenticação, criptografia de dados em trânsito e em repouso, controle de acesso granular e auditoria, ajudando também nas questões de segurança do nosso projeto.

Na figura 12 temos um exemplo da modelagem do banco de dados do nosso software, que já está normalizado.

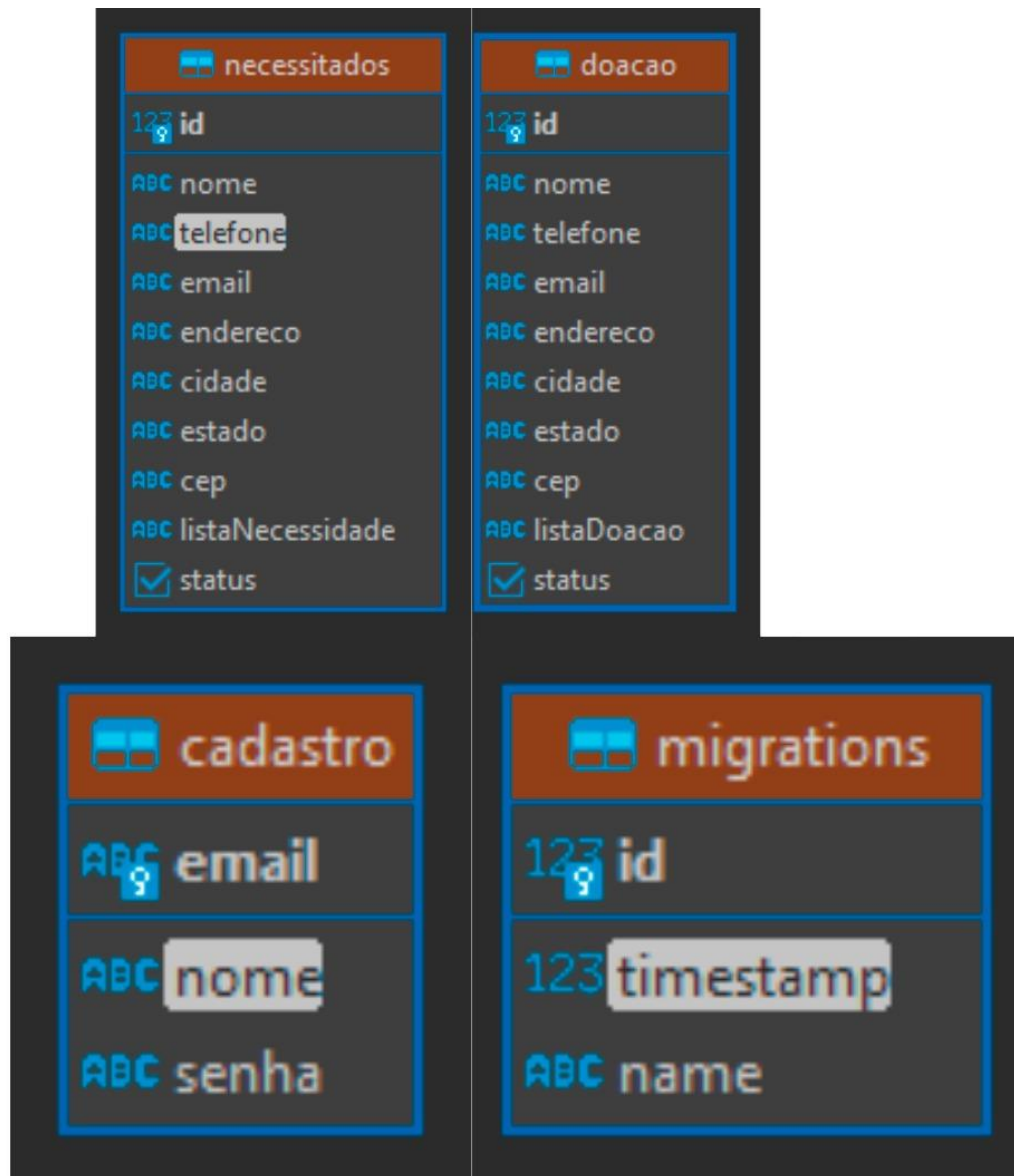


Figura 12. Diagrama do banco de dados simplificado. Autoria própria (2023).

6.4. Gerenciamento de Configuração e Versão

Para gerenciar configurações de versões e garantir sua evolução usamos uma tecnologia de software oferecida pelo GitHub.

O GitHub é uma plataforma de desenvolvimento colaborativo que oferece controle de versão utilizando o sistema Git. Ele é amplamente utilizado por desenvolvedores para

gerenciar e compartilhar código-fonte de projetos de software. Uma das razões principais para a escolha do GitHub para seus projetos reside em sua eficiência e facilidade de uso.

Primeiramente, o GitHub proporciona um ambiente centralizado para armazenar e organizar o código-fonte de maneira acessível. Isso facilita a colaboração entre membros da equipe, permitindo que eles contribuam, revisem e modifiquem o código de forma transparente. A funcionalidade de controle de versão oferecida pelo Git permite rastrear alterações, facilitando a identificação e resolução de conflitos.

A natureza distribuída do Git, que o GitHub adota, é outra vantagem significativa. Isso possibilita que cada desenvolvedor tenha uma cópia completa do repositório em seu ambiente local, promovendo a independência e a flexibilidade durante o desenvolvimento. Além disso, o GitHub oferece ferramentas poderosas para o gerenciamento de problemas, rastreamento de projetos e integração contínua, simplificando ainda mais o ciclo de vida do desenvolvimento de software.

A comunidade ativa e a vasta quantidade de repositórios públicos no GitHub também desempenham um papel crucial na escolha da plataforma. Isso permite que os desenvolvedores acessem uma variedade de bibliotecas, frameworks e soluções pré-existentes, acelerando o desenvolvimento e promovendo boas práticas de codificação.

Em resumo, a escolha do GitHub para seus projetos é respaldada pela eficiência, facilidade de colaboração, controle de versão robusto e a riqueza de recursos oferecidos pela plataforma. Esses aspectos contribuem para um ambiente de desenvolvimento mais ágil, transparente e colaborativo.

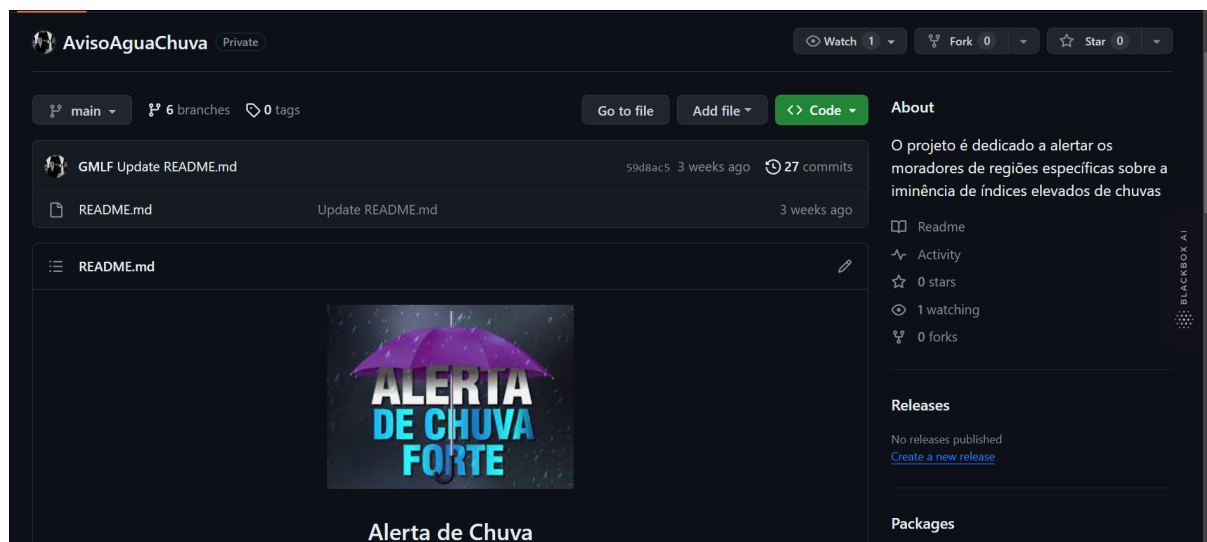


Figura 13. Exemplo do aplicativo GitHub. Autoria Própria (2023).

7. Privacidade e Segurança

Privacidade e segurança são conceitos relacionados à proteção de informações e dados pessoais, garantindo que sejam mantidos em sigilo, usados de forma adequada e protegidos contra acesso não autorizado, uso indevido ou divulgação não autorizada.

No momento nossas senhas estão entrando no banco sem alguma criptografia, o que é uma falha de segurança imensa, devido ao tempo essa estratégia foi adotada mas em uma segunda etapa desse projeto precisará ser ajustado.

Para o backup do software seria necessário realizar um backup por dia, a fim de minimizar a perda de dados visto que requisições ocorrem com um fluxo bem grande. Vale também ressaltar que os backups deverão ser armazenados em um local separado do servidor, um possível armazenamento criptografado e em nuvem será a solução.

Para a parte de autenticação utilizamos apenas uma comparação de email e senha que estão inseridos no banco nosso caso email e senha, nossa aplicação não obtém tokens de acesso para acessar os recursos protegidos.

Para seguir os Princípios da Privacy by Design e conformidade com a LGPD coletamos apenas os dados necessários para o funcionamento da app, com algumas poucas informações sensíveis somente disponibilizadas para administradores.

Privacidade e segurança estão intrinsecamente relacionadas e são fundamentais para garantir a confiança dos usuários e proteger seus dados pessoais contra abusos. Nosso software no momento não trata os dados com cautela e segurança.

8. Trabalhos Futuros

Considerando o desenvolvimento contínuo da aplicação de notificação de chuvas intensas, identificamos várias oportunidades de trabalhos futuros e melhorias que podem aprimorar significativamente a experiência dos usuários e a eficácia da plataforma. Abaixo estão algumas sugestões para prosseguimento no desenvolvimento:

Melhorias na Interface do Usuário (UI/UX): Durante a evolução da aplicação, é recomendável realizar pesquisas de usabilidade para identificar áreas passíveis de melhorias na experiência do usuário. Implementar feedbacks visuais e explorar temas escuros podem ser estratégias eficazes para aprimorar a acessibilidade e a estética geral da aplicação.

Integração com Redes Sociais: A adição de recursos de compartilhamento e suporte a login através de contas de redes sociais pode ampliar a visibilidade da aplicação, incentivando maior participação da comunidade e simplificando o processo de adesão.

Aprimoramento na Coordenação de Doações: Implementar um sistema de rastreamento de doações proporciona transparência aos doadores, permitindo-lhes acompanhar o destino de suas contribuições. Notificações automáticas ao concluir a entrega de doações podem fortalecer o engajamento dos doadores.

Aplicação de Tecnologias Emergentes: A exploração de tecnologias emergentes, como inteligência artificial para análise de padrões meteorológicos, e integração com dispositivos IoT para dados climáticos locais em tempo real, pode elevar a precisão das previsões e a relevância da aplicação.

Expansão para Outras Regiões: Avaliar a expansão da aplicação para outras cidades ou regiões, adaptando-a às necessidades específicas de cada local, e considerar traduções para outros idiomas para alcançar uma audiência mais diversificada.

Funcionalidades Adicionais para Administradores: Implementar ferramentas avançadas de análise e relatórios para administradores, juntamente com funcionalidades de comunicação interna para facilitar a coordenação entre diferentes organizações de ajuda.

Desenvolvimento de Aplicativo Móvel: Criar versões para dispositivos móveis (Android e iOS) e incorporar notificações push para oferecer acesso mais fácil e imediato, mantendo os usuários atualizados sobre alertas meteorológicos e atualizações.

Parcerias Estratégicas: Buscar parcerias com instituições meteorológicas, empresas de logística e organizações de ajuda para fortalecer a rede de apoio e ampliar os recursos disponíveis na aplicação.

Capacitação da Comunidade: Desenvolver módulos de treinamento online para capacitar a comunidade sobre como utilizar a aplicação de maneira eficaz e segura durante eventos climáticos adversos.

Avaliação de Impacto Social: Conduzir estudos para avaliar o impacto social da aplicação, medindo a eficácia das doações e a resiliência da comunidade em situações de emergência.

Essas sugestões representam um guia para direcionar os esforços futuros, alinhando a aplicação às necessidades da comunidade e garantindo sua relevância e utilidade contínuas.

9. Conclusão

Em conclusão, o projeto da aplicação de notificação de chuvas intensas representa uma iniciativa significativa na utilização da tecnologia para mitigar os impactos adversos de eventos climáticos extremos. Através da implementação de um sistema inovador, buscamos fornecer à comunidade de Apucarana uma ferramenta eficiente para o alerta precoce de chuvas críticas, coordenação de doações e auxílio em momentos de emergência.

O desenvolvimento desta aplicação reflete a dedicação à segurança e bem-estar da comunidade, abordando desafios reais enfrentados durante condições climáticas adversas. A integração de uma API externa para o cálculo de chuvas críticas, o sistema de login administrativo e a facilidade de integração para funcionalidades futuras destacam-se como pilares fundamentais do escopo do projeto.

Ao considerar futuros desenvolvimentos, é evidente que há espaço para aprimoramentos e expansões significativas. As sugestões propostas, como melhorias na interface do usuário, integração com redes sociais, coordenação avançada de doações e a exploração de tecnologias emergentes, fornecem um roadmap valioso para elevar a aplicação a novos patamares de utilidade e eficácia.

O projeto não apenas oferece uma solução para as necessidades imediatas da comunidade, mas também estabelece uma base sólida para o crescimento sustentável. O compromisso com requisitos não funcionais, como desempenho, segurança e escalabilidade, demonstra a busca pela excelência em todas as fases do ciclo de vida da aplicação.

Por fim, a colaboração contínua com a comunidade, a avaliação constante de seu impacto social e a disposição para explorar novas parcerias refletem a natureza dinâmica e adaptável deste projeto. À medida que avançamos, esperamos que esta aplicação não seja apenas uma ferramenta tecnológica, mas uma fonte de apoio e resiliência para a comunidade de Apucarana diante de desafios climáticos imprevistos. Este projeto é mais do que uma aplicação; é uma expressão tangível do comprometimento com a segurança e o apoio mútuo em tempos de necessidade.